

# Detecting Generated Images by Real Images Only

Xiuli Bi, Bo Liu, *Member, IEEE*, Fan Yang, Bin Xiao, Weisheng Li, *Member, IEEE*, Gao Huang, *Member, IEEE*, Pamela C. Cosman, *Fellow, IEEE*

**Abstract**—As deep learning technology continues to evolve, the images yielded by generative models are becoming more and more realistic, triggering people to question the authenticity of images. Existing generated image detection methods detect visual artifacts in generated images or learn discriminative features from both real and generated images by massive training. This learning paradigm will result in efficiency and generalization issues, making detection methods always lag behind generation methods. This paper approaches the generated image detection problem from a new perspective: Start from real images. By finding the commonality of real images and mapping them to a dense subspace in feature space, the goal is that generated images, regardless of their generative model, are then projected outside the subspace. As a result, images from different generative models can be detected, solving some long-existing problems in the field. Experimental results show that although our method was trained only by real images and uses 99.9% less training data than other deep learning-based methods, it can compete with state-of-the-art methods and shows excellent performance in detecting emerging generative models with high inference efficiency. Moreover, the proposed method shows robustness against various post-processing. These advantages allow the method to be used in real-world scenarios.

**Index Terms**—Generative model, image noise, generated image detection, one-class classification.

arXiv:2311.00962v1 [cs.CV] 2 Nov 2023

## 1 INTRODUCTION

CURRENTLY, the popularity of deep neural networks has driven the rapid development of digital forgery technology, making it easy to abuse AI synthesis algorithms. Various eye-popping technologies have entered our lives, from image content manipulation to scene synthesis, from face attribute tampering to face swapping. In Fig. 1, can you discern which images are captured by cameras and which are generated by neural networks? In fact, all images in Fig. 1 are generated by generative models, such as Generative Adversarial Networks (GAN), flow models, and diffusion models. These generated fake images can be used as fun plugins for applications such as face makeup [1] and attribute editing [2], but also to spread falsehoods. For example, unscrupulous people send LinkedIn job postings by impersonating real people with synthetic faces to conduct fraudulent activities. Since many images produced by generative models can reach the point of deceiving human

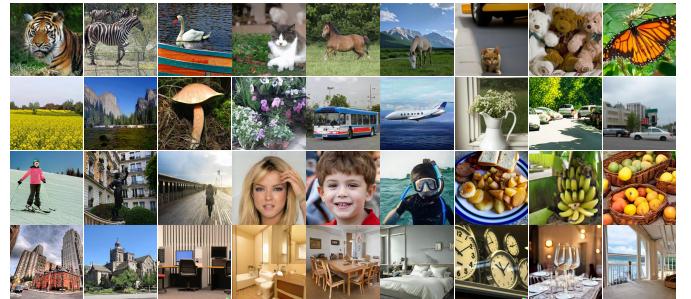


Fig. 1. Which are generated images and which are real images captured by cameras?

eyes, detection methods that can distinguish real and generated images are strongly needed. In this paper, we use the term “real images” to mean photographs of the real world as captured by digital cameras based on visible light. This definition excludes images of the real world from other sorts of detectors, such as CT and MR scans.

Existing detection methods [3–13] are designed for specific types of GAN-generated images. No matter what strategy these methods take for training, they try to find the decision boundaries between real and specific GAN-generated images, as shown in Fig. 2(a). In the feature space constructed by training balanced real and specific GAN-generated images, the classifier can accurately classify them; these detection methods are very effective in the types of GAN-generated images in their training set. When they encounter images generated by GAN models not seen in the training set, using images generated by ProGAN [14] for training can avoid a significant performance drop.

However, as shown in Fig. 2(b), the constructed feature

- Xiuli Bi, Bo Liu, Fan Yang, Bin Xiao and Weisheng Li are with the Department of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, China. email: bixl@cqupt.edu.cn, boliu@cqupt.edu.cn, cquptfan@gmail.com, xiabin@cqupt.edu.cn, liws@cqupt.edu.cn.
- Gao Huang is with the Department of Automation, Tsinghua University, Beijing 100084, China. email: gao.huang@tsinghua.edu.cn
- Pamela C. Cosman is with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093 USA. e-mail: pcosman@ucsd.edu.
- This work was supported in part by the National Key Research and Development Project under Grant 2019YFE0110800, in part by the National Natural Science Foundation of China under Grant 62172067 and Grant 61976031, the Natural Science Foundation of Chongqing for Distinguished Young Scholars under Grant CSTB2022NSCQ-JQX0001. Corresponding author: Bin Xiao.

Manuscript received April 19, 2005; revised August 26, 2015.

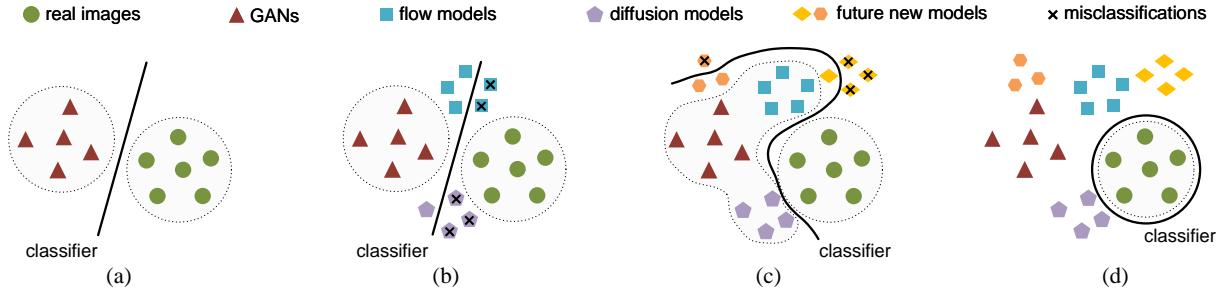


Fig. 2. Our method adopts a new perspective based on real images to detect generated images. The dashed circles indicate the samples in the training set. (a) The feature space determined by real and GAN-generated images; (b) The space does not fit for all generative models, such as flow and diffusion model-generated images; (c) The feature space becomes complicated to include more generative models: the classifier will be more complex and fail to handle unseen new models, and the training costs are high; (d) Our method only uses real images to construct a dense subspace and performs one-class classification to detect existing and possible future new generative models.

space and classifier, which are based on GAN-generated images, will not fit other model-generated images well, such as flow model- and diffusion model-generated images. Experimental results validated this view [3–13]: the classes of the generated images in the training set determine the detection performance. To enhance the versatility and detection performance of the classifier, one might think that adding more types of generated images to construct a proper feature space by re-training or fine-tuning will fit existing popular generative models. However, as shown in Fig. 2(c), the space becomes much more complicated, and such a classifier is required to fit the increasingly complicated feature space, resulting in a more difficult and time-consuming training process. By adopting this strategy, the classifier is chasing future new generative models, reducing the practical utilization value in realistic scenarios; this strategy also triggers more problems which will be discussed in Subsection 4.3.

These problems are caused by considering generated image detection from the perspective of generated images. In this paper, we analyze the characteristics of real and generated images, and find that the noise of real images is very similar while that of generated images is different. Based on this, we model the generated image detection problem as a one-class classification problem. By finding the commonality of real images and mapping them to a dense subspace in this noise-feature space, the goal is that generated images, regardless of their generative models, should ideally be mapped outside the subspace and consequently detected as shown in Fig. 2(d).

The contributions of the paper can be summarized as follows:

- We are the first to propose a new perspective on generated image detection: Start from real images. In this way, long-existing problems such as poor generalization, and high training and inference costs, can be handled.
- We have analyzed an appropriate way to map real images to a dense subspace theoretically and experimentally so that our method achieves good performance while using 99.9% less training data. It also has good robustness against various post-processing operations. Even for emerging generative models,

the detection performance is very good. Thus it can be used in realistic scenarios.

- This paper improves our previous work *Detecting Generated Images by Real Images* published in ECCV'22, which still needs real and generated images for massive training. This paper learns from real images only while keeping superior detection performance and achieves a much better inference efficiency.

In the remainder of the paper, we first analyze the work related to generated image detection in Section 2. Then in Section 3 we model generated image detection as a one-class classification problem and introduce a proper way to map real images to a subspace in feature space based on the commonality of real images. We present and analyze the experimental results in Section 4, and draw conclusions in Section 5.

## 2 RELATED WORK

### 2.1 Existing Generated Image Detection Methods

Existing generated image detection methods mainly rely on deep learning. They can be divided into image-artifact-detection-based and massive-training-based approaches. The artifact detection approach probes the traces left by the generative networks. Using these traces in the spatial or frequency domain, a trained classifier can discern the generated images. In contrast, by deploying massive training, the second group of approaches takes advantage of the strong learning ability of deep neural networks to search for the decision boundaries between real and generated images.

#### 2.1.1 Using Artifacts for Detection

For methods focusing on image artifacts, Yu et al. [12] argue that each GAN model has a unique fingerprint due to the influence of training data, network structure, loss function, parameter settings, and other factors. Liu et al. [5] notice a difference in texture between real and generated faces and used Gram matrices in the network to capture important and long-distance textural information. Dang et al. [6] recognize the importance of the spatial information of the tampered region and use the attention mechanism

to highlight informative regions to improve binary classification results. Zhao et al. [7] consider the generated image detection task as a fine-grained problem and improve detection performance by using the attention mechanism to amplify subtle differences in the shallow layers. It is argued in [8] that extracting forensic features from local areas is more effective than a global approach.

Zhang et al. [3] introduce a generator that can simulate sampling artifacts on several commonly seen GANs. This method shows that studying the artifacts in the frequency domain is an effective way to expose generated images. Durall et al. [9] find the distribution of high-frequency components of real images hard to simulate by generative models. Therefore generated images can be classified according to the characteristics of their high-frequency components. Dzanic et al. [13] argue that generated images do not decay at the highest frequencies while real images do, and images could be detected based on the degree of partial decay at high frequencies. Frank et al. [4] demonstrate that the artifacts in GAN-generated images are caused by upsampling operations, using DCT transforms to detect them.

The above methods analyze generative models and extract discriminative artifact features for detection. However, as generative models evolve and the quality of generated images improves, such as through the emergence of diffusion models, extracting discriminative artifacts from generated images becomes more difficult. Also, these spatial and frequency domain methods do not generalize well to unseen generative models, resulting in an unsatisfactory performance for cross-dataset experiments.

### 2.1.2 Deploying Massive Training for Detection

The massive training-based methods overcome some shortcomings of the artifact detection methods. By deploying massive training from generated and real images, discriminative features of generated images are automatically identified. In Wang's method [10], 720k real images were used with the generated images of ProGAN [14] to train ResNet50 [15], which can be generalized to detect some generative models. Based on this, Gragnaniello et al. [11] used a modified ResNet50 network by reducing two down-sampling layers to improve the detection performance. However, the training cost surges. Due to the use of neural networks to automatically find differences between real and generated images, such methods are easily influenced by the generated images in the training set. Thus, it does not work to detect other generative models such as flow-based models [16–20] and diffusion models [21–25].

## 2.2 One-class Classification Problems

Typical classifiers like SVM are two- or multi-class classifiers which generalize to new data by maximizing a margin between existing training data from different classes [26] in some feature space. One-class classifiers decide a boundary using one class of data. Any data outside the boundary will be classified as a different class. There are two implementations to construct the decision boundary. The first, proposed by Schölkopf et al. [27], maximizes the distance between a hyperplane decided by support vectors and the coordinate origin. The second, by Tax and Duin, restricts

training data to a hypersphere [28]. One-class classification problems commonly exist in novelty and outlier detection [29–32]. Successful application of one-class classifiers depends on whether the feature space is well constructed. An improper feature space contains large voids between the class instances, resulting in model failure or producing many false classifications.

## 3 USING REAL IMAGES TO DETECT GENERATED IMAGES

From Fig. 2, performing generated image detection from the perspective of generated images leads to many problems. In our definition, real images are produced by cameras, as a physical imaging process. In contrast, generated images are generated by deep learning networks, as a computational process. We note that the division into physical versus computational imaging processes is not a clean separation. On the one hand, camera images from physical imaging undergo various computational steps in the camera pipeline such as white balancing, demosaicking from a color filter array, or JPEG compression, and on the other hand, conditional generative models can take a real image as input to a neural network. Therefore, we consider this task from real images by starting with their properties. As shown in Fig. 2(d), from the perspective of real images, the classifier can distinguish various types of generated images. That is, it becomes a one-class classification problem.

### 3.1 Modeling the One-class Classification Problem

We consider a sequence  $\mathcal{T}$  consisting of some real images  $t_1, t_2, \dots, t_i, \dots, t_\ell$ , where  $\ell \in \mathbb{N}$  is the number of images. Suppose a mapping  $\Phi : \mathcal{T} \rightarrow \mathcal{S}$ , where the dot product in the feature of  $\Phi$  can be computed by a kernel  $K$ :

$$\Phi(t_i) \cdot \Phi(t_j) = K(t_i, t_j) \quad (1)$$

where  $K$  can be a simple kernel such as a Radial Basis Function (RBF) kernel:

$$K(t_i, t_j) = e^{-\|t_i - t_j\|^2 / 2\sigma^2}. \quad (2)$$

To create a dense subspace  $\mathcal{S}$  that contains real images only, we can limit these real images within a hypersphere. Or, more conveniently, we can separate the real images using a hyperplane parameterized by vector normal  $w$  and offset  $\rho$  from the origin by optimizing

$$\begin{aligned} & \min_{w, \eta, \rho} \frac{1}{2} \|w\|^2 + \frac{1}{v\ell} \sum_i \eta_i - \rho, \\ & \text{s.t. } w \cdot \Phi(t_i) \geq \rho - \eta_i, \eta_i \geq 0, \end{aligned} \quad (3)$$

to maximize the distance  $\theta$  between the origin and this hyperplane. Then  $\mathcal{S}$  will be the space which satisfies  $w \cdot \Phi(I) \geq \rho$ . In Eq. 3, the column vector  $\eta = [\eta_1, \eta_2, \dots, \eta_\ell]$  and each  $\eta_i$  is the slack variable corresponding to the  $i$ -th real image;  $v \in (0, 1)$  is a trade-off parameter between optimization terms. Finally, for an image  $I \notin \mathcal{T}$ , the decision function:

$$\varphi(I) = \text{sgn}[w \cdot \Phi(I) - \rho] \quad (4)$$

will decide whether the image  $I$  is in  $\mathcal{S}$  or not. For real images  $t_i$  in the sequence  $\mathcal{T}$ , the above sign function will be

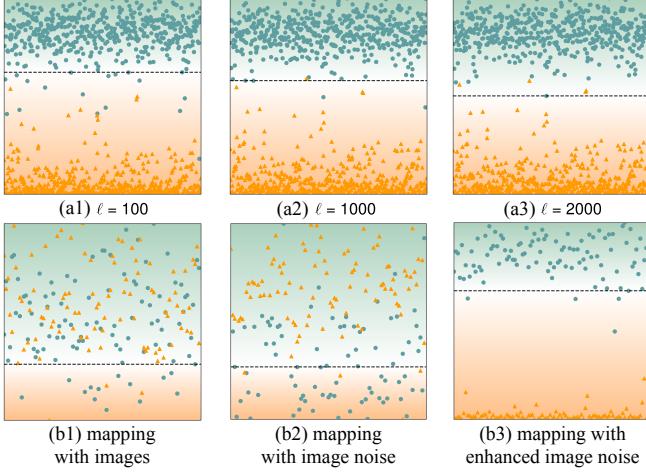


Fig. 3. Different values of  $\ell$  and mapping functions  $f \in \mathcal{F}$  affect the decision boundary and the classifier performance. Green circles and orange triangles represent real and generated images, respectively.

positive. Using the kernel trick in Eq. 1, the sign function in Eq. 4 can also be denoted in the form of support vector expansion:

$$\varphi(I) = \text{sgn} \left[ \sum_i \alpha_i K(I, t_i) - \rho \right]. \quad (5)$$

To determine whether real images are mapped to  $\mathcal{S}$  via  $\Phi$ , we analyze the probability  $P$  that a real image  $t_i \in \mathcal{T}$  falls outside  $\mathcal{S}$  after mapping. For any  $\theta \in \mathbb{R}$ , according to Theorem 7 in [33] (proved by Lemma 3.9 in [34]), all images  $t_1, t_2, \dots, t_i, \dots, t_\ell \in \mathcal{T}$  are mapped to  $\mathcal{S}$  with probability  $1 - \delta$  if  $f(t_i) \geq \theta + \gamma$  holds for any  $t_i$  (given that there is a class  $\mathcal{F}$  of functions which map  $\mathcal{T}$  to  $\mathbb{R}$ , and  $f \in \mathcal{F}$ ). The mapping methods are not limited to the one we introduced, and we use  $f$  as some function in the set. The probability  $P$  that the mapping of the real image  $t_i$  falls outside  $\mathcal{S}$  is

$$P\{f(t_i) < \theta - \gamma\} \leq \frac{2}{\ell} \left[ (\log \mathcal{N}(\gamma, \mathcal{F}, 2\ell)) + \log \frac{2\ell}{\delta} \right], \quad (6)$$

where  $\mathcal{N}(\gamma, \mathcal{F}, 2\ell) = \sup_{\mathcal{T}^{2\ell}} \mathcal{N}_d(\gamma, \mathcal{F})$ , and  $\mathcal{N}_d(\gamma, \mathcal{F})$  is the  $\gamma$ -covering number of function set  $\mathcal{F}$  with the distance  $d$  defined by

$$d(f, g) = \max_{i \in [2\ell]} |f(t_i) - g(t_i)|, \quad (7)$$

where  $f, g \in \mathcal{F}$ . From Eq. 6 we can see that if the real images can be mapped to  $\mathcal{S}$ ,  $P\{f(t_i) < \theta - \gamma\}$  should be small. The number of real images  $\ell$  in the sequence  $\mathcal{T}$  and the choice of mapping function  $f \in \mathcal{F}$ , or more specifically  $\Phi$  will affect the value of  $P$ .

We can also analyze the probability that a real image  $I \notin \mathcal{T}$  is falsely mapped outside the constructed space  $\mathcal{S}$ . We fix  $\theta \in \mathbb{R}$  and suppose that the range of the function set  $\mathcal{F}$  is  $[a, b]$ . According to Theorem 9 in [33], with probability  $1 - \delta$  over the sequence  $\mathcal{T}$ , for all  $\gamma > 0$  and any  $f \in \mathcal{F}$ ,

$$P\{f(I) < \theta - \gamma \text{ and } I \notin \mathcal{T}\} \leq \frac{2}{\ell} \left( \Omega + \log \frac{4\ell}{\delta} \right), \quad (8)$$

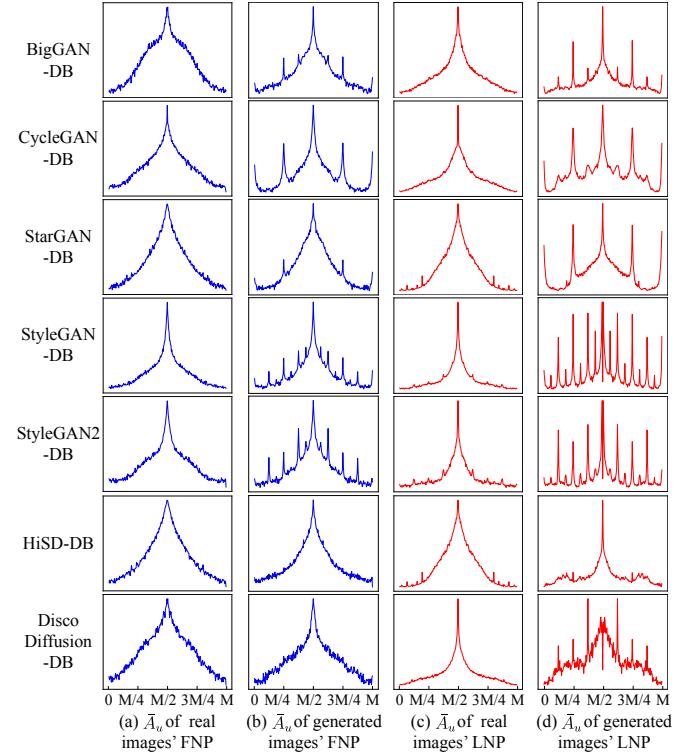


Fig. 4. The noise pattern (amplitude spectrum) of real images and generated images. The blue lines show the averaged FNP amplitude spectra  $\bar{A}_u$ . The red lines present the averaged LNP amplitude spectra (before feature enhancement).

where

$$\Omega = \left[ \log \mathcal{N} \left( \frac{\gamma}{2}, \mathcal{F}, 2\ell \right) \frac{64(b-a)\mathcal{D}(\mathcal{T}, f, \gamma)}{\gamma^2} \right. \\ \left. \cdot \log \frac{e\ell\gamma}{8\mathcal{D}(\mathcal{T}, f, \gamma)} \cdot \log \frac{32\ell(b-a)^2}{\gamma^2} \right], \quad (9)$$

and  $\mathcal{D}$  is defined as the sum of the distances from all real images  $t_i \in \mathcal{T}$  falsely classified as generated images to the decision boundary:

$$\mathcal{D}(\mathcal{T}, f, \gamma) = \sum_{t_i \in \mathcal{T}} \max\{0, \theta + \gamma - f(t_i)\}. \quad (10)$$

Eq. 8 shows that the probability  $P\{f(I) < \theta - \gamma \text{ and } I \notin \mathcal{T}\}$  is bounded by the term of the ratio of the logarithmic covering numbers which is at scale proportional to  $\gamma$ , the size  $\ell$  of the real image sequence  $\mathcal{T}$ .

As shown in Fig. 3(a1)~(a3), different values of  $\ell$  (the number of real images in  $\mathcal{T}$ ) will affect the decision boundary and consequently the number of false classifications. Also, the choice of mapping function  $f$  decides the term  $\mathcal{N}(\gamma, \mathcal{F}, 2\ell)$ . Fig. 3(b1)~(b3) show the classification results for using different  $f$  while  $\ell$  is fixed at 500. Therefore, for making the  $\gamma$ -covering  $\mathcal{N}_d$  take a smaller value,  $f$  should reflect the commonality of the real images so that it can map  $\mathcal{T}$  to a dense subspace  $\mathcal{S}$ .

### 3.2 The Commonality of Real Images

Previous research [4, 9] found that the upsampling operation in conditional generative models changes the noise

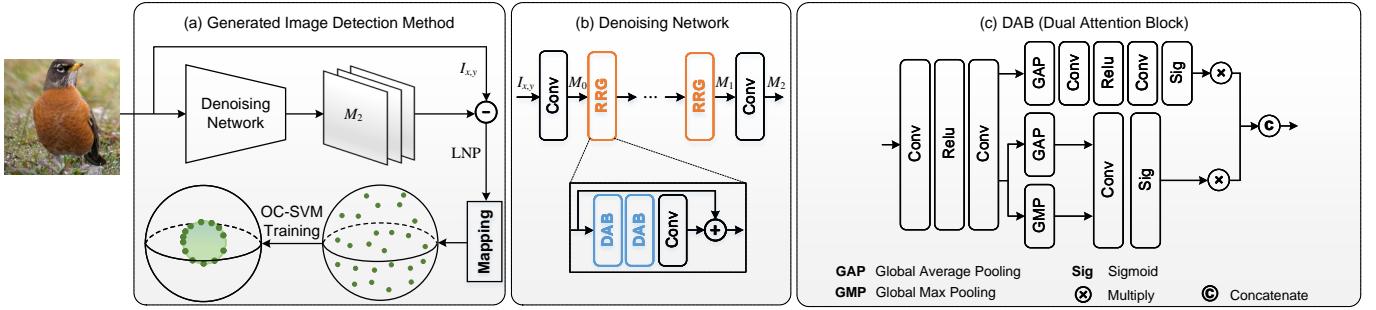


Fig. 5. (a) Overall structure of the generated image detection method; (b) Structure of the denoising network for LNP extraction; (c) Structure of the Dual Attention Block (DAB), which is a component of the denoising network.

characteristics of input real images. Moreover, noise transformation after initial sampling is also in the image generation process of unconditional models. Therefore, we will explore the commonality of real images from the noise characteristics.

To extract image noise  $n$  from the image  $I$ , applying a denoising filter  $\Xi(\cdot)$  is the most common practice:

$$n(x, y) = I(x, y) - \Xi(I(x, y)), \quad (11)$$

where  $(x, y)$  denotes coordinates. We adopted the denoising filters proposed in [35]. These filters keep image edges very effectively in practice, and obtain the noise relatively unaffected by image semantics. To better analyze the characteristics of the Filtered Noise Pattern (FNP)  $n(x, y)$ , we transform it by the Discrete Fourier Transform (DFT):

$$\zeta(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} n(x, y) e^{-i2\pi ux/M} e^{-i2\pi vy/N}, \quad (12)$$

where  $M \times N$  represents the image size. We then calculate the amplitude spectrum  $A(u, v)$  of  $\zeta(u, v)$ :

$$A(u, v) = \sqrt{Rp_{u,v}^2 + Ip_{u,v}^2}, \quad (13)$$

where  $Rp_{u,v}$  and  $Ip_{u,v}$  denote the real and imaginary parts of  $\zeta(u, v)$ , respectively.

Since the Fourier frequency spectrum is symmetric, we can average the amplitude spectra  $A(u, v)$  in either one dimension to show its characteristics. We took  $A_u$  as an instance and calculated it by

$$A_u = \sum_{v=0}^{N-1} A(u, v). \quad (14)$$

We then averaged  $A_u$  across all images in each image subset (the image subsets are detailed in the experiment section) and denoted it as  $\bar{A}_u$ . It can be seen in Fig. 4(b) that the  $\bar{A}_u$  of generated images' FNP show different distributions and periodic patterns in some subsets. However, the  $\bar{A}_u$  for the real images' FNP do not show periodic patterns and have similar distributions, as shown in Fig. 4(a).

To confirm this commonality of real images, considering the strong learning ability of deep neural networks, we used a denoising network to extract image noise. For convenience, the equations only show the processing of one

color channel, but we process all three channels of each color image similarly. The calculation can be formulated as

$$L(x, y) = I(x, y) - \Psi(I(x, y), \theta_\Psi), \quad (15)$$

where  $\Psi(\cdot)$  is a denoising network,  $\theta_\Psi$  represents the set of all parameters of  $\Psi(\cdot)$ , and  $\Psi(I(x, y))$  denotes the clean output image.  $L(x, y)$  is called the Learned Noise Pattern (LNP).

As shown in Fig. 4(c)(d), compared to FNP, LNP can better reflect the commonality of real images and their discrepancy from generated images. The  $\bar{A}_u$  of generated images' LNP show periodic peaks whereas the  $\bar{A}_u$  of real images' LNP do not. Moreover, the  $\bar{A}_u$  of generated images' LNP from different models are distinct and show various characteristics. This explains why learning from one particular type of generated image causes a generalization problem. In contrast, the  $\bar{A}_u$  of real images' LNP are much more similar. Therefore, mapping real images to the dense subspace constructed by noise features  $f : \mathcal{T} \rightarrow \mathcal{S}$  is feasible and better reflects the commonality of real images. Consequently, the probability that a real image  $t_i \in \mathcal{T}$  falls outside  $\mathcal{S} : P\{f(t_i) < \theta - \gamma\}$  can be made small.

### 3.3 Mapping Real Images to a Dense Subspace

The general form of the LNP is described in Eq. 15, and there are many choices for the denoising network  $\Psi(\cdot)$ . Early denoising networks, such as DNCNN [36], directly add Gaussian white noise (AWGN) to clean images to create noisy images and then use the clean/noisy pairs to train  $\Psi(\cdot)$ . However, in real scenarios, the noise may be non-Gaussian. To extract the noise in the real world, CycleISP [37] deploys networks to simulate the actual imaging pipeline of cameras by reconstructing RAW images from RGB images, adding noise to the RAW images, and then converting the RAW images to RGB images to get clean/noisy image pairs.

Inspired by the CycleISP [37] network, the LNP extraction in our proposed generated image detection method is shown in Fig. 5(b). Specifically, an image  $I$  of size  $M \times N$  will be processed as

$$M_0 = K_3(I(x, y)), \quad (16)$$

where  $K_3$  denotes a  $3 \times 3$  convolution. Many low-level applications including denoising [36] benefit from a residual learning framework [15], therefore we used the Recursive Residual Group (RRG) module to process the features. The

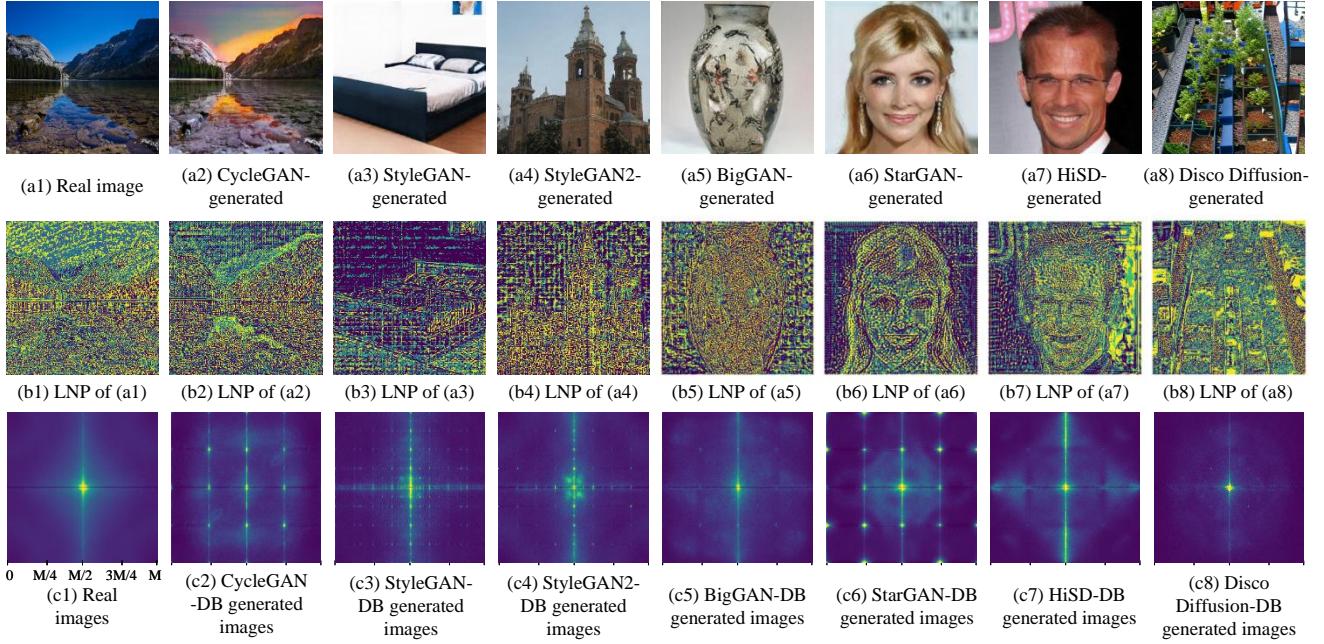


Fig. 6. The discrepancy between real and generated images in the extracted LNP. (a) Real and generated images in datasets; (b) Visualization of LNPs of images in row a; (c) Average amplitude spectrum of real images from different real image subsets and that of generated images from different GAN models.

RRG module is composed of two Dual Attention Blocks (DAB). Each DAB achieves calibration of the features by two types of channel attention and spatial attention, as shown in Fig. 5(c). Its dual attention structure suppresses less useful features and keeps informative ones, which is ideal for denoising. We used four consecutive RRG modules:

$$M_1 = RRG(RRG(RRG(RRG(M_0))). \quad (17)$$

Finally, by a convolution operation  $M_2 = K_3(M_1)$ , the three-channel feature map  $M_2$  can be obtained, and then the LNP  $L(x, y)$  in Eq. 15 is calculated by

$$L(x, y) = I(x, y) - M_2 \quad (18)$$

in our generated image detection method.

Some LNP examples obtained by Eq. 18 are shown in Fig. 6(b). We can see that the LNP in smooth regions of images from generative models exhibits grid artifacts, while the LNP of real images does not show that. Since a grid effect in the spatial domain will show periodicity in the frequency domain, we also transformed it by the DFT:

$$\zeta(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} L(x, y) e^{-i2\pi ux/M} e^{-i2\pi vy/N}, \quad (19)$$

and calculated amplitude spectra  $A(u, v)$  by Eq. 13.

Similar to  $\bar{A}_u$ , we also averaged  $A(u, v)$  of all images in each image subset and denote it as  $\bar{A}(u, v)$ . Fig. 6(c) shows  $\bar{A}(u, v)$  for each generated subset and real images from all real subsets. For the BigGAN [38] and HiSD [39] models using nearest-neighbor interpolation for upsampling, the period in the horizontal direction of  $\bar{A}(u, v)$  is  $M/4$ , and the period in the vertical direction of  $\bar{A}(u, v)$  is  $N/4$ . For StyleGAN [40] and StyleGAN2 [41] using bilinear interpolation for upsampling, the period in the horizontal direction of  $\bar{A}(u, v)$  is  $M/8$ , and the period in the vertical direction

of  $\bar{A}(u, v)$  is  $N/8$ . For CycleGAN [42] and StarGAN [43], upsampling is performed using deconvolution, and the  $\bar{A}(u, v)$  have strong vibrations with a period of  $M/4$  in the horizontal direction of  $\bar{A}(u, v)$  and a period of  $N/4$  in the vertical direction.  $\bar{A}(u, v)$  of all generated images' LNP show significant periodicity. In contrast, the  $\bar{A}(u, v)$  of real images' LNP does not present periodicity. Therefore, we can calculate the amplitude spectrum  $A(u, v)$  of LNP and then extract features from it to characterize the periodic signal.

To make the amplitude spectrum  $A(u, v)$  of LNP more discriminative, we performed spectrum enhancement by increasing the gap between the high and low energy of the spectrum. Since the discriminative high peaks in Fig. 4 are obtained by spectrum row averaging (Eq. 14), amplitudes less than the average row value are not bright spots, and we set their value to 0. Then the differences of remaining non-zero amplitudes are enlarged by squaring. The enhancement is formulated as

$$A'(u, v) = \begin{cases} 0 & A(u, v) < A_u \\ A^2(u, v) & \text{otherwise} \end{cases}, \quad (20)$$

where  $A_u$  is calculated by Eq. 14 on the amplitude spectrum  $A(u, v)$  of LNP.

Since the LNP amplitude spectra of generated images present periodicity, and the LNP amplitude spectra of real images are generally low in energy except at the center, we extract features based on this property. The extracted features can be sampled by

$$F(m, n) = A'(u, v) \delta(u - m\Delta M, v - n\Delta N), \quad (21)$$

where  $\delta(\cdot)$  is a 2-D impulse function,  $k$  is a sampling interval, and  $m$  and  $n$  are integers from  $[0, \lfloor (M-1)/k \rfloor]$  and  $[0, \lfloor (N-1)/k \rfloor]$ , respectively. The choice of  $k$  will be discussed in the experiment. At this point, we can map real

TABLE 1

Performance of a detection method that was sequentially fine-tuned by new model-generated images. Data in bold and underlined represents the best, and the evaluation metric is *ACC*.

Testing Training \	Style GAN -DB	Style GAN2 -DB	Cycle GAN -DB	Big GAN -DB	Star GAN -DB	HiSD -DB	Glow -DB	Latent Diffusion -DB	Disco Diffusion -DB	Stable Diffusion -DB	<i>ACC</i> <i>avg.</i>
StyleGAN-DB	<b>64.99</b>	60.35	50.08	51.40	49.97	49.92	49.97	50.25	50.00	49.86	52.68
+ StyleGAN2-DB	55.54	<b>66.54</b>	50.29	50.45	50.00	50.03	49.97	50.10	52.00	54.22	52.91
+ CycleGAN-DB	59.62	70.40	<b>91.94</b>	51.62	51.95	60.03	47.87	50.60	47.00	40.87	57.19
+ BigGAN-DB	60.16	65.66	72.90	<b>73.38</b>	44.72	49.40	44.32	52.60	56.00	44.14	56.33
+ StarGAN-DB	62.05	65.00	71.38	73.25	<b>92.74</b>	50.63	49.52	53.05	54.00	48.77	62.04
+ HiSD-DB	60.62	61.64	67.67	68.08	90.29	<b>53.13</b>	45.62	51.45	53.00	54.09	60.56
+ Glow-DB	61.16	62.05	68.85	67.41	92.39	52.28	<b>88.75</b>	49.75	52.00	47.82	64.25
+ Latent Diffusion-DB	63.25	66.31	66.19	63.73	89.84	52.73	86.16	<b>64.50</b>	54.00	51.09	65.78

images  $\mathcal{T}$  to a dense subspace  $\mathcal{S}$  by  $F(m, n)$  and finally optimize the one-class classifier.

## 4 EXPERIMENTS

We provide extensive experimental evidence to demonstrate the effectiveness of our method. This section introduces the datasets, describes the experimental setup and performance metrics, gives comparative experimental results, and presents a robustness study.

### 4.1 Datasets and Experimental Setup

We trained and tested the proposed detection method on publicly available datasets whenever possible. We used five datasets provided by [10] containing real images and GAN-generated images from conditional GANs (BigGAN [38], StarGAN [43], CycleGAN [42]), and unconditional GANs (StyleGAN [40], StyleGAN2 [41]) commonly used in generated image detection. To demonstrate the detection ability of our method on emerging generative models, we also included AE structure-based generative model HiSD [44], flow-based model Glow [16] whose network structure is reversible, Disco Diffusion [45] guided by CLIP [46], and the currently most advanced image generative diffusion models Stable Diffusion [25] and Latent Diffusion [47]. For the AE-based, Glow, and diffusion generative models, since public generated image datasets are not available, we used the officially published pre-trained models to produce generated image datasets without post-processing operations on the images. We denote the datasets using the model's name and a “-DB” suffix.

We randomly selected a certain number of real images from the real image subset provided by [10]. The experiment shows that  $\mathcal{T}$  with just 800 real images are enough to construct a dense subspace  $\mathcal{S}$  by  $F(m, n)$  and optimize the one-class classifier, which will be discussed in Subsection 4.4.3. During training, the proposed generated image detection method did not see any generated images. All other generated image detection methods were implemented with their best performance. In the testing phase, we randomly selected 500 real and 500 generated images from each generative model dataset to infer their authenticity and

evaluate our method and the other methods. All training was implemented on an NVIDIA GeForce GTX Titan X GPU and an Intel Xeon E5-2603 v4.

### 4.2 Evaluation Metrics

Generated image detection can be seen as a binary classification problem. Each image belongs to one of two classes, real or generated, and a decision must be made for each image. Since our ultimate goal is to discern the authenticity of an image, Accuracy (*ACC*) is a proper evaluation metric for this task.

Precision in a binary classification task is related to the selection of classification thresholds. Therefore, we use the Average Precision (*AP*), which calculates the precision for all possible classification thresholds and then takes the average.

*F1* score takes into account the Precision and Recall of the classification method. The *F1* score can be considered a harmonic average of the Precision and Recall, with its maximum value of 1 and minimum value of 0.

### 4.3 Verification of Fine-tuning with New Images

Existing generated image detection methods may achieve satisfactory detection results in the laboratory. In practice, a detection method may face ever-growing generative models. Re-training a deployed detection network with existing and new model-generated images is possible, but the cost is too high in practice. A feasible solution is to use new model-generated images to fine-tune deployed detection networks. We experimented with this scenario to study whether this learning paradigm works.

In our experiment, we used the generated image detection network proposed by [10], which uses ResNet50 as the backbone. 5000 images from each generative model were used for fine-tuning, and 1000 per model were used for testing. The network was first trained with 5000 images from StyleGAN-DB. Then, images from datasets of StyleGAN2, CycleGAN, BigGAN, StarGAN, HiSD, Glow, and Latent Diffusion Models were used to fine-tune the network sequentially. After the initial training and each fine-tuning, we tested the network's detection performance on every generative model. Results are in Table 1.

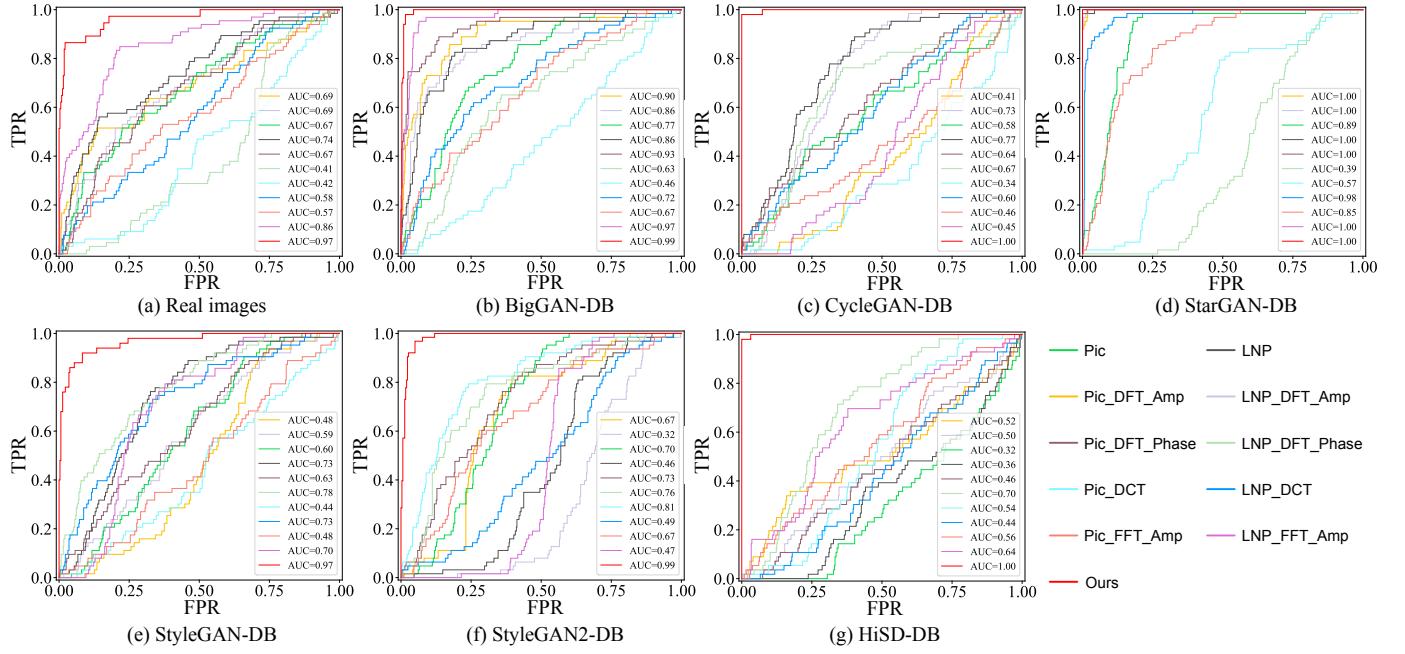


Fig. 7. Comparison of discriminative ability of different features in a multi-class classification experiment. The results are shown as ROC curves. The seven classes are six generative models (StarGAN, BigGAN, CycleGAN, StyleGAN, StyleGAN2, HiSD) and real images.

We can see that as the new model-generated images are used to fine-tune the network, it tends to favor the newly seen generative model, while the detection accuracy of the previously seen generative models decreases, resulting in catastrophic forgetting [48]. This experimental result is also consistent with our viewpoint depicted in Fig. 2 that simply increasing the type of training set does not maintain or improve the test accuracy. Therefore, fine-tuning with new model-generated images does poorly and does not improve the model's generalization ability.

#### 4.4 Discussion of the Proposed Method

##### 4.4.1 Effectiveness of LNP

Before mapping the real images  $\mathcal{T}$  to a more dense subspace  $\mathcal{S}$  by  $F(m, n)$ , we first verify the effectiveness of LNP. Using the enhanced LNP features to train common classifiers, such as SVM, can perform multi-class classification. In this way, we can verify the superiority of its discriminative ability over other image features. The experiment was set to perform seven-class classification on six generative models and real images. We randomly selected 500 generated images from each of the six generated image subsets and 500 real images from all six real image subsets, resulting in 3500 images in total.

We tested on many different types of image features: the original image (Pic), the amplitude spectrum after DFT of the original image (Pic\_DFT\_Amp), the phase spectrum after the DFT of the original image (Pic\_DFT\_Phase), the amplitude spectrum after the Fast Fourier Transform of the original image (Pic\_FFT\_Amp), the original image after Discrete Cosine Transform (Pic\_DCT), the proposed LNP (LNP), the amplitude spectrum after the DFT of LNP (LNP\_DFT\_Amp), the phase spectrum after DFT of LNP (LNP\_DFT\_Phase), the amplitude spectrum after FFT of

LNP (LNP\_FFT\_Amp), the LNP after DCT (LNP\_DCT), and the enhanced amplitude spectrum of LNP calculated by Eq. 20 (Ours). The experimental result of five-fold cross-validation is shown by ROC curves in Fig. 7. We see that using spatial features or phase spectra features cannot obtain stable performance in seven-class classification while using our proposed enhanced LNP feature extracted from amplitude spectra reaches over 0.97 in AUC and outperforms the other features mentioned above in detecting each generative model. The experiment shows that the proposed enhanced LNP features have stronger discriminative ability than the rest even in a more difficult classification task, laying a solid foundation for our generated image detection method.

##### 4.4.2 The Dimension of $F(m, n)$

We sampled from the enhanced LNP amplitude spectra  $A'(u, v)$  to construct  $F(m, n)$ . The value of  $k$  affects the sampling frequency and consequently determines the dimensionality of extracted features  $F(m, n)$ . Excessive dimensionality brings redundant information, while too low dimensionality will lead to insufficient information, affecting the classification results. Therefore, we experimented to determine the value of  $k$ . The results are in Table 2, and the choice of  $k = 32$  is reasonable.

For an image ( $M = N = 256$ ), when  $k$  is 32, the final extracted feature  $F(m, n)$  from the enhanced amplitude spectrum  $A'(u, v)$  has only 64 dimensions (the maximum value of  $m$  and  $n$  is 7). To confirm this choice of  $k$ , we used t-SNE [49] to observe the feature distribution of the real images versus the generated images under eight datasets. We randomly selected 100 real images and 100 generated images from each dataset. The visualization results are in Fig. 8. We see that the final sampled LNP feature  $F(m, n)$  has a good discrimination ability. Moreover, it suggests that

TABLE 2

Performance of our method under different dimension extracted feature  $F(m, n)$  (determined by  $k$ ). Data in bold and underlined represents the best.

Param	Testing Set						$ACC$
	Big GAN -DB	Cycle GAN -DB	HiSD -DB	Star GAN -DB	Style GAN -DB	Style GAN2 -DB	
$k$	Big GAN -DB	Cycle GAN -DB	HiSD -DB	Star GAN -DB	Style GAN -DB	Style GAN2 -DB	avg.
4	0.585	0.820	0.845	0.845	0.760	0.890	0.791
8	0.675	0.875	0.895	0.920	0.905	0.960	0.872
16	0.760	0.920	0.875	0.945	0.920	0.975	0.899
32	0.810	0.980	<b>0.900</b>	0.980	<b>0.905</b>	0.980	<b>0.926</b>
64	<b>0.820</b>	<b>0.990</b>	0.780	<b>0.985</b>	0.715	<b>0.990</b>	0.880

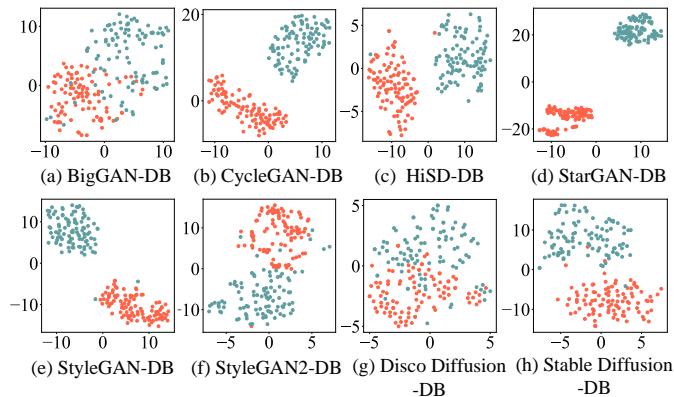


Fig. 8. Visualization of  $F(m, n)$  using t-SNE [49]. Red points denote real images, and green points denote generated images.

real images  $\mathcal{T}$  can be mapped to a more dense subspace  $\mathcal{S}$  by  $F(m, n)$ .

#### 4.4.3 Image Sequence $\mathcal{T}$

As discussed, we believe that real images have similarities, which enables one to construct a dense subspace  $\mathcal{S}$  for one-class classification. To validate this view, we randomly selected one hundred to one thousand real images from the real image subsets in [10] to construct  $\mathcal{S}$ . Then we tested it on every case and recorded the  $ACC$  value. We experimented one hundred times in each case to prevent the coincidence of results.

As shown in Fig. 9(a), there is an optimal range of  $\ell$  for mapping the real images  $\mathcal{T}$  to a more dense subspace  $\mathcal{S}$  by  $F(m, n)$  discussed in Eq. 6 in Subsection 3.1. Even when the number of real images in  $\mathcal{T}$  is only 100, the  $ACC$  of one-class classification reaches more than 80%. As the number of real images  $\ell$  increases, the results of one-class classification improve, and the best result is obtained when the number of real images  $\ell$  is 800. On the other hand, since generated images are so different, they cannot be used to construct a subspace  $\mathcal{S}$  for one-class classification, and the detection results are almost random guesses, as shown in Fig. 9(b).

We also used real images from each generative model's real image subset as  $\mathcal{T}$ . 800 real images were randomly selected from each model's real image subset, and 100 real

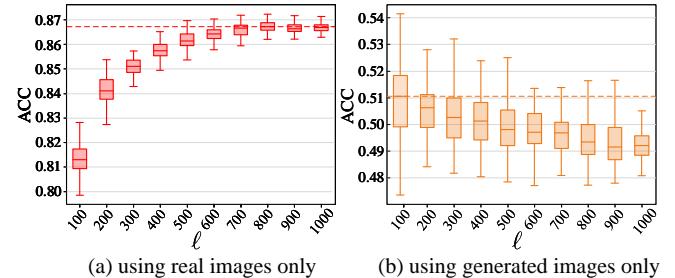


Fig. 9. Types of images and their numbers  $\ell$  in the image sequence  $\mathcal{T}$  to construct a dense subspace  $\mathcal{S}$  for one-class classification. (a) Only real images in  $\mathcal{T}$ . The dense subspace can be successfully constructed, and the best performance of one-class classification is achieved for  $\ell = 800$ . (b) Only generated images in  $\mathcal{T}$ . The constructed subspace is not sufficiently dense and one-class classification does not converge.

TABLE 3

Performance of our method when the image sequence  $\mathcal{T}$  comes from different real image subsets. Data in bold and underlined represents the best.

Image Sequence $\mathcal{T}$	Testing Set						$ACC$
	Big GAN -DB	Cycle GAN -DB	HiSD -DB	Star GAN -DB	Style GAN -DB	Style GAN2 -DB	
	Big	Cycle	HiSD	Star	Style	Style	avg.
BigGAN-DB	0.822	0.908	0.770	0.966	0.848	0.794	0.851
CycleGAN-DB	<b>0.834</b>	<b>0.944</b>	0.792	<b>0.976</b>	0.902	0.844	<b>0.882</b>
HiSD-DB	0.720	0.750	<b>0.902</b>	0.944	0.654	0.646	0.769
StarGAN-DB	0.716	0.748	0.900	0.946	0.654	0.655	0.770
StyleGAN-DB	0.812	0.934	0.624	0.946	<b>0.964</b>	0.920	0.867
StyleGAN2-DB	0.772	0.896	0.660	0.944	0.894	<b>0.962</b>	0.855

and 100 generated images were randomly selected from each model's real and generated image subsets to build the testing set. We repeated this process five times. The numbers in Table 3 are the average value of five experiments' results in the  $ACC$  metric.

Our method performs well in each case. We found that using the real images in the CycleGAN dataset as the image sequence  $\mathcal{T}$  shows better performance. This is because the real images in the CycleGAN dataset include various categories, such as natural and biological images. In all our later experiments, we use 800 real images from the CycleGAN real image subset as the real images  $\mathcal{T}$ .

## 4.5 Quantitative Evaluation and Comparison

### 4.5.1 Accuracy of Generated Image Detection

We now use our proposed generated image detection method to authenticate images and compare it with recently proposed state-of-the-art methods. Our method is compatible with different denoising modules. We used two more denoising networks in the experiment: PNGAN [52] and Restormer [53]. Results are shown in Table 4. Our method outperforms most methods in detection accuracy and is slightly behind our previous work [50]. However, it is worth noting that the proposed authentication method only takes 800 real images for training, using 0.1% training images of

TABLE 4

Our method is compared with other state-of-the-art methods. Real&Generated(ProGAN) indicates the number of real and generated images in the training set. Throughput indicates the number of images that can be authenticated per second. Data in bold and underlined represents the best, while data with parentheses represents the second best.

Methods	Training Set		Metrics	Testing Set										ACC avg.	Throughput image/s
	Real	Generated (ProGAN)		Big GAN -DB	Cycle GAN -DB	Star GAN -DB	Style GAN -DB	Style GAN2 -DB	HiSD -DB	Glow -DB	Disco Diffusion -DB	Latent Diffusion -DB	Stable Diffusion -DB		
Zhang(Spec) WIFS'19 [3]	1.3k	1.3k	ACC	0.80	0.70	<u>1.00</u>	0.62	0.55	0.56	0.65	0.37	0.65	0.62	0.65	69.8
			AP	0.90	0.98	<u>1.00</u>	0.68	0.59	<u>1.00</u>	(0.83)	0.62	0.97	0.84	0.84	
			F1	0.81	0.58	<u>1.00</u>	0.64	0.60	0.70	0.48	0.54	0.74	0.72	0.68	
Zhang et al. WIFS'19 [3](Img)	1.3k	1.3k	ACC	0.55	0.91	0.83	0.58	0.68	(0.96)	0.50	0.49	0.53	0.49	0.65	76.9
			AP	0.74	0.98	(0.99)	0.84	0.78	<u>1.00</u>	0.53	0.39	0.66	0.42	0.73	
			F1	0.67	0.90	0.86	0.70	0.70	0.96	0.45	0.65	0.67	0.65	0.72	
Frank et al. ICML'20 [4]	50k	50k	ACC	0.52	0.50	0.49	0.53	0.57	0.49	0.51	0.50	0.50	0.53	0.51	21.4
			AP	0.33	0.60	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.49	
			F1	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	
Wang et al. CVPR'20 [10]	360k	360k	ACC	0.76	0.84	0.92	0.90	0.80	0.82	0.28	0.56	0.57	0.51	0.70	105.0
			AP	(0.91)	0.95	(0.99)	(0.99)	<u>1.00</u>	(0.99)	0.35	0.55	0.94	0.83	0.85	
			F1	0.80	0.88	0.95	0.85	0.93	0.94	0.57	0.67	0.71	0.67	0.80	
Chai et al. ECCV'20 [8]	360k	360k	ACC	0.73	0.58	0.90	0.79	0.92	0.68	0.16	0.73	0.77	0.56	0.68	34.6
			AP	0.74	0.59	<u>1.00</u>	(0.99)	(0.99)	<u>1.00</u>	0.32	0.78	0.99	(0.93)	0.83	
			F1	0.74	0.60	0.89	0.73	0.91	0.53	0.08	0.70	0.70	0.22	0.61	
Grag et al. ICME'21 [11]	360k	360k	ACC	0.68	0.72	<u>1.00</u>	0.89	<u>0.99</u>	<u>0.98</u>	0.43	0.66	0.69	0.54	0.76	16.1
			AP	0.67	0.78	<u>1.00</u>	(0.99)	<u>1.00</u>	<u>1.00</u>	0.35	0.72	0.80	0.76	0.81	
			F1	0.57	0.62	<u>1.00</u>	0.90	<u>0.99</u>	<u>0.98</u>	0.16	0.69	0.72	0.71	0.73	
Liu et al. ECCV'22 [50]	360k	360k	ACC	<u>0.88</u>	(0.92)	<u>1.00</u>	<u>0.96</u>	0.92	0.94	<u>0.80</u>	0.80	0.86	0.73	<u>0.88</u>	95.4
			AP	<u>0.95</u>	0.98	<u>1.00</u>	(0.99)	(0.99)	<u>1.00</u>	0.69	<u>0.97</u>	0.97	<u>0.96</u>	<u>0.95</u>	
			F1	<u>0.88</u>	0.91	<u>1.00</u>	<u>0.96</u>	0.93	(0.96)	<u>0.80</u>	(0.83)	0.88	0.78	<u>0.89</u>	
Tan et al. CVPR'23 [51]	72k	72k	ACC	0.80	0.85	(0.99)	(0.92)	(0.94)	0.95	0.43	<u>0.87</u>	<u>0.98</u>	(0.78)	(0.85)	32.0
			AP	0.80	0.94	<u>1.00</u>	<u>1.00</u>	<u>1.00</u>	<u>1.00</u>	0.42	(0.86)	<u>1.00</u>	<u>0.96</u>	(0.90)	
			F1	0.83	0.85	(0.99)	(0.91)	(0.94)	0.95	0.43	<u>0.87</u>	<u>0.98</u>	0.72	0.85	
Ours -PNGAN [52]	0.8k		ACC	0.78	<u>0.95</u>	<u>1.00</u>	<u>0.96</u>	0.51	0.60	0.52	0.54	0.81	0.56	0.72	220.6
			AP	0.72	<u>1.00</u>	<u>1.00</u>	<u>1.00</u>	0.59	0.79	0.50	0.59	0.91	0.55	0.77	
			F1	0.80	<u>0.95</u>	<u>1.00</u>	<u>0.96</u>	0.66	0.72	0.67	0.68	0.83	0.70	0.80	
Ours -Restormer [53]	0.8k		ACC	0.69	0.91	0.83	(0.92)	0.52	0.72	(0.75)	0.53	0.83	<u>0.81</u>	0.75	286.7
			AP	0.67	(0.99)	<u>1.00</u>	(0.99)	0.62	0.80	<u>0.84</u>	0.56	0.90	0.80	0.82	
			F1	0.66	0.90	0.80	(0.91)	0.63	0.70	(0.78)	0.60	0.82	<u>0.82</u>	0.76	
Ours -CycleISP [37]	0.8k		ACC	(0.84)	<u>0.95</u>	0.98	0.90	0.85	0.79	0.66	(0.81)	(0.93)	0.74	(0.85)	413.2
			AP	0.85	<u>1.00</u>	<u>1.00</u>	<u>1.00</u>	0.98	0.84	0.74	0.78	(0.97)	0.67	0.88	
			F1	(0.85)	(0.94)	0.98	0.89	0.93	0.83	0.76	0.82	(0.93)	(0.79)	(0.87)	

that method, and avoids using any generated images while achieving good detection accuracy. We also note that Wang's approach [10] used 720k images as the training set and used ResNet50 [15] as the classifier. The approach in [11] used the same training set as [10] but used ResNet50 [15] with two layers of downsampling removed as the classifier, and the training time is four times longer than [10]. In contrast, our method is of high efficiency in both the training and inference phase, able to cope with huge numbers of generated images with its 413.2 images/s throughput when using the CycleISP denoising module.

Moreover, our method can also effectively detect more types of generative models other than GAN models, such as Glow [16], Disco Diffusion [45], Latent Diffusion [47], and Stable Diffusion [25]. Many existing generated image

detection methods cannot effectively detect these model-generated images. In contrast, our method maintains good detection accuracy even though it did not see any images generated by them. This may help with the problem that detection algorithms always lag behind generative algorithms.

#### 4.5.2 Robustness

In real-world scenes, images are subjected to various post-processing operations, such as blurring and noise corruption. We tested the robustness of our method under different post-processing operations applied to generated images. The post-processing operations we used include:

- Blurring: Gaussian filtering with a kernel size of 3 and sigma from 0.1 to 1.
- Brightness adjustment: the adjustment parameter is from 0.3 to 3.

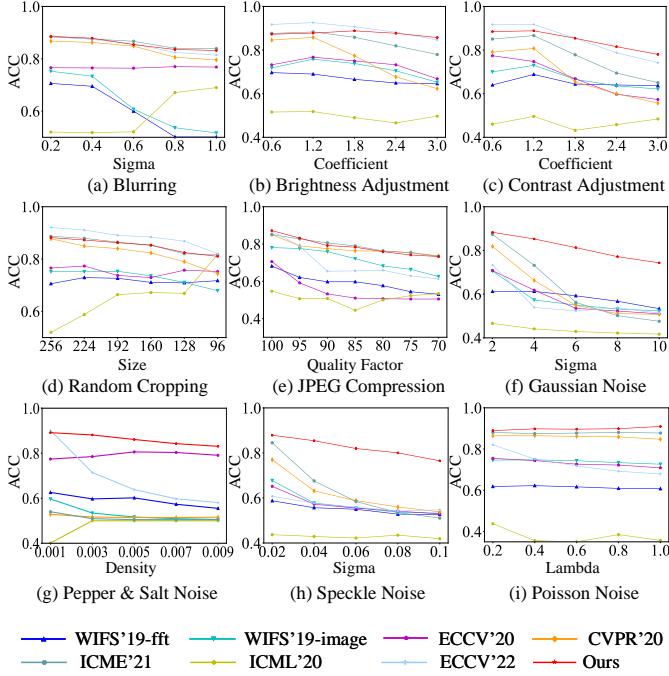


Fig. 10. Robustness of our method against image blurring, brightness and contrast adjustment, random cropping, JPEG compression, corruption by different types of noise.

- Contrast adjustment: Gamma transform with  $\gamma$  from 0.3 to 3.
- Random cropping: for a  $256 \times 256$  image, the cropping size was from 256 to 96, and we up-sampled the amplitude spectrum of the LNP of the final cropped image back to 256.
- JPEG compression: quality factors from 70 to 100.
- Gaussian noise: the sigma was set from 1 to 10, and the PSNR of the original image and the image after adding noise is from 26 to 47.
- Pepper & Salt noise: the ratio of pepper to salt is 1:1. The density of the added noise is 0.001 to 0.01, and the PSNR of the noisy images is from 18 to 31.
- Speckle noise: the sigma ranges from 0.01 to 0.1, and the PSNR of the noisy images is from 22 to 57.
- Poisson noise: the lambda is set from 0.1 to 1, and the PSNR of the noisy images is from 3 to 58.

The one-class classifier we used was still constructed with 800 real images from CycleGAN-DB without any data enhancement via post-processing operations. Fig. 10 shows the robustness of our model via testing by post-processed images. We can see that our model has good robustness under various circumstances. The LNP amplitude spectrum generally is stable in terms of adding noise, blurring, and clipping. For JPEG compression, our method works stably with higher-quality compression factors. Since the DCT in JPEG operates on each  $8 \times 8$  patch, high compression leaves bright spots similar to generated images on the amplitude spectrum of images, resulting in a performance decline.

In general, methods based on noise features are vulnerable to noise corruption. However, from the experimental results, our method shows robustness to different kinds of noise. The LNP is extracted by the denoising network and

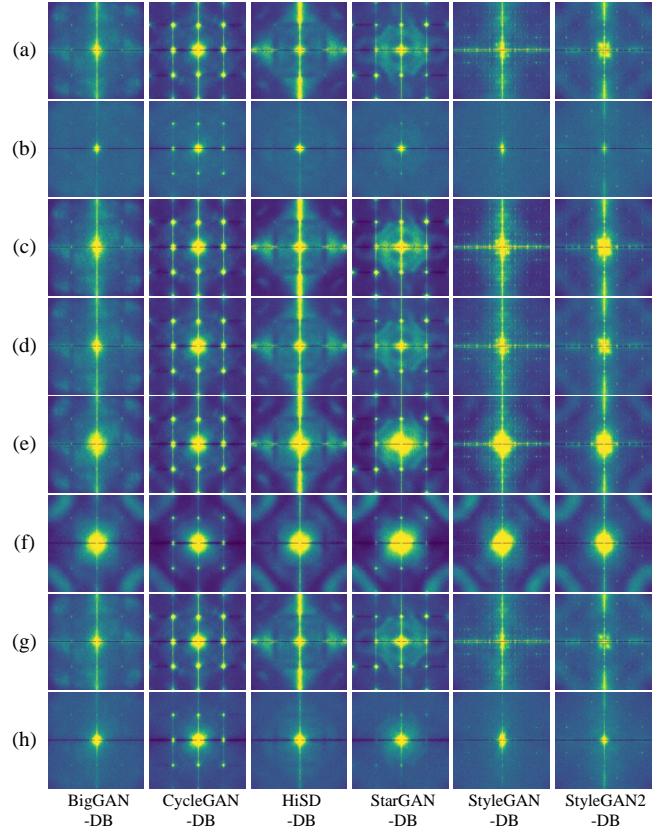


Fig. 11. LNP's amplitude spectra of images with different levels of added noise. (a) Gaussian noise with  $\sigma = 1$ ; (b) Gaussian noise with  $\sigma = 10$ ; (c) Poisson noise with  $\lambda = 0.1$ ; (d) Poisson noise with  $\lambda = 1$ ; (e) Pepper and Salt noise with the density of 0.001; (f) Pepper & Salt noise with density of 0.01; (g) Speckle noise with  $\sigma = 0.01$ ; (h) Speckle noise with  $\sigma = 0.1$ .

TABLE 5  
The robustness of our method against spot suppression on amplitude spectra. We tested on real and generated images.

scenario	All			Real			Generated		
	ACC	AP	F1	ACC	AP	F1	ACC	AP	F1
unprocessed	0.97	1.00	0.97	0.94	1.00	0.97	1.00	1.00	1.00
spot suppression	0.97	1.00	0.97	-	-	-	1.00	1.00	1.00

may contain added noise, as well as low-level features (grid artifacts) of generated images. However, the added noise is randomly distributed, so it does not mask the grid artifacts in the spatial domain, and the LNP remains highly periodic, which still differs from the characteristics of real images, as shown in Fig. 11.

We use the commonality of real images to detect generated images. However, future generative models will make generated images similar to real images as much as possible. Eliminating the bright spots in the amplitude spectrum can be used to reduce visual artifacts and deceive forensic algorithms. Durall's generative model [9] based on DCGAN [54] uses a loss function to suppress spectrum bright spots during the training phase. We tested the robustness of our methods by discerning 100 real and 100 generated images

randomly selected from their dataset [9]. The experimental results shown in Table 5 verify that spot suppression does not affect the detection accuracy.

## 5 CONCLUSION

In this paper, we started from real images to detect generated images. We demonstrated that real images are very similar in their noise amplitude spectra. In contrast, generated images are very different. Therefore, generated images can be detected by mapping them out of the dense subspace constructed by real images only. The experimental results show that the method has good detection performance. The proposed method's superior detection performance and high efficiency allow its use in realistic scenarios, even for detecting possible future new models. Most importantly, this paper provides a new perspective on generated image detection.

## REFERENCES

- [1] Si Liu, Wentao Jiang, Chen Gao, Ran He, Jiashi Feng, Bo Li, and Shuicheng Yan. Psgan++: Robust detail-preserving makeup transfer and removal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8538–8551, 2021.
- [2] Yujun Shen, Ceyuan Yang, Xiaou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2004–2018, 2020.
- [3] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in gan fake images. In *Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2019.
- [4] Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In *Proceedings of the International Conference on Machine Learning*, pages 3247–3258. PMLR, 2020.
- [5] Zhengzhe Liu, Xiaojuan Qi, and Philip HS 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.
- [6] Hao Dang, Feng Liu, Joel Stehouwer, Xiaoming Liu, and Anil K Jain. On the detection of digital face manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5781–5790, 2020.
- [7] Hanqing Zhao, Wenbo Zhou, Dongdong Chen, Tianyi Wei, Weiming Zhang, and Nenghai Yu. Multi-attentional deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2185–2194, 2021.
- [8] Lucy Chai, David Bau, Ser-Nam Lim, and Phillip Isola. What makes fake images detectable? understanding properties that generalize. In *Proceedings of the European Conference on Computer Vision*, pages 103–120. Springer, 2020.
- [9] Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7890–7899, 2020.
- [10] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8695–8704, 2020.
- [11] Diego Gragnaniello, Davide Cozzolino, Francesco Marra, Giovanni Poggi, and Luisa Verdoliva. Are gan generated images easy to detect? a critical analysis of the state-of-the-art. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.
- [12] Ning Yu, Larry S Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7556–7566, 2019.
- [13] Tarik Dzanic, Karan Shah, and Freddie Witherden. Fourier spectrum discrepancies in deep network generated images. *Advances in Neural Information Processing Systems*, 33:3022–3032, 2020.
- [14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [16] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.
- [17] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.
- [18] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.
- [19] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in Neural Information Processing Systems*, 30, 2017.
- [20] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems*, 29, 2016.
- [21] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.
- [22] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.
- [23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- [24] Robin Rombach, Andreas Blattmann, Dominik Lorenz,

- Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2021.
- [25] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.
- [26] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, 1992.
- [27] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in Neural Information Processing Systems*, 12, 1999.
- [28] David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004.
- [29] Wenbin Zhai, Liang Liu, Youwei Ding, Shanshan Sun, and Ying Gu. Etd: An efficient time delay attack detection framework for uav networks. *IEEE Transactions on Information Forensics and Security*, 18:2913–2928, 2023.
- [30] Hengrun Zhang, Kai Zeng, and Shuai Lin. Federated graph neural network for fast anomaly detection in controller area networks. *IEEE Transactions on Information Forensics and Security*, 18:1566–1579, 2023.
- [31] Shao-Yuan Lo, Poojan Oza, and Vishal M. Patel. Adversarially robust one-class novelty detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4167–4179, 2023.
- [32] Muhammad Zaigham Zaheer, Jin-Ha Lee, Arif Mahmood, Marcella Astrid, and Seung-Ik Lee. Stabilizing adversarially learned one-class novelty detection using pseudo anomalies. *IEEE Transactions on Image Processing*, 31:5963–5975, 2022.
- [33] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.
- [34] John Shawe-Taylor, Peter L Bartlett, Robert C Williamson, and Martin Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998.
- [35] Hui Yin, Yuanhao Gong, and Guoping Qiu. Side window filtering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8758–8766, 2019.
- [36] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.
- [37] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Cycleisp: Real image restoration via improved data synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2696–2705, 2020.
- [38] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.
- [39] Xinyang Li, Shengchuan Zhang, Jie Hu, Liujuan Cao, Xiaopeng Hong, Xudong Mao, Feiyue Huang, Yongjian Wu, and Rongrong Ji. Image-to-image translation via hierarchical style disentanglement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8639–8648, 2021.
- [40] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.
- [41] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo 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.
- [42] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2223–2232, 2017.
- [43] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.
- [44] Hisd. <https://github.com/imlixinyang/HiSD>. Accessed: 2023-07-12.
- [45] Disco diffusion. <https://github.com/alembics/disco-diffusion>. Accessed: 2023-07-12.
- [46] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.
- [47] Latent diffusion. <https://github.com/CompVis/latent-diffusion>. Accessed: 2023-07-12.
- [48] Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999.
- [49] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.
- [50] Bo Liu, Fan Yang, Xiuli Bi, Bin Xiao, Weisheng Li, and Xinbo Gao. Detecting generated images by real images. In *Proceedings of the European Conference on Computer Vision*, pages 95–110. Springer, 2022.
- [51] Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, and Yunchao 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.
- [52] Yuanhao Cai, Xiaowan Hu, Haoqian Wang, Yulun Zhang, Hanspeter Pfister, and Donglai Wei. Learning to generate realistic noisy images via pixel-level noise-aware adversarial training. *Advances in Neural Information Processing Systems*, 34:3259–3270, 2021.
- [53] Syed Waqas Zamir, Aditya Arora, Salman Khan, Mu-

- nawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5728–5739, 2022.
- [54] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2016.