A Sanity Check for AI-generated Image Detection

Shilin Yan[1]*, **Ouxiang Li**[2]*, **Jiayin Cai**[1]*, **Yanbin Hao**[2]
Xiaolong Jiang[1], **Yao Hu**[1], **Weidi Xie**[3]†

[1]Xiaohongshu Inc.
[2]University of Science and Technology of China
[3]Shanghai Jiao Tong University
tattoo.ysl@gmail.com weidi@sjtu.edu.cn

Abstract

With the rapid development of generative models, discerning AI-generated content has evoked increasing attention from both industry and academia. In this paper, we conduct a sanity check on ***"whether the task of AI-generated image detection has been solved"***. To start with, we present **`Chameleon`** dataset, consisting AI-generated images that are genuinely challenging for human perception. To quantify the generalization of existing methods, we evaluate 9 off-the-shelf AI-generated image detectors on **`Chameleon`** dataset. Upon analysis, almost all models classify AI-generated images as real ones. Later, we propose **AIDE** (**A**I-generated **I**mage **DE**tector with Hybrid Features), which leverages multiple experts to simultaneously extract visual artifacts and noise patterns. Specifically, to capture the high-level semantics, we utilize CLIP to compute the visual embedding. This effectively enables the model to discern AI-generated images based on semantics or contextual information; Secondly, we select the highest frequency patches and the lowest frequency patches in the image, and compute the low-level patchwise features, aiming to detect AI-generated images by low-level artifacts, for example, noise pattern, anti-aliasing, etc. While evaluating on existing benchmarks, for example, AIGCDetectBenchmark and GenImage, AIDE achieves **+3.5**% and **+4.6**% improvements to state-of-the-art methods, and on our proposed challenging **`Chameleon`** benchmarks, it also achieves the promising results, despite this problem for detecting AI-generated images is far from being solved. The dataset, codes, and pre-train models will be published at https://github.com/shilinyan99/AIDE.

1 Introduction

Recently, the vision community has witnessed remarkable advancements in generative models. These methods, ranging from generative adversarial networks (GANs) [22, 72, 13, 34] to diffusion models (DMs) [28, 47, 57, 58, 38, 42, 27, 46] have demonstrated unprecedented capabilities in synthesizing high-quality images that closely resemble real-world scenes. On the positive side, such generative models have enabled various valuable tools for artists and designers, democratizing access to advanced graphic design capabilities. However, it also raises concerns about the authenticity of visual content, posing significant challenges for image forensics [19], misinformation combating [65], and copyright protection [55]. In this paper, we consider the problem of distinguishing between images generated by AI models and those originating from real-world sources.

*Equal contribution.
†Corresponding author.

Preprint. Under review.

In the literature, although there are numerous AI-generated image detectors [62, 20, 49, 63, 70, 56, 69] and benchmarks [62, 63, 74, 29], the prevailing problem formulation typically involves training models on images generated solely by GANs (usually ProGAN [33]) and evaluating their performance on datasets including images from various generative models, including GANs and DMs. However, such formulation poses two fundamental issues in practice. *Firstly*, evaluation benchmarks are simple, as they often feature test sets composed of random images from generative models, rather than images that present genuine challenges for human perception; *Secondly*, confining models to train exclusively on images from certain type of generative models (GANs or Diffusions) imposes an unrealistic constraint, hindering the model's ability to learn from the diverse properties exhibited by more advanced generative models.

To address the aforementioned issues, we propose two pivotal strategies. *Firstly*, we introduce a novel test set for AI-generated image detection, named **`Chameleon`**, manually annotated to include images that genuinely challenge human perception. This dataset has three key features: (i) Deceptively real: all AI-generated images in the dataset have passed a human perception "Turing Test", *i.e.*, human annotators have misclassified them as real images. (ii) Diverse categories: comprising images of human, animal, object, and scene categories, the dataset depicts real-world scenarios across various contexts. (iii) High resolution: with most images having resolutions exceeding 720P and going up to 4K, all images in the dataset exhibit exceptional clarity. Consequently, this test set offers a more realistic evaluation of model performance. After evaluating 9 off-the-shelf AI-generated image detectors on **`Chameleon`**, unfortunately, all detectors suffer from significant performance drops, misclassifying the AI-generated images as real ones. *Secondly*, we reformulate the AI-generated image detection problem setup, which enables models to train across a broader spectrum of generative models, enhancing their adaptability and robustness in real-world scenarios.

Based on the above observation, it is clear that detecting AI-generated images remains challenging, and is far from being solved. Therefore, a fundamental question arises: what distinguishes AI-generated images from real ones? Intuitively, such cues may appear from various aspects, including low-level textures or pixel statistics (*e.g.*, the presence of white noise during image capturing), and high-level semantics (*e.g.*, penguins are unlikely to be appearing on the grassland in Africa), geometry principle (*e.g.*, perspective), physics (*e.g.*, lighting condition). To reflect such intuition, we propose a simple AI-generated image detector, termed as **AIDE** (**A**I-generated **I**mage **DE**tector with Hybrid Features). Specifically, AIDE incorporates a DCT [11] scoring module to capture low-level pixel statistics by extracting both high and low-frequency patches from the image, which are then processed through SRM (Spatial Rich Model) filters [21] to characterize the noise pattern. Additionally, to capture global semantics, we utilize the pre-trained OpenCLIP [31] to encode the entire image. The features from various levels are fused in the channel dimension for the final prediction. To evaluate the effectiveness of our model, we conduct extensive experiments on two popular benchmarks, including AIGCDetectBenchmark [62] and GenImage [74], for AI-generated image detection. On AIGCDetectBenchmark and GenImage benchmarks, AIDE surpasses state-of-the-art (SOTA) methods by **+3.5%** and **+4.6%** in accuracy scores, respectively. Moreover, AIDE also achieves competitive performance on our **`Chameleon`** benchmark.

Overall, our contributions are summarized as follows: (i) We present the **`Chameleon`** dataset, a meticulously curated test set designed to challenge human perception by including images that deceptively resemble real-world scenes. With thorough evaluation of 9 different off-the-shelf detectors, this dataset exposes the limitations of existing approaches. (ii) We present a simple mixture-of-expert model, termed as AIDE, that enables to discern AI-generated images based on both low-level pixel statistics and high-level semantics. (iii) Experimentally, our model achieves state-of-the-art results on public benchmarks for AIGCDetectBenchmark [62] and GenImage [74]. While on **`Chameleon`**, it acts as a competitive baseline on a realistic evaluation benchmark, to foster future research in this community.

2 Related Works

AI-generated image detection. The demand for detecting AI-generated images has long been present. Early studies primarily focus on spatial domain cues, such as color [43], saturation [44], co-occurrence [45], and reflections [48]. However, these methods often suffer from limited generalization capabilities as generators progress. To address this limitation, CNNSpot [62] discovers that an image classifier trained exclusively on ProGAN [33] generator could generalize effectively to other unseen

GAN architectures, with careful pre- and post-processing and data augmentation. FreDect [20] observes significant artifacts in the frequency domain of GAN-generated images, attributed to the upsampling operation in GAN architectures. More recent approaches have explored novel perspectives for superior generalization ability. UnivFD [49] proposes to train a universal liner classifier with pretrained CLIP-ViT [18, 52] features. DIRE [63] introduces DIRE features, which computes the difference between images and their reconstructions from pretrained ADM [17], to train a deep classifier. PatchCraft [70] compares rich-texture and poor-texture patches from images, extracting the inter-pixel correlation discrepancy as a universal fingerprint, which is reported to achieve the state-of-the-art (SOTA) generalization performance. AEROBLADE [56] proposes a training-free detection method for latent diffusion models using autoencoder reconstruction errors. However, these methods only discriminate real or fake images from a single perspective, often failing to generalize across images from different generators.

AI-generated image datasets. To facilitate AI-generated image detection, many datasets containing both real and fake images have been organized for training and evaluation. Early dataset from CNNSpot [62] collects fake images from GAN series generators [22, 72, 13, 34]. Particularly, this dataset generates fake images exclusively using ProGAN [33] as training data and evaluates the generalization ability on a set of GAN-based testing data. However, with recent emergence of more advanced generators, such as diffusion model (DM) [28] and its variants [17, 47, 57, 58, 38, 42, 27, 46], their realistic generations make visual differences between real and fake images progressively harder to detect. Subsequently, more datasets including DM-generated images have been proposed one after another, including DE-FAKE [66], CiFAKE [12], DiffusionDB [64], ArtiFact [53]. One representative dataset is GenImage [74], which comprises ImageNet's 1,000 classes generated using eight SOTA generators in both academia (*e.g.,* Stable Diffusion [6]) and industry (*e.g.,* Midjourney [4]). More recently, Hong *et al.* introduce a more comprehensive dataset, WildFake [29], which includes AI-generated images sourced from multiple generators, architectures, weights, and versions. However, existing benchmarks only evaluate AI-generated images using current foundational models with simple prompts and few modifications, whereas deceptively real images from online communities usually necessitate hundreds to thousands of manual parameter adjustments.

3 Chameleon Dataset

3.1 Problem Formulation

In this paper, our goal is to train a computational model that can distinguish the AI-generated images from the ones captured by the camera, *i.e.*, $y = \Phi_{\text{model}}(\mathbf{I}; \Theta) \in \{0, 1\}$, where $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ denotes an input RGB image, Θ refers to the learnable parameters. For training and testing, we consider the following two settings:

Train-Test Setting-I. In the literature, existing works on detecting AI-generated images [62, 20, 49, 63, 70] have exclusively considered the scenario of training on images from single generative model, for example, ProGAN [33], or Stable Diffusion [6], and then evaluated on images from various generative models. That is,

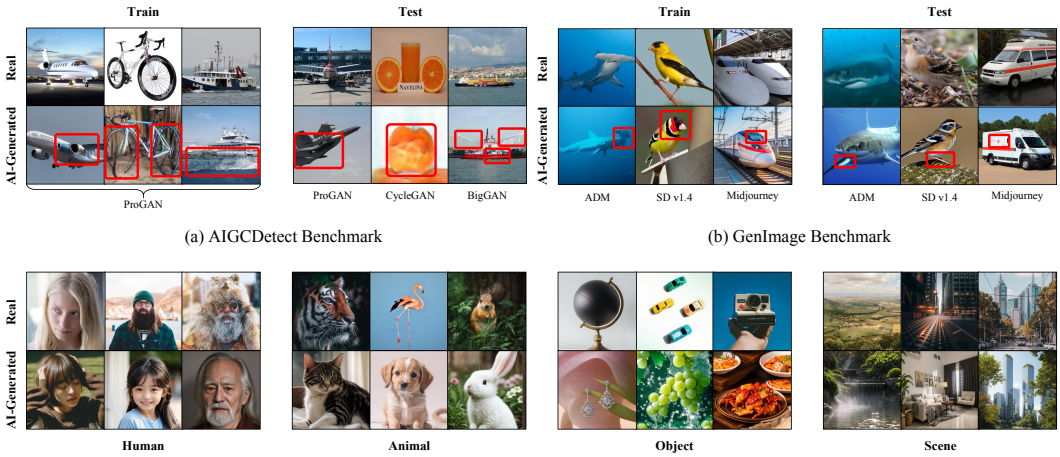
$$\mathcal{G}_{\text{train}} = \mathcal{G}_{\text{GAN}} \vee \mathcal{G}_{\text{DM}}, \mathcal{G}_{\text{test}} = \{\mathcal{G}_{\text{ProGAN}}, \mathcal{G}_{\text{CycleGAN}}, ..., \mathcal{G}_{\text{SD}}, \mathcal{G}_{\text{Midjourney}}\}. \tag{1}$$

Generally speaking, such problem formulation poses two critical issues: (i) evaluation benchmarks are simple, as these randomly sampled images from generative models, can be far from being photo-realistic, as shown in Figure 1; (ii) confining models to train exclusively on GAN-generated images imposes an unrealistic constraint, hindering the model's ability to learn from the diverse properties exhibited by more advanced generative models.

Train-Test Setting-II. Herein, we propose an alternative problem formulation, where the models are allowed to train on images generated from a wide spectrum of generative models, and then tested on images that are genuinely challenging for human perception.

$$\mathcal{G}_{\text{train}} = \{\mathcal{G}_{\text{GAN}}, \mathcal{G}_{\text{DM}}\}, \mathcal{G}_{\text{test}} = \{\mathcal{D}_{\texttt{Chameleon}}\}. \tag{2}$$

$\mathcal{D}_{\texttt{Chameleon}}$ refers to our proposed benchmark, as detailed below. We believe this setting resembles more practical scenario for future model development in this community.



(a) AIGCDetect Benchmark

(b) GenImage Benchmark

(c) Chameleon Testset

Figure 1: We visualize two contemporary AI-generated image benchmarks, namely (a) AIGCDetect Benchmark [62] and (b) GenImage Benchmark [74], where all images are generated from publicly available generators, including ProGAN (GAN-based), SD v1.4 (DM-based), and Midjourney (commercial API). These images are conditioned on simple prompts (e.g., *photo of a plane*) without delicate manual adjustments, thereby inclined to generate obvious anti-facts in consistency and semantics (marked with red boxes). In contrast, our Chameleon dataset in (c) aims to simulate real-world scenarios by collecting diverse images from online websites, where these online images are carefully adjusted by photographers and AI artists.

3.2 Chameleon Dataset

The primary objective of the **`Chameleon`** dataset is to evaluate the generalization and robustness of existing AI-generated image detectors, for a sanity check on the progress of AI-generated image detection. In this section, we outline the progression of the proposed dataset in three critical phases: (i) dataset collection, (ii) dataset curation, and (iii) dataset annotation. The statistical results of our dataset are illustrated in Table 1 and we compare our dataset with existing benchmarks in Fig. 1.

3.2.1 Dataset Collection

To simulate real-world cases on detecting AI-generated images, we structure our **`Chameleon`** dataset based on two main principles: (i) images must be deceptively real, and (ii) they should cover a diverse range of categories. Herein, we present the details of image collection.

Table 1: **Statistics of the Chameleon testset,** including over 11k high-fidelity AI-generated images from [1–3], as well as a similar scale of real-world photographs from [5].

	Real Images	Fake Images
Scene	3,574	2,976
Object	3,578	2,016
Animal	3,998	313
Human	3,713	5,865
Total	14,863	11,170

Fake Image Collection: To collect images that are deceptively real, and cover sufficiently diverse categories, we source user-created AI-generated images from popular AI-painting communities (*i.e.,* ArtStation [1], Civitai [2], and Liblib [3]), many of which utilize commercial APIs (*e.g.,* Midjourney [4] and DALLE-3 [54]) or various LoRA modules [30] with Stable Diffusion (SD) [6] that fine-tuned on their in-house data. Specifically, we initiate the process by utilizing GPT-4 [7] to generate diverse query words to retrieve AI-generated images. Throughout the collection process, we enforce stringent NSFW (Not Safe For Work) restrictions. Ultimately, our collection comprises over 150K fake images.

Real Image Collection: To ensure that real and fake images fall into the same distribution, we employ identical query words to retrieve real images, mirroring the approach used for gathering AI-generated images. Eventually, we collect over 20K images from platforms like Unsplash [5], which is an online community providing high-quality, free-to-use images contributed by photographers worldwide.

3.2.2 Dataset Curation

To ensure the collection of high-quality images, we implement a comprehensive pipeline for image cleaning: (i) we discard images with resolution lower than 448×448, as higher-resolution images generally provide better assessments of the robustness of existing models; (ii) due to the potential presence of violent and inappropriate content, we utilize SD's safety checker model [8] to filter out NSFW images; (iii) to avoid duplicated images, we compare their hash values to filter out duplicated images. In addition to this general cleaning pipeline, we introduce CLIP [52] to further filter out images with low image-text similarity. Specifically, for fake images, the online website provides prompts used to generate these images, and we calculate similarity using their corresponding prompts. For real images, we used the mean of the 80 prompt templates (*e.g., a photo of {category}* and *a photo of the {category}*) evaluated in CLIP's ImageNet zero-shot as the text embedding.

3.2.3 Dataset Annotation

At this stage, we establish an annotation platform and recruit 20 human workers to manually label each of the AI-generated images for their category and realism. For categorization, annotators are instructed to assign each image to one of four major categories: human, animal, object, and scene. Regarding realism assessment, workers are tasked with labeling the images as either **Real** or **AI-generated**, based on the criterion of ***"whether this image could be taken with a camera"***. It's important to note that as the annotators are not informed whether the images are generated by AI algorithms beforehand, those have been misclassified as real can thus be considered to pass the 'Perception Turing Test' and labeled as "highly realistic". Subsequently, we retain only those images judged as "highly realistic". Similarly, for real images, we follow the same procedure, retaining only those belonging to the four predefined categories, as we have done for AI-generated images.

4 Architecture

In this section, we present AIDE (**A**I-generated **I**mage **DE**tector with Hybrid Features), consisting of a module to compute patchwise low-level statistics of texture or smooth patches, a high-level semantic embedding module, and a discriminator to classify the image as being generated or photographed. The overview of our AIDE model is illustrated in Fig. 2.

4.1 Patchwise Feature Extraction

We leverage insights from the disparities in low-level patch statistics between AI-generated images and real-world scenes. Models like generative adversarial networks or diffusion models often yield images with certain artifacts, such as excessive smoothness or anti-aliasing effects. To capture such discrepancy, we adopt a Discrete Cosine Transform (DCT) score module to identify patches with the highest and lowest frequency. By focusing on these extreme patches, we aim to highlight the distinctive characteristics of AI-generated images, thus facilitating the discriminative power of our detection system.

Patch Selection via DCT Scoring. For an RGB image, we first divide this image into multiple patches with a fixed window size, $\mathbf{I} = \{x_1, x_2, \ldots, x_n\}$, $x_i \in \mathbb{R}^{N \times N \times 3}$. In our case, the patch size is set to be $N = 32$ pixels. We apply the discrete cosine transform to each of the image patches, obtaining the corresponding results in the frequency domain, $\mathcal{X}_f = \{x_1^{\text{dct}}, x_2^{\text{dct}} \ldots, x_n^{\text{dct}}\}$, $x_i^{\text{dct}} \in \mathbb{R}^{N \times N \times 3}$.

To acquire the highest and lowest image patches, we use the complexity of the frequency components as an indicator. From this, we design a simple yet effective scoring mechanism. Specifically, firstly, we design K different band-pass filters:

$$F_{k,ij} = \begin{cases} 1, & \text{if } \frac{2N}{K} \cdot k \leq i + j < \frac{2N}{K} \cdot (k+1) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $F_{k,ij}$ is the weight at the (i, j) position of the k-th filter. Next, for m-th patch $x_m^{\text{dct}} \in \mathbb{R}^{N \times N \times 3}$, we apply the filters $F_{k,ij} \in \mathbb{R}^{N \times N \times 3}$ to multiply the logarithm of the absolute DCT coefficients $x_m^{\text{dct}} \in \mathbb{R}^{N \times N \times 3}$ and sum all the positions to obtain the grade of the patch G^m. We formulated it as

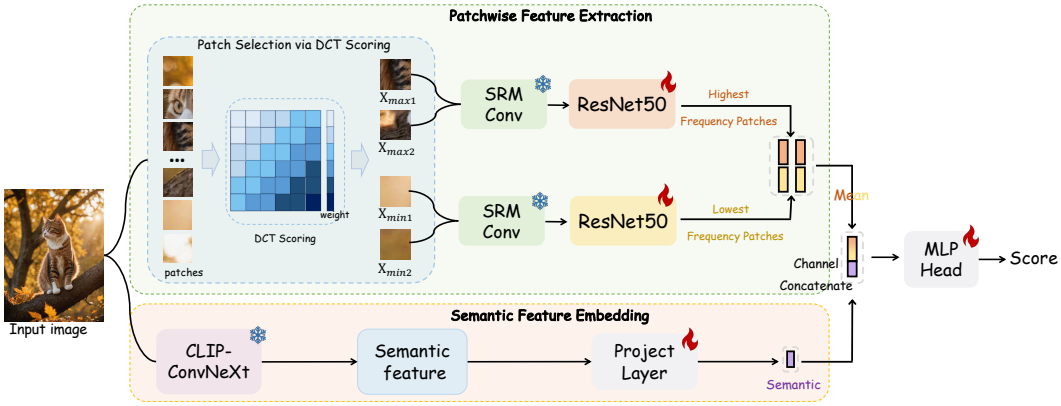


Figure 2: **Overview of AIDE.** It consists of a Patchwise Feature Extraction (PFE) module and a Semantic Feature Embedding (SFE) module in a mixture of experts manner. In PFE module, the DCT Scoring module first calculates the DCT coefficients for each smashed patch and then performs a weighted sum of these coefficients (weights gradually increase as the color goes from light to dark).

$$G^m = \sum_{k=0}^{K-1} 2^k \times \sum_{c=0}^{2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} F_{k,ij} \cdot \log(|x_m^{\text{dct}}| + 1) \tag{4}$$

where c is the number of patch channels. Thus, we acquire the grades of all patches $G = \{G^1, G^2, ..., G^n\}$, and then we sort them to identify the highest and lowest frequency patches.

Through the scoring module, we can obtain the top k patches $X_{\text{max}} = \{X_{\text{max}_1}, X_{\text{max}_2}, ..., X_{\text{max}_k}\}$ with the highest frequency and the top k patches $X_{\text{min}} = \{X_{\text{min}_1}, X_{\text{min}_2}, ..., X_{\text{min}_k}\}$ with the lowest frequency, where $X_{\text{max}_i} \in \mathbb{R}^{N \times N \times 3}$, $X_{\text{min}_i} \in \mathbb{R}^{N \times N \times 3}$.

Patchwise Feature Encoder. Next, firstly, these patches are resized to a size of 256×256. Secondly, they are input into the SRM [21] to extract their noise pattern. Lastly, these features are input into two ResNet50 [25] network ($f_1(\cdot)$ and $f_2(\cdot)$) to obtain the final feature map $F_{\text{max}} = \{f_1(X_{\text{max}_1}), f_1(X_{\text{max}_2}), ..., f_1(X_{\text{max}_k})\}$, $F_{\text{min}} = \{f_2(X_{\text{min}_1}), f_2(X_{\text{min}_2}), ..., f_2(X_{\text{min}_k})\}$. We acquire the highest frequency embedding and lowest frequency embedding on the mean-pooled feature:

$$F_{\text{max}} = \text{Mean}(\text{AveragePool}(F_{\text{max}})), \quad F_{\text{min}} = \text{Mean}(\text{AveragePool}(F_{\text{min}})). \tag{5}$$

4.2 Semantic Feature Embedding

To capture the rich semantic features within images, such as object co-occurrence and contextual relationships, we compute the visual embedding for input image with an off-the-shelf visual-language foundation model. Specifically, we adopt the ConvNeXt-based OpenCLIP model [31] to get the final feature map ($v \in \mathbb{R}^{h \times w \times c}$). To capture the global contexts, we append a linear projection layer followed by mean spatial pooling, $F_{\text{s}} = \text{avgpool}(g(v))$.

4.3 Discriminator

To distinguish between AI-generated images and real images, we utilize a mixture-expert-model for the final discrimination. At low-level, we take the average of the highest frequency feature F_{max} and the lowest frequency feature F_{min}. Then, we channel-wisely concatenate the representations between it and high-level embedding F_s. At last, the features are encoded into MLP to acquire the score, $y = f([\text{avgpool}(F_{\text{max}}, F_{\text{min}}); F_{\text{s}}])$, where $f(\cdot)$ denotes the MLP consisting of a linear layer, GELU [26] and classifier, $[;]$ refers to the operation of channel-wise concatenation.

Table 2: **Benchmark1**. Accuracy (%) of different detectors (rows) in detecting real and fake images from different generators (columns). DIRE-D indicates this result comes from DIRE detector trained over fake images generated by ADM following its official setup [63]. DIRE-G indicates this baseline is trained on the same ProGAN training data as others. GAN-Average and DM-Average are averaged over the first 8 and the last 8 test sets, respectively. The best result and the second-best result are marked in **bold** and underline, respectively.

Generator	CNNSpot [62]	FreDect [20]	Fusing [32]	GramNet [40]	LNP [37]	LGrad [59]	UnivFD [49]	DIRE-G [63]	DIRE-D [63]	PatchCraft [70]	Ours
ProGAN	**100.00**	99.36	**100.00**	99.99	99.67	99.83	99.81	95.19	52.75	**100.00**	99.99
StyleGAN	90.17	78.02	85.20	87.05	91.75	91.08	84.93	83.03	51.31	92.77	**99.64**
BigGAN	71.17	81.97	77.40	67.33	77.75	85.62	95.08	70.12	49.70	**95.80**	83.95
CycleGAN	87.62	78.77	87.00	86.07	84.10	86.94	98.33	74.19	49.58	70.17	**98.48**
StarGAN	94.60	94.62	97.00	95.05	99.92	99.27	95.75	95.47	46.72	**99.97**	99.91
GauGAN	81.42	80.57	77.00	69.35	75.39	78.46	**99.47**	67.79	51.23	71.58	73.25
StyleGAN2	86.91	66.19	83.30	87.28	94.64	85.32	74.96	75.31	51.72	89.55	**98.00**
WFIR	91.65	50.75	66.80	86.80	70.85	55.70	86.90	58.05	53.30	85.80	**94.20**
ADM	60.39	63.42	49.00	58.61	84.73	67.15	66.87	75.78	**98.25**	82.17	93.45
Glide	58.07	54.13	57.20	54.50	80.52	66.11	62.46	71.75	92.42	83.79	95.09
Midjourney	51.39	45.87	52.20	50.02	65.55	65.35	56.13	58.01	89.45	**90.12**	77.20
SD v1.4	50.57	38.79	51.00	51.70	85.55	63.02	63.66	49.74	91.24	**95.38**	93.00
SD v1.5	50.53	39.21	51.40	52.16	85.67	63.67	63.49	49.83	91.63	**95.30**	92.85
VQDM	56.46	77.80	55.10	52.86	74.46	72.99	85.31	53.68	91.90	88.91	**95.16**
Wukong	51.03	40.30	51.70	50.76	82.06	59.55	70.93	54.46	90.90	91.07	**93.55**
DALLE2	50.45	34.70	52.80	49.25	88.75	65.45	50.75	66.48	92.45	96.60	**96.60**
GAN-Average	87.94	78.78	84.21	84.87	86.76	85.28	77.39	50.79	91.90	88.21	**93.43**
DM-Average	53.61	49.28	52.55	52.48	80.91	65.41	59.97	**92.28**	64.95	90.42	92.11
Average	70.78	64.03	68.38	68.67	83.84	75.34	78.43	68.68	71.53	89.31	**92.77**

Table 3: **Benchmark2**. Accuracy (%) of different baselines (columns) in detecting real and fake images from different generators (rows). These methods are trained on real images from ImageNet and fake images generated by SD v1.4 and evaluated over eight generators. The best result and the second-best result are marked in **bold** and underline, respectively.

	Testing Dataset								Average
	Midjourney	SD v1.4	SD v1.5	ADM	GLIDE	Wukong	VQDM	BigGAN	
ResNet-50 [25]	54.90	**99.90**	99.70	53.50	61.90	98.20	56.60	52.00	72.09
DeiT-S [60]	55.60	**99.90**	99.80	49.80	58.10	98.90	56.90	53.50	71.56
Swin-T [41]	62.10	**99.90**	99.80	49.80	67.60	99.10	62.30	57.60	74.78
CNNSpot [62]	52.80	96.30	95.90	50.10	39.80	78.60	53.40	46.80	64.21
Spec [68]	52.00	99.40	99.20	49.70	49.80	94.80	55.60	49.80	68.79
F3Net [51]	50.10	**99.90**	**99.90**	49.90	50.00	**99.90**	49.90	49.90	68.69
GramNet [40]	54.20	99.20	99.10	50.30	54.60	98.90	50.80	51.70	69.85
DIRE [63]	60.20	**99.90**	99.80	50.90	55.00	99.20	50.10	50.20	70.66
UnivFD [49]	73.20	84.20	84.00	55.20	76.90	75.60	56.90	**80.30**	73.29
GenDet [73]	**89.60**	96.10	96.10	58.00	78.40	92.80	66.50	75.00	81.56
PatchCraft [70]	79.00	89.50	89.30	77.30	78.40	89.30	**83.70**	72.40	82.30
Ours	79.38	99.74	99.76	**78.54**	**91.82**	98.65	80.26	66.89	**86.88**

5 Experiments

5.1 Experimental Setup

Detectors. We evaluate 9 off-the-shelf detectors including CNNSpot [62], FreDect [20], Fusing [32], GramNet [40], LNP [37], LGrad [59], UnivFD [49], DIRE [63] and PatchCraft [70] for comparsion. More details can be found in Appendix.

Metrics. In accordance with existing AI-generated detection arpproaches [62, 61, 71], we report both classification accuracy (Acc) and average precision (AP) in our experiments. All results are averaged over both real and AI-generated images unless otherwise specified. We primarily report Acc for evaluation and comparison in the main paper, and AP results are presented in the Appendix.

5.2 Benchmarks

To comprehensively evaluate the generalization ability of existing approaches, we detail three benchmarks, (i) AIGCDetectBenchmark [62], (ii) GenImage [74], and (iii) our **`Chameleon`** dataset, where (i) and (ii) belong to **Setting-I** and (iii) belongs to **Setting-II** as summarized in Sec. 3.1. More detailed statistics of these benchmarks are shown in Appendix.

Benchmark 1 ($\mathcal{B}_1$): We follow the widely-used AIGCDetectBenchmark [62, 74, 70], which aims to constrain the detector to learn a universal fingerprint from one generator and generalize to others. Specifically, the training dataset consists of 360k real images from LSUN [67] and 360k fake images

generated only by ProGAN [33] with the same classes. In evaluation, we follow PatchCraft's [70] setup, which covers most existing foundational generators, collected from ForenSynths [62] and GenImage [74]. The test dataset includes 16 different generative models, with both GAN-based and DM-based architectures, providing a comprehensive evaluation of detection performance on these mainstream generators commonly utilized in both research and industry. This includes advanced GAN-based generators like StyleGAN2 [35] and commercial DM-based APIs like Midjourney [4].

Benchmark 2 ($\mathcal{B}_2$): GenImage is a newly proposed benchmark and has been evaluated in recent studies [70, 14, 23]. The two major improvements over $\mathcal{B}_1$ include incorporating advanced DM-based generators and expanding the training classes from LSUN's 20 classes to ImageNet's 1,000 classes. In particular, GenImage includes ImageNet as real images and generates images with ImageNet's labels as conditions. These images are generated using 8 generators, including BigGAN [13], ADM [17], Glide [46], Midjourney [4], SD v1.4 [6], SD v1.5 [6], VQDM [24], and Wukong [10]. Each generator corresponds to a training and testing set consisting of an equal number of real and generated images, resulting in 8 training sets and 8 testing sets. Following GenImage's setup, we utilize SD v1.4 as the training dataset and report the averaged results over 8 test datasets.

Benchmark 3 ($\mathcal{B}_3$): To test the models on our **`Chameleon`** testset, we train them under different training setups: (i) we adopt the training data from $\mathcal{B}_1$ as it is widely used among these baselines, which has demonstrated excellent generalization ability across on the test of $\mathcal{B}_1$; (ii) the models are trained on the training data from $\mathcal{B}_2$; (iii) the models are trained on the generated images from 8 generators in GenImage, including both the classic GAN-based models (*e.g.,* BigGAN) and the latest DM-series models (*e.g.,* SD and Midjourney).

5.3 Comparison with State-of-the-arts

On Benchmark 1: In Table 2, the quantitative results showcase the classification accuracies across various methods and generators within $\mathcal{B}_1$. In our evaluation, all methods except DIRE-D were trained exclusively on ProGAN's generations.

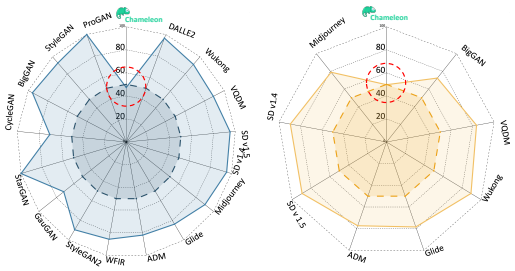


Figure 3: Performance of SOTA method, PatchCraft, under $\mathcal{B}_1$ (left), $\mathcal{B}_2$ (right), and our **`Chameleon`** testset. The boundary line for Acc = 50% is marked with a dashed line.

Our approach demonstrates a notable improvement over the existing SOTA method, PatchCraft, achieving a 3.5% higher average accuracy. Among the baseline methods, UnivFD incorporates CLIP semantic features for AI-generated image detection. It exhibits effectiveness in detecting GAN-generated images but experiences a performance drop when applied to DM-generated images, showing that with the growing generation quality, images generated by diffusion model tend not to lead discernible artifacts in semantics, as illustrated in Fig. 1 (a). While our method integrates semantic, low-frequency and high-frequency information at the feature level, leading to improved detection performance for both GAN-based (5.2% increase) and DM-based (1.7% increase) images compared to the SOTA method.

On Benchmark 2: While conducting experiments on $\mathcal{B}_2$, where all models were trained on SD v1.4 and evaluated across 8 up-to-date generators. Table 3 summarizes the results, highlighting our method's superior performance over the current state-of-the-art method, PatchCraft, by 4.6% in terms of average accuracy. It is worth noting the architectural similarities between SD v1.5, Wukong, and SD v1.4, as highlighted by GenImage [74]. This resemblance enables models to achieve nearly perfect accuracy, approaching 100% on such test sets. Consequently, the assessment of generalization performance on other generators becomes crucial, for example, Midjourney, ADM, and Glide, where our model demonstrates either the best or second-best results, with an average accuracy of 86.88%.

On Benchmark 3: As emphasized in Sec. 1, we argue that success on the existing public benchmarks may not truly reflect the progress of AI-generated image detection, as the testsets are typically randomly sampled from generative models.

To mitigate potential biases introduced by training setups, such as generator types and image categories, we assess the performance of existing baselines in diverse training settings. Despite their success on existing public benchmarks, as Fig. 3 shown, the state-of-the-art detector, PatchCraft,

Table 4: **Benchmark3**. Accuracy (%) of different detectors (rows) in detecting real and fake images of **Chameleon** testset (rows). For each training dataset, the first row indicates the **average Acc** evaluated on the **Chameleon** testset, and the second row gives "**fake image Acc / real image Acc**" for detailed analysis.

Training Dataset	CNNSpot [62]	FreDect [20]	Fusing [32]	GramNet [40]	LNP [37]	LGrad [59]	UnivFD [49]	DIRE [63]	PatchCraft [70]	Ours
ProGAN	49.88	50.42	49.90	52.21	50.03	50.63	50.51	51.37	47.30	49.55
	0.08/99.67	13.72/87.12	0.01/99.79	4.76/99.66	0.09/99.97	1.70/99.55	3.18/97.83	3.25/99.48	1.78/92.82	0.63/98.46
SD v1.4	56.25	49.97	49.98	55.58	48.79	55.49	58.03	53.77	49.71	**61.54**
	13.86/98.63	1.37/98.57	0.00/99.96	17.65/93.50	0.57/97.01	11.44/99.54	74.97/41.09	11.86/95.67	3.07/96.35	40.61/82.46
All GenImage	54.56	50.22	50.00	53.41	52.21	52.48	**63.54**	50.91	48.96	58.73
	9.86/99.25	0.89/99.55	0.02/99.98	8.23/98.58	7.72/96.70	5.19/99.77	85.52/41.56	2.09/99.73	1.39/96.52	22.40/95.06

suffers from significant performance drops. Moreover, the results presented in Table 4 unveil a significant performance drop across all methods. Most approaches struggle to achieve an average accuracy of around 50%, equivalent to random guessing, indicating their failure under this setting.

Although our method achieves state-of-the-art results on publicly available datasets, its performance on **Chameleon** remains unsatisfactory. This also validates the fact that our own dataset, **Chameleon**, which can challenge human perception, represents a genuine problem that needs to be addressed in this field.

5.4 Robustness to Unseen Perturbations

In real-world scenarios, images inevitably encounter unseen perturbations during transmission and interaction, which increases the difficulty in detecting AI-generated images. Herein, we evaluate the performance of different methods in handling potential unseen perturbations *i.e.,* JPEG compression (Quality Factor (QF) = 95, 90) and Gaussian blur (σ = 1.0, 2.0). As shown in Table 5, all methods are experiencing a decline in performance due to the disruption of the pixel distribution. This disruption diminishes the discriminative artifacts left by the generative model, making it more difficult to discern between real and AI-generated images. Consequently, the ability of these detectors to robustly identify AI-generated images is significantly compromised. From this table, our method still achieves SOTA performance in both perturbation scenarios, achieving 75.54% (QF = 95) and 81.88% ($\sigma = 1.0$). Despite the challenging conditions, our method consistently outperforms other approaches, maintaining a relatively higher Acc in detecting AI-generated images. This superior performance can be attributed to our model's ability to effectively capture and utilize multi-perspective features, *i.e.,* semantics and noise, between real and fake images, even when the pixel distribution is distorted.

Table 5: The classification accuracy (%) averaged over 16 test sets in $\mathcal{B}_1$ with specific perturbation.

Method	JPEG Compression		Gaussian Blur	
	QF=95	QF=90	$\sigma = 1.0$	$\sigma = 2.0$
CNNSpot [62]	64.03	62.26	68.39	67.26
FreDect [20]	66.95	67.45	65.75	66.48
Fusing [32]	62.43	61.39	68.09	66.69
GramNet [40]	65.47	64.94	68.63	68.09
LNP [37]	53.58	54.09	67.91	66.42
LGrad [59]	51.55	51.39	71.73	69.12
DIRE-G [63]	66.49	66.12	64.00	63.09
UnivFD [49]	74.10	74.02	70.31	68.29
PatchCraft [70]	72.48	71.41	75.99	74.90
Ours	**75.54**	**74.21**	**81.88**	**80.35**

5.5 Ablation Studies

Our method focuses on detecting AI-generated images with mixture of experts, namely patchwise feature extraction (PFE-H and PFE-L for high-frequency and low-frequency patches, respectively) and semantic feature extraction (SFE). These modules collectively contribute to comprehensively identifying AI-generated images from different perspectives. Herein, we conduct ablation studies on each module on $\mathcal{B}_1$ and draw the following conclusions from Table 6: (i) The first 3 rows show the results of ablating each module individually, indicating that while these features are useful for detection, they still suffer from inferior performance in accuracy (around 75% Acc). This aligns with our analysis that finding a universal fingerprint for detection is challenging, and multiple features should be considered simultaneously. (ii) The second 3 rows sequentially combine two modules, and this integration of different features can improve performance to some extent. No-

Table 6: **Ablation study** of different modules in our method.

Module			Average
PFE-H	PFE-L	SFE	
✓	✗	✗	76.09
✗	✓	✗	75.24
✗	✗	✓	75.26
✓	✓	✗	76.70
✓	✗	✓	80.69
✗	✓	✓	84.20
✓	✓	✓	92.77

tably, PFE-L + SFE achieves the best result among the 3 combinations. This result aligns with the motivation of recent SOTA methods, UnivFD and PatchCraft, highlighting the importance of low-frequency patches and semantic features in AI-generated image detection. (iii) On top of above findings, our method with mixture of the three experts achieves the SOTA performance, proving that the simultaneous use of all three features can achieve excellent results. This comprehensive extraction captures subtle discrepancies and underscores the effectiveness of our ensemble method in detecting AI-generated images.

6 Conclusion

In this paper, we have conducted a sanity check on detecting AI-generated images. Specifically, we re-examined the unreasonable assumption in existing training and testing settings and suggested new ones. In terms of benchmarks, we propose a novel, challenging benchmark, termed as **`Chameleon`**, which is manually annotated to challenge human perception. We evaluate 9 off-the-shelf models and show that all detectors suffered from huge performance drops. In terms of architecture, we propose a simple yet effective model, that simultaneously incorporates low-level patch statistics and high-level semantics for AI-generated image detection. Despite our approach demonstrates state-of-the-art performance on existing (AIGCDetectBenchmark [62] and GenImage [74]) and our proposed benchmarks (**`Chameleon`**), it leaves significant room for future improvement.

References

[1] Artstation. https://www.artstation.com.

[2] Civitai. https://civitai.com.

[3] Liblib. https://www.liblib.art.

[4] Midjourney. https://www.midjourney.com/home.

[5] Unsplash. https://unsplash.com.

[6] Stable diffusion. https://github.com/Stability-AI/StableDiffusion, 2022.

[7] Chatgpt. https://chatgpt.com, 2022.

[8] Stable diffusion safety checker. https://huggingface.co/CompVis/stable-diffusion-safety-checker, 2022.

[9] whichfaceisreal. https://www.whichfaceisreal.com, 2023.

[10] Wukong. https://xihe.mindspore.cn/modelzoo/wukong, 2023.

[11] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[12] J. J. Bird and A. Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images. *IEEE Access*, 2024.

[13] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[14] J. Chen, J. Yao, and L. Niu. A single simple patch is all you need for ai-generated image detection. *arXiv preprint arXiv:2402.01123*, 2024.

[15] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[17] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[18] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[19] W. D. Ferreira, C. B. Ferreira, G. da Cruz Júnior, and F. Soares. A review of digital image forensics. *Computers & Electrical Engineering*, 85:106685, 2020.

[20] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz. Leveraging frequency analysis for deep fake image recognition. In *International conference on machine learning*, pages 3247–3258. PMLR, 2020.

[21] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on information Forensics and Security*, 7(3):868–882, 2012.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[23] P. Grommelt, L. Weiss, F.-J. Pfreundt, and J. Keuper. Fake or jpeg? revealing common biases in generated image detection datasets. *arXiv preprint arXiv:2403.17608*, 2024.

[24] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[26] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[27] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[28] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[29] Y. Hong and J. Zhang. Wildfake: A large-scale challenging dataset for ai-generated images detection. *arXiv preprint arXiv:2402.11843*, 2024.

[30] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[31] G. Ilharco, M. Wortsman, R. Wightman, C. Gordon, N. Carlini, R. Taori, A. Dave, V. Shankar, H. Namkoong, J. Miller, H. Hajishirzi, A. Farhadi, and L. Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

[32] Y. Ju, S. Jia, L. Ke, H. Xue, K. Nagano, and S. Lyu. Fusing global and local features for generalized ai-synthesized image detection. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3465–3469. IEEE, 2022.

[33] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[34] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[35] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[36] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[37] B. Liu, F. Yang, X. Bi, B. Xiao, W. Li, and X. Gao. Detecting generated images by real images. In *European Conference on Computer Vision*, pages 95–110. Springer, 2022.

[38] L. Liu, Y. Ren, Z. Lin, and Z. Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

[39] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[40] Z. Liu, X. Qi, and P. H. Torr. Global texture enhancement for fake face detection in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8060–8069, 2020.

[41] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[42] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[43] S. McCloskey and M. Albright. Detecting gan-generated imagery using color cues. *arXiv preprint arXiv:1812.08247*, 2018.

[44] S. McCloskey and M. Albright. Detecting gan-generated imagery using saturation cues. In *2019 IEEE international conference on image processing (ICIP)*, pages 4584–4588. IEEE, 2019.

[45] L. Nataraj, T. M. Mohammed, S. Chandrasekaran, A. Flenner, J. H. Bappy, A. K. Roy-Chowdhury, and B. Manjunath. Detecting gan generated fake images using co-occurrence matrices. *arXiv preprint arXiv:1903.06836*, 2019.

[46] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[47] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[48] J. F. O'brien and H. Farid. Exposing photo manipulation with inconsistent reflections. *ACM Trans. Graph.*, 31(1):4–1, 2012.

[49] U. Ojha, Y. Li, and Y. J. Lee. Towards universal fake image detectors that generalize across generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24480–24489, 2023.

[50] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019.

[51] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In *European conference on computer vision*, pages 86–103. Springer, 2020.

[52] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[53] M. A. Rahman, B. Paul, N. H. Sarker, Z. I. A. Hakim, and S. A. Fattah. Artifact: A large-scale dataset with artificial and factual images for generalizable and robust synthetic image detection. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 2200–2204. IEEE, 2023.

[54] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[55] J. Ren, H. Xu, P. He, Y. Cui, S. Zeng, J. Zhang, H. Wen, J. Ding, H. Liu, Y. Chang, et al. Copyright protection in generative ai: A technical perspective. *arXiv preprint arXiv:2402.02333*, 2024.

[56] J. Ricker, D. Lukovnikov, and A. Fischer. Aeroblade: Training-free detection of latent diffusion images using autoencoder reconstruction error. *arXiv preprint arXiv:2401.17879*, 2024.

[57] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[58] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[59] C. Tan, Y. Zhao, S. Wei, G. Gu, and Y. Wei. Learning on gradients: Generalized artifacts representation for gan-generated images detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12105–12114, 2023.

[60] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[61] S.-Y. Wang, O. Wang, A. Owens, R. Zhang, and A. A. Efros. Detecting photoshopped faces by scripting photoshop. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10072–10081, 2019.

[62] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros. Cnn-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8695–8704, 2020.

[63] Z. Wang, J. Bao, W. Zhou, W. Wang, H. Hu, H. Chen, and H. Li. Dire for diffusion-generated image detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22445–22455, 2023.

[64] Z. J. Wang, E. Montoya, D. Munechika, H. Yang, B. Hoover, and D. H. Chau. Diffusiondb: A large-scale prompt gallery dataset for text-to-image generative models. *arXiv preprint arXiv:2210.14896*, 2022.

[65] D. Xu, S. Fan, and M. Kankanhalli. Combating misinformation in the era of generative ai models. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9291–9298, 2023.

[66] Q. Xu, H. Wang, L. Meng, Z. Mi, J. Yuan, and H. Yan. Exposing fake images generated by text-to-image diffusion models. *Pattern Recognition Letters*, 176:76–82, 2023.

[67] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[68] X. Zhang, S. Karaman, and S.-F. Chang. Detecting and simulating artifacts in gan fake images. In *2019 IEEE international workshop on information forensics and security (WIFS)*, pages 1–6. IEEE, 2019.

[69] Y. Zhang and X. Xu. Diffusion noise feature: Accurate and fast generated image detection. *arXiv preprint arXiv:2312.02625*, 2023.

[70] N. Zhong, Y. Xu, Z. Qian, and X. Zhang. Rich and poor texture contrast: A simple yet effective approach for ai-generated image detection. *arXiv preprint arXiv:2311.12397*, 2023.

[71] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis. Learning rich features for image manipulation detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1053–1061, 2018.

[72] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[73] M. Zhu, H. Chen, M. Huang, W. Li, H. Hu, J. Hu, and Y. Wang. Gendet: Towards good generalizations for ai-generated image detection. *arXiv preprint arXiv:2312.08880*, 2023.

[74] M. Zhu, H. Chen, Q. Yan, X. Huang, G. Lin, W. Li, Z. Tu, H. Hu, J. Hu, and Y. Wang. Genimage: A million-scale benchmark for detecting ai-generated image. *Advances in Neural Information Processing Systems*, 36, 2024.

A Experimental Details

A.1 Implementation Details

Our method includes two key modules: Patchwise Feature Extraction (PFE) and Semantic Feature Embedding (SFE). For PFE channel, we first patchify each image into patches and the patch size is set to be $N = 32$ pixels. Then these patches are sorted using our DCT Scouring module with $K = 6$ different band-pass filters in the frequency domain. Subsequently, we select two highest-frequency and two lowest-frequency patches using the calculated DCT scores. These selected patches are then resized to 256×256 and extracted their noise pattern using SRM [21]. For SFE channel, we use the pre-trained OpenCLIP [31] to extract semantic features. We adopt data augmentations including random JPEG compression (QF $\sim$ Uniform$(30, 100)$) and random Gaussian blur ($\sigma \sim$ Uniform$(0.1, 3.0)$) to improve the robustness of detectors. Each augmentation is conducted with 10% probability. During the training phase, we use AdamW optimizer with the learning rate of 1×10^{-4} in $\mathcal{B}_1$ and 5×10^{-4} in $\mathcal{B}_2$, respectively. The batch size is set to 32 and the model is trained on 8 NVIDIA A100 GPUs for only 5 epochs. Our method trains very quickly, only 2 hours are sufficient.

A.2 Baseline Detectors

We choose a set of representative methods in AI-generated detection as baselines for comparison, including frequency-based [20, 37, 70], gradient-based [59], semantic-based [49], reconstruction-based [63], etc.

- CNNSpot (CVPR'2020) [62] proposes that a naïve image classifier with simple data augmentations (*i.e.,* JPEG compression and Gaussian blur) can generalize surprisingly to images generated by unknown GAN-based architectures.
- FreDect (ICML'2020) [20] observes significant artifacts in the frequency domain of GAN-generated images and makes use of these artifacts for classification.
- Fusing (ICIP'2022) [32] designs a two-branch model to fuse global spatial information and local informative features for training the classifier.
- GramNet (CVPR'2020) [40] leverages global image texture representations to improve the robustness and generalization in detecting AI-generated images.
- LNP (ECCV'2022) [37] proposes to extract the noise pattern of images with a learnable denoising network and uses noise patterns to train a classifier.
- LGrad (CVPR'2023) [59] employs gradients computed by a pretrained CNN model to present the generalized artifacts for classification.
- UnivFD (CVPR'2023) [49] uses CLIP features to train a binary liner classifier.
- DIRE (ICCV'2023) [63] observes obvious differences in discrepancies between images and their reconstruction by DMs and uses this feature to train a classifier.
- PatchCraft (Arxiv'2024) [70] compares rich-texture and poor-texture patches from images and extracts the inter-pixel correlation discrepancy as a universal fingerprint for classification.

A.3 Statistics of Public Benchmarks

Table 7 provides a detailed explanation of Benchmark 1 & 2 introduced in our main paper. There are two main benchmarks here: AIGCDetectBenchmark [62] and GenImage [74]. **AIGCDetectBenchmark**: It is trained on ProGAN and then tested on 16 different test sets, including data generated by both GAN and Stable Diffusion models. **GenImage**: It is trained on Stable Diffusion V1.4 and tested on a large amount of data generated by Stable Diffusion, with only a small portion of GAN data included. The test sets related to Stable Diffusion in AIGCDetectBenchmark are consistent with those used in GenImage.

Table 7: **Statistics of Benchmark 1 & 2.** SD and WFIR refer to Stable Diffusion and whichfaceisreal, respectively. The term "Number" only counts on fake images and an equal number of real images is added for each generative model from the same source. The BigGAN test sets in $\mathcal{B}_1$ and $\mathcal{B}_2$ are different, from ForenSynths [62] and GenImage [74], respectively.

	Benchmark 1				Benchmark 2			
	Generator	**Image Size**	**Number**	**Source**	**Generator**	**Image Size**	**Number**	**Source**
Train	ProGAN [33]	256×256	360.0k	LSUN [67]	SD v1.4 [6]	512×512	324.0k	ImageNet [16]
Test	ProGAN [33]	256×256	8.0k	LSUN [67]	BigGAN [13]	256×256	12.0k	ImageNet [16]
	StyleGAN [34]	256×256	12.0k	LSUN [67]				
	BigGAN [13]	256×256	4.0k	ImageNet [16]	ADM [17]	256×256	12.0k	ImageNet [16]
	CycleGAN [72]	256×256	2.6k	ImageNet [16]				
	StarGAN [15]	256×256	4.0k	CelebA [39]	Glide [46]	256×256	12.0k	ImageNet [16]
	GauGAN [50]	256×256	10.0k	COCO [36]				
	StyleGAN2 [35]	256×256	15.9k	LSUN [67]	Midjourney [4]	1024×1024	12.0k	ImageNet [16]
	WFIR [9]	1024×1024	2.0k	FFHQ [34]				
	ADM [17]	256×256	12.0k	ImageNet [16]	SD v1.4 [6]	512×512	12.0k	ImageNet [16]
	Glide [46]	256×256	12.0k	ImageNet [16]				
	Midjourney [4]	1024×1024	12.0k	ImageNet [16]	SD v1.5 [6]	512×512	16.0k	ImageNet [16]
	SD v1.4 [6]	512×512	12.0k	ImageNet [16]				
	SD v1.5 [6]	512×512	16.0k	ImageNet [16]	VQDM [24]	256×256	12.0k	ImageNet [16]
	VQDM [24]	256×256	12.0k	ImageNet [16]				
	Wukong [10]	512×512	12.0k	ImageNet [16]	Wukong [10]	512×512	12.0k	ImageNet [16]
	DALLE 2 [54]	256×256	2.0k	ImageNet [16]				

Table 8: **Benchmark1**. The average precision (AP %) of different baselines (rows) in detecting real and fake images from different generators (columns). GAN-Average and DM-Average are averaged over the first 8 and the last 8 test sets, respectively. The best result and the second-best result are marked in **bold** and underline, respectively.

Generator	**CNNSpot [62]**	**FreDect [20]**	**Fusing [32]**	**GramNet [40]**	**LNP [37]**	**LGrad [59]**	**UnivFD [49]**	**DIRE-G [63]**	**DIRE-D [63]**	**PatchCraft [70]**	**Ours**
ProGAN [33]	**100.00**	99.99	**100.00**	**100.00**	99.99	**100.00**	99.08	58.79	**100.00**	**100.00**	**100.00**
StyleGAN [34]	99.83	88.98	99.50	99.23	98.60	98.31	91.74	56.68	97.56	98.96	**99.99**
BigGAN [13]	85.99	93.62	90.70	81.79	84.32	92.93	75.25	46.91	99.27	**99.42**	94.44
CycleGAN [72]	94.94	84.78	95.50	95.33	92.83	95.01	80.56	50.03	99.80	85.26	**99.89**
StarGAN [15]	99.04	99.49	99.80	99.23	**100.00**	**100.00**	99.34	40.64	99.37	**100.00**	99.99
GauGAN [50]	90.82	82.84	88.30	84.99	78.85	95.43	72.15	47.34	**99.98**	81.33	97.69
StyleGAN2 [35]	99.48	82.54	99.60	99.11	99.59	97.89	88.29	58.03	97.90	97.74	**99.96**
WFIR [9]	**99.85**	55.85	93.30	95.21	91.45	57.99	60.13	59.02	96.73	95.26	99.27
ADM [17]	75.67	61.77	94.10	73.11	94.20	72.95	85.84	**99.79**	86.81	93.40	98.77
Glide [46]	72.28	52.92	77.50	66.76	88.86	80.42	78.35	**99.54**	83.81	94.04	98.94
Midjourney [4]	66.24	46.09	70.00	56.82	76.86	71.86	61.86	**97.32**	74.00	96.48	88.13
SD v1.4 [6]	61.20	37.83	65.40	59.83	94.31	62.37	49.87	98.61	86.14	**99.06**	98.26
SD v1.5 [6]	61.56	37.76	65.70	60.37	93.92	62.85	49.52	98.83	85.84	**99.06**	98.20
VQDM [24]	68.83	85.10	75.60	61.13	87.35	77.47	54.57	98.98	96.53	96.26	**99.27**
Wukong [10]	57.34	39.58	64.60	55.62	92.38	62.48	55.38	98.37	91.07	97.54	**98.62**
DALLE2 [54]	53.51	38.20	68.12	49.82	96.14	82.55	74.48	**99.71**	63.04	99.56	99.41
GAN-Average	96.24	86.01	95.84	94.36	93.20	92.20	83.32	52.18	98.83	94.75	**98.90**
DM-Average	64.58	49.91	72.63	60.43	90.50	71.62	63.73	**98.89**	83.41	96.93	97.45
Average	80.41	67.96	84.23	77.40	91.85	81.91	73.53	75.54	91.12	95.84	**98.18**

B More Experimental Results

B.1 AP Result

We additionally provide classification results regarding AP in Table 8. It is important to highlight that the AP (Average Precision) metric emphasizes different aspects compared to Acc (Accuracy). While Acc focuses on the overall correctness of predictions across all samples, AP provides a more comprehensive evaluation of a model's performance across various thresholds, particularly in handling imbalanced datasets. On top of that, our method still achieves SOTA performance among these baselines on AP metric, which underscores the superiority of our approach. This indicates that our method not only excels in general prediction accuracy but also maintains robust performance across different decision thresholds, demonstrating its effectiveness in distinguishing between classes even in challenging scenarios.