Training-free Detection of AI-generated images via Cropping Robustness

Anonymous Author(s)
Affiliation
Address
email

Abstract

AI-generated image detection has become crucial with the rapid advancement of vision-generative models. Instead of training detectors tailored to specific datasets, we study a training-free approach leveraging self-supervised models without requiring prior data knowledge. These models, pre-trained with augmentations like `RandomResizedCrop`, learn to produce consistent representations across varying resolutions. Motivated by this, we propose **WaRPAD**, a training-free AI-generated image detection algorithm based on self-supervised models. Since neighborhood pixel differences in images are highly sensitive to resizing operations, WaRPAD first defines a base score function that quantifies the sensitivity of image embeddings to perturbations along high-frequency directions extracted via Haar wavelet decomposition. To simulate robustness against cropping augmentation, we rescale each image to a multiple of the model's input size, divide it into smaller patches, and compute the base score for each patch. The final detection score is then obtained by averaging the scores across all patches. We validate WaRPAD on real datasets of diverse resolutions and domains, and images generated by 23 different generative models. Our method consistently achieves competitive performance and demonstrates strong robustness to test-time corruptions. Furthermore, as invariance to `RandomResizedCrop` is a common training scheme across self-supervised models, we show that WaRPAD is applicable across self-supervised models.

1 Introduction

AI-generated image detection aims to design reliable metrics that can distinguish between real and synthetically generated images. This task has become increasingly critical with the advent of highly capable text-to-image (T2I) generative models that can produce photorealistic outputs, which may be exploited for vicious purposes (*e.g.*, fake news [1], deepfakes [2]). Most existing detection approaches [3, 4] are trained to recognize specific real data distributions (*e.g.*, ImageNet [5], LSUN [6]). However, the scope of real image distributions that current detection approaches can effectively cover remains extremely limited compared to the diversity of generated images. Furthermore, a training dataset for detection may contain artifacts (*e.g.*, WebP compression in LSUN), which may lead the detector to overfit the artifacts and may fail to be robust in test-time corruptions [7]. Consequently, there is a growing need for detection methods that can operate universally across diverse domains without relying on the constraints of predefined real image distributions.

In response to this growing demand, this paper focuses on *training-free* detection methods where no prior knowledge of real image distributions is given. Prior training-free detection methods have design score functions based on representations extracted from large-scale pre-trained foundation models. One representative line of work leverages the representation of latent diffusion models (LDMs) (*e.g.*, Stable Diffusion [8], Midjourney [9]). For instance, AEROBLADE [10] detects LDM-generated

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

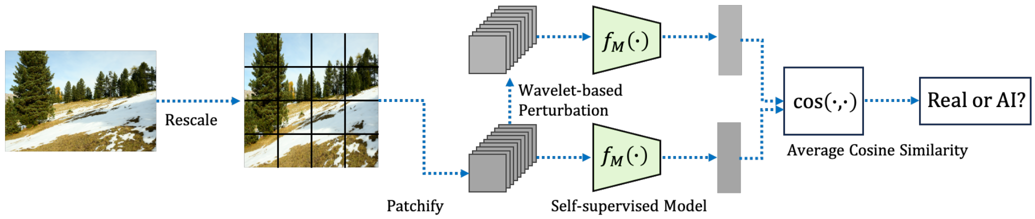


Figure 1: **Conceptual illustration of our method WaRPAD.** We first rescale and patchify the given image to the batch of patches. Then, we perturb the patches on the high-frequency direction of Haar wavelet decomposition. Our final score function is the averaged cosine similarity between the perturbed and non-perturbed patches' features through the self-supervised model.

images by measuring the autoencoder reconstruction loss, while Manifold Bias [11] proposes a curvature-based metric in the latent space of the LDM. However, the performance of these approaches is often tied closely to the choices of the LDMs, and their generalizability to other generative models remains uncertain.

Another stream of training-free research utilizes representations from self-supervised models, such as DINOv2 [12]. These models benefit from pre-training on a wide range of real-world images and are often applied as generalist models for various downstream tasks [13, 14]. Some prior works [15, 16] attempt to utilize self-supervised models on AI-generated image detection by introducing simple perturbations (*e.g.*, Gaussian noise or Gaussian blurring) and measure the cosine similarity between the original and perturbed image embeddings. However, their performance often lags behind the diffusion-based training-free baselines (see Section 4.2).

Contribution. In this study, we approach AI-generated image detection from a data augmentation perspective, utilizing foundation models that have been pre-trained on real images. Our primary motivation stems from the observation that these models are trained to produce consistent embedding representations between an original image and its randomly cropped and resized variants. We hypothesize that embeddings of AI-generated images exhibit lower robustness to such `RandomResizedCrop` (RRC) transformations than those of real images.

To investigate this hypothesis, we examine how RRC affects the high-frequency components of images, as obtained through wavelet decomposition. Notably, we observe that even when the cropped image closely matches the original in size, RRC introduces substantial variations in the difference in the neighborhood pixels. This indicates that RRC acts as an effective perturbation on the high-frequency components extracted via Haar wavelet decomposition. Given that foundation models are typically trained to be invariant to such perturbations on the real images, we propose a detection score function that quantifies embedding sensitivity to high-frequency distortions as a signal for distinguishing real and synthetic images.

Furthermore, to simulate the effects of RRC in a more structured and deterministic manner, we resize each image to a multiple ($\times K^2$) of the default input resolution and partition it into K^2 patches of the default resolution. We then apply the proposed score function to each patch and aggregate the results. This patch-based perturbation consistently reveals a greater discrepancy in AI-generated images, demonstrating the effectiveness of our method. Figure 1 shows the computation of our unified method, **WaRPAD**: **Wa**velet, **R**esizing, and **P**atchifying for **A**I-generated image **D**etection.

We conduct extensive evaluations of WaRPAD across multiple AI-generated image detection benchmarks. These benchmarks span various generative model types, including LDMs, proprietary models (*e.g.*, Firefly [17], Dall-E [18]), and generative adversarial network (GAN) [19] architectures, as well as multiple image domains. Our method consistently outperforms other training-free baselines in all settings. Notably, we observe an improvement of **6.5 $\sim$ 24.7% in AUROC** over prior methods based on the same DINOv2 model. Furthermore, we evaluate the robustness of our method against various image corruptions and show that it maintains competitive performance under such conditions, surpassing other detection methods.

In brief, our contributions are summarized as follows.

- We propose WaRPAD that applies a self-supervised foundation model's robustness on RRC augmentation for detecting AI-generated images (Section 3).

- WaRPAD outperforms existing training-free AI-generated image detection methods in every benchmark consistently (Section 4.2). Furthermore, WaRPAD is robust to test-time corruption of the examined images (Section 4.3).
- Our analysis suggests that a self-supervised model trained to be invariant under RRC can be applied for AI-generated image detection, supporting the generalizability of WaRPAD (Section 4.3).

2 Preliminary

First, we introduce the AI-generated image detection framework. Furthermore, we discuss the self-supervised models and their key data augmentation strategy. For the broader overview, we refer to Deng et al. [20] and Uelwer et al. [21] for the comprehensive survey.

2.1 AI-generated Image Detection

AI-generated image detection aims to design a scoring function $S(\mathbf{x})$ that determines whether a given image $\mathbf{x}$ has been acquired from the real world (*i.e.*, $S(\mathbf{x}) \geq \tau$) or generated by a generative model (*i.e.*, $S(\mathbf{x}) < \tau$). While most existing approaches train $S(\mathbf{x})$ using labeled datasets containing both real and synthetic images, we assume a more practical setting in which such training data is not available. This setup allows us to assess the generalizability of detection methods to previously unseen data distributions.

Recently, this generalizability to unseen distributions has been discussed in training-based approaches as well. ZED [3] proposes an entropy-based score that can be trained solely on real image distributions without requiring any synthetic examples. Cozzolino et al. [22] train a Linear SVM classifier on CLIP [23] embeddings extracted from image pairs (*e.g.*, MS-COCO [24] and LDM-generated [8]) and evaluate its performance on unseen images (*e.g.*, Raise-1K [25] and Firefly [17]). Rajan and Lee [7] improve robustness to post-processed images by retraining the final layer of the pre-trained detection model. Extending beyond these approaches, we focus on a training-free detection setting in which no real or fake data is provided during the design of the detection score.

Training-free detection aims to construct a universal score based on the outputs of a pre-trained foundation model. Such a foundation model may either be a generative model itself (*e.g.*, LDM [8]) or a model trained on diverse real-world images (*e.g.*, DINOv2 [12], CLIP [23]). In this work, we adopt the latter approach, leveraging models pre-trained on real data to guide our detection framework.

2.2 Self-supervised Models

Self-supervised models aim to learn robust representations from large-scale unannotated images that can be generalized to a wide range of downstream tasks. Self-supervised learning methods apply various augmentations, often referred to as views, to a single image and train the model to maximize the similarity between outputs corresponding to different views. This paradigm has been implemented in various forms, including contrastive learning (*e.g.*, SimCLR [26], Moco [27]), clustering-based (*e.g.*, SwaV [28]), and knowledge distillation (*e.g.*, DINO [29]). Since providing a comprehensive overview of self-supervised learning (SSL) is beyond the scope of this paper, we focus on the DINOv2 [12] model and the augmentation strategies.

DINOv2 is a Vision Transformer [30] model trained on a web-scale LVD-142M dataset, building upon the iBOT [31] framework. DINOv2 tokenizes each input image and outputs patch-level embeddings along a `[CLS]` token embedding that summarizes the entire image. The model is trained using a teacher-student framework, where the student network is optimized to maximize the similarity between its output and the output of the teacher network given relatively mild augmentations. The teacher network is updated via an exponential moving average of the student parameters.

A core component of the data augmentation applied to both networks is `RandomResizedCrop` (RRC), which randomly crops the region from the input image and resizes it to the given resolution. This operation enforces spatial invariance by encouraging the model to recognize that different subregions of an image correspond to the same underlying semantic content. Due to its effectiveness, RRC has become a standard augmentation technique across many SSL frameworks. In this work, we adopt RRC as the key motivation for designing our AI-generated image detection method.

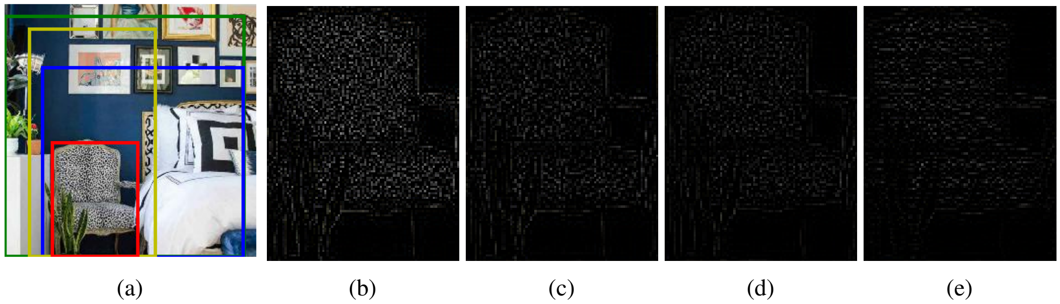


Figure 2: **Motivation for the Haar-wavelet perturbation sensitivity score.** **(a)**: The original image along with the designated region for high-frequency visualization, marked in **red**. To simulate the effect of `RandomResizedCrop` (RRC), we apply cropping regions indicated in green, blue, and yellow. **(b)**: The high-frequency component of the original (uncropped) image obtained via Haar wavelet decomposition. **(c)**, **(d)**, **(e)**: The corresponding high-frequency components of the RRC-transformed images, where the cropping regions are defined by the green, blue, and yellow boxes, respectively.

3 Method

This section introduces our method, WaRPAD, inspired by the `RandomResizedCrop` (RRC) operation. We first propose a score function that measures the sensitivity of self-supervised model features to perturbations along high-frequency directions induced by wavelet decomposition (Section 3.1). Subsequently, we simulate local robustness by introducing a rescaling-and-patching paradigm that partitions resized images into subregions for evaluation (Section 3.2).

3.1 Base Detection Score

We hypothesize that measuring an image's robustness to RRC can serve as an effective score for detecting AI-generated images. However, RRC involves random cropping over a broad hyperparameter space, which introduces variance and may discard key image content of the original image. To address this, we do not apply RRC directly but instead propose a score function that approximates its effect. Specifically, we examine how RRC alters the high-frequency components of the image using wavelet decomposition.

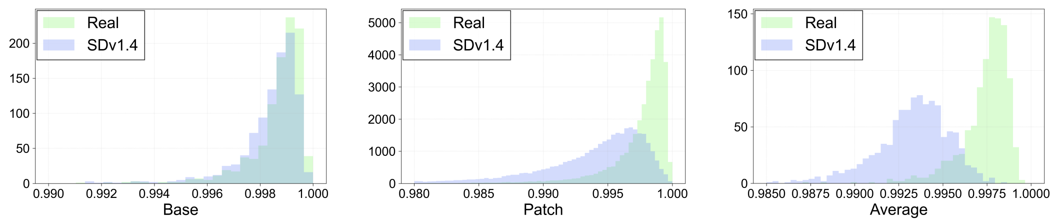
For the analysis, we visualize the high-frequency components obtained via Haar wavelet decomposition under multiple instances of RRC, as illustrated in Figure 2a. Figure 2b presents the high-frequency component within the red-marked region of the original image, whereas Figures 2c, 2d, and 2e show the corresponding components after applying RRC with green, blue, and yellow cropping regions, respectively. These comparisons reveal that RRC introduces substantial variations in the high-frequency components, even when the cropped region closely matches the original image in size. Based on this observation, we define our base score function as the model's sensitivity to perturbations along the high-frequency directions, formally expressed as follows:

$$\mathrm{HFwav}(\mathbf{x}) = \frac{\mathbf{f}_{\mathrm{M}}(\mathbf{x}) \cdot \mathbf{f}_{\mathrm{M}}(\mathbf{x} - \alpha \mathrm{HF}(\mathbf{x}))}{\|\mathbf{f}_{\mathrm{M}}(\mathbf{x})\| \|\mathbf{f}_{\mathrm{M}}(\mathbf{x} - \alpha \mathrm{HF}(\mathbf{x}))\|}, \tag{1}$$

where $\mathbf{f}_{\mathrm{M}}$ denotes the feature output of the self-supervised model M (e.g., the `[CLS]` token output of DINOv2), HF represents the high-frequency component derived from wavelet decomposition, and $0 < \alpha < 1$ is the perturbation weight. The sensitivity score function HFwav quantifies the change of model M to perturbations along high-frequency directions. Our central hypothesis is that model M, having been trained on real images, is encouraged to be invariant to such high-frequency perturbations. Accordingly, we expect the HFwav score to be higher for real images than the AI-generated images.

3.2 WaRPAD

We further propose a test-time augmentation strategy inspired by RRC. Our main idea is to deterministically simulate multiple instances of RRC by explicitly rescaling the image and thereby patchifying



(a) (b) (c)

Figure 3: **Effect of `RescaleNPatchify`. (a)**: Histogram of real and SDv1.4-generated data examined by HFwav. **(b)**: Histogram of real and SDv1.4-generated data examined on patches augmented by `RescaleNPatchify`. **(c)**: Histogram of real and SDv1.4-generated data examined on our WaRPAD score function.

Table 1: **Benchmarks in the main experiment.**

Benchmark	Real Dataset	# of Generative Models	Input Resolution	# of Test Images per Dataset
Synthbuster [32]	Raise-1K [25]	9 (LDM, Proprietary Model)	Varying ($256 \times 256 \sim 4928 \times 3264$)	1000
GenImage [33]	ImageNet [5]	8 (GAN, Diffusion Model, LDM)	Varying ($128 \times 128 \sim 1024 \times 1024$)	$6000 \sim 8000$
Deepfake-LSUN-Bedroom [34]	LSUN [6]	10 (GAN, Diffusion Model)	256×256	10000

the image to patches with the same sizes as follows:

$$\texttt{RescaleNPatchify}(\mathbf{x}) = \texttt{Patchify}\left(\texttt{Rescale}(\mathbf{x}, d_{\text{rescale}}), d_{\text{patch}}\right), \tag{2}$$

where d_{rescale} and d_{patch} are the dimension of the rescaled image and the dimension of the patch, $\texttt{Rescale}(\mathbf{x}, d)$ is the image rescaling operation to dimension $d \times d$, and $\texttt{Patchify}(\mathbf{x}, d)$ is the image patchifying operation to patch dimension $d \times d$, respectively.

The `RescaleNPatchify` operation results in $n_{\text{patch}} = (\frac{d_{\text{rescale}}}{d_{\text{patch}}})^2$ number of patches with $d_{\text{patch}} \times d_{\text{patch}}$ dimension. We now examine the proposed sensitivity score in Section 3.1 and average the sensitivity score to output the unified score function as follows:

$$\text{WaRPAD}(\mathbf{x}) = \frac{1}{n_{\text{patch}}} \sum_{\mathbf{x}_{\text{patch}} \in \texttt{RescaleNPatchify}(\mathbf{x})} \text{HFwav}(\mathbf{x}_{\text{patch}}) \tag{3}$$

Our proposed WaRPAD examines the sensitivity score in the resized subregion of the images, which are multiple instances of RRC with the area of $\frac{1}{n_{\text{patch}}}$ resized to a dimension of $d_{\text{patch}} \times d_{\text{patch}}$. Since the model M is trained to be invariant under various RRC instances that include crops similar to the patch, we expect the base score HFwav evaluated on the patch to be still robust. On the other hand, We hypothesize the model output of the AI-generated image will deviate in the patches, hence our WaRPAD further improves on detecting AI-generated images. We verify our hypothesis by examining the WaRPAD and our base score in 1000 real RAISE-1k [25] data and 1000 SDv1.4-generated data in the Synthbuster benchmark [32]. We select $d_{\text{rescale}} = 1344$ and $d_{\text{patch}} = 224$.

We show the histogram of real and AI-generated images under our base score on the image, base score on the patch, and the averaged WaRPAD score in Figure 3a, 3b, and 3c, respectively. AI-generated images lose their robustness in our wavelet-based high-frequency perturbation when examined in patches generally. The discrimination between AI-generated images and real images is strengthened in our final score induced by averaging the score function across patches.

4 Experiment

We now evaluate WaRPAD's efficacy on AI-generated image detection. We first introduce the benchmarks and generative models for each benchmark as well as the baseline methods (Section 4.1). We report the performance of WaRPAD across these benchmarks (Section 4.2). Finally, we present detailed ablation studies of WaRPAD and test its robustness in corruptions (Section 4.3).

4.1 Experiment Settings

Datasets. We first test WaRPAD in Synthbuster [32] benchmark based on RAISE-1k dataset [25] where 9 generative models are applied: Firefly [17], GLIDE [35], SDXL [36], SDv2, SDv1.3, SDv1.4

Table 2: **AI-generated image detection performance (AUROC) of WaRPAD and baselines in the Synthbuster [32] benchmark. Bold** and <u>underline</u> denotes the best method and the second best methods, respectively.

Method	Firefly	GLIDE	SDXL	SDv2	SDv1.3	SDv1.4	DALL-E 3	DALL-E 2	Midjourney	Mean
RIGID [15]	0.519	0.868	<u>0.757</u>	0.615	0.448	0.446	<u>0.442</u>	0.596	0.593	0.587
MINDER [16]	0.440	0.568	0.472	0.721	0.656	0.668	0.346	0.445	0.345	0.518
AEROBLADE [10]	<u>0.592</u>	<u>0.954</u>	0.668	0.567	<u>0.950</u>	<u>0.950</u>	**0.486**	0.392	**0.769**	<u>0.703</u>
Manifold Bias [11]	0.493	0.779	0.562	<u>0.749</u>	0.544	0.549	0.379	<u>0.607</u>	0.424	0.565
WaRPAD **(ours)**	**0.927**	**0.999**	**0.830**	**0.775**	**0.959**	**0.958**	0.422	**0.930**	<u>0.702</u>	**0.834**

Table 3: **AI-generated image detection performance (AUROC) of WaRPAD and baselines in the GenImage [33] benchmark. Bold** and <u>underline</u> denotes the best and second best methods, respectively.

Method	ADM	BigGAN	GLIDE	Midjourney	SDv1.4	SDv1.5	VQDM	Wukong	Mean
RIGID [15]	<u>0.874</u>	0.974	0.952	0.778	0.682	0.682	<u>0.915</u>	0.699	0.820
MINDER [16]	0.768	0.681	0.582	0.450	0.607	0.596	0.882	0.676	0.655
AEROBLADE [10]	0.856	<u>0.981</u>	<u>0.989</u>	**0.918**	**0.982**	**0.984**	0.732	**0.983**	<u>0.928</u>
Manifold Bias [11]	0.727	0.925	0.852	0.510	0.675	0.673	0.874	0.653	0.736
WaRPAD **(ours)**	**0.986**	**0.998**	**0.991**	<u>0.810</u>	<u>0.940</u>	<u>0.936</u>	**0.981**	<u>0.924</u>	**0.946**

Table 4: **AI-generated image detection performance (AUROC) of WaRPAD and baselines in the Deepfake-LSUN-Bedroom [34] benchmark. Bold** and <u>underline</u> denotes the best method and the second best methods, respectively.

Method	ADM	DDPM	Diff-ProjectedGAN	Diff-StyleGAN2	IDDPM	LDM	PNDM	ProGAN	ProjectedGAN	StyleGAN	Mean
RIGID [15]	0.742	0.887	0.937	0.914	0.855	0.846	0.843	0.957	0.944	0.681	0.861
MINDER [16]	0.706	0.796	<u>0.973</u>	0.942	0.782	0.844	<u>0.896</u>	0.970	0.973	0.805	0.869
AEROBLADE [10]	0.545	0.741	0.488	0.534	0.656	0.595	0.382	0.454	0.490	0.342	0.522
Manifold Bias [11]	**0.788**	<u>0.905</u>	0.968	<u>0.943</u>	<u>0.888</u>	<u>0.928</u>	0.891	**0.996**	<u>0.978</u>	**0.912**	<u>0.920</u>
WaRPAD **(ours)**	<u>0.785</u>	**0.937**	**0.988**	**0.965**	**0.908**	**0.940**	**0.970**	<u>0.995</u>	**0.986**	<u>0.870</u>	**0.934**

[8], DALL-E 3 [37], DALL-E 2 [18], and Midjourney [9]. We also test WaRPAD in GenImage [33] benchmark where the real data is from the ImageNet [5] dataset and 8 generative models are applied for fake image generation: ADM [38], BigGAN [39], GLIDE, Midjourney, SDv1.4, SDv1.5, VQDM [40], and Wukong [41]. Finally, we test on the deepfake-LSUN-bedroom benchmark [34] containing 10 generative models: ADM, DDPM [42], Diff-ProjectedGAN, Diff-StyleGAN2 [43], IDDPM [44], LDM [8], PNDM [45], ProGAN [46], ProjectedGAN [47], and StyleGAN [48]. We summarize the main information of these benchmarks in Table 1.

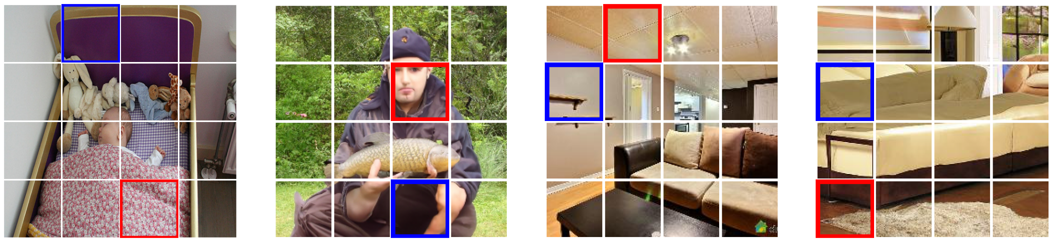
Baselines. We consistently compare against available training-free baselines. AEROBLADE [10] and Manifold Induced Bias [11] apply SD for the examination. Consistent with the proposal of Brokman et al. [11], we apply SDv1.4 for the inspection model. We also compare against RIGID [15] and MINDER [16], which apply DINOv2 for AI-generated image detection. We follow the original hyperparameter settings from the paper.

Implementation Details. The performance of all methods is reported by the area under the ROC curve (AUROC). Consistent with RIGID and MINDER, we use the DINO-ViT-L14 model as the base model. We use the Haar wavelet with a 2-level decomposition to extract the high-frequency information. We also set d_{patch} and α to 224 and 0.1 throughout all experiments. For the rescaling dimension d_{rescale}, we set 896 for the GenImage and Deepfake-LSUN-bedroom benchmark and 1344 for the Synthbuster benchmark. We implement our code in the Pytorch [49] framework. All experiments are done on a single A100 GPU.

4.2 Main Results

Tables 2, 3, and 4 report the performance of WaRPAD compared to other training-free approaches in Synthbuster, GenImage, and Deepfake-LSUN-bedroom benchmark, respectively. WaRPAD achieves the best performance on all benchmarks on average. Notably, WaRPAD consistently outperforms RIGID and MINDER by a wide margin of over 6% in AUROC.

On the other hand, diffusion-model-based methods fail to show consistent performance compared to WaRPAD. For example, while AEROBLADE is competitive to WaRPAD in the GenImage benchmark, where some generative models share the same autoencoder with the inspected SDv1.4



(a) Real-GenImage (b) AI-GenImage (c) Real-LSUN (d) AI-LSUN

Figure 4: **Visualization of the HFwav score across patches.** We show the patch with the highest score in red and lowest score in blue. Each image is from ImageNet **(a)**, ADM-generated GenImage **(b)**, LSUN **(c)**, and ADM-generated Deepfake-LSUN-Bedroom dataset **(d)**, respectively.

Table 5: **Ablation Study on each component of WaRPAD.** We report AUROC. Gains are computed against RIGID [15].

Method	Synthbuster	GenImage	Deepfake-LSUN-Bedroom
RIGID [15]	0.587	0.820	0.861
RIGID $\times n_{\text{patch}}$	0.589 (+0.2%)	0.823 (+0.3%)	0.872 (+1.1%)
HFwav	0.636 (+4.9%)	0.809 (-1.1%)	0.890 (+2.9%)
RIGID + `RescaleNPatchify`	0.656 (+6.9%)	0.800 (-2.0%)	0.861 (+0.0%)
WaRPAD	0.834 (+24.7%)	0.946 (+12.6%)	0.934 (+7.3%)

model (*e.g.*, SDv1.5, Wukong), its performance deteriorates on detecting proprietary models (*e.g.*, Firefly, DALL-E 2) or GAN-based models. On the other hand, Brokman et al. [11] can efficiently perform in the Deepfake-LSUN-Bedroom benchmark but fails on the GenImage and Synthbusters benchmarks.

We also visualize the patch-wise score information of the real and AI-generated images. Figure 4 shows the patch with the highest HFwav score (denoted as red) and the patch with the lowest score (denoted as blue) in the real and AI-generated data in GenImage and Deepfake-LSUN-Bedroom benchmark, respectively. While HFwav assigns high scores on patches with rich texture information, the region does not always align with the semantics of the image (*e.g.*, Figure 4c).

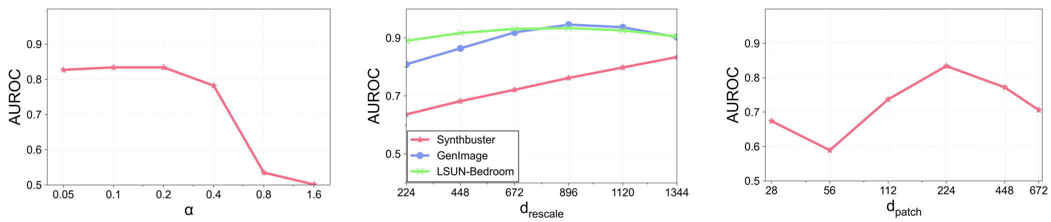
4.3 Analysis

We further analyze the effect of each component of WaRPAD and investigate the contribution of each hyperparameter. We further study the robustness of WaRPAD under test-time corruptions against other training-free AI-generated image detection methods.

Effect of each Component. We first analyze the effect of our proposed HFwav and `RescaleNPatchify` operation. We also investigate the synergy of HFwav and `RescaleNPatchify` by experimenting with RIGID [15] with the same `RescaleNPatchify` procedure. Further, we experiment with the n_{patch}-ensemble version of RIGID that takes the same computational cost as our WaRPAD. We show the results in Table 5 with a comparison against RIGID [15]. Our perturbation-based metric improves over RIGID in two out of 3 benchmarks. Moreover, RIGID combined with our proposed `RescaleNPatchify` shows relatively little improvement from RIGID compared to WaRPAD, highlighting the efficacy of both components.

Hyperparameter Analysis. We analyze the effect of the hyperparameters of the WaRPAD independently. Figure 5a analyzes the effect of perturbation weight α in the Synthbuster benchmark. We also analyze the effect of the rescaling dimension d_{rescale} on separate benchmarks in Figure 5b. All hyperparameter choices consistently improve over the base dimension, 224. Finally, as DINOv2 can accept arbitrary patch sizes, we also test WaRPAD in the different patch dimensions d_{patch}. We show the result in Figure 5c where the base choice of 224 performs the best.

Design Choices. We first show the AUROC result of WaRPAD across different aggregation rules of the computed patch-wise metric and the backbone DINOv2 version. For the aggregation rule, we test the mean, median, minimum, and maximum across patches. For the backbone model, we experiment with 'ViT-S14', 'ViT-B14', 'ViT-L14', and 'ViT-g14'. We also experiment with the 'ViT-L14' and 'ViT-g14' with register tokens [14].



(a) (b) (c)

Figure 5: **Hyperparameter analysis of WaRPAD (a)**: AUROC performance of WaRPAD with respect to α in the Synthbuster benchmark. **(b)**: AUROC result with respect to the d_{rescale}. **(c)**: AUROC result with respect to the d_{patch} in the Synthbuster benchmark.

Table 6: **Ablation Study on the DINOv2 backbone and patch-wise aggregation rule on WaRPAD.** We report AUROC. **Bold** denotes the best choice.

Agg Rule	ViT-S14	ViT-B14	ViT-L14	ViT-L14-reg	ViT-g14	ViT-g14-reg
Mean	0.76/0.86/0.83	0.82/0.92/0.90	0.83/**0.95**/0.93	0.84/0.94/0.94	0.86/0.94/**0.95**	**0.87**/0.94/**0.95**
Median	0.77/0.85/0.81	0.83/0.91/0.89	0.84/0.94/0.92	0.85/0.94/0.93	0.85/0.93/**0.95**	**0.87**/0.94/**0.95**
Min	0.71/0.85/0.79	0.75/0.89/0.84	0.76/0.91/0.88	0.77/0.91/0.89	0.80/0.90/0.91	0.80/0.90/0.90
Max	0.68/0.74/0.63	0.77/0.81/0.77	0.78/0.90/0.79	0.78/0.90/0.76	0.79/0.90/0.87	0.80/0.91/0.87

Table 7: **AI-generated image detection performance (AUROC) to wavelet choice and decomposition level in the Synthbuster benchmark. Bold** and <u>underline</u> denotes the best and the second best choice. We group the wavelets by the number of vanishing moments on a wavelet function ψ.

	1 vanishing moment			>1 vanishing moments								
Level	Haar	bior 1.3	bior 1.5	db2	db3	db4	bior 2.2	bior 2.4	bior 3.1	coif1	coif2	coif3
1	0.745	0.715	0.701	0.458	0.505	0.527	0.474	0.483	0.486	0.480	0.481	0.487
2	**0.834**	0.591	<u>0.827</u>	0.512	0.491	0.508	0.512	0.487	0.472	0.533	0.506	0.501
3	0.787	0.585	0.569	0.466	0.490	0.460	0.476	0.490	0.486	0.498	0.457	0.467

Table 8: **AI-generated image detection performance (AUROC) in the Synthbuster benchmark under different backbones.**

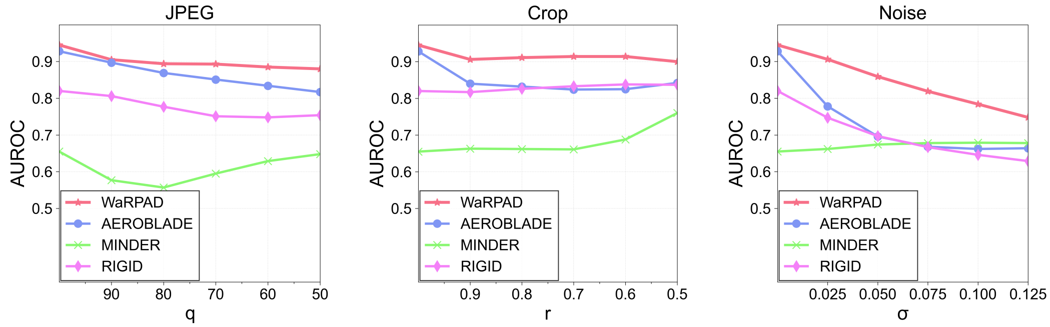
	Invariant to RRC				Others	
Method	DINOv2 [12]	CLIP [23]	SwaV [28]	DINO [29]	BeiT [50]	ViTMAE [51]
RIGID	0.587	0.561	0.542	0.480	0.619	0.531
MINDER	0.518	0.583	0.542	0.478	0.473	0.545
WaRPAD	0.834	0.802	0.743	0.707	0.486	0.620

We show the result in Table 6. Note that Mean and 'ViT-L14' correspond to the results in Section 4.2. In the perspective of the aggregation rule, the mean or median aggregation rule achieves the best performance consistently. On the other hand, concerning the DINOv2 backbone, a larger model size shows better results with the slight exception of the 'ViT-L14' backbone in the GenImage benchmark.

We further experiment with the different choices of the wavelet and the decomposition level of the wavelet decomposition. Apart from the Haar wavelet, we experiment with Daubeches wavelets (db2, db3, db4), Biorthogonal wavelets (bior1.3, bior1.5, bior2.2, bior2.4, bior3.1), and Coiflet wavelets (coif1, coif2, coif3) with decomposition levels from 1 to 3.

We present the results in Table 7, grouping wavelets by the number of vanishing moments in the (synthesis) wavelet function ψ. While the Haar wavelet achieves the highest performance, our results indicate that other wavelets with one vanishing moment also perform competitively. In contrast, wavelets with higher vanishing moments tend to induce more structured perturbations on the real images, and we find that DINOv2 is no longer robust to the perturbation. We further report this phenomenon in the Appendix, where we include histograms of the score distributions for other wavelets for both real and AI-generated images.

Robustness to Corruptions. Our WaRPAD is based on the self-supervised vision model, which may learn robust representations due to training on a wide range of perturbations (*e.g.*, `ColorJitter`, `RandomSolarize`). Hence, we test WaRPAD's robustness on the corruption of the input images,



(a) JPEG Compression (b) Center Crop (c) Gaussian Noise

Figure 6: **Robustness of WaRPAD in corruptions.** We test the AUROC performance of WaRPAD, AEROBLADE, MINDER, and RIGID in corrupted test images in the GenImage benchmark.

both real and AI-generated. For comparison, we also test RIGID, MINDER, and AEROBLADE on the same corruption. We report the average performance on the GenImage benchmark with JPEG compression, center crop and resizing, and Gaussian noise corruptions with varying degrees.

Figure 6 shows the performance of WaRPAD as well as the training-free baselines under corruption in the GenImage benchmark. WaRPAD achieves competitive performance over the baseline in every corruption consistently. On the other hand, AEROBLADE's performance quickly degrades compared to WaRPAD in high-level corruption. Experiments in other benchmarks in the Appendix exhibit consistent behaviors.

Extension to other backbones. The result in DINOv2 shows that our proposed metric can perform not only in in-domain datasets (*e.g.*, ImageNet) but also in datasets unobserved in the training phase (*e.g.*, RAISE-1k, LSUN-Bedroom). We further experiment WaRPAD with other backbones of self-supervised models trained to be invariant under RRC. For the model, we select CLIP [23], SwaV [28], and DINO [29]. We also test other models that use RRC only as data augmentation: BeiT [50] and ViTMAE [51]. We also specify the details of the tested models in the Appendix. We further include MINDER and RIGID for each backbone as a comparison. We do not change any hyperparameters.

We show the result in Table 8. Our WaRPAD consistently improves over RIGID and MINDER when the backbone model is trained to be invariant under RRC. However, the gain of WaRPAD is less prominent when tested in the masked-image-modeling-based backbones and even underperforms over RIGID when the backbone model is BeiT.

5 Conclusion

We propose WaRPAD, an effective training-free AI-generated image detection method motivated by `RandomResizedCrop`, the core data augmentation scheme of self-supervised methods. WaRPAD shows improved performance and robustness against existing training-free methods. The main advantage of the WaRPAD is the ubiquity of the `RandomResizedCrop`, which enables WaRPAD applicable to various self-supervised models as the backbone. Since WaRPAD applies to vision-text trained encoders (*e.g.*, CLIP [23]), our method can be extended to detecting multimodal AI-generated data. We leave this direction to future work.

Broader Impact. WaRPAD offers effective training-free detection of AI-generated images in the wild. This provides a *tabula rasa* defense against the improper use of generative models, including fraud or manipulation of AI-generated images. Furthermore, our method can be applied to filter out AI-generated images from the web-scale image data.

Limitations. Our `RescaleNPatchify` procedure induces extra computation costs due to the number of patches, although these patches can be computed in a batch-wise manner. Furthermore, WaRPAD is dependent on the choice of the backbone self-supervised model, and may not generalize to high-resolution real/AI-generated images outside the scope of the backbone model. However, this may be the limitation of other training-free detection methods, too.

References

[1] Elizabeth Howcroft. Ai-generated content raises risks of more bank runs, uk study shows, 2025. URL https://www.reuters.com/technology/artificial-intelligence/ai-generated-content-raises-risks-more-bank-runs-uk-study-shows-2025-02-14/. Accessed: 2025-04-13.

[2] Samantha Murphy Kelly. Hollywood celebrities are being deepfaked into porn. congress is considering a new law to stop it, March 2025. URL https://edition.cnn.com/2025/03/08/tech/hollywood-celebrity-deepfakes-congress-law/index.html. Accessed: 2025-04-13.

[3] Davide Coozolino, Giovanni Poggi, Matthias Nießner, and Luisa Verdoliva. Zero-shot detection of ai-generated images. In *ECCV*, 2024.

[4] Shilin Yan, Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, and Weidi Xie. A sanity check for ai-generated image detection. In *ICLR*, 2025.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[6] LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. Understanding and improving training-free ai-generated image detections with vision foundation models. *https://arxiv.org/abs/1506.03365*, 2015.

[7] Anirudh Sundara Rajan and Yong Jae Lee. Stay-positive: A case for ignoring real image features in fake image detection, 2025. URL https://arxiv.org/abs/2502.07778.

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[9] MidJourney. MidJourney AI Art Generator, 2022. URL https://www.midjourney.com/home/.

[10] Jonas Ricker, Denis Lukovnikov, and Asja Fischer. Aeroblade: Training-free detection of latent diffusion images using autoencoder reconstruction error. In *CVPR*, 2024.

[11] Jonathan Brokman, Amit Giloni, Omer Hofman, Roman Vainshtein, Hisashi Kojima, and Guy Gilboa. Manifold induced biases for zero-shot and few-shot detection of generated images. In *ICLR*, 2025.

[12] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt.

[13] Yang Liu, Chenchen Jing, Hengtao Li, Muzhi Zhu, Hao Chen, Xinlong Wang, and Chunhua Shen. A simple image segmentation framework via in-context examples. In *NeurIPS*, 2024.

[14] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024.

[15] 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.

[16] Chung-Ting Tsai, Ching-Yun Ko, I-Hsin Chung, Yu-Chiang Frank Wang, and Pin-Yu Chen. Understanding and improving training-free ai-generated image detections with vision foundation models. *arXiv preprint arXiv:2411.19117*, 2024.

[17] Adobe. Adobe firefly - free generative ai for creatives, 2025. URL https://www.adobe.com/products/firefly.html.

[18] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. URL https://arxiv.org/abs/2204.06125.

[19] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NIPS*, 2014.

[20] Jingyi Deng, Chenhao Lin, Zhengyu Zhao, Shuai Liu, Qian Wang, and Chao Shen. A survey of defenses against ai-generated visual media: Detection, disruption, and authentication. *arXiv preprint arXiv:2407.10575*, 2024. URL https://arxiv.org/abs/2407.10575.

[21] Tobias Uelwer, Jan Robine, Stefan Sylvius Wagner, Marc Höftmann, Eric Upschulte, Sebastian Konietzny, Maike Behrendt, and Stefan Harmeling. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9052–9071, 2024. doi: 10.1109/TPAMI.2024.3415112. URL https://www.computer.org/csdl/journal/tp/2024/12/10559458/1XR0ep31Wr6.

[22] Davide Cozzolino, Giovanni Poggi, Riccardo Corvi, Matthias Nießner, and Luisa Verdoliva. Raising the bar of ai-generated image detection with clip. In *CVPR Workshop*, 2024.

[23] Alec Radford, Jong Wook Kim, Luke 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 *ICML*, 2021.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[25] Duc-Tien Dang-Nguyen, Cecilia Pasquini, Valentina Conotter, and Giulia Boato. Raise: a raw images dataset for digital image forensics. In *Proceedings of the 6th ACM Multimedia Systems Conference*, MMSys '15, page 219–224, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450333511. doi: 10.1145/2713168.2713194. URL https://doi.org/10.1145/2713168.2713194.

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[27] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[28] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

[29] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[30] 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 *ICLR*, 2021.

[31] Xiuji Zhou, Chen Wei Yang, Chen Change Loy, and Ziwei Liu. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022.

[32] Quentin Bammey. Synthbuster: Towards detection of diffusion model generated images. *IEEE Open Journal of Signal Processing*, 5:1–9, 2024. doi: 10.1109/OJSP.2023.3337714.

[33] 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 *NIPS*, 2023.

[34] Jonas Ricker, Simon Damm, Thorsten Holz, and Asja Fischer. Towards the detection of diffusion model deepfakes. In *VISAPP*, 2024.

[35] 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. In *ICML*, 2022.

[36] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024.

[37] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Pamela Mishkin, Brooks Chan, Christopher Hesse, Alec Radford, and Ilya Sutskever. Dall·e 3, 2023. URL https://cdn.openai.com/papers/dall-e-3.pdf. OpenAI Technical Report.

[38] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. In *NIPS*, 2021.

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

[40] 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 *CVPR*, 2022.

[41] Jiaxi Gu, Xiaojun Meng, Guansong Lu, Lu Hou, Minzhe Niu, Xiaodan Liang, Lewei Yao, Runhui Huang, Wei Zhang, Xin Jiang, Chunjing Xu, and Hang Xu. Wukong: A 100 million large-scale chinese cross-modal pre-training benchmark. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2022.

[42] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[43] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusiongan: Training gans with diffusion. In *ICLR*, 2023.

[44] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021.

[45] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *ICLR*, 2022.

[46] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

[47] Axel Sauer, Kashyap Chitta, Jens Muller, and Andreas Geiger. Projected gans converge faster. In *NeurIPS*, 2021.

[48] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*, 2019.

[50] Hang Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022.

[51] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: we clearly stated the contributions and the scope in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: we include the limitation section in the conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: Our paper is not based on the theoretical results but on the intuitions for AI-generated image detection.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: we clearly stated our hyperparameter (our algorithm is deterministic) in the experiment section. We followed the author's hyperparameters when reproducing the baseline.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [No]

Justification: We will publish the code after the paper gets accepted. However, as our algorithm is deterministic and the benchmark is public, it would be easy to reproduce the results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We include all the details on the hyperparameters in the experiment section.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: Our algorithm is deterministic and does not require multiple seeds. While one baseline is based on random noise, we also experimented with the ensemble version of this.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We clearly state the resource of A100GPU in the experiment section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We believe that we have faithfully practiced the code of ethics in the guideline.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss the broader impact in the conclusion section.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: We do not release any data or model that pose high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite the respective benchmarks for AI-generated image detection. Furthermore, we also include the methods that create the dataset and the original datasets. License of these datasets are provided in the Appendix.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: The paper does not release new assets. We will publish our code if accepted.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: We do not involve any crowdsourcing or research with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: We do not report any result that involves human subject.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: Our methodology is not based on any LLMs.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Algorithm 1 WaRPAD (PyTorch-like Pseudo-code)

```
# f(x): normalized [cls] token output of self-supervised model
# alpha: weight of perturbation
# DWTForward, DWTInverse: forward and inverse discrete wavelet transform
# Sim: cosine similarity function


def HFwav(x):
    x_low, x_high = DWTForward(x)
    N_perturb = DWTInverse([torch.zeros_like(x_low), x_high])
    feat_original = f(x)
    feat_perturb = f(x - alpha * N_perturb)
    return Sim(feat_original, feat_perturb)

def WaRPAD(x):
    x_patch = RescaleNPatchify(x)
    f_patch = HFwav(x_patch)
    return f_patch.mean()
```

A Appendix

A.1 Pseudocode of WarPAD

We show the Pytorch-like pseudocode of WaRPAD in Algorithm 1. Note that all operations allow batch-wise computation, hence we can process the input patches in a batch-wise manner.

A.2 Further information of the Experiment settings.

Synthbuster. The Synthbuster benchmark consists of 1000 real RAISE-1k images and 9000 AI-generated images consisting of scene and art images under the 'CC BY-NC-SA 4.0' license. We download all real [1] and AI-generated datasets [2] in the URL via the author's official repository.

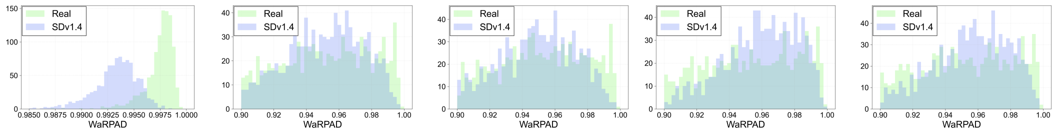
GenImage. The GenImage benchmark consists of ImageNet real data and AI-generated data consisting of 8 different generative models under the 'CC BY-NC-SA 4.0' license. Each test consists of pairs of real and AI-generated image pairs, where the size is 6000+6000 except of SDv1.5, where the size is 8000+8000. We download the datasets via the author's official repository [3].

Deepfake-LSUN-Bedroom. The Deepfake-LSUN-Bedroom benchmark consists of 10000 real LSUN-Bedroom images and 10×10000 AI-generated data where the model is trained to generate LSUN-Bedroom-like images. We download the datasets via the author's official repository [4] under the 'CC BY 4.0' license.

Baselines. We follow the author's implementation for the AEROBLADE [5] and Manifold Bias [6], respectively. Since the original implementation of the AEROBLADE operates on the fixed dimension, extension to data with arbitrary size (*e.g.*, Synthbuster, GenImage) is not trivial. Our finding is that the preservation of the original dimension is crucial for the performance, hence we chose to center-crop or resize the image to the fixed dimension of the autoencoder dimension whether the image is larger or smaller than the autoencoder default dimension, respectively. On the other hand, we have not found any official implementation of the authors on the RIGID and MINDER. Instead, we manage to reproduce the RIGID in the consistent setting of our WaRPAD. Note that RIGID and MINDER propose to resize the image to the default resolution of DINOv2, which is 224×224.

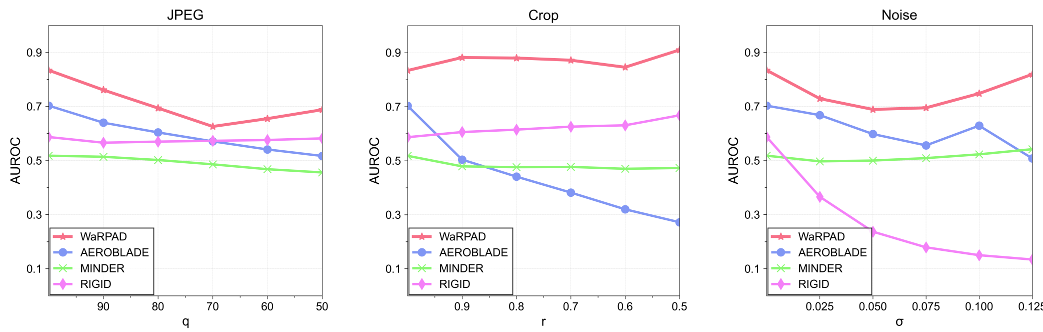
Experiment Settings. Most experiments are deterministic since they operate on deterministic operations. A slight exception is RIGID, where the Gaussian noise augmentation is done on the

[1] https://loki.disi.unitn.it/RAISE/download.html
[2] https://zenodo.org/records/10066460
[3] https://github.com/GenImage-Dataset/GenImage
[4] https://zenodo.org/records/7528113
[5] https://github.com/jonasricker/aeroblade
[6] https://tinyurl.com/zeroshotimplementation



(a) Haar (b) db2 (c) coif1 (d) bior3.1 (e) coif3

Figure 7: **Histogram of other wavelet.** We show the results on Haar, db2, coif1, bior3.1, and coif3 wavelet, respectively.



(a) JPEG Compression (b) Center Crop (c) Gaussian Noise

Figure 8: **Robustness of WaRPAD in corruptions.** We test the AUROC performance of WaRPAD, AEROBLADE, MINDER, and RIGID in corrupted test images in the Synthbuster benchmark.

image. However, our experiments in Table 5 show that RIGID with multiple runs does not change much performance against RIGID with a single seed.

Additional Backbones. All pre-trained backbones are accessible and downloadable. For the CLIP model, we use "clip-vit-base-32" [7] for the base model. We use SwaV on the Resnet50 [52] backbone [8] and DINO of "vit-s16" version [9] pre-trained on the ImageNet dataset. We use the "vit-mae-base" model for the ViTMAE backbone [10] and "beit-base-patch16-224" model for the BeiT backbone [11].

A.3 Histogram of other wavelet choices.

We show the computed histogram of WaRPAD on other wavelets (db2, coif1, bior3.1, and coif3), on real and SDv1.4-generated images computed in the Synthbuster benchmark in Figure 7. Results show DINOv2 model loses its robustness in other wavelet choices, especially wavelets with more vanishing moments.

A.4 Further robustness experiments

We further include the performance of WaRPAD, AEROBLADE, MINDER, and RIGID in the Synthbuster benchmark in Figure 8 consistent to Figure 6. The trend is consistent, where the WaRPAD performs the best.

[7] https://huggingface.co/openai/clip-vit-base-patch32
[8] https://github.com/facebookresearch/swav
[9] https://github.com/facebookresearch/dino
[10] https://huggingface.co/docs/transformers/en/model_doc/vit_mae
[11] https://huggingface.co/docs/transformers/en/model_doc/beit