What can Discriminator do? Towards Box-free Ownership Verification of Generative Adversarial Networks

Ziheng Huang\textsuperscript{1†}, Boheng Li\textsuperscript{1†}, Yan Cai\textsuperscript{1}, Run Wang\textsuperscript{1∗}, Shangwei Guo\textsuperscript{2}, Liming Fang\textsuperscript{3}, Jing Chen\textsuperscript{1}, Lina Wang\textsuperscript{1}

\textsuperscript{1} Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, China
\textsuperscript{2} College of Computer Science, Chongqing University, China
\textsuperscript{3} College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China

\textsuperscript{†} Equal contribution * Corresponding author. E-mail: wangrun@whu.edu.cn

Abstract

In recent decades, Generative Adversarial Network (GAN) and its variants have achieved unprecedented success in image synthesis. However, well-trained GANs are under the threat of illegal steal or leakage. The prior studies on remote ownership verification assume a black-box setting where the defender can query the suspicious model with specific inputs, which we identify is not enough for generation tasks. To this end, in this paper, we propose a novel IP protection scheme for GANs where ownership verification can be done by checking outputs only, without choosing the inputs (i.e., box-free setting). Specifically, we make use of the unexploited potential of the discriminator to learn a hypersphere that captures the unique distribution learned by the paired generator. Extensive evaluations on two popular GAN tasks and more than 10 GAN architectures demonstrate our proposed scheme to effectively verify the ownership. Our proposed scheme shown to be immune to popular input-based removal attacks and robust against other existing attacks. The source code and models are available at https://github.com/AbstractTeen/gan_ownership_verification.

1. Introduction

With the rapid development of GANs, we have witnessed fruitful applications of GAN in many fields, such as realistic facial images synthesis [45], fine-grained attribute editing [55], etc. Unlike the classification model with specified label prediction, the GANs learn a data distribution and output the synthesized data sample within a certain distribution. In GANs, the discriminator and generator are two essential components, where the discriminator works as a judge to discriminate whether the sample is produced by the genera-
To tackle this problem, we leverage the Pearson correlation to quantify the implicit reconstruction ability of the discriminator, and prevent the degradation via adding the term into the loss function of the discriminator.

To comprehensively evaluate the effectiveness and robustness of our proposed method, we conduct experiments on two popular GAN tasks (i.e., entire image synthesis, image-to-image translation) and 10 state-of-the-art GAN architectures to demonstrate the effectiveness of our scheme in verifying the ownership of generator. We also show qualitatively and quantitatively that our scheme is immune to popular and powerful removal attacks (e.g., input transformation-based and reverse engineering-based attacks) and robust to other existing attacks.

Our main contributions are summarized as follows:

- We identify a fundamental limitation of black-box setting-based ownership verification schemes on generation tasks, i.e., choosing deterministic inputs is not allowed for applications like unconditioned image synthesis.

- We reveal the unexploited potential of the well-trained discriminator for capturing the unique distribution learned by the paired generator. Based on this finding, we make the first attempt towards box-free verification scheme of GANs, which does not require specifying the input and does not rely on additional models.

- Extensive evaluations on two popular GAN applications and more than 10 GAN architectures demonstrate our proposed scheme to effectively verify the ownership. Through qualitative and quantitative analysis, we show that our proposed scheme is immune to popular removal attacks and robust to other existing attacks.

2. Related Work

2.1. Ownership Verification

Black-box Ownership Verification. The black-box ownership verification scheme assumes the defender can verify the ownership via querying the suspicious model with specific inputs [1, 64, 28, 54]. Towards this end, the owner either predetermines and trains the model to learn a set of abnormal input-output behaviours (model watermarks), or extract a set of boundary samples (e.g., adversarial examples) that can identify the model (model fingerprint). Then, this hidden behavior can be extracted remotely by querying the model with these special inputs. Recently, Ong et al. [37] proposed the first work towards ownership verification of GANs. In their work, the IP information was embedded through backdooring the generator, after which trigger inputs (i.e., carefully-crafted inputs with trigger noises, as shown in Fig. 1 top panel) will result in a visible watermark like a company’s logo on generated outputs. However, for some GAN applications (e.g., unconditional image synthesis), black-box is not enough since deterministic inputs are not allowed [62]. Moreover, the recent removal attacks have shown great threats to the survival of backdoor or adversarial input-based black-box verification paradigms.

Removal Attacks. The black-box methods rely on hidden functionalities in the model. Therefore, current attacks...
Table 1: Comparison of our approach with prior works. In the “Img. Syn./Trans.” column, ✓ denotes not applicable or not evaluated in the original paper.

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
<th>Technique</th>
<th>Target model</th>
<th>Purpose</th>
<th>Box-free?</th>
<th>Img. Syn./Trans.?</th>
<th>Ambiguity attack?</th>
<th>No external model?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uchida et al. [52]</td>
<td>2017</td>
<td>Model Watermarking</td>
<td>DNN</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Adi et al. [1]</td>
<td>2018</td>
<td>Model Watermarking</td>
<td>DNN</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Zhang et al. [64]</td>
<td>2018</td>
<td>Model Watermarking</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Lukas et al. [34]</td>
<td>2019</td>
<td>Model Fingerprinting</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Le Merrer et al. [28]</td>
<td>2020</td>
<td>Model Watermarking</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Zhao et al. [65]</td>
<td>2020</td>
<td>Model Fingerprinting</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Jia et al. [22]</td>
<td>2021</td>
<td>Model Watermarking</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cao et al. [8]</td>
<td>2021</td>
<td>Model Fingerprinting</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Bansal et al. [4]</td>
<td>2022</td>
<td>Model Watermarking</td>
<td>DNN</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Peng et al. [19]</td>
<td>2022</td>
<td>Model Fingerprinting</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Yang et al. [55]</td>
<td>2022</td>
<td>Model Fingerprinting</td>
<td>Classifiers</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Yu et al. [61]</td>
<td>2019</td>
<td>Fingerprint Extraction</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Yu et al. [62]</td>
<td>2021</td>
<td>GAN Fingerprinting</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Girish et al. [15]</td>
<td>2021</td>
<td>Fingerprint Extraction</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Asnani et al. [3]</td>
<td>2021</td>
<td>Fingerprint Extraction</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Yu et al. [63]</td>
<td>2022</td>
<td>GAN Fingerprinting</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Guanera et al. [17]</td>
<td>2022</td>
<td>Fingerprint Extraction</td>
<td>GAN</td>
<td>Model Attribution</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ong et al. [37]</td>
<td>2021</td>
<td>GAN Watermarking</td>
<td>GAN</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ours</td>
<td>2023</td>
<td>Distribution Capturing</td>
<td>GAN</td>
<td>Ownership Verification</td>
<td>✓</td>
<td>✓/✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

A straightforward attack is to eliminate the hidden functionality through model modifications such as pruning [35]. A more targeted attack is first to reverse-engineer the verification samples and then invalidate the watermarks/fingerprints through adversarial training [56, 48]. Observing the verification samples are less robust than normal inputs, another attack is to evade verification via preprocessing the input [19, 56]. This attack has become the recent trend since it is a model architecture-careless and not limited to a specific watermarking technique. It is also intractable as the input transformations can be diverse and usually hard to consider in advance. In contrast, our proposed box-free ownership verification scheme is free from choosing verification samples thus totally immune to the intractable input transformation-based attacks and reverse engineering-based attacks.

**Ambiguity Attack.** Recent works [12, 13, 37] revealed the concept of ambiguity attack, where it is proved that unless an irreversible verification scheme is adopted, the adversary can forge his/her own vouch using exactly the same technique the owner adopted. Subsequently, when one claims ownership of the model, the adversary can also claim ownership due to the existence of his/her own vouch. Finally, the ownership is in doubt. To ensure the owner is free from this concern, a feasible technique used for verifying ownership should necessarily be non-reproducible, even if the adversary has full control of the stolen model and acquires knowledge of the adopted ownership verification paradigm. A practical and robust ownership verification scheme should well survive these two threats.

### 2.2. Model Attribution

The model attribution was initially developed to combat DeepFakes, where researchers focus on attributing certain fake images to the specific types of GAN that generated them [61, 62, 63, 2, 17]. Generally, the attribution techniques aim to analyse the unique fingerprints carried by the GAN-generated images, or proactively watermark the output images through methods like steganography [57, 65, 66]. Due to the merit that these techniques can usually work in the box-free setting where only generated images are available, it has shown potential or even applications in ownership verification of GANs. However, the existing studies on GAN model attribution are not ready for this challenging task, since i) most works are limited to classifying the model architecture and/or datasets [6, 15, 59] rather than a specific GAN model; ii) nearly all attribution techniques require training a powerful external classifier, which is both time and resource consuming; and iii) most works’ external classifiers can be trained with only real/fake images [61, 3, 11], and the steganography-based techniques could be easily reproduced by the attacker [57, 65], which denotes that the adversary can easily forge the verification vouch and perform ambiguity attacks or overwriting attacks. A comparison between our approach with prior studies is illustrated in Tab. 1.

### 3. The Proposed Verification Scheme

Motivated by the unstable training phenomenon of GAN training, the insight behind our proposal is to capture the model-specific distribution by exploiting the potential of the paired discriminator. Before stepping into the details of our proposed scheme, we first introduce the organization of this section. First, we formalize our threat model. Then, we describe the details of the essential loss terms used in our proposed scheme. Finally, we introduce the pipeline and training flow of our proposed approach.

#### 3.1. Threat Model

In our threat model, two opposing parties are considered. A model owner (also the defender) who trains the source
model, and an adversary who has stolen the source model through illegal means and deploys the piracy model on Internet APIs publicly accessible for profit.

**Defender’s Goals and Capabilities.** The goal of the defender is to identify the stolen models that are remotely deployed by the attacker. The defender (1) has white-box access to the source model, (2) can query the suspicious model but, (3) cannot query the query input (i.e., box-free setting). This setting is more challenging than the black-box setting widely adopted by prior works [1, 37, 54], as specifying carefully-crafted inputs is not allowed.

**Adversary’s Goals and Capabilities.** The adversary’s goal is not to be verified as pirated while keeping the piracy a similar performance as the source model. For this purpose, the adversary may modify the model (model modification) or manipulate the inputs/outputs (sample transformation). Observe that our scheme does not suffer from the intractable input transformation-based attacks since our inputs are totally equivalent to those of normal users. Adopting such attacks would only impair the performance yet does not bring any benefits. The assumptions we make are standard and are also widely adopted by prior works [1, 64, 22, 37].

### 3.2. Compactness Loss

The discriminator witnessed the gradual development of the generator and has comparable parameters to the latter. It learns a hyperplane in its embedding space to distinguish between real and generated images. Intuitively, it potentially learned how to extract special representations in images generated by the paired generator. However, in real-world scenarios, suspicious images are usually generated by unknown and unseen GANs. Since embedding spaces of different generators are not aligned, it is difficult to harness the discriminator and find maximum margin hyperplanes to distinguish between different GAN instances.

Inspired by previous works on data description [49, 50], we propose to separate the data via optimizing a hypersphere instead of a hyperplane. Specifically, let \( \mathcal{X} \subseteq \mathbb{R}^{c \times h \times w} \) be the data space, and \( \phi: \mathbb{R}^{c \times h \times w} \to \mathbb{R}^d \) be the “encoder” part of the discriminator (explained later), which maps the data to a \( d \)-dimensional feature space. Our objective is to find the smallest hypersphere specified by a center \( c \in \mathbb{R}^d \) and radius \( R > 0 \) that encloses the majority of the data distribution of the paired generator in feature space. Therefore, our main objective is to minimize the “compactness” of representations [40, 44, 42]. Given the data \( \{x_1, \ldots, x_n\} \) from the paired generator, our objective is defined as:

\[
\min_{W,R} R^2 + \frac{1}{n \mu} \sum_{i=1}^{n} \max\{0, ||\phi(x_i; W) - c||^2 - R^2\} \tag{1}
\]

where \( W \) indicates the weights of \( \phi \). This objective contains two terms. The first term aims to minimize the radius thus volume of the hypersphere, while the second term penalizes the points outside the hypersphere. \( \nu \in (0, 1) \) is a hyperparameter that balances the importance of the two terms. We empirically set \( \nu \) to 0.35 in the following experiments.

The benefits of our approach are as follows. First, our training is unsupervised, i.e., only requires data generated by the paired GAN, analogous to the training of GAN itself and does not introduce additional annotation overhead. Second, prior works utilizing the compactness loss as their objective are shown vulnerable to “hypersphere collapse” where the hypersphere radius collapses to zero and the network converges to trivial solutions [51]. Fixing \( c \) as the mean of the network representations is shown to be helpful to avoid overfitting and hypersphere collapse [44, 10]. However, it is still difficult to define a proper hypersphere center \( c \) with an initial network whose parameters are random. In contrast, our well-trained discriminator network potentially provides us with a robust network representation thus a robust \( c \). Empirical results show that this strategy makes the convergence faster and more robust, the hypersphere collapse hardly exist.

### 3.3. Pearson Correlation Loss

Recall that we expect our discriminator to provide a robust initial center for training. This is feasible since we assume the discriminator extracts data representations well, thus network representations are useful. However, in reality, we observe that the discriminator usually converges to a constant and no helpful network representation is preserved. This is not surprising since the more the GAN is trained, the less possible to distinguish between samples of real and generated data. The optimal discriminator converges to 1/2 and thus no extraction ability is preserved, as many existing literature has pointed out [16, 67].

To tackle this challenge, we propose to preserve the useful network representations of the discriminator via encouraging it to implicitly reconstruct the ground truth latent representation \( z \). Our key insight is that this additional task encourages the discriminator to fit the generator’s latent distribution and prohibits its trivial convergences. However, we empirically found that the straightforward MSE loss makes the training extremely unstable. In our approach, we leverage the pearson correlation loss, which is inspired by the pearson Correlation Coefficient (PCC) [38], to measure the quality of the reconstructed latent representation. We define our term as:

\[
\rho(z, \hat{z}) = \frac{\sum_{i=1}^{n_z} (z_i - \mu_z)(\hat{z}_i - \mu_{\hat{z}})/n_z}{\sigma_z \sigma_{\hat{z}}} \tag{2}
\]

where \( \hat{z} \) is the reconstructed latent representation and \( z \) is the ground truth. \( n_z \) is the dimension of \( z \). \( \mu \) and \( \sigma \) indicates the mean value and standard deviation, respectively. \( \rho(\cdot, \cdot) \in [-1, 1] \) measures the linear correlation between the variables. The higher \( \rho \) indicates better reconstruction performance.

PCC is “milder” than MSE [46]. The pearson correlation
loss ensures that the discriminator can implicitly reconstruct the latent representation from its corresponding generated image while avoiding making the training unstable. For training, obviously, this additional task can be easily cooperated with the BCE loss of the native GAN training. There is no annotation cost and no notable training overhead. We show in the supplementary materials that this additional loss does not bring any degradation to the original generation task.

Note that this loss is optional for GANs that are trained in supervised setting (e.g., StarGAN). This is because these tasks usually require the discriminator to do an additional classification task, therefore the aforementioned convergence problem does not necessarily exist.

3.4. Training Pipeline

The whole training pipeline of our scheme is presented in Algorithm 1. We utilize the pearson correlation loss to preserve the network representations of the well-trained discriminator and harness these representations to train a robust hypersphere in the embedding space which captures the unique data distribution of the paired generator. We show the whole training pipeline as follows.

Step 1. Redefine the Training Objective: For unsupervised GANs, we introduce an additional task (i.e., reconstructing the latent representation) to ensure the network representations of the well-trained discriminator are useful. The native GAN is composed of a generator mapping \( \mathbb{R}^d \rightarrow \mathbb{R}^{c \times h \times w} \) and a discriminator mapping \( \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R} \), which can be divided into an “encoder” \( \phi : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^d \) and a “classifier” that maps \( \mathbb{R}^d \rightarrow \mathbb{R} \). As explained in Sec. 3.3, we wish the reconstructed latent representation \( \hat{z} = \phi(G(z); \mathcal{W}) \in \mathbb{R}^d \) to be close to the ground truth representation \( z \in \mathbb{R}^d \). This is done via adding the pearson correlation loss to the adversarial objective of both \( G \) and \( D \). The training objective of the generator is:

\[
\min_{G} \mathcal{L}_G = \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z))) - \lambda \rho(z, \hat{z})]
\]  

(3)

The training objective of the discriminator is:

\[
\max_{D} \mathcal{L}_D = \mathbb{E}_{x \sim p_d(x)}[\log D(x) - \lambda \rho(z, \hat{z})]
\]  

(4)

where \( \lambda \) is a hyper-parameter that adjusts the pearson correlation strength. We empirically set \( \lambda = 0.5 \) in experiments.

Step 2. Train the GAN Models: We then initialize and train the GAN models \( G \) and \( D \) with some training data \( \mathcal{D} = \{x_1, \ldots, x_n\} \subseteq \mathcal{X} \) until we reach the maximum epochs. Except for the additional pearson correlation loss, the training of the GANs follow the settings in their original papers. After this, the generator is capable to tackle the task distribution and can be deployed.

Step 3. Empower the Discriminator: The final step is to utilize the discriminator’s well-trained network representation and optimizes the network which forms a hypersphere to enclose the learned distribution. We first harness the generator to generate a few data points sampled from the distribution, forming a set of training data \( \mathcal{D}' = \{x'_1, \ldots, x'_n\} \subseteq \mathcal{X} \), where \( x'_i = G(z_i) \sim p_z(z) \). As explained in Sec. 3.2, we fix \( c \) as the mean of the network representations of the discriminator and train the network with data \( \mathcal{D}' \) and the objective described in Eq. (1).

We use stochastic gradient descent (SGD) to optimize the parameters \( \mathcal{W} \) of the neural network with backpropagation. Noticeably, using one common SGD learning rate may be inefficient to optimize \( \mathcal{W} \) and \( R \) simultaneously since they usually have different scales, as Ruff et al. [44] pointed out. Therefore, we optimize \( \mathcal{W} \) and \( R \) alternatively as suggested. In detail, we first fix the radius \( R \) and train the network parameters \( \mathcal{W} \) for every interval \( k \in \mathbb{N} \) epochs. Then, after every \( k \) epoch, we solve for radius \( R \) via line search with the current network parameters. We train the network parameters \( \mathcal{W} \) and radius \( R \) until convergence.

3.5. Ownership Verification

In predicting a given input, we can calculate a score which measures the representation proximity to the captured unique distribution using the network parameters. Given an input \( x \), the representation proximity score is calculated by the distance of the point to the center of the hypersphere:

\[
s(x) = \|\phi(x; \mathcal{W}) - c\|^2 - R^2
\]  

(5)
Therefore, we feed a batch of images produced by the suspicious proximity score threshold. Through extreme results on each model independently trained in similar settings) as piracy as well, even only small factors (e.g., initial seeds) are different. Fig. 2 and Fig. 3 present the experimental results on verifying the two popular GAN tasks (i.e., entire images synthesis, image-to-image translation) measured with AUC score.

The closer the AUC score is to 100%, the greater the difference between two batches of synthesized outputs exposed. On the contrary, the closer the AUC score is to 50%, the more likely the two batches of images are from the same model because φ judges that there is no notable difference in the data distribution represented by the two batches.

### 4.1. Setup

Models. For image synthesis, we use DCGAN [41] and its two variants: SNDCGAN [36] and DCGAN established with residual block [20]. We also conducted experiments on SOTA architectures including ProGAN [24], StyleGAN [25], StyleGAN2 [26] and StyleGAN3 [27]. For image-to-image translation, we consider three popular GANs that edit face attributes: StarGAN [9], AttGAN [21], and STGAN [32].

Datasets. We evaluate our method on two popular datasets, LSUN [60] and CelebA [33]. We use LSUN bedroom for image synthesis and CelebA for both image synthesis and image-to-image translation.

### 4.2. Effectiveness

We first show preliminarily that our method can effectively find out the piracy and would not mistakenly recognize homogeneous models (i.e., the model independently trained in similar settings) as piracy as well, even only small factors (e.g., initial seeds) are different. Fig. 2 and Fig. 3 present the experimental results on verifying the two popular GAN tasks (i.e., entire images synthesis, image-to-image translation) measured with AUC score.

The closer the AUC score is to 100%, the greater the difference between two batches of synthesized outputs exposed. On the contrary, the closer the AUC score is to 50%, the more likely the two batches of images are from the same model because φ judges that there is no notable difference in the data distribution represented by the two batches.
Model Architectures. We consider the case where GANs with different architectures are trained on the same dataset and initial seeds. Experimental results in Fig. 2 and 3 illustrated that our method can identify the differences between GANs with different model architectures with an AUC score larger than 70% and determine a piracy GAN with the same elements with an AUC score close to 50%.

Training Datasets. We also investigate the effect of training a GAN on different datasets. We respectively split CelebA and LSUN into 3 disjoint subsets with 50k images each. This ensures the training data are from the similar distribution. Fig. 2 (c) shows that the AUC scores of two GANs trained on different datasets are all above 75%. The AUC score is higher than the other two elements, which indicates that the GAN training is sensitive to the training datasets.

Initialization seeds. We finally investigate the effects of initial random seeds in ownership verification. Fig. 2 shows that our method maintains AUC around 60%. Meanwhile, all the AUC scores of two identical models are less than 53%. We note that the initialization random seed is the smallest variable in training a homogeneous GAN. This extreme situation (i.e., the source model and the piracy only differ in random seeds) is almost impossible to happen in reality. However, our scheme still achieves an AUC score of ~ 60%, notably margins from the AUC score of two same models (~ 50%). This is why we set the suspicious AUC score to 60%.

<table>
<thead>
<tr>
<th></th>
<th>StyleGAN2</th>
<th>StyleGAN3</th>
<th>StyleGAN</th>
<th>ProGAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>StyleGAN2</td>
<td>53.39</td>
<td>83.12</td>
<td>84.91</td>
<td>85.66</td>
</tr>
<tr>
<td>StyleGAN3</td>
<td>82.57</td>
<td>53.28</td>
<td>83.52</td>
<td>86.34</td>
</tr>
</tbody>
</table>

Table 2: Evaluation on unknown and SOTA GANs. The AUC means the classifier is trained with \{column\}'s discriminator, measuring the images generated by the \{column\}'s G. We mark the results from the paired G and D in bold font and unpaired in normal font.

4.3. Scalability to SOTA Architectures

Table 2 shows the experimental results on the SOTA GAN Architectures. The results show that on SOTA architectures the performances are even better (AUC > 80%). This is because the more complex the task is (e.g., larger datasets, higher resolution, more sophisticated network architecture), the discriminator has larger parameters and the distributions learned by different GAN instances are more complex thus more different from each other. This shows that our method generalizes well on SOTA and potential future architectures.

4.4. Robustness

In this section, we mainly evaluate the robustness of our method in tackling the three existing attacks, i.e., model pruning, output transformations, and ambiguity attack. The experiments here are done on DCGAN trained on CelebA with size 128 x 128. Specifically, we note that the discriminator plays a similar role to private keys and should be kept secret by the owner. Therefore, according to our threat model, attacks that require the presence of the discriminator (e.g., fine-tuning) are not considered.

Pruning. The model pruning aims to reset the unimportant weights to 0 without introducing any performance degradation to the main task, like image synthesis in GANs. In experiments, we randomly reset the weights in GANs to explore whether our proposed method is sensitive to the model pruning. Experimental results in Fig. 4 (left) shows the AUC score for measuring the similarity of two models is less than 60% when the pruning rate reaches almost 40%. This indicates that our proposed method could verify two models in high confidence in this pruning rate settings. Here, the pruning rate less than 40% is a region of operation (ROO) as the quality of synthesized image is almost acceptable, where the FID score is larger than 60. Fig. 4 (right) visualizes the synthesized images when the pruning rate is 40%, which exhibits obvious damage. Thus, our method could survive the model pruning well.

<table>
<thead>
<tr>
<th>Image transformation (magnitude)</th>
<th>CelebA AUC</th>
<th>CelebA SSIM</th>
<th>LSUN AUC</th>
<th>LSUN SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>noise (σ=0.05)</td>
<td>56.41</td>
<td>0.81</td>
<td>55.93</td>
<td>0.84</td>
</tr>
<tr>
<td>blur (k=3, σ=2)</td>
<td>57.16</td>
<td>0.78</td>
<td>57.24</td>
<td>0.80</td>
</tr>
<tr>
<td>JPEG (factor=60)</td>
<td>54.90</td>
<td>0.73</td>
<td>56.77</td>
<td>0.69</td>
</tr>
<tr>
<td>crop (15%)</td>
<td>54.15</td>
<td>0.25</td>
<td>56.71</td>
<td>0.27</td>
</tr>
</tbody>
</table>

Table 3: Evaluation for the image transformation attacks measured by AUC scores and the similarity of images is measured by SSIM. The value of magnitude indicates the ROO, where the image has acceptable quality.

Image Transformations. Though input transformations are vain attempts, the adversary may conduct various output transformations to evade the verification. In experiments, we explore whether the sample transformation brings any degradation to our verification. Fig. 6 shows the performance of our method measured by AUC score in identifying suspicious models under four types of image transformations. The dotted line in Fig. 6 indicates a ROO where the synthesized images have no serious damage to human eyes. Fig. 5 visualizes the images under the ROO setting and Tab. 3 shows the corresponding magnitudes for the four types of image transformation. Experimental results illustrated
that our method could identify the ownership effectively in the ROO of image transformations. We attribute this to the distribution-capturing property of our proposed method, which potentially learned some high-level features that are robust against these low-level transformations.

**Ambiguity Attacks.** In a real scenario, the adversary launches an ambiguity attack by obtaining a classifier that performs as well as the owner’s one after replicating the model illegally. We simulate this attack by training a classifier using exactly the same technique described in Sec. 3 but without the help of the discriminator. Tab. 4 shows that random initialization will significantly degrade the performance of the learned hypersphere. The reason, as we have mentioned earlier, is that the discriminator provides a strong and unique center \( c \), compared to naive random initialization. Thus, the attacker failed to obtain a well-performed classifier even if he obtains the philosophy of our proposed method.

**5. Conclusion and Discussion**

In this paper, we propose a novel ownership verification scheme for GANs, which working in box-free manner, universal to popular GAN tasks, and resisting the powerful ambiguity attack well. Inspired by the power of discriminator in witnessing the development of generators in synthesizing images gradually, we empower the discriminator to capture the unique GAN training which is important for ownership verification. Evaluation experiment results show that our method is highly effective, general and robust.

**Limitations and Discussions.** Though there seem to be no trivial adaptive attacks, our method relies on the empowered discriminator to capture the unique distribution learned by the generator. Therefore, if the discriminator is disclosed, the adversary may train a same classifier to confuse the verification, or leverage the discriminator to perform adversarial attacks. However, the discriminator is conventionally considered useless and will not be used after training. In the popular MLaaS scenario, there is also no reason for the model owner to open it to a third party. The adversary could not even access the discriminator through APIs. In our proposed scheme, the discriminator plays a similar role to private keys and should be kept secret by the owner.

There are many spaces worth discovering for future works. For example, one may extend our insight of distribution capturing to other generative models like diffusion models. It is also interesting to explore effective methods to evade our box-free verification, which could be our future work.

**6. Acknowledgment**

This research was supported in part by the National Key Research and Development Program of China under No.2021YFB3100700, the National Natural Science Foundation of China (NSFC) under Grants No. 62202340, the Open Foundation of Henan Key Laboratory of Cyberspace Situation Awareness under No. HNTS2022004, Wuhan Knowledge Innovation Program under No. 2022010801020127, the Fundamental Research Funds for the Central Universities under No. 2042023k0121, the Natural Science Foundation of Hubei Province under No. 2021CFB089.
References


[12] Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. Advances in neural information processing systems, 32, 2019. 3


[29] Yiming Li, Yang Bai, Yong Jiang, Yong Yang, Shu-Tao Xia, and Bo Li. Untargeted backdoor watermark: Towards harmless and stealthy dataset copyright protection. In NeurIPS, 2022. 2


[59] Tianyun Yang, Ziyao Huang, Juan Cao, Lei Li, and Xirong Li. Deepfake network architecture attribution. In Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI 2022), 2022. 3


