

GENERALIZATION THROUGH DISCREPANCY: LEVERAGING DISTRIBUTIONAL FITTING GAPS FOR AI-GENERATED IMAGE DETECTION

006 **Anonymous authors**

007 Paper under double-blind review

ABSTRACT

013 The generalization of detectors for AI-generated images remains a critical challenge, as methods trained on one generative family often fail when tested on un-
 014 seen architectures. To tackle this generalization challenge, we dive into the in-
 015 herent distribution approximation nature of generative modeling and posit that
 016 a universal forensic signal lies in the discrepancy between mathematically pre-
 017 cise image rescaling traces and the imperfect approximations learned by genera-
 018 tive models through training data. We introduce a novel contrastive pre-training
 019 framework that sensitizes a feature extractor to these subtle rescaling artifacts by
 020 leveraging their inherent periodic patterns and position shift properties, using only
 021 real images for training. Our method sets a new state-of-the-art on both GAN and
 022 diffusion-generated benchmarks, validating the efficacy of our method. We intro-
 023 duce a new and robust perspective on detection generalization through the lens
 024 of distributional fitting divergence. The code and models will be made publicly
 025 available.

1 INTRODUCTION

029 The emergence of high-fidelity generative mod-
 030 els, from Generative Adversarial Networks
 031 (GANs) (Goodfellow et al., 2014) to modern
 032 diffusion processes (Ho et al., 2020), has en-
 033 abled the synthesis of images possessing un-
 034 precedented realism and diversity. As these
 035 synthetic contents become increasingly indis-
 036 tinguishable from real photographs, they not
 037 only foster creativity across artistic and com-
 038 mercial fields but also introduce profound
 039 threats to digital trust and authenticity. The
 040 unchecked spread of AI-generated forgeries ex-
 041 acerbates issues such as large-scale disinfor-
 042 mation and the undermining of evidentiary in-
 043 tegrity in legal and journalistic contexts. These
 044 challenges highlight the critical demand for re-
 045 liable detection methods in the broader effort to
 046 secure AI-generated media.

046 Although detection methods achieve high accuracy in in-domain detection tasks, they often suffer
 047 from substantial performance degradation when exposed to samples from unseen generative models.
 048 Current research on improving detection generalization largely follows two paradigms: prior-based
 049 methods and feature-based methods. Prior-based approaches leverage explicit prior hypotheses to
 050 isolate discriminative forensic traces. For instance, Tan et al. (2023) distinguished generated im-
 051 ages from real ones in gradient feature space through gradient mapping, while Tan et al. (2024b)
 052 enhanced detection by analyzing the upsampling process in generative models and amplifying its
 053 artifacts via residual learning. Feature-based methods leverage the powerful feature extraction capa-
 054 bilities of large pre-trained models to enhance detection generalization. For instance, several studies

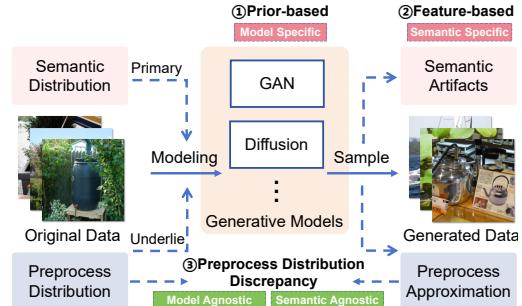


Figure 1: Instead of learning model-specific or semantic-specific features, our framework focuses on the discrepancy of approximated preprocess distribution.

(Ojha et al., 2023; Tan et al., 2025; Cozzolino et al., 2024; Khan & Dang-Nguyen, 2024) utilize the representational power of large pre-trained models to extract image features, thereby amplifying the artifacts in generated images and improving generalization performance.

Although methods based on prior assumptions perform well on specific generative models, their detection performance degrades significantly when confronted with unseen models that violate those assumptions. For instance, Durall et al. (2020) detected images by leveraging high-frequency anomalies caused by upsampling operations in GANs, which generalizes well across various GAN variants but fails against diffusion-generated samples. Similarly, Wang et al. (2023) discriminated images by analyzing reconstruction residuals specific to the diffusion process, yet exhibits limited effectiveness when applied to GAN-generated content. Such performance drops stem from the inherent limitation of prior-based approaches: observations designed for one class of generative models often do not transfer effectively to others. Recent methods avoid manual prior engineering by leveraging the feature extraction capabilities of large-scale pre-trained models. However, since these pre-trained models are often optimized for high-level semantic tasks (Radford et al., 2021) rather than forensic detection, they tend to overlook critical low-level or high-frequency artifacts, thereby impairing generalization performance (Ojha et al., 2023). Although techniques such as orthogonal decomposition (Yan et al., 2025) attempted to disentangle semantic and forensic features, they still fall short of fully addressing this fundamental mismatch. Therefore, improving detection generalization necessitates solving two core challenges: broadening the coverage of prior-based assumptions to encompass diverse generative classes, and aligning pre-trained feature representations with semantically agnostic, forensically relevant cues.

Towards more generalizable prior assumption, we take inspiration from the intrinsic nature of data distribution fitting in generative models. Generative models approximate the semantic distribution of the given training data, yet inevitably introduce discernible discrepancies in the process. Most existing detection methods focus on this semantic-level discrepancies, such as unnatural texture smoothness (Chen & Yashtini, 2024) or anomalies in high-frequency details (Tan et al., 2024b). However, such semantic discrepancies are highly dependent on the training data distribution and exhibit distinct artifact patterns. Moreover, as generative models continue to evolve, semantic-level discrepancies will gradually diminish, ultimately rendering detection methods that rely solely on such features ineffective. Besides semantic distribution approximations, as indicated by Corvi et al. (2023a), generative modeling also captures underlying distributional characteristics of the training data. For instance, when a generative model is trained on JPEG-compressed images, it tends to produce samples exhibiting similar compression artifacts. This phenomenon is consistently observed across diverse generative frameworks, including both GANs and diffusion models. These underlying distributions include JPEG compression, rescaling operation, and other image processing transformations. Since such processing operations possess rigorous mathematical formulations, the inevitable approximation errors introduced by generative models when fitting these processes create measurable discrepancies from the true mathematical distribution.

In this work, we propose to leverage the discrepancies in generative models' fitting of the underlying distributions. Among these, rescaling stands out as the most widely adopted preprocessing step in generative model pipelines. The prevalence of rescaling in generative training stems from two primary factors: 1) widely-used training datasets are web-sourced like ImageNet (Russakovsky et al., 2015) and LSUN (Yu et al., 2015) which inherently undergo rescaling during collection; 2) model constraints (e.g., fully-connected layers, batching) require fixed input sizes, as Stable Diffusion (Rombach et al., 2022) initially resizes images to 256×256 before upscaling to 512×512 . Consequently, these rescaling distributions become intrinsically embedded within the generated images. We therefore select rescaling distribution discrepancy as the primary entry point for generalization detection. Crucially, this discrepancy remains largely invariant to variations in semantic content across datasets, thereby offering stronger generalization for detection tasks. Through further analysis of rescaling distribution properties, we construct a contrastive learning task sensitive to these characteristics to pre-train the feature extractor. The trained extractor becomes highly responsive to rescaling distribution discrepancies, enabling feature extraction from a non-semantic perspective. This approach amplifies fitting divergences in generated images, thereby significantly improving generalization in detection tasks. Our main contributions are as follows:

- We are the first to identify and exploit the pre-process distributional discrepancies between real and generated data, which are independent from specific generative architectures, increasing generalization inherently.

- 108 • We reveal the periodic patterns and position shift properties of the rescaling operation,
109 which derives a novel semantic-agnostic self-supervised contrastive pre-training task to extract
110 authenticity-oriented features, further enhancing our generalization capability.
- 111
- 112 • We establish new state-of-the-art on comprehensive benchmarks, outperforming recent methods
113 by a substantial margin. Extensive experiments in few-shot fine-tuning scenarios indicate that the
114 pre-trained feature extractor effectively focuses on authenticity-relevant characteristics.

116 2 RELATED WORK

119 2.1 GENERATIVE MODELS

121 Generative models have evolved beyond the capabilities of classical autoencoders (Masci et al.,
122 2011; Vincent et al., 2008; Salah et al., 2011), primarily through their capacity to synthesize novel
123 data instances that reflect the true underlying distribution. Although GANs (Goodfellow et al., 2014)
124 once set the standard for image generation with variants such as ProGAN (Karras et al., 2017) for
125 multi-scale learning, StyleGAN (Karras et al., 2019) for style-based control, BigGAN (Brock et al.,
126 2018) for high-capacity synthesis, and StarGAN (Choi et al., 2018) for cross-domain adaptation,
127 they suffer from persistent limitations in output fidelity. Despite advantages in speed and flexi-
128 bility, GANs frequently introduce perceptible semantic distortions that undermine their practical
129 utility. A paradigm shift occurred with the introduction of diffusion models (Ho et al., 2020; Song
130 et al., 2020a;b), which offer a principled probabilistic framework that not only stabilizes training
131 but also significantly enhances output diversity and coherence. Subsequent large-scale implemen-
132 tations (Rombach et al., 2022; Ramesh et al., 2022; Saharia et al., 2022) have further established
133 the superiority of diffusion processes in producing high-resolution, semantically consistent imagery.
134 Consequently, the marked reduction in generative artifacts has rendered many conventional detec-
135 tion mechanisms increasingly inadequate, particularly those reliant on semantic inconsistencies.

136 2.2 GENERALIZATION DETECTION OF AI-GENERATED IMAGE

138 Generalization methods for detection can be broadly categorized into two paradigms: the first lever-
139 ages explicit priors assumptions to extract discriminative forensic artifacts. For instance, Wang
140 et al. (2020) employed data augmentation techniques to effectively enhance cross-model detection
141 performance against GAN-generated images. Subsequent analysis by Durall et al. (2020) demon-
142 strated that the upsampling mechanisms inherent in GAN-based generators produce characteristic
143 high-frequency artifacts, which can be reliably identified via spectral-domain analysis. Tan et al.
144 (2023) leveraged gradient-based analysis to reveal distributional discrepancies between generated
145 and real data, enabling detection through classification in gradient feature space. Zheng et al. (2024)
146 utilized patch-shuffling method to reduce the influence of semantic biases. Liu et al. (2020) and
147 Shiohara & Yamasaki (2022) leveraged the characteristics of underlying textures and features. Al-
148 though methods based on prior assumptions perform effectively on specific models, their detection
149 efficacy diminishes significantly as new generative architectures emerge, since such priors may not
150 consistently generalize across model architectures.

151 The second approach utilizes the remarkable feature extraction capability of large-scale pre-trained
152 models for generalization detection. Ojha et al. (2023) first proposed the use of the pre-trained
153 CLIP (Radford et al., 2021) model for generalizable detection of generated images. Since CLIP is
154 trained on large-scale real-world data, it extract features effectively for discrimination. Furthermore,
155 Cozzolino et al. (2024) and Tan et al. (2025) incorporated the textual module into the visual branch
156 of the CLIP framework, capitalizing on its multimodal nature to further enhance the generalization
157 capability of pre-trained models in detecting generated data. However, pre-trained models tend
158 to prioritize high-level semantic information, often lacking in low-level or high-frequency features
159 that are critical for generated detection. Although methods such as Yan et al. (2025) attempted to
160 disentangle semantic and detection-relevant features via orthogonal decomposition, this limitation
161 still adversely affects detection performance. In this work, we explicitly modeling the inherent
162 characteristics of generative models through pre-training on real data, effectively addressing the
163 generalization challenges across diverse generative models.

162 3 PRELIMINARY

164
165 In this section, we provide a detailed analysis of rescaling distribution properties. Our examination
166 reveals the rigorous mathematical foundation underlying rescaling operation, which motivates
167 the proposed contrastive learning framework. The rescaling process can be implemented through
168 various interpolation methods. In this work, we focus on bilinear interpolation to elucidate the
169 underlying mechanisms, noting that other interpolation techniques exhibit analogous properties.

170 **Rescaling via Bilinear Interpolation.** The bilinear interpolation
171 operation consists of two primary stages: 1) mapping
172 of the pixel position, and 2) interpolation of pixel values. The
173 mapping process involves both row and column dimensions
174 and is defined as follows:

$$175 \begin{aligned} x_{\text{src}} &= (x_{\text{dst}} + 0.5) \cdot \frac{w_{\text{src}}}{w_{\text{dst}}} - 0.5 \\ 176 \quad y_{\text{src}} &= (y_{\text{dst}} + 0.5) \cdot \frac{h_{\text{src}}}{h_{\text{dst}}} - 0.5 \end{aligned} \quad (1)$$

179 where $(x_{\text{src}}, y_{\text{src}})$ denote the corresponding coordinates in the
180 original source image for a pixel located at $(x_{\text{dst}}, y_{\text{dst}})$ in the
181 rescaled image. $(w_{\text{src}}, h_{\text{src}})$ and $(w_{\text{dst}}, h_{\text{dst}})$ represent the
182 width and height of the original image and rescaled image,
183 respectively.

184 Based on the results of the pixel position mapping, the subse-
185 quent step involves interpolating the pixel values. The bilinear interpolation process computes the
186 value for a mapped pixel in the source image by performing linear interpolation using its four near-
187 est neighboring pixels, weighted by their relative positional relationships. As illustrated in Figure
188 2, r_1 and r_2 represent the relative positional relationships between the mapped pixel and its four
189 nearest neighbors in the original image. They are computed as the fractional parts of the mapped
190 coordinates, obtained by:

$$191 \quad r_1 = x_{\text{src}} - \lfloor x_{\text{src}} \rfloor \quad r_2 = y_{\text{src}} - \lfloor y_{\text{src}} \rfloor, \quad (2)$$

193 where $\lfloor \cdot \rfloor$ denotes the floor function. These fractional components quantify the relative offsets within
194 the unit pixel cell along the horizontal and vertical directions, respectively. The detailed computa-
195 tional procedure is given by the formula shown in the figure.

197 **Local Distribution Properties of Bilinear Inter-**
198 **polation.** Owing to the mapping and interpolation
199 relationships of pixel positions, our further analysis
200 reveals two key local distribution properties of bi-
201 linear interpolation: 1) the periodic distributions in
202 interpolated pixel relationships, and 2) the local de-
203 pendence among adjacent interpolated pixels. We
204 illustrate both properties in detail using the exam-
205 ple provided in Figure 3 which illustrates a bilinear
206 interpolation diagram from a 6×6 pixel grid to a
207 4×4 pixel grid. The blue regions represent pixels from the
208 original image, while the red regions correspond to
209 pixels in the rescaled image. Numeric values within
210 the circles indicate corresponding pixel intensities.
211 Row and column indices for both grids are annotated
212 along the top and left sides of the diagram.

212 From Equation (2), it can be observed that when x_{src}
213 and y_{src} are perturbed by integer offsets, the corre-
214 sponding interpolation ratios r_1 and r_2 remain un-
215 changed. As a result, the same neighboring pixel interpolation relationships are maintained during
the pixel value computation. Returning to Equation (1), when x_{dst} and y_{dst} are varied by multiples

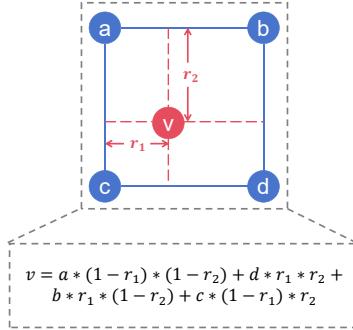


Figure 2: The interpolation re-
lationship between neighboring
pixels in bilinear interpolation
method.

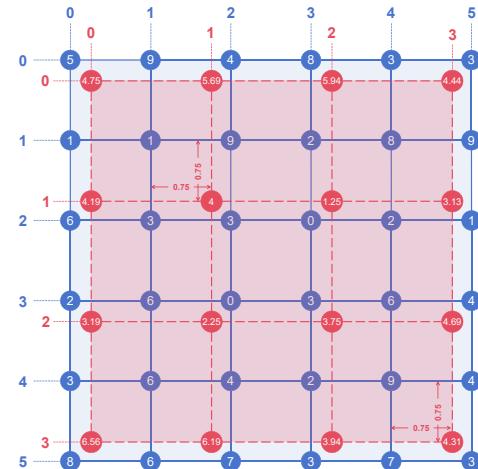


Figure 3: Bilinear interpolation from a 6×6 (blue) to 4×4 (red) pixel grid.

216 of the denominators of $\frac{w_{\text{src}}}{w_{\text{dst}}}$ and $\frac{h_{\text{src}}}{h_{\text{dst}}}$ in their reduced forms, the corresponding changes in x_{src} and
 217 y_{src} will be integer-valued. Consequently, when x_{dst} and y_{dst} vary periodically with a specific pe-
 218 riod, the resulting interpolation distribution also exhibits periodic behavior. As illustrated in Figure
 219 3 by the points $(1, 1)$ and $(3, 3)$ in the rescaled image, the scaling ratios $\frac{w_{\text{src}}}{w_{\text{dst}}} = \frac{h_{\text{src}}}{h_{\text{dst}}} = \frac{6}{4}$ simplify to
 220 $\frac{3}{2}$. Therefore, when x_{dst} and y_{dst} are altered with a period of 2, identical interpolation relationships
 221 are maintained. Similarly, this property holds for the points $(1, 3)$ and $(3, 1)$.
 222

223 The second characteristic of bilinear interpolation is its dependency on neighboring pixels during
 224 the interpolation process. As shown in Figure 3, the interpolation of both column 0 and column 1 in
 225 the rescaled image depends on the pixel values from column 1 of the original image. In contrast, the
 226 interpolation of column 1 and column 2 in the rescaled image exhibits no connection and they are
 227 entirely independent. This phenomenon arises from accumulated deviations during pixel position
 228 mapping, leading to the position shift. The interval of these shifts are determined by the rescaling
 229 ratio between original and rescaled images. Such variations in local dependency constitute one of
 230 the distinctive features of rescaling distributions under different ratios.
 231

232 4 METHODOLOGY

233 Through a detailed analysis of the rescaling process, we identify characteristic properties of inter-
 234 polation distributions. Based on this, we leverage contrastive learning to enable classifiers to model
 235 these interpolation distributions, thereby achieving generalized detection of generated images from
 236 the perspective of distributional fitting discrepancies.
 237

238 4.1 PROBLEM DEFINITION

239 Generalized detection of generated images is a binary classification problem aimed at determining
 240 whether a given image is model-generated. Typically, the task is formulated as fine-tuning a clas-
 241 sifier on data synthesized by a single generative model, while generalizing to detect images from
 242 diverse unseen models (Wang et al., 2020). This cross-model generalization encompasses both vari-
 243 ants within the same model family (e.g., different GAN architectures) and transfers across distinct
 244 families (e.g., from GANs to diffusion models).
 245

246 Concretely, let $\mathbf{x} \in \mathbb{R}^{h \times w \times 3}$ denotes an RGB input image with height h and width w . The source
 247 label y of \mathbf{x} belongs to the set $\mathcal{Y} = \{R_1, G_1\}$. A classifier f_θ is trained on the dataset $\mathcal{D}_{\text{train}} =$
 248 (\mathbf{x}_i, y_i) where $y_i \in \mathcal{Y}$. The trained classifier is then required to generalize to test images \mathbf{x}' drawn
 249 from unseen sources $\{R_2, R_3, \dots, R_N, G_2, G_3, \dots, G_N\}$, and make predictions according to:
 250

$$251 \hat{y} = \begin{cases} \text{real,} & \text{if } f_\theta(\mathbf{x}') \geq \tau \\ 252 \text{fake,} & \text{otherwise} \end{cases} \quad (3)$$

253 where τ is a decision threshold. Here, $\{G_i\}_{i=1}^N$ denote N distinct generative models, and $\{R_i\}_{i=1}^N$
 254 denote N different real sources.
 255

256 4.2 RESCALING-CONTRASTIVE PRE-TRAINING FOR GENERALIZED DETECTION

257 **Pre-training via Rescaling Contrastive Learning.** The pre-training process of the classifier con-
 258 sists of three main stages: 1) applying bilinear interpolation to images according to given rescaling
 259 ratios, thereby generating images with diverse rescaling distributions; 2) selecting image patches
 260 to form positive and negative sample pairs based on their rescaling distributions and relative pos-
 261 itional relationships; and 3) pre-training the model on the constructed sample pairs using supervised
 262 contrastive learning. The detailed descriptions of the main stages of the pre-training procedure are:
 263

264 During the image rescaling stage, we randomly select scaling ratios $s \in (1, 2)$, and rescaling each
 265 image according to its assigned ratio, as illustrated in Figure 4(a). Here, s_1 and s_2 denote distinct
 266 scaling factors. Although input images may have different original resolutions, those rescaled with
 267 the same ratio exhibit identical local rescaling distributions. Due to the pixel position mapping
 268 and local interpolation dependencies as analyzed in Section 3, patches from images rescaled with
 269 different ratios exhibit distinct interpolation distributions at any location.

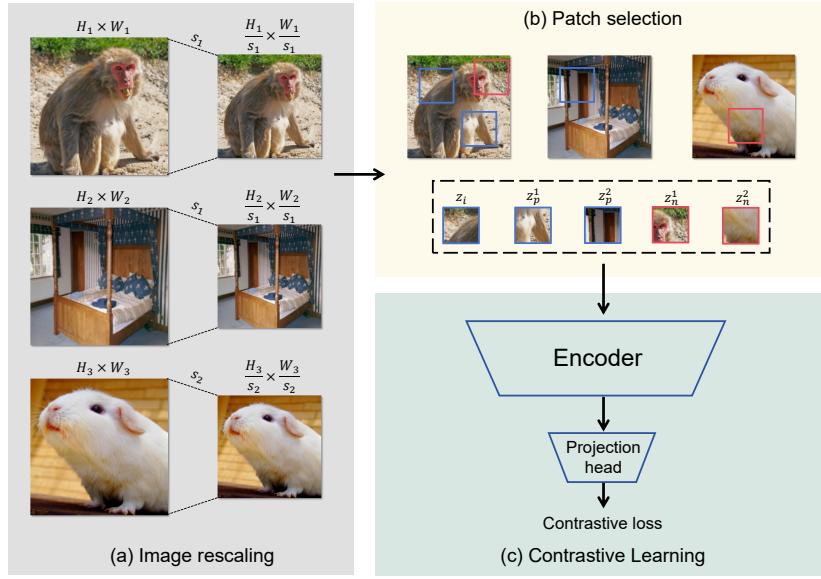


Figure 4: **Pipeline for constructing a contrastive learning setting via image rescaling.** (a) Image rescaling process using different ratio combinations. (b) Selection of positive and negative samples based on rescaling ratios and positional relationships. (c) Pre-training a classification model with contrastive learning to enhance sensitivity to rescaling distributions.

After rescaling different images with assigned ratios, the periodic property of rescaling distributions necessitates that the selection of positive sample pairs must consider both the rescaling ratio and the relative positional relationships of patches. Positive pairs are constructed by selecting patches from images with identical rescaling ratios according to periodic positional relationships. Negative samples comprise two types: 1) patches from images with different rescaling ratios (regardless of position), and 2) patches from images with the same rescaling ratio but at aperiodic positions.

As shown in Figure 4(b), Image 1 and Image 2 share the same rescaling ratio s_1 , while Image 3 has a different ratio s_2 . For an anchor patch z_i in Image 1, we compute its periodic positional relationship based on ratio s_1 , and randomly select another patch z_p^1 at a position corresponding to an integer multiple of this period. Similarly, a patch z_p^2 is selected from Image 2 following the same periodic relationship. Negative samples are constructed by selecting z_n^1 from aperiodic positions in Image 1 and z_n^2 from an arbitrary position in Image 3.

We employ a contrastive learning approach to pre-train the model for rescaling distribution modeling, with the details shown in Figure 4(c). The positive and negative sample pairs obtained during the patch selection stage are fed into an encoder, followed by a projection head. The projection head is implemented as a simple Multi-Layer Perceptron (MLP), which maps the encoded features to a latent space where the supervised contrastive loss (Khosla et al., 2020) is computed. The objective function is defined as follows:

$$\mathcal{L}_{\text{con}} = \sum_{i=1}^N \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{j=1}^N \mathbf{1}_{[j \neq i]} \exp(z_i \cdot z_j / \tau)}, \quad (4)$$

where z_i denotes the projected feature of the i -th sample, $P(i)$ represents the set of indices belonging to the same class as sample i within the batch, τ is a temperature parameter, and N is the batch size.

Inference. During inference, the projection head is discarded and replaced with a binary classification head for fine-tuning. The fine-tuning process can be performed either by freezing the encoder parameters and updating only the classification head, or by jointly optimizing the entire network. Beyond generalization detection, since the features extracted by the encoder, which are grounded in rescaling distributions, already capture the discrepancy between the rescaling distributions approximated by generative models and those derived through mathematical modeling, few-shot fine-tuning proves highly effective. This enables incremental learning with minimal samples when encountering unseen generative models.

324 Table 1: Cross-GAN performance (ACC./A.P.) comparison on the **Self-Synthesis** dataset (9 GAN
 325 variants). **Bold** and underline indicate the best and second-best results, respectively.

Method	Ref	AttGAN	BEGAN	CramerGAN	InfoMaxGAN	MMDGAN	RelGAN	S3GAN	SNGAN	STGAN	Mean
CNNSpot	CVPR2020 (Wang et al., 2020)	51.1 / 83.7	50.2 / 44.9	81.5 / 97.5	71.1 / 94.7	72.9 / 94.4	53.3 / 82.1	55.2 / 66.1	62.7 / 90.4	63.0 / 92.7	62.3 / 82.9
Frank	PMLR2020 (Frank et al., 2020)	65.0 / 74.4	39.4 / 39.9	31.0 / 36.0	41.1 / 41.0	38.4 / 40.5	69.2 / 96.2	69.7 / 81.9	48.4 / 47.9	25.4 / 34.0	47.5 / 54.7
Durall	CVPR2020 (Durall et al., 2020)	39.9 / 38.2	48.2 / 30.9	60.9 / 67.2	50.1 / 51.7	59.5 / 65.5	80.0 / 88.2	<u>87.3</u> / 97.0	54.8 / 58.9	62.1 / 72.5	60.3 / 63.3
Patchfor	ECCV2020 (Chai et al., 2020)	68.0 / 92.9	97.1 / 100.0	<u>97.8</u> / 99.9	93.6 / 98.2	97.9 / 100.0	<u>99.6</u> / 100.0	66.8 / 68.1	97.6 / 99.8	92.7 / 99.8	90.1 / 95.4
F3Net	ECCV2020 (Qian et al., 2020)	<u>85.2</u> / 94.8	87.1 / 97.5	89.5 / <u>99.8</u>	67.1 / 83.1	73.7 / 99.6	98.8 / 100.0	65.4 / 70.0	51.6 / 93.6	60.3 / <u>99.9</u>	75.4 / 93.1
GANDetect	ICIP2022 (Mandelli et al., 2022)	57.4 / 75.1	67.9 / 100.0	67.8 / 99.7	67.6 / 92.4	67.7 / 99.3	60.9 / 86.2	69.6 / 83.5	66.7 / 90.6	69.6 / 97.2	66.1 / 91.6
LGraD	CVPR2023 (Tan et al., 2023)	68.6 / 93.8	69.9 / 89.2	50.3 / 54.0	71.1 / 82.0	57.5 / 67.3	89.1 / <u>99.1</u>	78.5 / 86.0	78.0 / 87.4	54.8 / 68.0	68.6 / 80.8
UnivFD	CVPR2023 (Ojha et al., 2023)	78.5 / <u>98.3</u>	72.0 / 98.9	77.6 / <u>99.8</u>	77.6 / <u>98.9</u>	77.6 / <u>99.7</u>	78.2 / 98.7	85.2 / 98.1	77.6 / 98.7	74.2 / 97.8	77.6 / 98.8
NPR	CVPR2024 (Tan et al., 2024b)	83.0 / 96.2	<u>99.0</u> / <u>99.8</u>	98.7 / 99.0	94.5 / 98.3	98.6 / 99.0	<u>99.6</u> / 100.0	79.0 / 80.0	88.8 / 97.4	98.0 / 100.0	<u>93.2</u> / 96.6
Ours	-	98.9 / 100.0	100.0 / 100.0	96.9 / 99.5	96.9 / 99.9	96.8 / <u>99.7</u>	99.7 / 100.0	94.7 / 98.9	<u>93.8</u> / 98.3	<u>97.4</u> / 100.0	97.2 / 99.6

334 5 EXPERIMENTS

335 5.1 EXPERIMENT SETUP

339 **Datasets.** To evaluate the generalization performance of the proposed method in practical scenarios, our dataset encompasses a diverse range of GANs, diffusion models, and various real image sources. To assess cross-model generalization across GAN variants, we follow the setting of the 340 NPR (Tan et al., 2024b): classifier is trained on the ForenSynths (Wang et al., 2020) dataset and 341 evaluated on the Self-Synthesis (Tan et al., 2024a) dataset. The ForenSynths dataset contains 20 semantic 342 categories, though only four (i.e., car, cat, chair, horse) are used during training to maintain 343 consistency with prior works. The Self-Synthesis dataset includes multiple GAN variants such as 344 AttGAN (He et al., 2019) and BEGAN (Berthelot et al., 2017). We further examine generalization 345 capability across different diffusion models following C2P-CLIP (Tan et al., 2025), with validation 346 performed on the GenImage (Zhu et al., 2023) dataset. The training subset consists of data generated 347 by the SDv1.4 model (Rombach et al., 2022), while the test set includes samples from multiple 348 diffusion models (e.g., ADM (Dhariwal & Nichol, 2021), GLIDE (Nichol et al., 2021)) as well as 349 the BigGAN model (Brock et al., 2018). Real images are sourced from the LSUN (Yu et al., 2015) 350 and ImageNet (Russakovsky et al., 2015) datasets.

352 **Implementation Details.** Our method is implemented utilizing the PyTorch (Paszke et al., 2019) 353 framework with 8 NVIDIA 3090 GPUs. The encoder architecture adopts the Xception (Chollet, 354 2017) backbone. During training, images are randomly cropped to 128×128 patches, while center 355 cropping is applied during testing. We use the Adam (Kingma & Ba, 2014) optimizer with an initial 356 learning rate of 2×10^{-4} . First order moment decay rate and second order moment decay rate are 357 set to 0.9 and 0.999, respectively, and weight decay is set to 2×10^{-4} . The classifier is pre-trained 358 for 200 epochs with a batch size of 128 on ImageNet dataset. Unless otherwise specified, the entire 359 network is jointly fine-tuned by default in subsequent experiments.

360 **Evaluation Metrics.** Following existing works (Ojha et al., 2023; Tan et al., 2024b; 2025), we 361 compare the effectiveness of different methods using Accuracy (Acc) and Average Precision (AP). 362 The Acc metric is computed with a fixed threshold of 0.5 across all benchmarks, ensuring a fair and 363 consistent comparison of detection performance.

365 5.2 MAIN RESULTS

367 **Evaluation on Self-Synthesis Dataset.** Table 1 presents the cross-model generalization accuracy 368 across nine GAN architectures. Following the training setting of NPR (Tan et al., 2024b), all 369 competing methods are fine-tuned using ProGAN data across four semantic categories. Our method 370 significantly outperforms the UnivFD baseline (Ojha et al., 2023) by 19.6% and exceeds the 371 current state-of-the-art NPR by 4.0% in classification accuracy, demonstrating strong generalization 372 capability across diverse GAN architectures.

374 **Evaluation on GenImage Dataset.** Table 2 summarizes the detection accuracy across multiple 375 methods, including those reported in GenImage (Zhu et al., 2023), C2P-CLIP (Tan et al., 2025), 376 DRCT (Chen et al., 2024) and Effort (Yan et al., 2025). All methods are trained using SDv1.4 377 in the GenImage dataset. The GenImage benchmark incorporates synthetic images produced by 378 advanced diffusion models, including commercial systems such as MidJourney and WuKong. A

378 Table 2: Accuracy comparison on the **GenImage** dataset with SDv1.4 as the training dataset. **Bold**
 379 and underline denote the best and second-best performance, respectively.
 380

381 Methods	382 Ref	383 SDv1.4	384 SDv1.5	385 Midjourney	386 ADM	387 GLIDE	388 Wukong	389 VQDM	390 BigGAN	391 mAcc
382 ResNet-50	383 CVPR2016 (He et al., 2016)	384 99.9	385 99.7	386 54.9	387 53.5	388 61.9	389 98.2	390 56.6	391 52.0	392 72.1
383 DeiT-S	384 ICML2021 (Touvron et al., 2021)	385 99.9	<u>386 99.8</u>	387 55.6	388 49.8	389 58.1	390 98.9	391 56.9	392 53.5	393 71.6
384 Swin-T	385 ICCV2021 (Liu et al., 2021)	386 99.9	<u>387 99.8</u>	388 62.1	389 49.8	390 67.6	<u>391 99.1</u>	392 62.3	393 57.6	394 74.8
385 CNNSpot	386 CVPR2020 (Wang et al., 2020)	387 96.3	388 95.9	389 52.8	390 50.1	391 39.8	392 78.6	393 53.4	394 46.8	395 64.2
386 Spec	387 WIFS2019 (Zhang et al., 2019)	388 99.4	389 99.2	390 52.0	391 49.7	392 49.8	393 94.8	394 55.6	395 49.8	396 68.8
387 F3Net	388 ECCV2020 (Qian et al., 2020)	389 99.9	390 99.9	391 50.1	392 49.9	393 50.0	<u>394 99.9</u>	395 49.9	396 49.9	397 68.7
388 GramNet	389 CVPR2020 (Liu et al., 2020)	390 99.2	391 99.1	392 54.2	393 50.3	394 54.6	395 98.9	396 50.8	397 51.7	398 69.9
389 UnivFD	390 CVPR2023 (Ojha et al., 2023)	391 96.4	392 96.2	393 93.9	394 71.9	395 85.4	396 94.3	397 81.6	398 90.5	399 88.8
390 DIRE	391 ICCV2023 (Wang et al., 2023)	392 100.0	393 99.9	394 50.4	395 52.3	396 67.2	397 100.0	398 50.1	399 50.0	400 71.2
391 FreqNet	392 AAAI2024 (Tan et al., 2024a)	393 98.8	394 98.6	395 89.6	396 66.8	397 86.5	398 97.3	399 75.8	400 81.4	401 86.8
392 NPR	393 CVPR2024 (Tan et al., 2024b)	394 98.2	395 97.9	396 81.0	397 76.9	398 89.8	399 96.9	400 84.1	401 84.2	402 88.6
393 FatFormer	394 CVPR2024 (Liu et al., 2024)	395 100.0	396 99.9	<u>397 92.7</u>	398 75.9	399 88.0	<u>400 99.9</u>	<u>401 98.8</u>	402 55.8	403 88.9
394 DRCT	395 ICML2024 (Chen et al., 2024)	396 95.0	397 94.4	<u>398 91.5</u>	399 79.4	400 89.2	<u>401 94.7</u>	402 90.0	403 81.7	404 89.5
395 C2P-CLIP	396 AAAI2025 (Tan et al., 2025)	397 90.9	398 97.9	399 88.2	400 96.4	401 99.0	402 98.8	403 96.5	<u>404 98.7</u>	<u>405 95.8</u>
396 Effort	397 ICML2025 (Yan et al., 2025)	398 99.8	399 99.8	400 82.4	401 78.7	402 93.3	403 97.4	404 91.7	<u>405 77.6</u>	406 91.1
397 Ours	398 -	399 99.9	400 99.9	401 92.1	<u>402 94.2</u>	403 98.8	404 99.7	405 99.7	406 99.9	407 98.0

393 Table 3: Evaluation on other setups for the proposed method, including ablation study, linear layer
 394 fine-tuning and few-shot fine-tuning tasks.
 395

396 Methods	397 SDv1.4	398 SDv1.5	399 Midjourney	400 ADM	401 GLIDE	402 Wukong	403 VQDM	404 BigGAN	405 mAcc
<i>Ablation Study</i>									
Xception	93.1	91.9	65.6	54.2	73.6	88.3	61.5	64.2	74.1
Ours+Pre-training(Ours)	99.9	99.9	<u>92.1</u>	94.2	98.8	99.7	99.7	99.9	98.0
<i>Linear layer Fine-tuning</i>									
UnivFD(fc)	96.4	96.2	93.9	71.9	85.4	94.3	81.6	90.5	88.8
Ours(fc)	99.1	98.9	90.8	91.4	95.6	98.5	98.1	96.5	96.1
<i>Few-shot Fine-tuning</i>									
UnivFD+4-shot	96.4	97.8	94.3	88.9	96.2	96.2	90.5	97.4	94.7
NPR+4-shot	98.2	98.6	94.5	92.3	95.7	97.5	88.6	81.8	93.4
Ours+4-shot	99.9	99.9	98.7	98.9	99.3	100.0	99.8	99.8	99.5
UnivFD+8-shot	96.4	97.8	95.7	91.6	97.3	96.7	93.9	99.0	96.1
NPR+8-shot	98.2	98.7	96.7	96.1	97.2	97.9	89.1	86.3	95.0
Ours+8-shot	99.9	99.9	99.3	99.5	99.8	100.0	100.0	100.0	99.8

409 notable characteristic of this dataset is the inclusion of high-resolution imagery (e.g., 1024×1024
 410 pixels from MidJourney), whose divergence from conventional resolutions introduces significant
 411 resolution bias, further challenging detection robustness. Our approach, based on rescaling distribution
 412 discrepancy, achieves a new state-of-the-art average accuracy of 98.0%, surpassing the UnivFD
 413 baseline and prior best method C2P-CLIP by margins of 9.2% and 2.2%, respectively. The results
 414 demonstrate the efficacy of our proposed method in detecting images generated by diffusion models.
 415

416 **Proposed Pre-training Improves Baseline Performance.** We evaluate the generalization performance
 417 of both the baseline method and our approach incorporating the proposed pre-training strategy,
 418 as summarized in Table 3. All models are trained on SDv1.4. The results indicate that the
 419 Xception model, similar to the ResNet-50 results reported in Table 2, fails to achieve generalization
 420 in the baseline setting. However, when enhanced with our pre-training procedure, it attains strong
 421 generalization by effectively leveraging distributional discrepancies.
 422

423 **Linear Layer Fine-tuning.** We further evaluate the scenario where the pre-trained model parameters
 424 are frozen and only the linear classification head is fine-tuned. As shown in Table 3, we compare
 425 our method with UnivFD, which also fine-tunes a linear layer on top of a frozen CLIP model. The
 426 comparative results demonstrate that our pre-training approach learns features more relevant to de-
 427 tection tasks, leading to superior performance.
 428

429 **Few-shot Fine-tuning.** To address scenarios where limited samples from unseen models available
 430 for adaptation, we further evaluate few-shot incremental learning performance, as reported in Table
 431 3. We compare our approach with UnivFD and NPR, which represent pre-training fine-tuning and
 432 prior-based feature extraction paradigms, respectively. All models are first trained on SDv1.4, then
 433 fine-tuned with either 4-shot or 8-shot samples from each target model before evaluation. The results
 434

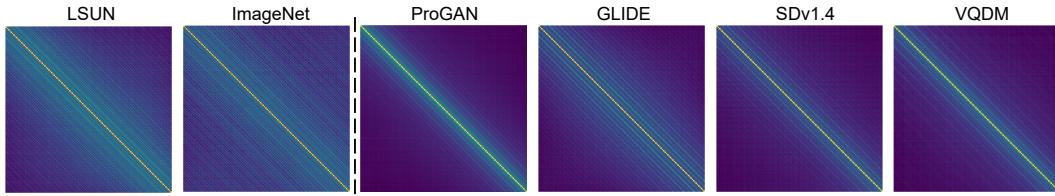


Figure 5: Average cosine similarity map between image patches extracted by the pre-trained model.

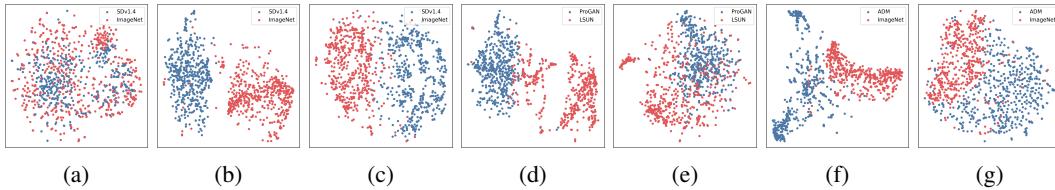


Figure 6: t-SNE visualization of features extracted by different encoders across varying data distributions and post-processing strategies.

show that while UnivFD and NPR exhibit improvements, they struggle to achieve high accuracy on specific categories. In contrast, our method consistently attains superior accuracy across all target models, demonstrating the efficacy of the features learned by our pre-training framework.

Generated Data Exhibits Distinct Cosine Similarity Map. To visualize the relationships of the local rescaling distributions within an image, we randomly crop a 256×256 region from each image sample. Along the diagonal direction, we extract 128 patches of size 128×128 with a stride of 1. Each patch is processed by our proposed pre-trained model to extract features, resulting in a feature matrix of size 128×2048 , where 2048 is the output dimension of the model. By computing the pairwise cosine similarity between all patch features, we obtain a 128×128 similarity matrix. This matrix is averaged over 1000 images to produce the final similarity map, as shown in Figure 5. The resulting visualization reveals that real images exhibit multiple bright bands parallel to the diagonal, indicating the periodic nature of local interpolation distributions across different rescaling operations. In contrast, generated images lack such distinctive patterns, revealing a distributional fitting discrepancy which enables effective generalization detection.

Pre-trained Extractor Remains Robust against Post-rescaling. Figure 6 presents t-SNE visualizations of features extracted by different models under varying data categories and post-processing conditions. (a) shows features from the CLIP used in UnivFD, which struggles to distinguish generative images. (b)-(g) show the features extracted from our proposed pre-trained model. Specifically, (b), (d) and (f) display evaluations on different generative models using randomly cropped 256×256 patches from original images. (c), (e) and (g) show corresponding features of images rescaled before feature extracting, where images are rescaled from 256×256 to 224×224 (c), 192×192 (e), and 160×160 (g), respectively. The results demonstrate that our pre-trained model maintains clear separability across both original and rescaled images, indicating that our approach does not rely merely on the presence or absence of rescaling artifacts. Instead, it operates on more fine-grained distributional discrepancies inherent in the approximations of generative models.

6 CONCLUSION

This paper proposes a novel detection method that leverages distributional discrepancies in rescaling operations. By analyzing interpolation properties, we identify consistent fitting gaps between generative models and mathematical rescaling. Our contrastive pre-training framework enables models to learn these fine-grained discrepancies rather than semantic features. Experiments show state-of-the-art performance across GANs and diffusion models, with strong generalization in various settings. Visualizations confirm the method captures fundamental distributional properties beyond superficial artifacts. This work provides a new perspective of distributional discrepancy for generalization detection against evolving generative AI.

486 REFERENCES
487

488 Quentin Bammey. Synthbuster: Towards detection of diffusion model generated images. *IEEE Open
489 Journal of Signal Processing*, 5:1–9, 2023.

490 Lorenzo Baraldi, Federico Cocchi, Marcella Cornia, Lorenzo Baraldi, Alessandro Nicolosi, and Rita
491 Cucchiara. Contrasting deepfakes diffusion via contrastive learning and global-local similarities.
492 In *European Conference on Computer Vision*, pp. 199–216. Springer, 2024.

493 David Berthelot, Thomas Schumm, and Luke Metz. Began: Boundary equilibrium generative ad-
494 versarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

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

497 Bar Cavia, Eliahu Horwitz, Tal Reiss, and Yedid Hoshen. Real-time deepfake detection in the real-
498 world. *arXiv preprint arXiv:2406.09398*, 2024.

499 Lucy Chai, David Bau, Ser-Nam Lim, and Phillip Isola. What makes fake images detectable? under-
500 standing properties that generalize. In *Computer vision–ECCV 2020: 16th European conference,
Glasgow, UK, August 23–28, 2020, proceedings, part XXVI 16*, pp. 103–120. Springer, 2020.

501 You-Ming Chang, Chen Yeh, Wei-Chen Chiu, and Ning Yu. Antifakeprompt: Prompt-tuned vision-
502 language models are fake image detectors. *arXiv preprint arXiv:2310.17419*, 2023.

503 Baoying Chen, Jishen Zeng, Jianquan Yang, and Rui Yang. Drct: Diffusion reconstruction con-
504 trastive training towards universal detection of diffusion generated images. In *Forty-first Interna-
505 tional Conference on Machine Learning*, 2024.

506 Yuming Chen and Maryam Yashtini. Detecting ai generated images through texture and frequency
507 analysis of patches. In *2024 4th International Conference on Artificial Intelligence, Virtual Reality
508 and Visualization*, pp. 103–110. IEEE, 2024.

509 Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Star-
510 gan: Unified generative adversarial networks for multi-domain image-to-image translation. In
511 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797,
512 2018.

513 François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings
514 of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

515 Riccardo Corvi, Davide Cozzolino, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. Intriguing
516 properties of synthetic images: from generative adversarial networks to diffusion models. In
517 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 973–
518 982, 2023a.

519 Riccardo Corvi, Davide Cozzolino, Giada Zingarini, Giovanni Poggi, Koki Nagano, and Luisa Ver-
520 doliva. On the detection of synthetic images generated by diffusion models. In *ICASSP 2023-2023
521 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5.
522 IEEE, 2023b.

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

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

528 Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: Cnn based gener-
529 ative deep neural networks are failing to reproduce spectral distributions. In *Proceedings of the
530 IEEE/CVF conference on computer vision and pattern recognition*, pp. 7890–7899, 2020.

531 Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten
532 Holz. Leveraging frequency analysis for deep fake image recognition. In *International conference
533 on machine learning*, pp. 3247–3258. PMLR, 2020.

540 Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,
 541 Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information*
 542 *processing systems*, 27, 2014.

543

544 Patrick Grommelt, Louis Weiss, Franz-Josef Pfreundt, and Janis Keuper. Fake or jpeg? revealing
 545 common biases in generated image detection datasets. *arXiv preprint arXiv:2403.17608*, 2024.

546

547 Fabrizio Guillaro, Giada Zingarini, Ben Usman, Avneesh Sud, Davide Cozzolino, and Luisa Verdoliva.
 548 A bias-free training paradigm for more general ai-generated image detection. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18685–18694, 2025.

549

550 Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recogni-
 551 tion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

552

553 Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute
 554 editing by only changing what you want. *IEEE transactions on image processing*, 28(11):5464–
 555 5478, 2019.

556

557 Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in*
 558 *neural information processing systems*, 33:6840–6851, 2020.

559

560 Zexi Jia, Chuanwei Huang, Yeshuang Zhu, Hongyan Fei, Xiaoyue Duan, Zhiqiang Yuan, Ying Deng,
 561 Jiapei Zhang, Jinchao Zhang, and Jie Zhou. Secret lies in color: Enhancing ai-generated images
 562 detection with color distribution analysis. In *Proceedings of the Computer Vision and Pattern*
 563 *Recognition Conference*, pp. 13445–13454, 2025.

564

565 Yan Ju, Shan Jia, Lipeng Ke, Hongfei Xue, Koki Nagano, and Siwei Lyu. Fusing global and local
 566 features for generalized ai-synthesized image detection. In *2022 IEEE International Conference*
 567 *on Image Processing (ICIP)*, pp. 3465–3469. IEEE, 2022.

568

569 Dimitrios Karageorgiou, Symeon Papadopoulos, Ioannis Kompatsiaris, and Efstratios Gavves. Any-
 570 resolution ai-generated image detection by spectral learning. In *Proceedings of the Computer*
 571 *Vision and Pattern Recognition Conference*, pp. 18706–18717, 2025.

572

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

575

576 Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative
 577 adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern*
 578 *recognition*, pp. 4401–4410, 2019.

579

580 Sohail Ahmed Khan and Duc-Tien Dang-Nguyen. Clipping the deception: Adapting vision-
 581 language models for universal deepfake detection. In *Proceedings of the 2024 International*
 582 *Conference on Multimedia Retrieval*, pp. 1006–1015, 2024.

583

584 Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron
 585 Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural*
 586 *information processing systems*, 33:18661–18673, 2020.

587

588 Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*
 589 *arXiv:1412.6980*, 2014.

590

591 Christos Koutlis and Symeon Papadopoulos. Leveraging representations from intermediate encoder-
 592 blocks for synthetic image detection. In *European Conference on Computer Vision*, pp. 394–411.
 593 Springer, 2024.

594

595 Romeo Lanzino, Federico Fontana, Anxhelo Diko, Marco Raoul Marini, and Luigi Cinque. Faster
 596 than lies: Real-time deepfake detection using binary neural networks. In *Proceedings of the*
 597 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3771–3780, 2024.

598

599 Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, and Fuli Feng. Improving synthetic
 600 image detection towards generalization: An image transformation perspective. In *Proceedings of*
 601 *the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 2405–
 602 2414, 2025.

594 Huan Liu, Zichang Tan, Chuangchuang Tan, Yunchao Wei, Jingdong Wang, and Yao Zhao. Forgery-
 595 aware adaptive transformer for generalizable synthetic image detection. In *Proceedings of the*
 596 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10770–10780, 2024.

597

598 Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo.
 599 Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the*
 600 *IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

601 Zhenghe Liu, Xiaojuan Qi, and Philip HS Torr. Global texture enhancement for fake face detec-
 602 tion in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern*
 603 *recognition*, pp. 8060–8069, 2020.

604

605 Sara Mandelli, Nicolò Bonettini, Paolo Bestagini, and Stefano Tubaro. Detecting gan-generated
 606 images by orthogonal training of multiple cnns. In *2022 IEEE International Conference on Image*
 607 *Processing (ICIP)*, pp. 3091–3095. IEEE, 2022.

608 Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-
 609 encoders for hierarchical feature extraction. In *Artificial neural networks and machine learning—*
 610 *ICANN 2011: 21st international conference on artificial neural networks, espoo, Finland, June*
 611 *14–17, 2011, proceedings, part i 21*, pp. 52–59. Springer, 2011.

612

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

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

619

620 Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, JP Bradbury, Gregory Chanan, Trevor
 621 Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. An imperative style, high-
 622 performance deep learning library. *Advances in neural information processing systems*, 32:8026,
 623 2019.

624 Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. Thinking in frequency: Face
 625 forgery detection by mining frequency-aware clues. In *European conference on computer vision*,
 626 pp. 86–103. Springer, 2020.

627

628 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal,
 629 Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual
 630 models from natural language supervision. In *International conference on machine learning*, pp.
 631 8748–8763. PmLR, 2021.

632

633 Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-
 634 conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

635

636 Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-
 637 resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF confer-*
 638 *ence on computer vision and pattern recognition*, pp. 10684–10695, 2022.

639

640 Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng
 641 Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual
 642 recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

643

644 Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar
 645 Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic
 646 text-to-image diffusion models with deep language understanding. *Advances in neural informa-*
 647 *tion processing systems*, 35:36479–36494, 2022.

648

649 Rifai Salah, P Vincent, X Muller, X Glorot, and Y Bengio. Contractive auto-encoders: Explicit
 650 invariance during feature extraction. In *Proc. of the 28th International Conference on Machine*
 651 *Learning*, pp. 833–840, 2011.

648 Zeyang Sha, Zheng Li, Ning Yu, and Yang Zhang. De-fake: Detection and attribution of fake
 649 images generated by text-to-image generation models. In *Proceedings of the 2023 ACM SIGSAC*
 650 *conference on computer and communications security*, pp. 3418–3432, 2023.

651

652 Kaede Shiohara and Toshihiko Yamasaki. Detecting deepfakes with self-blended images. In *Pro-
 653 ceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18720–
 654 18729, 2022.

655

656 Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv*
 657 *preprint arXiv:2010.02502*, 2020a.

658

659 Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben
 660 Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint*
 661 *arXiv:2011.13456*, 2020b.

662

663 Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, and Yunchao Wei. Learning on gradients:
 664 Generalized artifacts representation for gan-generated images detection. In *Proceedings of the
 665 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12105–12114, 2023.

666

667 Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Frequency-
 668 aware deepfake detection: Improving generalizability through frequency space domain learning.
 669 In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5052–5060,
 670 2024a.

671

672 Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Rethinking
 673 the up-sampling operations in cnn-based generative network for generalizable deepfake detection.
 674 In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp.
 675 28130–28139, 2024b.

676

677 Chuangchuang Tan, Renshuai Tao, Huan Liu, Guanghua Gu, Baoyuan Wu, Yao Zhao, and Yunchao
 678 Wei. C2p-clip: Injecting category common prompt in clip to enhance generalization in deepfake
 679 detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7184–
 680 7192, 2025.

681

682 Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and
 683 Hervé Jégou. Training data-efficient image transformers & distillation through attention. In
 684 *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

685

686 Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and
 687 composing robust features with denoising autoencoders. In *Proceedings of the 25th international
 688 conference on Machine learning*, pp. 1096–1103, 2008.

689

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

693

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

697

698 Shilin Yan, Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, and Weidi Xie. A sanity
 699 check for ai-generated image detection. *arXiv preprint arXiv:2406.19435*, 2024.

700

701 Zhiyuan Yan, Jiangming Wang, Peng Jin, Ke-Yue Zhang, Chengchun Liu, Shen Chen, Taiping Yao,
 702 Shouhong Ding, Baoyuan Wu, and Li Yuan. Orthogonal subspace decomposition for generaliz-
 703 able ai-generated image detection. In *Forty-second International Conference on Machine Learn-
 704 ing*, 2025.

705

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

702 Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in gan fake
 703 images. In *2019 IEEE international workshop on information forensics and security (WIFS)*, pp.
 704 1–6. IEEE, 2019.

706 Chende Zheng, Chenhao Lin, Zhengyu Zhao, Hang Wang, Xu Guo, Shuai Liu, and Chao Shen.
 707 Breaking semantic artifacts for generalized ai-generated image detection. *Advances in Neural*
 708 *Information Processing Systems*, 37:59570–59596, 2024.

709 Nan Zhong, Yiran Xu, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. Patchcraft: Exploring texture
 710 patch for efficient ai-generated image detection. *arXiv preprint arXiv:2311.12397*, 2023a.

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

715 Mingjian Zhu, Hanting Chen, Qiangyu Yan, Xudong Huang, Guanyu Lin, Wei Li, Zhijun Tu, Hailin
 716 Hu, Jie Hu, and Yunhe Wang. Genimage: A million-scale benchmark for detecting ai-generated
 717 image. *Advances in Neural Information Processing Systems*, 36:77771–77782, 2023.

719 A APPENDIX

721 A.1 THE USE OF LARGE LANGUAGE MODELS (LLMs)

723 We employ large language models (LLMs) solely for text polishing and revision of the writing in our
 724 paper. Their use is strictly limited to linguistic refinement and does not extend to the methodological
 725 contributions, experimental design, or any substantive technical content presented in this work.

727 A.2 IMPLEMENTATION DETAILS OF OTHER INTERPOLATION APPROACHES

729 PyTorch provides several interpolation methods, including NEAREST, BOX, BILINEAR, BICU-
 730 BIC, LANCZOS, and HAMMING. We have elaborated on the specific procedure of BILINEAR
 731 interpolation and identified two key characteristics: periodic distributions and local dependency. In
 732 this part, we analyze the detailed mechanisms of other commonly used interpolation methods and
 733 demonstrate that, with the exception of NEAREST interpolation, all other methods exhibit these two
 734 characteristics. Although NEAREST interpolation does not share these properties, it is not used in
 735 data preprocessing of generative models due to the severe artifacts it introduces during rescaling.
 736 Thus, NEAREST interpolation is not discussed in this section.

737 A.2.1 BOX INTERPOLATION

739 Given an output pixel at integer coordinates $(x_{\text{dst}}, y_{\text{dst}})$, its value is computed solely from the por-
 740 tion of the input image that maps to it.

742 **Compute the Input-Space Region of This Output Pixel.** Let the input image size be $(w_{\text{src}}, h_{\text{src}})$
 743 and the output size be $(w_{\text{dst}}, h_{\text{dst}})$. The scaling factors are

$$745 \quad s_x = \frac{w_{\text{src}}}{w_{\text{dst}}}, \quad s_y = \frac{h_{\text{src}}}{h_{\text{dst}}}.$$

747 The continuous region in the input domain corresponding to this single output pixel is

$$749 \quad R_d = [x_{\text{dst}}s_x, (x_{\text{dst}} + 1)s_x] \times [y_{\text{dst}}s_y, (y_{\text{dst}} + 1)s_y].$$

751 **Find All Input Pixels Overlapping This Region.** An input pixel at integer coordinates (x, y)
 752 spans the unit square

$$753 \quad P_{xy} = [x, x + 1] \times [y, y + 1].$$

754 It contributes to the output pixel $(x_{\text{dst}}, y_{\text{dst}})$ if

$$755 \quad P_{xy} \cap R_d \neq \emptyset.$$

756 **Compute the Overlap Area.** For each overlapping input pixel (x, y) , compute the width and
 757 height of the intersection:

$$759 \quad w_{xy} = \max(0, \min(x + 1, (x_{\text{dst}} + 1)s_x) - \max(x, x_{\text{dst}}s_x)),$$

$$761 \quad h_{xy} = \max(0, \min(y + 1, (y_{\text{dst}} + 1)s_y) - \max(y, y_{\text{dst}}s_y)).$$

763 The intersection area is

$$764 \quad A_{xy} = w_{xy} \cdot h_{xy}.$$

765 **Compute the Output Pixel Value.** Let $I(x, y)$ be the input pixel value. The BOX interpolation
 766 result of this single output pixel is

$$768 \quad O(x_{\text{dst}}, y_{\text{dst}}) = \frac{1}{s_x s_y} \sum_{\text{overlapping } (x, y)} I(x, y) A_{xy}.$$

771 **Interpolation Characteristics.** In the BOX interpolation technique, each output pixel is first
 772 mapped back to the source image according to the scaling factors. The contribution of each source
 773 pixel is then determined by its proportional overlap area with the pixel region, which serves as the
 774 interpolation weight. Consequently, similar to BILINEAR interpolation, BOX interpolation exhibits
 775 both periodicity and local dependency.

777 A.2.2 BICUBIC INTERPOLATION

779 Given an output pixel at integer coordinates $(x_{\text{dst}}, y_{\text{dst}})$, BICUBIC interpolation computes its value
 780 by sampling a 4×4 neighborhood around the corresponding position in the source image.

781 **Coordinate Mapping.** Let the input image have size $(w_{\text{src}}, h_{\text{src}})$ and the output size be
 782 $(w_{\text{dst}}, h_{\text{dst}})$. The scaling factors are

$$784 \quad s_x = \frac{w_{\text{src}}}{w_{\text{dst}}}, \quad s_y = \frac{h_{\text{src}}}{h_{\text{dst}}}.$$

786 The output pixel $(x_{\text{dst}}, y_{\text{dst}})$ corresponds to the input coordinate

$$788 \quad x_{\text{src}} = (x_{\text{dst}} + 0.5) s_x - 0.5, \quad y_{\text{src}} = (y_{\text{dst}} + 0.5) s_y - 0.5.$$

789 Let

$$791 \quad x_0 = \lfloor x_{\text{src}} \rfloor, \quad y_0 = \lfloor y_{\text{src}} \rfloor.$$

792 **Bicubic Kernel.** The cubic convolution kernel with parameter a (typically $a = -0.5$) is

$$794 \quad k(t) = \begin{cases} (a+2)|t|^3 - (a+3)|t|^2 + 1, & 0 \leq |t| < 1, \\ a|t|^3 - 5a|t|^2 + 8a|t| - 4a, & 1 \leq |t| < 2, \\ 0, & |t| \geq 2. \end{cases}$$

799 **Compute Horizontal and Vertical Weights.** For the horizontal direction:

$$801 \quad w_i = k(x_{\text{src}} - (x_0 + i)), \quad i \in \{-1, 0, 1, 2\}.$$

802 For the vertical direction:

$$804 \quad v_j = k(y_{\text{src}} - (y_0 + j)), \quad j \in \{-1, 0, 1, 2\}.$$

806 **Bicubic Combination Over a 4×4 Neighborhood.** Let $I(x, y)$ denote the source image pixel
 807 value (per channel). The bicubic interpolated output pixel is

$$808 \quad O(x_{\text{dst}}, y_{\text{dst}}) = \sum_{j=-1}^2 \sum_{i=-1}^2 I(x_0 + i, y_0 + j) w_i v_j.$$

810 **Interpolation Characteristics.** In BICUBIC interpolation, the mapping of pixel positions follows
 811 the same procedure as in BILINEAR interpolation. The key difference is that each mapped pos-
 812 ition is reconstructed using a weighted combination of a 4×4 neighborhood of surrounding pixels.
 813 Consequently, BICUBIC interpolation exhibits the same interpolation characteristics as BILINEAR
 814 interpolation.

815 A.2.3 LANCZOS INTERPOLATION

816 Given an output pixel at integer coordinates $(x_{\text{dst}}, y_{\text{dst}})$, LANCZOS interpolation reconstructs its
 817 value by a separable, windowed-sinc filter with finite support parameter a (commonly $a = 2$ or
 818 $a = 3$).
 819

820 **Coordinate Mapping.** Let the source image size be $(w_{\text{src}}, h_{\text{src}})$ and the destination size be
 821 $(w_{\text{dst}}, h_{\text{dst}})$. Define the scaling factors
 822

$$823 \quad s_x = \frac{w_{\text{src}}}{w_{\text{dst}}}, \quad s_y = \frac{h_{\text{src}}}{h_{\text{dst}}}.$$

824 Map the integer destination pixel to a source coordinate using the commonly used center-preserving
 825 formula
 826

$$827 \quad x_{\text{src}} = (x_{\text{dst}} + 0.5) s_x - 0.5, \quad y_{\text{src}} = (y_{\text{dst}} + 0.5) s_y - 0.5.$$

828 Let
 829

$$830 \quad x_0 = \lfloor x_{\text{src}} \rfloor, \quad y_0 = \lfloor y_{\text{src}} \rfloor.$$

831 **Lanczos Kernel.** Define the normalized sinc function
 832

$$833 \quad \text{sinc}(t) = \begin{cases} \frac{\sin(\pi t)}{\pi t}, & t \neq 0, \\ 1, & t = 0. \end{cases}$$

834 The Lanczos kernel with window parameter $a > 0$ is
 835

$$836 \quad L_a(t) = \begin{cases} \text{sinc}(t) \text{ sinc}(\frac{t}{a}), & |t| < a, \\ 0, & |t| \geq a. \end{cases}$$

837 **Determine the Contributing Source Samples.** The kernel is nonzero only for offsets satisfying
 838 $|t| < a$. Thus the integer source indices contributing in the horizontal direction are the set
 839

$$840 \quad \mathcal{I} = \{ i \in \mathbb{Z} : |x_{\text{src}} - i| < a \},$$

841 and in the vertical direction
 842

$$843 \quad \mathcal{J} = \{ j \in \mathbb{Z} : |y_{\text{src}} - j| < a \}.$$

844 Equivalently one can enumerate
 845

$$846 \quad i \in \{ x_0 - (a - 1), \dots, x_0 + a \}, \quad j \in \{ y_0 - (a - 1), \dots, y_0 + a \}.$$

847 **Compute Separable Weights.** For each contributing horizontal index i and vertical index j , com-
 848 pute the separable weights
 849

$$850 \quad w_i = L_a(x_{\text{src}} - i), \quad v_j = L_a(y_{\text{src}} - j).$$

851 The combined 2D weight for source sample (i, j) is the product
 852

$$853 \quad W_{ij} = w_i v_j.$$

854 **Compute the Output Pixel Value.** Let $I(i, j)$ denote the source image sample value (per channel).
 855 The Lanczos interpolated value for this single destination pixel is the normalized weighted sum over
 856 the contributing neighborhood:
 857

$$858 \quad O(x_{\text{dst}}, y_{\text{dst}}) = \frac{\sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} I(i, j) W_{ij}}{\sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} W_{ij}}.$$

864 **Interpolation Characteristics.** In LANCZOS interpolation, the mapping of pixel positions
 865 follows the same procedure as in BILINEAR interpolation. The key difference is that each mapped
 866 position is reconstructed using a weighted combination of a $2a \times 2a$ neighborhood of surrounding
 867 pixels. Consequently, LANCZOS interpolation exhibits the same interpolation characteristics as
 868 BILINEAR interpolation.

869 **A.2.4 HAMMING INTERPOLATION**

870 HAMMING interpolation is a windowed-sinc interpolation method in which the ideal sinc kernel
 871 is multiplied by a Hamming window. For a single destination pixel located at integer coordinates
 872 $(x_{\text{dst}}, y_{\text{dst}})$, the value is computed by sampling a finite neighborhood in the source image using this
 873 kernel.

874 **Coordinate Mapping.** Let the source image have size $(w_{\text{src}}, h_{\text{src}})$ and the destination size be
 875 $(w_{\text{dst}}, h_{\text{dst}})$. The scaling factors are

$$876 \quad s_x = \frac{w_{\text{src}}}{w_{\text{dst}}}, \quad s_y = \frac{h_{\text{src}}}{h_{\text{dst}}}.$$

877 Map the destination pixel center to a continuous source location:

$$878 \quad x_{\text{src}} = (x_{\text{dst}} + 0.5) s_x - 0.5, \quad y_{\text{src}} = (y_{\text{dst}} + 0.5) s_y - 0.5.$$

879 Define

$$880 \quad x_0 = \lfloor x_{\text{src}} \rfloor, \quad y_0 = \lfloor y_{\text{src}} \rfloor.$$

881 **Hamming Windowed Sinc Kernel.** Define the normalized sinc:

$$882 \quad \text{sinc}(t) = \begin{cases} \frac{\sin(\pi t)}{\pi t}, & t \neq 0, \\ 1, & t = 0. \end{cases}$$

883 The Hamming window is

$$884 \quad w(t) = 0.54 + 0.46 \cos\left(\frac{\pi t}{a}\right), \quad |t| < a,$$

885 where a is the window radius (typically $a = 2$).

886 The Hamming interpolation kernel is the product

$$887 \quad H_a(t) = \begin{cases} \text{sinc}(t) w(t), & |t| < a, \\ 0, & |t| \geq a. \end{cases}$$

888 **Contributing Source Samples.** The kernel has finite support $|t| < a$. Thus, contributing horizontal
 889 and vertical indices are

$$890 \quad \mathcal{I} = \{i \in \mathbb{Z} \mid |x_{\text{src}} - i| < a\}, \quad \mathcal{J} = \{j \in \mathbb{Z} \mid |y_{\text{src}} - j| < a\}.$$

891 Equivalently,

$$892 \quad i \in \{x_0 - (a - 1), \dots, x_0 + a\}, \quad j \in \{y_0 - (a - 1), \dots, y_0 + a\}.$$

893 **Separable Weights.** Compute the 1D horizontal and vertical weights:

$$894 \quad w_i = H_a(x_{\text{src}} - i), \quad v_j = H_a(y_{\text{src}} - j).$$

895 The 2D separable weight is

$$896 \quad W_{ij} = w_i v_j.$$

897 **Compute the Output Pixel Value.** Let $I(i, j)$ denote the source pixel values. The Hamming-
 898 interpolated destination pixel is

$$899 \quad O(x_{\text{dst}}, y_{\text{dst}}) = \frac{\sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} I(i, j) W_{ij}}{\sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} W_{ij}}.$$

918 Table 4: Accuracy and AUC comparison on the **GenImage** (unbiased) dataset with SDv1.4 as the
 919 training dataset. **Bold** and underline denote the best and second-best performance, respectively.
 920

Methods	SDv1.4	SDv1.5	Midjourney	ADM	GLIDE	Wukong	VQDM	BigGAN	Mean
CNNDetect(Wang et al., 2020)	50.1 / 65.1	49.9 / 66.4	50.1 / 79.3	49.9 / 51.8	50.7 / 59.4	50.2 / 62.6	51.2 / 63.4	58.4 / 70.9	51.3 / 64.8
DMID(Corvi et al., 2023b)	99.9 / 100.	<u>99.8 / 100.</u>	97.4 / 100.	51.3 / 78.5	56.6 / 94.9	99.6 / 100.	75.1 / 97.6	52.3 / 74.6	79.0 / 93.2
LGrad(Tan et al., 2023)	49.8 / 50.0	49.1 / 49.2	50.6 / 50.5	30.5 / 24.6	30.0 / 22.2	46.9 / 47.6	30.8 / 23.9	28.9 / 18.7	39.6 / 35.8
UnivFD(Ojha et al., 2023)	55.5 / 78.7	56.6 / 78.1	54.2 / 74.0	64.4 / 85.2	63.9 / 88.8	63.7 / 86.9	79.7 / 94.8	86.1 / 96.7	65.5 / 85.4
DeFake(Shi et al., 2023)	85.1 / 93.3	85.4 / 93.4	79.2 / 87.7	48.5 / 49.3	80.4 / 87.9	81.8 / 89.8	64.4 / 71.1	64.4 / 72.6	73.7 / 80.6
DIRE(Wang et al., 2023)	47.3 / 41.7	47.3 / 39.8	47.5 / 38.0	46.7 / 25.3	47.0 / 29.9	47.7 / 45.4	47.7 / 35.0	46.9 / 26.6	47.3 / 35.2
AntifakePrompt(Chang et al., 2023)	77.1 / -	76.6 / -	70.4 / -	81.6 / -	81.8 / -	77.6 / -	81.1 / -	81.7 / -	78.5 / -
NPR(Tan et al., 2024b)	49.4 / 54.3	49.7 / 53.3	47.4 / 42.3	50.5 / 46.9	48.3 / 42.1	50.2 / 52.4	53.9 / 52.3	56.3 / 56.9	50.7 / 50.1
FatFormer(Liu et al., 2024)	52.0 / 49.8	53.3 / 48.7	51.6 / 46.2	60.4 / 69.1	65.1 / 78.4	58.1 / 61.6	71.5 / 84.5	80.1 / 88.5	61.5 / 65.9
FasterThanLies(Lanzino et al., 2024)	92.2 / 97.8	92.3 / 97.9	69.7 / 83.1	77.2 / 88.6	66.1 / 83.0	88.1 / 95.4	76.6 / 86.3	54.1 / 78.9	77.0 / 88.9
RINE(Koutris & Papadopoulos, 2024)	60.5 / 93.9	61.1 / 94.1	52.4 / 86.3	63.9 / 93.8	74.7 / 98.1	70.0 / 95.7	81.4 / 98.4	88.5 / 99.4	69.1 / 95.0
AIDE(Yan et al., 2024)	74.5 / 98.2	75.9 / 98.5	57.4 / 88.1	50.1 / 61.2	52.3 / 80.4	69.3 / 95.9	51.0 / 78.0	50.7 / 73.1	60.2 / 84.2
LaDeDaCavia et al. (2024)	54.8 / 55.6	53.0 / 53.6	52.1 / 51.3	34.6 / 6.8	34.5 / 8.8	57.7 / 61.6	34.8 / 10.8	80.3 / 93.1	50.2 / 42.7
C2P-CLIP(Tan et al., 2025)	80.5 / 94.4	79.1 / 94.3	55.9 / 76.3	71.3 / 86.7	74.8 / 93.6	81.0 / 93.1	74.1 / 92.2	87.5 / 97.2	75.5 / 91.0
CoDE(Baraldi et al., 2024)	96.6 / 99.4	96.5 / 99.2	69.6 / 86.0	51.9 / 53.7	58.0 / 78.1	95.0 / 99.1	56.0 / 66.8	50.0 / 70.2	71.7 / 81.6
B-Free(Guillaro et al., 2025)	98.8 / 100.	<u>98.8 / 100.</u>	95.7 / 99.2	79.8 / 93.0	<u>85.3 / 95.8</u>	99.0 / 100.	<u>88.7 / 97.0</u>	68.7 / 94.1	<u>89.3 / 97.4</u>
Ours	<u>99.2 / 99.8</u>	<u>98.7 / 99.8</u>	<u>96.7 / 99.5</u>	98.7 / 99.7	98.9 / 99.8	<u>98.2 / 99.6</u>	98.5 / 99.7	98.7 / 99.8	<u>98.4 / 99.7</u>

932 Table 5: Average precision comparison on GANs from the **UniversalFakeDetect** dataset with Pro-
 933 GAN as the training dataset.
 934

Methods	ProGAN	StyleGAN	StyleGAN2	BigGAN	CycleGAN	StarGAN	GauGAN	Deepfake	Mean
CNNDetect(Wang et al., 2020)	99.2	91.4	96.7	73.3	88.2	90.7	92.2	62.3	86.7
FreDect(Frank et al., 2020)	85.2	72.2	71.4	86.5	71.7	99.5	77.4	49.2	76.6
LGrad(Tan et al., 2023)	100.0	99.9	99.9	90.5	94.7	100.0	79.2	67.8	91.6
UFID(Ojha et al., 2023)	100.0	98.8	98.6	99.1	99.6	100.0	99.2	90.2	98.2
PatchCraft(Zhong et al., 2023b)	100.0	98.7	97.7	99.3	85.1	100.0	81.8	79.6	92.7
FreqNet(Tan et al., 2024a)	100.0	99.6	95.5	95.5	99.7	100.0	98.6	94.5	97.9
NPR(Tan et al., 2024b)	100.0	100.0	100.0	94.5	95.7	100.0	88.2	86.1	95.6
FatFormer(Liu et al., 2024)	100.0	99.6	99.8	100.0	99.8	100.0	100.0	97.6	99.5
SAFE(Li et al., 2025)	100.0	99.8	100.0	95.4	99.8	100.0	97.0	97.5	98.7
CoD(Jia et al., 2025)	100.0	99.9	99.9	98.0	99.9	100.0	99.9	98.5	99.6
Ours	100.0	100.0	100.0	99.3	99.2	100.0	98.6	91.7	98.6

945 **Interpolation Characteristics.** In HAMMING interpolation, the mapping of pixel positions fol-
 946 lows the same procedure as in BILINEAR interpolation. The key difference is that each mapped
 947 position is reconstructed using a weighted combination of a $2(a-1) \times 2(a-1)$ neighborhood of
 948 surrounding pixels. Consequently, HAMMING interpolation exhibits the same interpolation char-
 949 acteristics as BILINEAR interpolation.
 950

951 A.3 ADDITIONAL GENERALIZATION ANALYSIS

953 **Evaluation on GenImage (unbiased) Dataset.** Table 4 presents the evaluation results on the Gen-
 954 Image (unbiased) dataset. Following the experimental settings in Guillaro et al. (2025) and Grom-
 955 melt et al. (2024), we test the classifier’s detection performance on JPEG-compressed generated
 956 images from the GenImage dataset. Our method significantly outperforms current state-of-the-art
 957 approaches, achieving an average accuracy improvement of 9.1% over the B-Free method. This
 958 demonstrates that our approach does not rely on JPEG compression artifacts as shortcuts during
 959 detection.

960 **Evaluation on UniversalFakeDetect Dataset.** Tables 5 and 6 present the evaluation results on the
 961 GAN and Diffusion models from the UniversalFakeDetect dataset, respectively. Following the ex-
 962 perimental settings in Ojha et al. (2023) and Jia et al. (2025), the model is trained on ProGAN-
 963 generated data and tested on various other models. The results demonstrate that our method achieves
 964 competitive performance across both GAN and Diffusion models, maintaining consistency with cur-
 965 rent state-of-the-art results.
 966

967 **Evaluation on Synthbuster Dataset.** Table 7 presents the evaluation results on the Synthbuster
 968 dataset. Following the experimental settings in Bammey (2023) and Karageorgiou et al. (2025),
 969 we test our method on various high-resolution, high-quality diffusion model samples. The exper-
 970 imental results demonstrate that our approach remains effective in detecting images generated by
 971 high-quality diffusion models. Moreover, the detection performance on original images without
 972 post-processing remains robust, as our method maintains the capability to capture the rescaling dis-

972 Table 6: Average precision comparison on Diffusions from the **UniversalFakeDetect** dataset with
 973 ProGAN as the training dataset.
 974

Methods	DALL-E	Glide_100_10	Glide_100_27	Glide_50_27	ADM	LDM_100	LDM_200	LDM_200_cfg	Mean
CNNDet(Wang et al., 2020)	61.2	72.9	71.3	76.1	66.6	63.7	64.5	63.1	67.5
FreDect(Frank et al., 2020)	62.5	44.3	40.8	42.3	52.5	51.3	50.9	52.4	49.6
LGrad(Tan et al., 2023)	97.3	94.9	93.2	95.0	99.8	99.2	99.1	99.2	97.3
UFD(Ojha et al., 2023)	96.5	96.5	97.0	97.2	84.5	97.0	97.0	88.6	94.3
PatchCraft(Zhong et al., 2023b)	93.0	92.0	93.9	88.7	90.5	97.7	97.9	96.9	93.8
FreqNet(Tan et al., 2024a)	99.5	96.1	96.6	95.0	74.5	99.6	99.0	99.0	94.9
NPR(Tan et al., 2024b)	99.5	99.8	99.7	99.8	81.0	99.0	99.9	99.9	97.4
FatFormer(Liu et al., 2024)	99.8	99.5	99.3	99.1	91.8	99.8	99.8	99.0	98.4
SAFER(Li et al., 2025)	99.7	99.4	98.9	99.2	95.7	100.0	100.0	99.8	99.0
CoD(Jia et al., 2025)	99.6	99.6	99.5	99.5	97.4	99.8	100.0	99.8	99.4
Ours	100.0	99.3	98.6	98.8	99.9	100.0	100.0	99.9	99.6

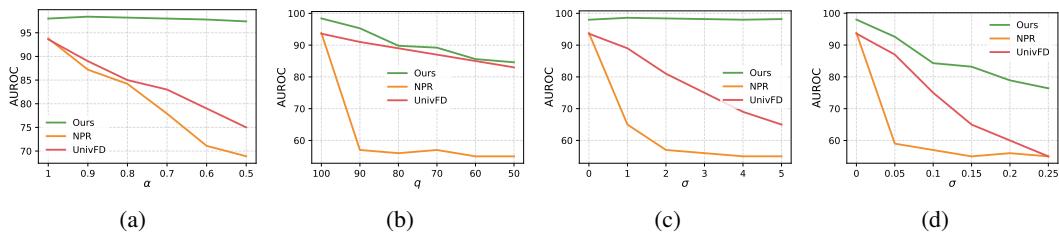
983 Table 7: AUC comparison on the **Synthbuster** dataset. **Bold** and underline denote the best and
 984 second-best performance, respectively.
 985

Methods	Glide	SD1.3	SD1.4	SD2	SDXL	MJv5	DALLE2	DALLE3	Firefly	Mean
NPR(Tan et al., 2024b)	72.2	89.6	60.5	12.5	18.1	15.3	3.9	97.1	38.0	45.2
Dire(Wang et al., 2023)	33.3	59.9	61.3	68.5	46.9	41.9	52.2	65.2	49.9	53.2
CNNDet(Wang et al., 2020)	59.2	59.0	61.2	57.5	67.4	48.8	71.5	23.5	73.4	57.9
FreqNet(Tan et al., 2024a)	43.6	92.3	92.7	42.5	66.5	36.9	47.4	42.2	80.9	60.6
Fusing(Ju et al., 2022)	63.0	62.8	62.2	66.9	62.1	64.0	76.7	25.2	76.3	62.1
LGrad(Tan et al., 2023)	76.5	82.4	83.4	60.7	70.2	69.2	85.7	30.0	42.0	66.7
UnivFD(Ojha et al., 2023)	63.3	80.8	81.2	84.3	78.3	57.1	91.4	31.0	<u>95.5</u>	73.7
GramNet(Liu et al., 2020)	78.2	83.9	84.3	66.7	77.8	63.8	85.2	42.9	38.0	69.0
DeFake(Sha et al., 2023)	86.1	64.2	63.6	66.2	52.3	67.0	41.4	93.3	39.4	63.7
PatchCr(Zhong et al., 2023a)	78.4	95.7	96.2	95.7	96.7	79.0	81.8	28.1	79.1	81.2
DMID(Corvi et al., 2023b)	73.1	100.0	100.0	<u>99.7</u>	<u>99.6</u>	<u>99.9</u>	54.3	41.3	90.2	84.2
RINE(Koultis & Papadopoulos, 2024)	95.6	99.9	<u>99.9</u>	96.6	99.3	96.4	<u>93.0</u>	41.8	82.9	89.5
SPAI(Karageorgiou et al., 2025)	90.2	99.6	99.6	96.5	97.4	94.5	91.1	90.2	96.0	<u>95.0</u>
Ours	92.1	99.7	99.7	100.0	100.0	100.0	97.1	93.0	91.5	97.0

999 tribution characteristics from the source data. Consequently, our approach surpasses current state-
 1000 of-the-art methods.

1001 **Robustness to Perturbations.** In addition to evaluating on clean images, we also evaluate the
 1002 classifier’s detection capability under various image degradation scenarios. In real-world applica-
 1003 tions, images may undergo multiple perturbations during propagation, making robust detection
 1004 under degraded conditions crucial for practical deployment. Following prior works Wang et al.
 1005 (2020) and Ojha et al. (2023), we test four types of perturbations: Rescaling (with scaling factor
 1006 α), JPEG compression (with quality parameter q), Gaussian blur (with standard deviation σ), and
 1007 Gaussian noise (with standard deviation σ). As shown in Figure 7, our method maintains strong de-
 1008 tection performance across different perturbation scenarios. Although our approach relies on rescal-
 1009 ing distributions, its fine-grained pretraining enables the model to remain unaffected by rescaling
 1010 post-processing while still effectively discerning differences in image rescaling distributions.

1012 **Consistent Performance under Patch Size Variations.** Table 8 presents the generalization re-
 1013 sults of the proposed method under different patch sizes, with experimental settings consistent with
 1014 Table 2. The model was pre-trained using a patch size of 128, while fine-tuning and testing were



1024 Figure 7: Robustness to perturbations: (a) Rescaling; (b) JPEG compression; (c) Gaussian blur; (d)
 1025 Gaussian noise.

Table 8: Accuracy comparison with varying patch sizes on the **GenImage** dataset.

Patch-size	SDv1.4	SDv1.5	Midjourney	ADM	GLIDE	Wukong	VQDM	BigGAN	Mean
128	99.9	99.9	92.1	94.2	98.8	99.7	99.7	99.9	98.0
64	99.9	99.8	92.1	89.7	99.6	99.6	99.1	99.3	97.4
32	99.6	99.7	91.3	87.8	97.7	99.2	98.0	99.2	96.6
16	95.6	95.8	88.1	84.4	88.8	93.8	89.8	92.4	91.1
8	89.7	91.3	83.6	73.9	83.2	86.7	78.7	79.8	83.4

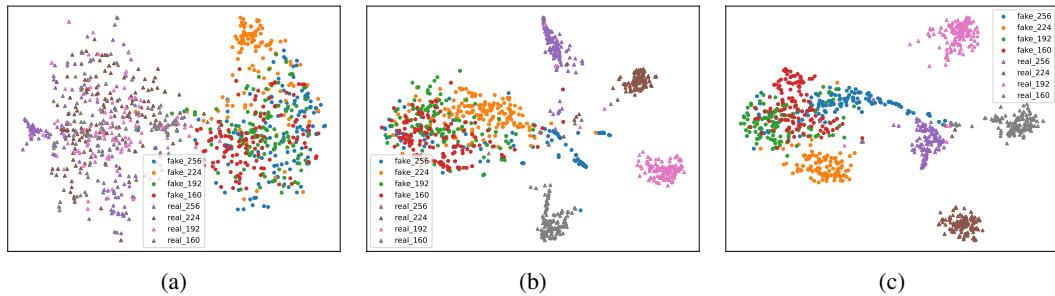
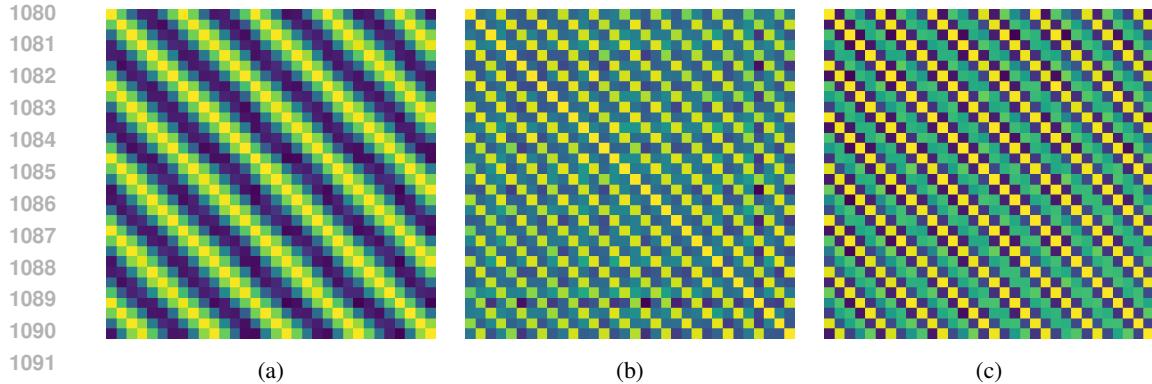


Figure 8: Feature classification performance between generated and real images under different rescaling post-processings: (a) ProGAN; (b) ADM; (c) SDv1.4.

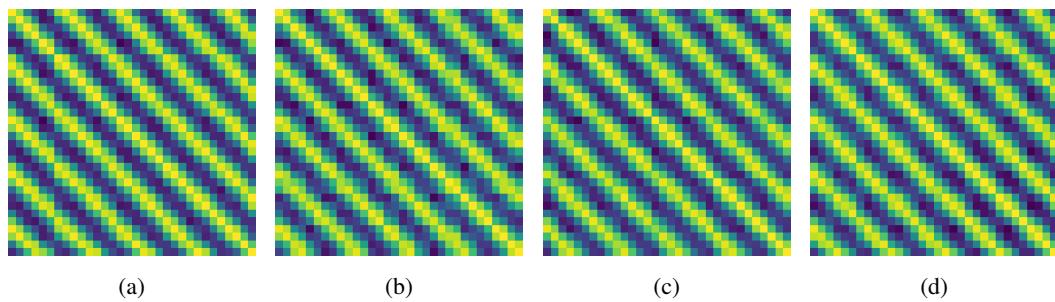
conducted with smaller patch sizes. As shown in the results, the detection performance remains relatively stable when the patch size is reduced to 64 or 32. However, further reduction in patch size leads to a rapid decline in detection effectiveness. This is primarily because smaller patches contain limited rescaling distribution information, making it difficult to extract features that characterize the approximated distribution of generated images. Overall, the pre-trained model demonstrates robustness to variations in patch size, maintaining remarkable detection performance across different configurations.

t-SNE Visualizations under Mixed Post-rescaling. Figure 8 presents a unified t-SNE visualization incorporating all post-rescaling images from Figure 6. The three subfigures display generated images and their real counterparts from ProGAN, ADM, and SDv1.4 models, respectively. Each image group comprises the original images along with three variants processed with different rescaling factors. The visualization reveals that real images maintain coherent clustering across various rescaling factors while exhibiting clear separation from rescaled generated images. Furthermore, generated images form distinct clusters corresponding to their respective rescaling factors. Although different rescaling factors differentially affect the approximated rescaling distribution of synthetic images, they fail to obscure the inherent distributional discrepancies. Consequently, the pre-trained model effectively discriminates between post-rescaling generated images and real images. The robustness experiment in Figure 7(a) further confirms our method’s resilience to rescaling-based post-processing perturbations.

Visualization of Periodicity in Rescaled Images. Rescaled images exhibit distinct periodic distribution patterns, which can be observed through cosine similarity maps derived from features extracted by the pre-trained model. As shown in Figure 9, we present cosine similarity maps of three randomly selected real images after bilinear rescaling with different factors. For clearer visualization, we magnify the top-left 32×32 region to highlight these characteristics. Consistent with our theoretical derivation in the preliminary section, the minimal distribution period of rescaling corresponds to the denominator of the rescaling factor’s reduced fractional form. In subfigure (a), where 256/224 simplifies to 8/7, the minimal period is 7, and bright stripes indeed appear at 7-pixel intervals. Similarly, subfigures (b) and (c) demonstrate periods of 3 and 5, respectively. These observations validate our theoretical analysis regarding the periodic distribution properties of bilinear interpolation and confirm the pre-trained model’s capability to effectively capture such periodic features.



1093 Figure 9: Cosine similarity maps of single images under different rescaling ratios, exhibiting distinct
1094 periodicity. (a) Rescaling from 256×256 to 224×224 ; (b) Rescaling from 256×256 to 192×192 ; (c)
1095 Rescaling from 256×256 to 160×160 .



1107 Figure 10: Consistent periodicity across interpolation methods, compatible with BILINEAR-
1108 pretrained feature extraction. Rescaling 256×256 to 224×224 via: (a) BOX; (b) BICUBIC;
1109 (c) LANCZOS; (d) HAMMING interpolation.

1110
1111 **Consistent Periodic Distribution Across Interpolation Methods.** We employed various inter-
1112 polation methods to process images and computed their corresponding cosine similarity maps. As
1113 shown in Figure 10, we present results from several PyTorch-provided interpolation approaches:
1114 BOX, BICUBIC, LANCZOS, and HAMMING. Although the pre-trained model was exclusively
1115 trained on bilinear interpolation distributions, it successfully captures periodic features from other
1116 interpolation techniques. This observation aligns with our theoretical analysis that different inter-
1117 polation methods share similar periodic distribution characteristics. Furthermore, these results validate
1118 our model’s generalization capability across diverse interpolation schemes.

1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133