Stealing Machine Learning Models: Attacks and Countermeasures for Generative Adversarial Networks

Hailong Hu
SnT, University of Luxembourg
Esch-sur-Alzette, Luxemburg
hailong.hu@uni.lu

Jun Pang
FSTM & SnT, University of Luxembourg
Esch-sur-Alzette, Luxemburg
jun.pang@uni.lu

ABSTRACT

Model extraction attacks aim to duplicate a machine learning model through query access to a target model. Early studies mainly focus on discriminative models. Despite the success, model extraction attacks against generative models are less well explored. In this paper, we systematically study the feasibility of model extraction attacks against generative adversarial networks (GANs). Specifically, we first define fidelity and accuracy on model extraction attacks against GANs. Then we study model extraction attacks against GANs from the perspective of fidelity extraction and accuracy extraction, according to the adversary's goals and background knowledge. We further conduct a case study where the adversary can transfer knowledge of the extracted model which steals a state-of-the-art GAN trained with more than 3 million images to new domains to broaden the scope of applications of model extraction attacks. Finally, we propose effective defense techniques to safeguard GANs, considering a trade-off between the utility and security of GAN models.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

KEYWORDS

Model extraction; Generative adversarial networks; Transfer learning; Perturbation-based defenses

ACM Reference Format:
Hailong Hu and Jun Pang. 2021. Stealing Machine Learning Models: Attacks and Countermeasures for Generative Adversarial Networks. In *Annual Computer Security Applications Conference (ACSAC '21), December 6–10, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3485832.3485838

1 INTRODUCTION

Over the past few years, machine learning, deep learning in particular, has gained significant advances in a variety of areas, such as computer vision [6, 32, 33, 63] and natural language processing (NLP) [15, 35]. In general, machine learning models are often considered as the intellectual property of model owners and are closely safeguarded. The reasons are from at least two aspects. First, obtaining a practical deep learning model is non-trivial. This is because training a model requires a large number of training data, intensive computing resources and human resources [7, 15, 33, 55, 63, 71]. Second, deep learning models themselves are confidential, and exposure of deep learning models to potential adversaries poses a threat to security and privacy [42, 43, 51, 56, 60, 64]. However, model extraction attack — a novel attack surface targeting at duplicating a model only through query access to a target model, has recently emerged and gained significant attention from the research community.

In the early study, Tramèr et al. [64] first attempt model extraction on traditional machine learning models and shallow neural networks, such as logistic regression, decision tree, support vector machine and multilayer perceptrons. Since then, Jagielski et al. [27] further mount the attack against a million of parameters model trained on billions of Instagram images [44], which makes model extraction attack more practical. In addition to model extraction on deep convolutional neural networks about image classification, there are some works studying the problem of model extraction in NLP tasks [34, 61]. For instance, with the assumption that victim models are trained based on the pretrained BERT model, Krishna et al. [34] show that an adversary can effectively extract language models whose performance is only slightly worse than that of the victim models. However, to the best of our knowledge, these model extraction attacks mainly focus on discriminative models. The attack against generative models, GANs in particular, is still an open question.

Comparing to model extraction attacks on discriminative models, we observe that there exist some differences for generative models. First, adversaries can leverage output information from target models such as labels, probabilities and logits, to mount model extraction attacks on discriminative models [27, 42, 49, 64], while generative models do not provide such information but only return images. Second, model extraction attacks on discriminative models are evaluated on a test dataset. In contrast, unsupervised generative models aiming to learn the distribution of training data are evaluated by quantitative measures such as Fréchet Inception Distance (FID) [23] and multi-scale structural similarity (MS-SSIM) [48], or qualitative measures such as preference judgment [26, 72]. Therefore, these differences indicate that model extraction strategies, evaluations and defenses on generative models are very different from these on discriminative models.

In this paper, we aim to systematically study the feasibility of model extraction attacks against GANs from the perspective of fidelity extraction and accuracy extraction. First, we define *fidelity* and *accuracy* of model extraction on GANs. More specifically, when

ACSAC '21, December 6–10, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8579-4/21/12.
https://doi.org/10.1145/3485832.3485838

an adversary mounts model extraction attacks against GANs, *fidelity* measures the difference of data distribution between the attack model and the target model, while *accuracy* ensures the distribution of the attack model is consistent with the distribution of the training set of the target model. In the next step, according to the adversary's goals and the background information that they can have access to (see Figure 2), we systematically study two different types of attacks on GANs: fidelity extraction attack and accuracy extraction attack, which are shown in Figure 1.

Fidelity Extraction Attack. Adversaries mounting fidelity extraction focus on *fidelity* and they aim to steal the distribution of a target model. For this attack, we assume adversaries have no knowledge of the architecture of target models, and they either obtain a batch of generated data that the model owner has publicly released or query the target model to obtain generated data. It can be considered as a black-box fidelity extraction. After obtaining the generated data, adversaries can train a copy of the target GAN model. We study two different target models: Progressive GAN (PGGAN) [31] and Spectral Normalization GAN (SNGAN) [47]. Extensive experimental evaluations show that fidelity extraction can achieve an excellent performance with only about 50K queries (i.e., 50K generated samples). When we continue to increase the number of queries, we find that it cannot bring significant improvement of the *accuracy* of attack models. This is mainly because the discriminator of a target GAN model is often better than its corresponding generator and it is very hard to reach global optimum [3]. In other words, directly querying the target model enables the attack model to be more consistent with the target generator rather than the real data distribution of the target model (see Figure 5 for an example). Therefore, it motivates us to perform accuracy extraction to improve the *accuracy* of attack models.

Accuracy Extraction Attack. Adversaries mounting accuracy extraction concentrate on *accuracy* and they target at stealing the distribution of the training set of a target model. In order to achieve a high accuracy model extraction attack, we propose to utilize subsampling techniques where generated samples far away from the true distribution are rejected and only samples that are closer to the true distribution are retained (see Figure 5). To achieve this goal, we assume that adversaries can obtain more background knowledge. In particular, we assume adversaries can obtain the discriminator from the target GAN model and partial real data. We utilize the discriminator to subsample generated samples. These refined samples are more close to real data distribution, compared to samples are directly generated by the target model (see Figure 5(e)). Then, we use these refined samples and partial real data to train our attack model. Extensive experimental evaluations show that our accuracy extraction attack indeed brings improvement of the *accuracy* of attack models, compared to fidelity extraction attacks (Figure 6). This indicate that the risks of partially releasing training data can be further exacerbated under this type of attack.

Case Study. We perform one case study to further demonstrate the impact of model extraction attacks on a large-scale scenario. In this case study — model extraction based transfer learning (Section 7), we show that stealing a state-of-the-art GAN model can enable adversaries to enhance the performance of their own GAN model by transfer learning. Specifically, for the target model StyleGAN trained on the 3 million bedroom images [71], the adversary first launches fidelity extraction attack, and the attack performance with 4.12 FID on *fidelity* and 6.97 FID on *accuracy* can be achieved under 50K queries. Furthermore, the adversary transfers the extracted knowledge to new domains, and experimental evaluations show that compared with training from scratch on LSUN-Classroom dataset with 20.34 FID [31], model extraction based transfer learning achieves 16.47 FID, which is the state-of-the-art performance on the LSUN-Classroom dataset.

Defenses. Both fidelity extraction and accuracy extraction attacks on GANs compromise the intellectual property of model providers. In particular, accuracy extraction aiming to steal the distribution of the training set of a target model can further severely breach the privacy of the training set. Therefore, we propose possible defense techniques by considering two aspects: *fidelity* and *accuracy* (Section 8). In terms of *fidelity* of model extraction, limiting the number of queries is an effective method. In terms of *accuracy* of model extraction, we believe that a high accuracy attack model requires adversaries to have access to generated data which can be much closer to real data distribution. The performance of model extraction attacks will be attenuated if adversaries only obtain a partial or distorted distribution of generated data. Thus, we propose two types of perturbation-based defense strategies: input and output perturbation-based approaches, to reveal less distribution information by increasing the similarity of samples or lowering the quality of samples [2]. The input perturbation-based approaches include linear and semantic interpolation perturbation while the output perturbation-based approaches include random noise, adversarial example noise, filtering and compression perturbation. Extensive experimental evaluations show that, compared to queries from the prior distribution of the target model, the equal amount of queries by perturbation-based defenses can effectively degrade the *accuracy* of attack models (Figure 8).

Summary of Contributions. Our contributions in the current work are threefold:

(1) we conduct the first systematic study of model extraction attacks against GANs and devise fidelity extraction attacks and accuracy extraction attacks for GANs;
(2) we preform one case study to illustrate the impact of model extraction attacks against GANs on a large-scale scenario;
(3) we propose new effective defense measures to mitigate model extraction attacks against GANs.

Organization. The rest of the paper is organized as following. The next section 2 reviews related work. Section 3 introduces the preliminary knowledge, and Section 4 taxonomizes the space of model extraction attacks on GANs. Section 5 and Section 6 introduce the fidelity extraction and accuracy extraction, respectively. Section 7 presents one case study. In Section 8, we discuss possible defense mechanisms. Section 9 concludes this paper.

2 RELATED WORK

Generative Adversarial Networks (GANs). GANs have achieved impressive performance in a variety of areas, such as image synthesis [6, 31–33, 39, 47, 53, 57], image-to-image translation [40, 52, 73], and texture generation [37, 69], since a

framework of GAN was first proposed by Goodfellow et al. in 2014 [19]. For image synthesis tasks, the current state-of-the-art GANs [6, 31, 32, 47] are able to generate highly realistic and diverse images. For instance, SNGAN [47] generates realistic images by a spectral normalization method to stabilize the training process. PGGAN [31] proposed by Karras et al. is the first GAN that successfully generates real-like face images at a high resolution of 1024×1024, applying a progressive training strategy. Unlike the PGGAN training in an unsupervised method, BigGAN [6] proposed by Brock et al. aims to generate high-quality images from a multi-class dataset by conditional GANs which leverage information about class labels. Recently, StyleGAN [32] has further improved the performance of GANs on high-resolution images through adding neural style transfer [25]. In this paper, *we choose SNGAN and PGGAN as the target models to be attacked by model extraction, considering their impressive performance on image generation. StyleGAN is also used as a target model in a case study in Section 7.*

Model Extraction Attacks. With the availability of machine learning as a service (MLaaS), model extraction attack has received much attention from the research community [9, 14, 27, 34, 64], which aims to duplicate (i.e., 'steal') a machine learning model. This type of attack can be categorized into two classes: accuracy model extraction and fidelity model extraction. In terms of accuracy model extraction, it was first proposed by Tramèr et al. [64], where the objective of the attack is to gain similar or even better performance on the test dataset for the extracted model. Since then, various methods attempting to reduce the number of queries have been developed for further improving the attack efficiency, such as model extraction using active learning [12, 50] or semi-supervised learning [27]. In terms of fidelity model extraction, it requires the attack model to faithfully reproduce predictions of the target model, including the errors which occur in the target model. Typical works include model reconstruction from model explanation [45], functionally equivalent extraction [27] and cryptanalytic extraction [9]. In addition to model extraction attacks on images, there are several work about model extraction in natural language processing [34, 61]. Krishna et al. [61] mount model extraction attacks against BERT-based models and the performance of the extracted model is slightly worse than that of the target model. Overall, these studies mainly focus on discriminative models, such as regression and convolutional neural networks for classification, and recurrent neural networks for natural language processing. *Unlike the existing studies, our work aims to study model extraction attacks against GANs.*

In addition to model extraction attacks, there are other types of attacks in relation to privacy and security [10, 22, 68, 70], such as membership inference attacks [13, 20, 56, 59, 60] and property inference attacks [18]. Some efforts have been also made to investigate membership inference attacks against GANs, where queries to a GAN model can reveal information about the training dataset [13, 20, 24]. Overall, these studies mainly focus on privacy on the training dataset, *while model extraction attacks in our paper concentrate on machine learning model itself.*

Model Extraction Defenses. Defense for model extraction can be broadly classified into two categories: restricting the information returned by models [36, 64] and differentiating malicious adversaries from normal users [30]. Tramèr et al. propose a defense where the model should only return class labels instead of class probabilities [64]. Recently, a technique PRADA has proposed to guard machine learning models by detecting abnormal query patterns [30]. Watermarking ML models as a passive defense mechanism recently has been proposed to claim model's ownership [8, 29, 38]. However, these defense techniques are used to protect discriminative models where models return probabilities or labels. In this paper, *we focus on defense approaches safeguarding generative adversarial networks where models return images.*

3 PRELIMINARIES

In this section, we begin with the general structure of GANs. Then, we proceed with discussing model extraction attacks in a general machine learning setting. Finally, we describe datasets used in this paper.

3.1 Generative Adversarial Networks

GAN is a generative model where it adversarially learns the unknown true distribution p_r on the training data $\mathbf{X}$. As shown in Figure 2, a GAN generally consists of two components: a generator G and a discriminator D. G is responsible for generating fake data $x_g = G(z)$, where the latent code z is sampled from a prior distribution p_z, such as Gaussian distribution or uniform distribution, while D takes the role of a binary classifier which differentiates real-like samples x_g from real samples $x_r \in \mathbf{X}$ as accurately as possible. The seminal GAN [19] is trained through optimizing the following loss functions:

$$L_D = -\mathbb{E}_{x\sim p_r}[\log D(x)] - \mathbb{E}_{z\sim p_z}[1 - \log D(G(z))] \quad (1)$$

$$L_G = -\mathbb{E}_{z\sim p_z}[\log D(G(z))] \quad (2)$$

If D and G converge and reach global equilibrium, then $p_r(x) = p_g(x)$, where $p_g(x)$ is the generator's distribution. For a fixed G, the optimal discriminator D^* can be obtained by:

$$D^*(x) = \frac{p_r(x)}{p_r(x) + p_g(x)} \quad (3)$$

In the course of employment, only G is utilized to produce new synthetic data while D is usually discarded.

3.2 Model Extraction Attacks against Machine Learning Models

A machine learning model is essentially a function f that maps input data $\mathcal{X}$ to output data $\mathcal{Y}$: $\mathcal{Y} = f(\mathcal{X})$. In general, machine learning models can be categorized as two classes [4]: discriminative models and generative models. For discriminative models on image classification tasks, the input data corresponds to an image while the output data can be interpreted as a probability distribution over categorical labels. *A key goal of discriminative models is to find an optimal set of parameters which minimizes the errors on the test dataset.* For generative models on image generation tasks, the input data is represented by a latent code and the output data is an image. *A core goal of generative models is to adjust the parameters to learn a distribution which is similar to the training data distribution p_r.*

A model extraction attack in the machine learning setting emerges when an adversary aims to obtain a copy model $\tilde{f}$ through

Table 1: Dataset description

Dataset	LSUN-Bedroom	LSUN-Kitchen	CelebA
Size of dataset	3,033,042	2,212,277	202,599
Dataset	LSUN-Classroom	LSUN-Church	
Size of dataset	168,103	126,277	

Table 2: Notations

Notation	Description
p_r	distribution of training set of a GAN
p_g	implicit distribution of a target generator
$\tilde{p}_g$	implicit distribution of an attack generator
fidelity	FID ($\tilde{p}_g$, p_g)
accuracy	FID ($\tilde{p}_g$, p_r)

querying the target model f. In general, there are two types of attacks around model extraction based on adversary's objective: accuracy extraction and fidelity extraction [27]. For discriminative models, accuracy extraction requires the extracted model to match or exceed the accuracy of the target model on the test dataset, while fidelity extraction requires the extracted model not only to achieve the same accuracy as the target model on the test dataset but also to replicate the errors of the target model. The limit of fidelity extraction is the functionally-equivalent model extraction [27]. Considering different goals and evaluations between discriminative models and generative models, we redefine model extraction on GANs in Section 4.1.

3.3 Dataset Description

We utilize five different datasets in this paper, which are all widely adopted in image generation. Among them, four datasets are from the LSUN dataset [71] which includes 10 scene categories and 20 object categories and we define them as LSUN-Bedroom, LSUN-Church, LSUN-Classroom, and LSUN-Kitchen, respectively. CelebA dataset [41] consists of about 200K high-quality human face images. Datasets including LSUN-Bedroom, LSUN-Classroom, and LSUN-Kitchen are only used in Section 7 to illustrate the attack effects in a case study. The details of the datasets are shown in Table 1.

4 TAXONOMY OF MODEL EXTRACTION AGAINST GANS

In this section, we start with adversary's goal and formally elaborate on our attacks. Next, we illustrate adversary's background knowledge where an adversary can mount attacks according to the obtained information. Finally, we detail the metrics to evaluate the attack performance.

4.1 Adversary's Goals

In general, model extraction based on adversary's goals can be categorized into either fidelity extraction or accuracy extraction. Unlike supervised discriminative models aiming at minimizing errors on a test set, unsupervised generative models target at learning the distribution of a data set.

Therefore, for model extraction attacks on GANs, fidelity extraction aims to minimize the difference of data distribution between attack models and target models, while accuracy extraction aims to minimize the distribution between attack models and the training set of target models.

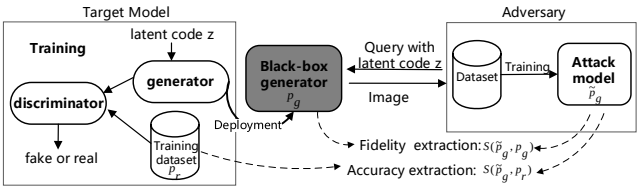


Figure 1: Fidelity extraction and accuracy extraction.

Specifically, as shown in Figure 1, the goal of fidelity extraction is to construct a $\tilde{G}$ minimizing $S(\tilde{p}_g, p_g)$, where S is a similarity function, $\tilde{p}_g$ is the implicit distribution of the attack generator $\tilde{G}$, and p_g is the implicit distribution of the target generator G. In contrast, accuracy extraction's goal is to construct a $\tilde{G}$ minimizing $S(\tilde{p}_g, p_r)$, where p_r is the distribution of the training set of the target generator G. In this work, we use Fréchet Inception Distance (FID) to evaluate the similarity between two data distributions, mainly considering its computational efficiency and robustness [23]. It is elaborated in Section 4.3. In our work, we study the fidelity extraction in Section 5, and accuracy extraction in Section 6.

4.2 Adversary's Background Knowledge

Adversaries can mount model extraction attacks at different levels based on their obtained information about the target GAN. The more background knowledge adversaries acquire, the more effective they should be in achieving their goal. In general, four components of a GAN can be considered by an adversary. As shown in Figure 2, they are respectively: (1) generated data; (2) latent codes used by interactively querying a generator; (3) partial real data from the training dataset of the target GAN; (4) a discriminator from the target GAN.

In the following attack settings, we assume an adversary obtains different levels of background knowledge to achieve accuracy extraction or fidelity extraction.

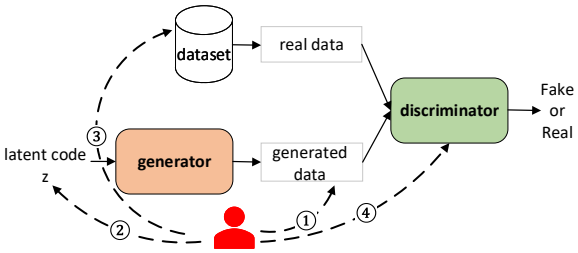


Figure 2: Adversary's background knowledge.

4.3 Metrics

Metrics for GANs. We use the widely adopted FID [23] to evaluate the performance of GANs. FID measures the similarity between p_g and p_r. Specifically, on the basis of features extracted by the pretrained Inception network ϕ, it models $\phi(p_r)$ and $\phi(p_g)$ using Gaussian distribution with mean μ and covariance Σ, and the value of FID between real data p_r and generated data p_g in convolutional

features is computed as: $FID(p_r, p_g) = \left\|\mu_r - \mu_g\right\|^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2})$, where Tr refers to the trace of a matrix in linear algebra. A lower FID indicates that the distribution's discrepancy between the generated data and real-world data is smaller and the generated data is more realistic. In our work, FID is computed with all real samples and 50K generated samples.

Metrics for Attack Performance. In this work, we use two FID-based metrics: *fidelity* and *accuracy*, to evaluate the attack performance. *Fidelity* measures the consistency between p_g which is an implicit distribution of a target generator and $\tilde{p}_g$ which is an implicit distribution of an attack generator. Note that, *fidelity* not only measures how close the attack model and the target model are, but also indicates how well the performance of model itself is. In contrast, *accuracy* measures the consistency of data distribution between p_r and $\tilde{p}_g$. Similar to FID, the smaller the *fidelity* and *accuracy* values are, the better performance attack models achieve. When it is clear from the context, we refer to *accuracy* and *fidelity* as *accuracy* value and *fidelity* value, respectively. The summarized notations can be seen in Table 2.

Fidelity extraction focuses on *fidelity* and adversaries aim to steal the distribution of a target model. After obtaining an attack model which steals from a target model, they can directly utilize it to generate new samples. Additionally, they can also transfer knowledge of the stolen model to their own domains through transfer learning. In contrast, accuracy extraction concentrates on *accuracy* and adversaries target at stealing the distribution of the training set of a target model. This type of attacks can severely violate the privacy of the training data and it also means that adversaries may steal valuable commercial datasets from a trained GAN. Additionally, adversaries can utilize the stolen high-accuracy model to mount other novel attacks and we leave it for future work.

5 FIDELITY EXTRACTION

In this section, we instantiate our fidelity extraction attack strategy. we assume that adversaries have access to either generated samples provided by the model producer or querying the target model to obtain data (see Figure 2). We start with target models and attack models. Then, we describe our attack performance. Next, we study the effect of the number of queries. In the end, we perform experiments to deeply understand model extraction on GANs.

5.1 Target Models and Attack Models

We choose representative GANs: Progressive GAN (PGGAN) [31] and Spectral Normalization GAN (SNGAN) [47] as our target models, which both show pleasing performances in image generation. The implementation details can be seen in Appendix A.1. For training sets LSUN-Church and CelebA, we first resize them to 64×64 and use all records of each dataset to train our target models. As shown in Table 3, target GAN models achieve an excellent performance on these dataset and the performance of PGGAN is better than that of SNGAN.

We use GANs as our attack models to extract target models. In practice, adversaries may not know the target model's architecture. Therefore, we study the performance of attack models with different architectures. Specifically, we choose SNGAN and PGGAN as our attack models. There are four different situations for their combinations. For simplification, we define each situation as an attack-target model pair, and they are respectively SNGAN-SNGAN, SNGAN-PGGAN, PGGAN-SNGAN and PGGAN-PGGAN. The reason why we choose SNGAN and PGGAN as the research object is that: 1) they both show good performance in image generation; and 2) they have significant difference in the aspects of training, loss function and normalization, which all facilitate us to study the performance of attack models with different architectures.

Table 3: Performance of target GANs.

Target model	Dataset	FID
SNGAN	LSUN-Church	12.72
SNGAN	CelebA	7.60
PGGAN	LSUN-Church	5.88
PGGAN	CelebA	3.40

5.2 Methodology

As shown in Figure 1, for fidelity extraction, we assume that an adversary obtains the generated data by the model provider or querying the target GAN. This scenario is practical, because some model owners need to protect their models through providing the public with some generated data or a black-box GAN model API. In this case, the adversary uses the generated data to retrain a GAN to extract the target model. We do not distinguish whether generated data is from queries or model providers, because our approach only relies on these generated data. However, in Appendix A.3, we also present the attack performance on queries with different prior distributions.

Note that model extraction on GANs is different from machine learning on GANs. This is because machine learning on GANs requires users to train a GAN on real samples which are collected from the real world. In contrast, model extraction on GANs enables users to train a GAN on generated data from a target GAN model. In essence, model extraction on GANs approximates the target GAN which is a much simpler deterministic function, compared to real samples which usually represents a more complicated function.

5.3 Results

5.3.1 ***Attack Performance on Different Models.*** Table 4 shows the fidelity extraction's performance with 50K queries to the target model. In general, attack models can achieve an excellent performance[1]. For instance, our attack performance of PGGAN-PGGAN on the CelebA achieves 1.02 FID on *fidelity*, which means that the attack model can achieve a perfect extraction attack for the target model. It is noticeable that the the attack model achieves such performance only on 50K generated images while the target model is trained on more than 200K images. In Section 7, our case study further illustrates that even for a GAN model trained on 3 million samples, our attack still can achieve 4.12 *fidelity* with only 50K queries. In other words, adversaries are able to obtain a good GAN model only by access to the generated data from the target model

[1]We say model extraction attacks achieve an excellent performance because we choose the state-of-the-art StyleGAN [32] trained on the LSUN-Bedroom dataset as a reference, where it has the lowest FID 2.65.

Table 4: The performance of fidelity extraction with 50K queries to the target model.

Target model	Attack model	Dataset	*Fidelity* FID($\tilde{p}_g$, p_g)	*Accuracy* FID ($\tilde{p}_g$, p_r)
PGGAN	SNGAN	LSUN-Church	6.11	14.05
	SNGAN	CelebA	4.49	9.29
	PGGAN	LSUN-Church	1.68	8.28
	PGGAN	CelebA	1.02	4.93
SNGAN	SNGAN	LSUN-Church	8.76	30.04
	SNGAN	CelebA	5.34	17.32
	PGGAN	LSUN-Church	2.21	14.56
	PGGAN	CelebA	1.39	9.57

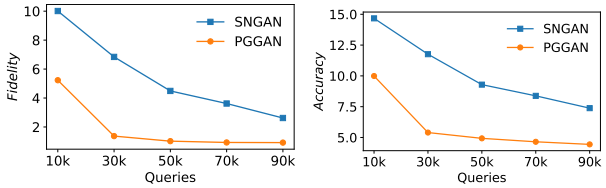
instead of collecting their own data which is usually labor-intensive and time-consuming.

For the target model PGGAN, if the attack model is SNGAN, we observe that the performance of model extraction is very efficient on both CelebA and LSUN-Church dataset and the attack model SNGAN can learn more from the target model PGGAN, compared to the SNGAN-SNGAN case, which indicates that attacking a state-of-the-art GAN is valuable and viable for an adversary. Furthermore, this case SNGAN-PGGAN is the most common situation in the actual attack scenarios, because generally we implicitly assume that performance of the adversary's model may often be weaker than that of the target model and the structure of the attack model is inconsistent with that of the target model.

We also report *accuracy* in Table 4 and find that for model extraction on GAN models, the *accuracy* of attack models is always higher than that of target model, in which *accuracy* of attack models represents similarity between distribution of real dataset p_r and distribution of the attack model $\tilde{p}_g$ and for *accuracy* of a target model, also called FID of target model, it represents similarity between distribution of real dataset p_r and distribution of the target model p_g. For example, when the target model SNGAN has 12.72 FID on the LSUN-Church dataset, *accuracy* of the attack model SNGAN will increase to 30.04. Even for the PGGAN-PGGAN case, its *accuracy* increases from 3.40 to 4.93 on the CelebA dataset. This is mainly because although theoretically, the distribution of the target model p_g is equal to that of the real training dataset p_r, it is actually not equal because GAN cannot achieve the global optimum. However, we will discuss how to reduce *accuracy* values and achieve high accuracy extraction in Section 6.

For the target model SNGAN, if the attack model is PGGAN, the *fidelity* of model extraction is lower than that of the attack model SNGAN. It is mainly because the PGGAN model itself is stronger and able to more accurately approximate the target model. Similarly, PGGAN as an attack model has more lower *accuracy*, in contrast with SNGAN as an attack model. For instance, compared to SNGAN-SNGAN with 17.32 of *accuracy* on CelebA dataset, the *accuracy* of PGGAN-SNGAN is only 9.57, which largely improves the attack performance on accuracy. This indicates that using an attack model which is larger than the target model is an efficient approach to improve attack performance.

Overall, fidelity extraction can achieve an excellent performance in terms of *fidelity*. In general, adversaries can steal an fidelity model, and then use the extracted model for their own purpose. However, unlike discriminative models where adversaries can directly utilize their extracted model, the extracted model of a GAN only generates target model's images. Therefore, in Section 7, we will perform a case study where adversaries can effectively leverage the extracted model to generate images for their own applications rather than target GANs' images through transfer learning.



(a) *Fidelity* on CelebA (b) *Accuracy* on CelebA

Figure 3: Attack performance on the number of queries.

5.3.2 ***Attack Performance on the Number of Queries.*** We choose PGGAN trained on CelebA dataset as the target model to study the effect of the number of queries due to the best performance among our target models. Figure 3 plots the attack performance with respect to the number of queries which are also the size of training dataset of attack models. As expected, we observe that the attack performance increases with an increase in the number of queries. This indicates that releasing a small number of data by the model owner or restricting the number of queries is a relatively safe measure.

We estimate the monetary cost of the number of queries. Taking the Google Cloud Vision API[2] as an example, the price is $1.50 per 1K queries with the first 1K queries are free for each month. Thus, the price of the number of queries from 10K to 90K is from $13.50 to $133.50. Although the attack cost is not high in our attacks, designing a more powerful attack to reduce the number of queries is still an interesting research direction. We leave it as future work.

5.3.3 ***Understanding Fidelity Extraction on GANs In-depth.*** We further dissect the difference of distributions between target models and attack models to understand the nature of model extraction on GANs. Specifically, we first transform the training data into 2048-dimension feature vectors by the pretrained Inception-v3 model[3] which is widely utilized in the evaluation of a GAN model [23]. Then these feature vectors are clustered into k classes by a standard K-means algorithm. Finally, we calculate the proportions of each class, which can be also considered as a distribution of the training data [5, 54]. The blue bar in Figure 4 shows the distribution of the training data where we set k to 30. For target models and attack models, we query the model to obtain 50K images, then perform the same procedures as the training data.

Figure 4 shows distribution differences among the training data, the target model PGGAN and attack models. We observe that for the high proportions of classes, which can be considered as prominent features of a distribution, target models can learn more features about these classes while attack models further learn more features by querying the target models. In contrast, for the low proportions of classes, target models learn less features about these classes while

[2]https://cloud.google.com/vision/pricing
[3]https://pytorch.org/hub/pytorch_vision_inception_v3/

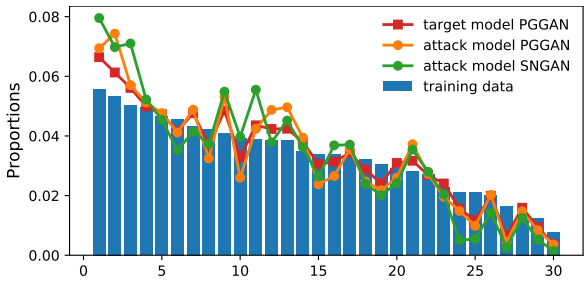


Figure 4: Class distribution differences among the training data, the target model PGGAN, and attack models.

Table 5: JS distances between models. A smaller value indicates a better performance. The JS distance between the training data and the target model PGGAN is 4.14×10^{-3}. The JS values below show a consistent trend with Figure 4.

Target model	Attack model	$JS_{fidelity}$ ($\times10^{-3}$)	$JS_{accuracy}$ ($\times10^{-3}$)
PGGAN	SNGAN	5.88	15.95
	PGGAN	1.83	9.10

attack models further learn less features about these classes. This is one reason why attack models always have higher *accuracy* values than target models. In terms of *fidelity*, we observe that there is a consistent trend on proportions of classes for target models and attacks models. This is the reason why we can achieve a satisfying performance about *fidelity*. We also analyze the target model SNGAN, and similar results are shown in Figure 9 in Appendix.

We also summarize this difference in a single number by computing the Jensen-Shannon (JS) divergence on this representation of distributions, which is shown in Table 5. Note that, based on *accuracy* and *fidelity* defined in Section 4.1, we mark $JS_{fidelity}$ as the JS divergence between the target model and the attack model, and $JS_{accuracy}$ as the JS divergence between the training data and the attack model.

6 ACCURACY EXTRACTION

In this section, we instantiate our accuracy extraction attack strategy. In addition to fidelity extraction's assumptions, we also assume that adversaries have more background knowledge in order to achieve accuracy extraction, such as partial real data and the target model's discriminator. We start with the motivation and problem formulation of accuracy extraction. Then, we describe the methodology of accuracy extraction. In the end, we present the performance of accuracy extraction.

6.1 Motivation and Problem Formulation

As shown in Figure 1, fidelity extraction can be implemented through querying the generator of the target GAN, because p_g is the generator's distribution. As for accuracy extraction, it is much more difficult due to the lack of availability of real data distribution p_r. Although an approach is to use p_g as an approximation of p_r, we observe that with the increase in the number of queries, *accuracy* of attack models reaches its saturation point and is hard to be improved, which is shown in Figure 3(b). For instance, as we increase the number of queries from 50K to 90K for the PGGAN-PGGAN case on CelebA dataset, *accuracy* of the attack model has smaller and smaller improvements from 4.93 to 4.44, while the ideal *accuracy* is 3.40 which is also the performance of the target model. Note that the case PGGAN-PGGAN is the best for the attacker; the attack will perform even worse if the attackers do not choose the same architectures and hyperparameters as the target model.

The reason why there exists a gap between the attack model and the target model in terms of *accuracy* is that the target GAN model is hard to reach global equilibrium and the discriminator is often better than the generator in practice [3]. As a result, real data distribution p_r is not completely learned by the generator of the target model, which means that $p_g \neq p_r$. Therefore, directly using the generator's distribution p_g does not guarantee the high accuracy and it only minimizes the distribution discrepancy between the attack model and the target model. We explain this by a simple example on Figure 5, which is popular in the GAN literature [3, 16, 65].

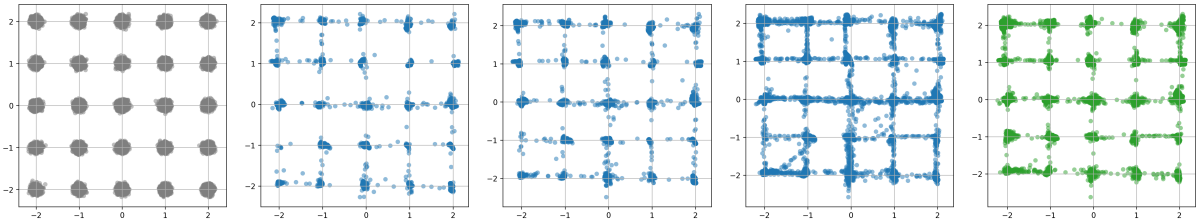
Figure 5(a) presents real samples drawn from a mixture of 25 two-dimensional Gaussian distributions (each with standard deviation σ of 0.05). Figure 5(b) - Figure 5(d) show samples which are generated by a target GAN with different queries. We define a generated sample as "high-quality" if its Euclidean distance to its corresponding mixture component is within four standard deviations ($4\sigma = 0.2$) [3]. The architecture and setup information of the target GAN is shown in Appendix A.1. Overall, we can observe that target GAN's distribution is not completely the same as the training set's distribution, which means that directly extracting a model from the generator of the target GAN makes its distribution similar to the target model's distribution rather than its training dataset's distribution.

Therefore, a natural approach to achieving accuracy extraction is that the adversary can get more high-quality samples that are closer to the real data distribution.

6.2 Methodology

Our approach to obtaining high-quality samples is based on subsampling. The key insight here is that we can reject some poor samples from generated samples based on some prior knowledge. In order to achieve it, we suppose that adversaries can obtain additional background information. This is a common assumption that can be found in many works in relation to the security and privacy of machine learning [20, 28, 60]. As shown in Figure 2, we assume that adversaries can have limited auxiliary knowledge of the discriminator of the target model and partial training samples. This is because the discriminator from the target model can reveal the distribution information of the training data [3]. Thus, using the information provided by the discriminator, we can subsample the generated data to make the obtained data closer to the real dataset's distribution, which improves accuracy extraction.

Specifically, for accuracy extraction, we first leverage the discriminator of the target model to subsample the generated samples. As a result, these refined samples are much closer to the true distribution. In this work, we use Metropolis-Hastings (MH) subsampling algorithm [65] to subsample the generated data. See Algorithm 1 in Appendix for details. MH subsampling algorithm utilizes the discriminator through Metropolis-Hastings algorithm [62] to refine



(a) Real Samples 50K (b) Direct Sampling 5k (c) Direct Sampling 10K (d) Direct Sampling 50K (e) MH Subsampling 50K

Figure 5: Difference of distribution between training data and generators. The percentage of "high-quality" samples for Figure 5(b), Figure 5(c), Figure 5(d) and Figure 5(e) is 94.36%, 94.31%, 94.15% and 95.64%, respectively. The more we query, the more bad-quality samples we obtain, which affects the performance of model extraction. But if we reduce the number of queries, the performance of attack models still be poor due to insufficient training samples.

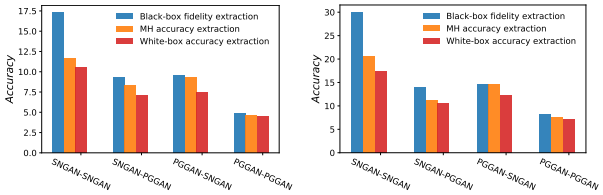
samples which are generated by the generator. The discriminator generally needs to be calibrated by partial real samples from training set of the target GAN model, considering that some discriminators of GANs output a score rather than a probability. In our experiments, all discriminators are calibrated through logistic regression. Then we train the attack model on those refined samples. After the training process of the attack model is stable, we add partial real data to further train the attack model.

In this scenario, although the number of queries will increase due to subsampling samples, we assume that adversaries eventually obtain 50K refined samples in order to make a comparison with fidelity extraction. Partial real samples used to calibrate the discriminator are fixed to 10% of training data. In addition, these partial real samples will be added into training process of the attack models. Here, we refer the former where only refined samples are used to train the attack model to MH accuracy extraction which is also considered as an indicator to show how well these refined samples are beneficial to *accuracy*. We refer the latter where both refined samples and partial real data are used to train the attack model to white-box accuracy extraction. We refer fidelity attack in Section 5 to black-box fidelity extraction.

It is worth noting that we cannot directly choose the lowest *accuracy* value in real attack scenarios due to unavailability of training dataset from target models. Therefore, the *accuracy* value reported in this paper is chosen when its corresponding *fidelity* value is the lowest in the training process.

6.3 Results

Figure 6 plots not only the results of the MH accuracy extraction and the white-box accuracy extraction on both CelebA and LSUN-Church datasets, but also the black-box fidelity extraction for comparison. We can observe that MH subsampling is an effective approach to improve *accuracy* of attack models. For example, when target model is SNGAN, the MH accuracy extraction can significantly improve attack model's accuracy on both datasets because MH subsampling algorithm selects high-quality samples from generated samples of the target model SNGAN. For the MH accuracy extraction and the white-box accuracy extraction which both leverage the refined samples in the training process, the white-box accuracy extraction can further improve *accuracy*. This is because partial real data can further correct the distribution of the attack



(a) *Accuracy* on CelebA (b) *Accuracy* on LSUN-Church

Figure 6: Comparison on *accuracy* for different attack approaches.

model and make it closer to the real distribution. Similar to fidelity extraction in Section 5.3.3, we also analyze distribution differences for accuracy extraction, which is shown in Figure 10 in Appendix.

7 CASE STUDY: MODEL EXTRACTION BASED TRANSFER LEARNING

In this section, we present one case study where the extracted model serves as a pretrained model and adversaries transfer knowledge of the extracted model to new domains by means of fine-tuning to broaden the scope of applications based on extracted models. We start with methods of transfer learning on GAN and demonstrate how adversaries can benefit from model extraction, in addition to directly leveraging the extracted model to generate images.

We consider the state-of-the-art GAN model StyleGAN [32] that was trained on more than 3 million bedroom images as the target model. StyleGAN produces high-quality images at a resolution of 256×256, with 2.65 FID on LSUN-Bedroom dataset [71]. We suppose adversaries only query the target model StyleGAN and have no any other background knowledge, which is also called black-box fidelity extraction in our paper. Although an adversary can obtain an extracted model, the model only generates images which are similar to the target model. In this case, the extracted model can only generate bedroom images due to target model trained on LSUN-Bedroom dataset. Therefore, the adversary's goal is to use the PGGAN as the attack model to extract the target model StyleGAN and leverage transfer learning to obtain a more powerful GAN which generates images that the adversary wishes. *The attack is successful if the performance of models training by transfer learning based on the extracted GAN outperforms models training from scratch.*

Table 6: Comparison between transfer learning based on model extraction and training from scratch. The target model is StyleGAN trained on LSUN-Bedroom dataset, and the attack model uses PGGAN.

Target dataset	Methods	FID
LSUN-Kitchen	Transfer Learning	7.59
LSUN-Kitchen	Training from Scratch	8.83
LSUN-Classroom	Transfer Learning	16.47
LSUN-Classroom	Training from Scratch	20.34

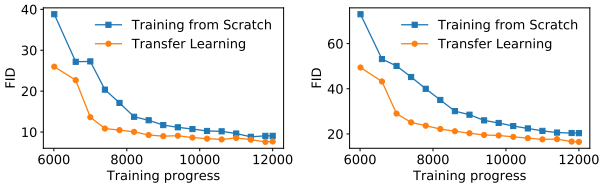
Transferring knowledge of models which steal the state-of-the-art models to new domains where adversaries wish the GAN model can generate other types of images can bring at least two benefits: 1) if adversaries have too few images for training, they can easily obtain a better GAN model on limited dataset through transfer learning; 2) even if adversaries have sufficient training data, they can still obtain a better GAN model through transfer learning, compared with a GAN model training from scratch. Therefore, we consider two variants of this attack: one where the adversary owns a small target dataset (i.e., about 50K images in our work) and the other one where the adversary has enough images (i.e., about 168k images in our work).

More specifically, after querying the target model StyleGAN and obtaining 50K generated images, adversaries train their attack model PGGAN on the obtained data, as illustrated in Section 5.2. Here, *fidelity* of the attack model PGGAN is 4.12 and its *accuracy* is 6.97. Then, we use the extracted model's weights as an initialization to train a model on adversary's own dataset which is also called target dataset in the section. We conduct the following two experiments:

(1) We first randomly select 50K images from LSUN-Kitchen dataset as a limited dataset. Then, we train the model on these selected data by transfer learning and from scratch, respectively.
(2) We train a model on the LSUN-Classroom dataset including about 168k images by transfer learning and from scratch, respectively.

Results. Table 6 shows the performance of models trained by transfer learning and training from scratch. We can observe that the performance of training by transfer learning is always better than that of training from scratch on both large and small target dataset. To be specific, on the limited LSUN-Kitchen dataset which contains 50K images, the FID of model trained by transfer learning decreases from 8.83 to 7.59, compared with the model trained from scratch. It indicates that the extracted model is useful for models trained on other types of images. On the large LSUN-Classroom dataset which contains more than 168k classroom images, the performance of model significantly improves from model training from scratch with 20.34 FID[4] to training by transfer learning with 16.47 FID. This is also the best performance for PGGAN on LSUN-Classroom dataset, in contrast with 20.36 FID reported by Karras et al. [31]. We also plot the process of training for both settings on the two

[4]This value is not equal to 20.36 [31] due to randomness.



(a) FID on LSUN-Kitchen (b) FID on LSUN-Classroom

Figure 7: Comparison between transfer learning based on model extraction and training from scratch on LSUN-Kitchen and LSUN-Classroom dataset.

datasets, which is shown in Figure 7. We can obviously and consistently observe that training by transfer learning based on model extraction is always better than training from scratch during the training process, which indicates that the extracted model PGGAN which duplicates the state-of-the-art StyleGAN on LSUN-Bedroom dataset can play a significant role in other applications rather than only on generating bedroom images. That reminds us that model extraction on GANs severely violates intellectual property of the model owners.

8 DEFENSES

Model extraction attacks on GANs leverage generated samples from a target GAN model to retrain a substitutional GAN which has similar functions to the target GAN. In this section, we introduce defense techniques to mitigate model extraction attacks against GANs.

According to adversary's goals as defined in Section 4.1, we discuss defense measures from two aspects: *fidelity* and *accuracy*. In terms of *fidelity* of model extraction, it is difficult for model owners to defend except for limiting the number of queries. This is because adversaries can always design an attack model to learn the distribution based on their obtained samples. The more generated samples adversaries obtain, the more effective they achieve.

In terms of *accuracy* of model extraction, its effectiveness is mainly because adversaries are able to obtain samples generated by latent codes draw from a prior distribution of the target model, and these samples generated through the prior distribution are close to real data distribution [2]. However, if adversaries obtain some generated samples which are only representative for partial real data distribution or a distorted distribution, *accuracy* of attack models becomes poor. Based on this, we propose two types of perturbation-based defense mechanisms: input perturbation-base and output perturbation-based approaches. In the rest of this section, we focus on defense approaches which are designed to mitigate *accuracy* of attack models.

8.1 Methodology

8.1.1 Input Perturbation-base Defenses. For this type of defenses, we propose two approaches based on perturbing latent codes: linear interpolation defense and semantic interpolation defense.

Linear Interpolation Defense. For n latent codes queried from users, model providers randomly select two queried points and interpolate k points between the two points. This process is repeated for $\lceil n/k \rceil$ times to get n modified latent codes. These modified latent

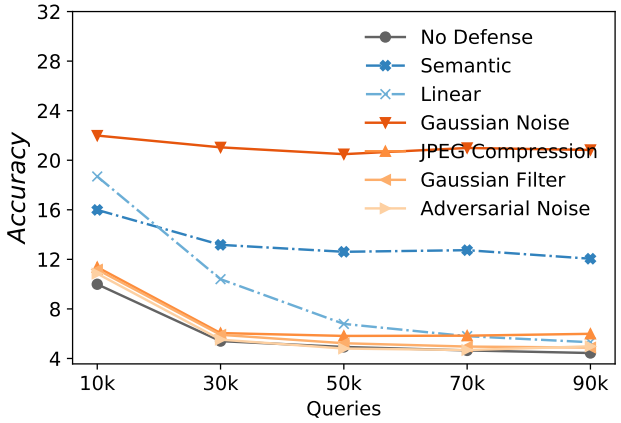


Figure 8: The performance of attack model PGGAN under various defenses.

codes are used to query the target model. In our experiments, we interpolate 9 points. See Figure 14(a) in Appendix for visualization.

Semantic Interpolation Defense. Unlike linear interpolation defense where target models return a batch of random images, semantic interpolation defense returns various semantic images that are predefined by model providers, which restricts the space of images the adversary queries. Generally, semantic information can be any information that humans can perceive. For instance, for a human face image, it includes gender, age and hair style. We adopt the semantic interpolation algorithm proposed by Shen et al. [58]. The details of this defense are presented in Appendix A.5.1. In our experiments, we totally explore 12 semantic information on CelebA dataset. See Figure 14(b) in Appendix for visualization.

8.1.2 Output Perturbation-base Defenses. Instead of perturbing latent codes, this type of defenses directly perturbs the generated samples. Specifically, we propose four approaches: random noise, adversarial noise, filtering and compression. See Figure 15 in Appendix for visualization.

Random Noise. Adding random noises on generated samples is a straightforward method. In our experiments, we use Gaussian-distributed additive noises (mean = 0, variance = 0.001).

Adversarial Noise. We generate adversarial examples through mounting targeted attacks where all images are misclassified into a particular class by the classifier ResNet-50 trained on ImageNet dataset. In our experiments, all face images are misclassified into the class — goldfish and the C&W algorithm [11] based on L_2 distance are used.

Filtering. The Gaussian filter is used to process generated samples. In our experiments, we use Gaussian filter (sigma = 0.4) provided by the skimgae package [66].

Compression. The JPEG compression algorithm is used to process generated samples. In our experiments, we use the JPEG compression (quality = 85) provided by the simplejpeg package [17].

8.2 Results

In this experiment, we choose PGGAN trained on CelebA dataset as the target model to evaluate our defense techniques, considering its excellent performance among our target models. We only show the effectiveness of defense techniques on black-box fidelity extraction, considering its more practical assumption: adversaries obtain samples by model providers or queries.

8.2.1 ***Defense on Black-box Fidelity Extraction.*** Figure 8 plots results of attack model PGGAN on defenses. We observe that attack performance is weakened when the target model PGGAN uses these defense approaches, compared to the target model without any defenses. Gaussian noise and semantic interpolation defenses show stable performance while other defense techniques' performance is gradually weakened with an increase in the number of queries. Figure 12 in Appendix also shows similar defense performance for the attack model SNGAN. We further evaluate the defense utility, i.e. the quality of generated images after deploying defense measures. Our quantitative and qualitative measures show that these defense techniques do not impact the visual quality of generated images (see Figure 14, Figure 15 and Table 10). More details are shown in Appendix A.5.2.

8.2.2 ***Discussion.*** The reason why input perturbation-based defenses can work is at least explained from two aspects: increasing the similarity of generated samples and a distribution mismatch between latent codes produced by interpolation and drawn from prior distribution. For the former, we can see that interpolation operations increase the similarity of images from Figure 14. For the latter, latent codes produced by interpolation operations are different from latent codes drawn from the prior distribution that the target model was trained on. This is because latent codes produced by linear operation do not obey the prior distribution of the target model, which also bring a benefit in disguising the true data distribution [2].

Output perturbation-based defenses can work because they directly perturb these generated samples. In practice, this type of defense requires model providers to trade-off image quality and the model's security through magnitudes of changes. Although Gaussian noise defense shows the best performance, it is possible for adversaries to remove noise.

9 CONCLUSION

In this paper, we have systematically studied the problem of model extraction attacks on generative adversarial networks, and devised, implemented, and evaluated this attack from the perspective of fidelity extraction and accuracy extraction. For fidelity extraction, extensive experimental evaluations show that adversaries can achieve an excellent performance with about 50K queries. For accuracy extraction, adversaries further improve the accuracy of attack models after obtaining additional background knowledge, such as partial real data from the training set or the discriminator of the target model. Furthermore, we have also performed a case study where the attack model which steals a state-of-the-art target model can be transferred to new domains to broaden the scope of applications based on extracted models.

These effective attacks also motivate us to design two types of defense techniques: input and output perturbation-based defense. They mitigate model extraction attacks through perturbing latent codes and generated samples, receptively. Extensive experimental

evaluations show that semantic interpolation and Gaussian noise defenses achieve stable performance.

Finally, we also identify a number of directions for future work. Because GAN models generally are considered as intellectual properties of model owners, protecting GANs through verifying the ownership is an interesting direction. In addition, stealing a GAN model also means the leakage of distribution of the training set. Therefore, training with differential privacy techniques can be utilized to protect the privacy of training data of a model [1]. However, training time and stability of the training process are big challenges for GANs. For further work, we plan to design new methods based on differential privacy techniques to mitigate *accuracy* of model extraction.

ACKNOWLEDGMENTS

This work is supported by the National Research Fund, Luxembourg (Grant No. 13550291).

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 308–318.

[2] Eirikur Agustsson, Alexander Sage, Radu Timofte, and Luc Van Gool. 2019. Optimal Transport Maps For Distribution Preserving Operations on Latent Spaces of Generative Models. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[3] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. 2019. Discriminator Rejection Sampling. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[4] Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S. Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. 2020. Statistical Mechanics of Deep Learning. *Annual Review of Condensed Matter Physics* 11, 1 (2020), 501–528.

[5] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. 2019. Seeing what a gan cannot generate. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 4502–4511.

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2019. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[7] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[8] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2021. IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. In *Proceedings of ACM Asia Conference on Computer and Communications Security (ASIA CCS)*. 14–25.

[9] Nicholas Carlini, Matthew Jagielski, and Ilya Mironov. 2020. Cryptanalytic Extraction of Neural Network Models. In *Proceedings of Annual International Cryptology Conference (CRYPTO)*. Springer.

[10] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting Training Data from Large Language Models. In *Proceedings of USENIX Security Symposium (USENIX Security)*. USENIX Association, 2633–2650.

[11] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*. IEEE, 39–57.

[12] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. 2020. Exploring Connections Between Active Learning and Model Extraction. In *Proceedings of USENIX Security Symposium (USENIX Security)*. USENIX Association.

[13] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. 2020. Gan-leaks: A taxonomy of membership inference attacks against generative models. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 343–362.

[14] Kangjie Chen, Shangwei Guo, Tianwei Zhang, Xiaofei Xie, and Yang Liu. 2021. Stealing Deep Reinforcement Learning Models for Fun and Profit. In *Proceedings of ACM Asia Conference on Computer and Communications Security (ASIA CCS)*. 307–319.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[16] Xin Ding, Z Jane Wang, and William J Welch. 2020. Subsampling Generative Adversarial Networks: Density Ratio Estimation in Feature Space With Softplus Loss. *IEEE Transactions on Signal Processing* 68 (2020), 1910–1922.

[17] Joachim Folz. 2020. simplejpeg 1.4.0. https://gitlab.com/jfolz/simplejpeg

[18] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. 2018. Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 619–633.

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2672–2680.

[20] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. 2019. LOGAN: Membership inference attacks against generative models. In *Proceedings on Privacy Enhancing Technologies*, Vol. 2019. Sciendo, 133–152.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 770–778.

[22] Yingzhe He, Guozhu Meng, Kai Chen, Xingbo Hu, and Jinwen He. 2019. Towards Privacy and Security of Deep Learning Systems: A Survey. *arXiv preprint arXiv:1911.12562* (2019).

[23] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 6626–6637.

[24] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. 2019. Monte carlo and reconstruction membership inference attacks against generative models. In *Proceedings on Privacy Enhancing Technologies*, Vol. 2019. Sciendo, 232–249.

[25] Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 1501–1510.

[26] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. 2017. Stacked generative adversarial networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 5077–5086.

[27] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. 2020. High Accuracy and High Fidelity Extraction of Neural Networks. In *Proceedings of USENIX Security Symposium (USENIX Security)*. USENIX Association.

[28] Shouling Ji, Weiqing Li, Neil Zhenqiang Gong, Prateek Mittal, and Raheem A Beyah. 2015. On Your Social Network De-anonymizablity: Quantification and Large Scale Evaluation with Seed Knowledge. In *Proceedings of Network and Distributed Systems Security Symposium (NDSS)*. Internet Society.

[29] Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. 2021. Entangled watermarks as a defense against model extraction. In *Proceedings of USENIX Security Symposium (USENIX Security)*. USENIX Association, 1937–1954.

[30] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. 2019. PRADA: protecting against DNN model stealing attacks. In *Proceedings of IEEE European Symposium on Security and Privacy (Euro S&P)*. IEEE, 512–527.

[31] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2018. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[32] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 4401–4410.

[33] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and improving the image quality of stylegan. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 8110–8119.

[34] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. 2020. Thieves on Sesame Street! Model Extraction of BERT-based APIs. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[35] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (2020), 1234–1240.

[36] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. 2019. Defending against model stealing attacks using deceptive perturbations. In *Proceedings of IEEE Security and Privacy Workshops*. IEEE, 43–49.

[37] Chuan Li and Michael Wand. 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *Proceedings of European conference on computer vision (ECCV)*. Springer, 702–716.

[38] Huiying Li, Emily Wenger, Ben Y Zhao, and Haitao Zheng. 2019. Piracy Resistant Watermarks for Deep Neural Networks. *arXiv preprint arXiv:1910.01226* (2019).

[39] Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, and Hwann-Tzong Chen. 2019. COCO-GAN: generation by parts via conditional coordinating. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 4512–4521.

[40] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. 2019. Few-shot unsupervised image-to-image translation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 10551–10560.

[41] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 3730–3738.

[42] Daniel Lowd and Christopher Meek. 2005. Adversarial learning. In *Proceedings of ACM SIGKDD international conference on Knowledge discovery in data mining (KDD)*. ACM, 641–647.

[43] Mario Lučić, Michael Tschannen, Marvin Ritter, Xiaohua Zhai, Olivier Bachem, and Sylvain Gelly. 2019. High-Fidelity Image Generation With Fewer Labels. In *Proceedings of International Conference on Machine Learning (ICML)*. 4183–4192.

[44] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. 2018. Exploring the Limits of Weakly Supervised Pretraining. In *Proceedings of European conference on computer vision (ECCV)*. Springer, 181–196.

[45] Smitha Milli, Ludwig Schmidt, Anca D Dragan, and Moritz Hardt. 2019. Model reconstruction from model explanations. In *Proceedings of Conference on Fairness, Accountability, and Transparency*. ACM, 1–9.

[46] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. 2012. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters* 20, 3 (2012), 209–212.

[47] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. Spectral Normalization for Generative Adversarial Networks. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[48] Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of International Conference on Machine Learning (ICML)*. 2642–2651.

[49] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 4954–4963.

[50] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2020. ACTIVETHIEF: Model Extraction Using Active Learning and Unannotated Public Data. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 34. AAAI, 865–872.

[51] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of ACM on Asia conference on computer and communications security (ASIA CCS)*. ACM, 506–519.

[52] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2337–2346.

[53] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[54] Eitan Richardson and Yair Weiss. 2018. On gans and gmms. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 31. Curran Associates, Inc., 5847–5858.

[55] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International journal of computer vision* 115, 3 (2015), 211–252.

[56] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2019. ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models. In *Proceedings of Network and Distributed Systems Security Symposium (NDSS)*. Internet Society.

[57] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training GANs. In *Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2234–2242.

[58] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. 2020. Interpreting the latent space of gans for semantic face editing. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 9243–9252.

[59] Reza Shokri, Martin Strobel, and Yair Zick. 2019. Privacy risks of explaining machine learning models. *arXiv preprint arXiv:1907.00164* (2019).

[60] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*. IEEE, 3–18.

[61] Tatsuya Takemura, Naoto Yanai, and Toru Fujiwara. 2020. Model Extraction Attacks against Recurrent Neural Networks. *arXiv preprint arXiv:2002.00123* (2020).

[62] Luke Tierney. 1994. Markov chains for exploring posterior distributions. *The Annals of Statistics* (1994), 1701–1728.

[63] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. 2019. Fixing the train-test resolution discrepancy. In *Proceedings of Annual Conference on Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 8250–8260.

[64] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction APIs. In *Proceedings of USENIX Security Symposium (USENIX Security)*. USENIX Association, 601–618.

[65] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. 2019. Metropolis-hastings generative adversarial networks. In *Proceedings of International Conference on Machine Learning (ICML)*. 6345–6353.

[66] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. 2014. scikit-image: image processing in Python. *PeerJ* 2 (6 2014), e453.

[67] N Venkatanath, D Praneeth, Maruthi Chandrasekhar Bh, Sumohana S Channappayya, and Swarup S Medasani. 2015. Blind image quality evaluation using perception based features. In *2015 Twenty First National Conference on Communications*. IEEE, 1–6.

[68] Binghui Wang and Neil Zhenqiang Gong. 2018. Stealing hyperparameters in machine learning. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*. IEEE, 36–52.

[69] Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. 2018. Texturegan: Controlling deep image synthesis with texture patches. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 8456–8465.

[70] Pan Xudong, Zhang Mi, Ji Shouling, and Yang Min. 2020. Privacy Risks of General-Purpose Language Models. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*. IEEE, 1471–1488.

[71] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365* (2015).

[72] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. 2017. StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. IEEE, 5907–5915.

[73] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. 2016. Generative visual manipulation on the natural image manifold. In *Proceedings of European conference on computer vision (ECCV)*. Springer, 597–613.

A APPENDIX

A.1 Implementation Details

We implement PGGAN[5] and SNGAN[6] based on following codes indicated in the footnotes. We choose the ResNet architecture for SNGAN and the architecture of PGGAN is the same as the official implementation. We use hinge loss for SNGAN and WGAN-GP loss for PGGAN. For target GAN on synthetic data in Figure 5, we use four fully connected layers with ReLU activation for both generator and discriminator and the prior is a 2-dimensional standard normal distribution. The training data is a mixture of 25 2-D Gaussian distributions (each with standard deviation of 0.05). We train it using standard loss function [19]. In Section 7 about case study, we directly use the pretrained StyleGAN[7] trained on LSUN-Bedroom dataset as our target model. We resize all images used in our paper to 64×64, except for the case study where images with a resolution of 256×256 are used. The dimension of latent space of SNGAN, PGGAN and StyleGAN is 256, 512 and 512, respectively, and their latent codes are all draw from standard Gaussian distribution. For attack models, we use suggested hyperparameters provided by original models and only modify some related to computing resources.

In Section 8 about semantic interpolation defense, the semantic information is from attributes of CelebA dataset[8], which has labeled for each image. we only choose 12 (male, smiling, wearing lipstick, mouth slightly open, wavy hair, young, eyeglasses, wearing hat, black hair, receding hairline, bald, mustache) out of 40 facial attributes to learn semantic hyperplanes, because the number of images for each attribute varies largely and some attributes is hard to distinguish when they are applied in target GAN model. We train the prediction model for each attribute based on ResNet-50 model pretrained on ImageNet[9]. The magnitude of semantic interpolation is set as 3.

A.2 MH Algorithm

Algorithm 1 shows the MH subsampling algorithm [65]. Inputs of this algorithm are a target generator (only used to query), a white-box discriminator which is used to subsample generated samples and partial real samples which are used to calibrate the discriminator (Algorithm 1, line 2). Outputs are refined samples whose distribution is much closer to distribution of real training data.

A.3 Attack Performance on Queries From Different Prior Distributions.

Adversaries can query the target model via trying common prior distributions to generate latent codes if they do not know the prior distribution of a target model. Gaussian distribution and uniform distribution are widely used in almost all GANs [6, 31–33, 47, 53]. Table 7 shows the attack performance with two prior distributions. We choose PGGAN trained on CelebA dataset with standard normal prior distribution as the target model. From Table 7, we find that adversaries can obtain a similar attack performance no matter what the prior distribution of latent codes is.

[5]https://github.com/tkarras/progressive_growing_of_gans
[6]https://github.com/christiancosgrove/pytorch-spectral-normalization-gan
[7]https://github.com/NVlabs/stylegan
[8]http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html
[9]https://download.pytorch.org/models/resnet50-19c8e357.pth

Algorithm 1 MH subsampling

Input: target generator G, target discriminator D, partial real samples $X_r = \{x_{r1}, x_{r2}, \ldots, x_{rm}\}$
Output: N refined images
1: Sample m fake images $X_g = \{x_{g1}, x_{g2}, \ldots, x_{gm}\}$ from G
2: Train a calibrated classifier: $C \leftarrow LogisticRegression(D(X_r), D(X_g))$
3: $images \leftarrow \emptyset$
4: **while** $|images| < N$ **do**
5: $x \leftarrow$ a real image from X_r
6: **for** $i = 1$ to K **do**
7: Sample x' from G
8: Sample u from Uniform$(0, 1)$
9: Compute real image's density ratio: $r(x) = \frac{C(D(x))}{1-C(D(x))}$
10: Compute fake image's density ratio: $r(x') = \frac{C(D(x'))}{1-C(D(x'))}$
11: $p = min(1, \frac{r(x')}{r(x)})$
12: **if** $u \leq p$ **then**
13: $x \leftarrow x'$
14: **end if**
15: **end for**
16: **if** x is not a real images **then**
17: Append$(x, images)$
18: **end if**
19: **end while**

Table 7: Performance of fidelity extraction attack with different prior distributions. We use standard normal distribution and uniform distribution over an interval -1 and 1 to generate latent codes. The number of queries is fixed to 50K.

Attack model	Prior distribution	*Fidelity* FID($\tilde{p}_g, p_g$)	*Accuracy* FID ($\tilde{p}_g, p_r$)
SNGAN	Gaussian	4.49	9.29
SNGAN	Uniform	4.29	9.16
PGGAN	Gaussian	1.02	4.93
PGGAN	Uniform	0.98	4.85

A.4 Additional Results for Analyzing Distribution Differences

A.4.1 ***Understanding Fidelity Extraction for the Target Model SNGAN.*** Figure 9 shows distribution differences for the target model SNGAN trained on CelebA dataset. Table 8 summarizes these differences statistically.

A.4.2 ***Understanding Accuracy Extraction on GANs In-depth.*** Following the same procedure illustrated in Section 5.3.3, we also dissect distribution differences for accuracy extraction. Specifically, we choose the PGGAN-PGGAN case as an example (see Figure 6) and the attack models is PGGAN. From the Figure 10, we observe that for CelebA, white-box accuracy extraction which has

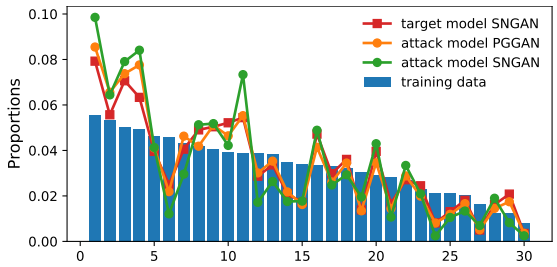
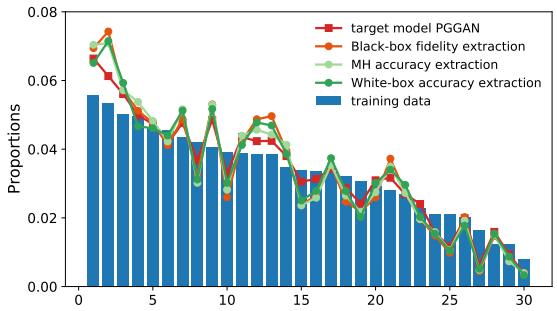


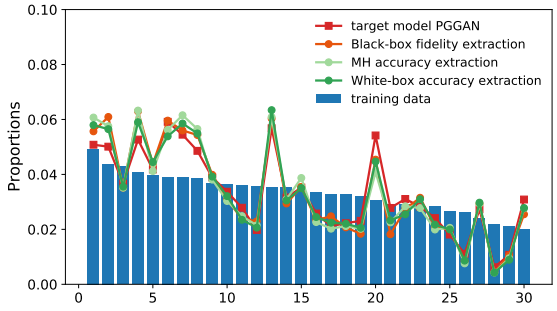
Figure 9: Class distribution differences among the training data, the target model SNGAN, and attack models.

Table 8: JS distances between models. For the JS distance between training data and the target model, and the target model SNGAN is 16.36×10^{-3}. The JS value shows a consistent trend with Figure 9.

Target model	Attack model	$JS_{fidelity}$ ($\times10^{-3}$)	$JS_{accuracy}$ ($\times10^{-3}$)
SNGAN	SNGAN	8.90	34.12
	PGGAN	1.60	18.56



(a) The target model PGGAN trained on CelebA.



(b) The target model PGGAN trained on LSUN-Church.

Figure 10: Distribution differences for accuracy extraction.

minimal *accuracy* values among these methods is more consistent with the distribution of the training data by lowering the highest proportions of classes. For LSUN-Church, similar results also can be observed. Table 9 summarizes these differences statistically.

Table 9: JS distances between models. For the JS distance between training data and the target model, the target model PGGAN on CelebA is 4.14×10^{-3} and the target model PGGAN on LSUN-Church is 14.78×10^{-3}.

Dataset	Methods	$JS_{fidelity}$ ($\times10^{-3}$)	$JS_{accuracy}$ ($\times10^{-3}$)
CelebA	Black-box fidelity extraction	1.83	9.10
	MH accuracy extraction	1.42	8.17
	White-box accuracy extraction	1.17	7.53
LSUN-Church	Black-box fidelity extraction	2.32	19.14
	MH accuracy extraction	2.28	19.89
	White-box accuracy extraction	1.61	18.65

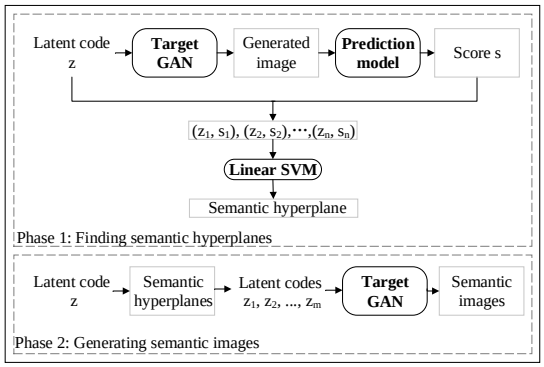


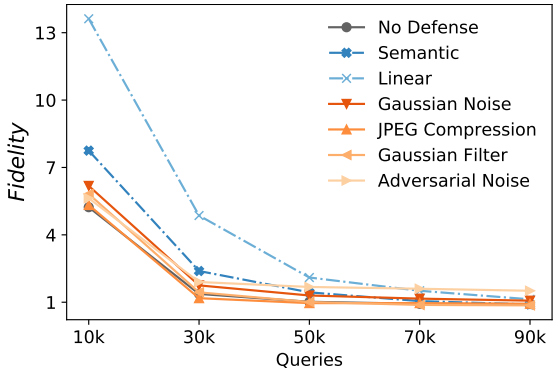
Figure 11: Semantic interpolation defense.

A.5 Defense Techniques

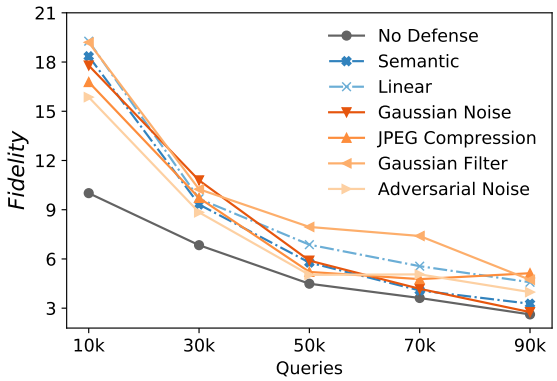
A.5.1 Semantic Interpolation Defense. The process of semantic interpolation defense is shown in Figure 11. Semantic interpolation defense consists of two phases: finding semantic hyperplanes and generating semantic images. In the first phase, we first train a prediction model for each semantic information. Then the trained prediction model is used to predict semantic score s for each image generated through latent code z. As a result, we get latent code-score pairs and label the highest k scores as positive and the lowest k scores as negative. Finally, we train a linear support vector machine (SVM) on dataset where latent codes as training data and scores as labels. A trained linear SVM contains a hyperplane which separates one semantic information. In the second phase, we can obtain a semantic image for each semantic hyperplane through interpolation. A latent code interpolates points along the normal vector of the hyperplane and corresponding semantic images can be obtained.

In our experiments, we train each prediction model for each semantic information, and prediction model is built on the basis of ResNet-50 network [21] trained on ImageNet dataset [55].

A.5.2 Defense Utility. We quantitatively and qualitatively evaluate the defense utility, i.e. the quality of generated images after deploying defense measures. Figure 14 and Figure 15 show returned images for input perturbation-based and output perturbation-based defenses. Table 10 shows image quality scores. We use two widely-adopted no-reference image quality scores: Naturalness Image Quality Evaluator (NIQE) [46] and Perception based Image Quality Evaluator (PIQE) [67]. Overall, our defense measures do not impact the quality of generated images.



(a) The performance of attack model PGGAN



(b) The performance of attack model SNGAN

Figure 13: *Fidelity* of attack models under different defenses for black-box fidelity extraction scenarios. *Fidelity* values of attack models can be largely decreased with an increase in the number of queries.

Table 10: Defense utility. Each score is an average of 50K image score. Lower is better.

Metrics	No Defense	Semantic	Linear	Gaussian Noise
NIQE	18.87	18.87	18.87	18.87
PIQE	42.66	40.04	42.62	23.64

Metrics	JPEG Compression	Gaussian Filter	Adversarial Noise
NIQE	18.87	18.87	18.87
PIQE	35.99	47.80	37.91

A.5.3 Attack Performance for the Attack Model SNGAN under Various Defense Techniques. Figure 12 shows that the performance of attack model SNGAN under various defenses for the black-box fidelity extraction scenario.

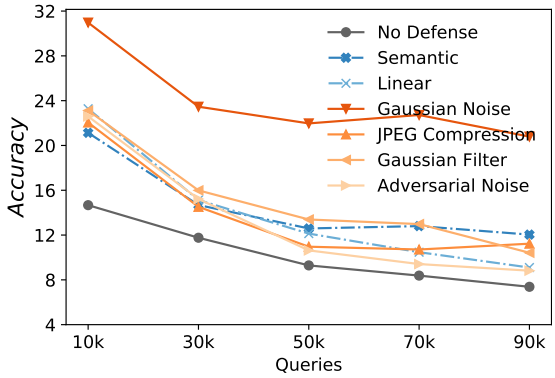
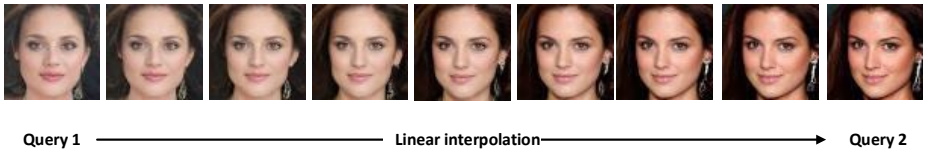
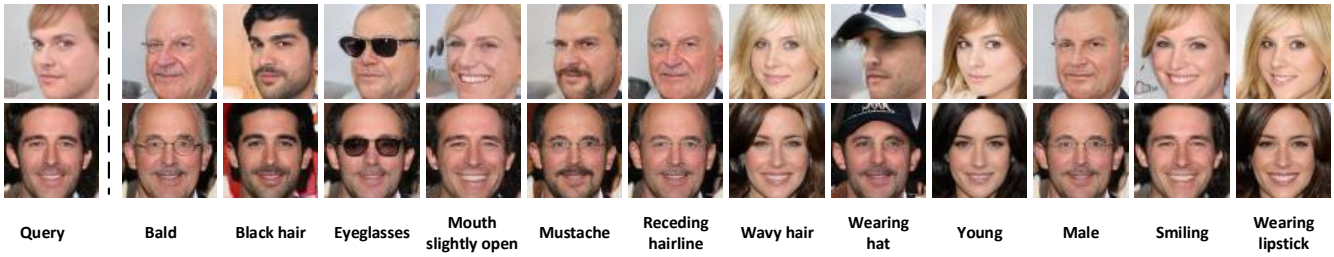


Figure 12: The performance of attack model SNGAN under various defenses for the black-box fidelity extraction scenario.

A.5.4 Fidelity on Various Defense Techniques. Figure 13 shows *fidelity* of attack models under different defenses for black-box fidelity extraction scenario. We observe that *fidelity* values of attack models can be largely decreased with an increase in the number of queries.

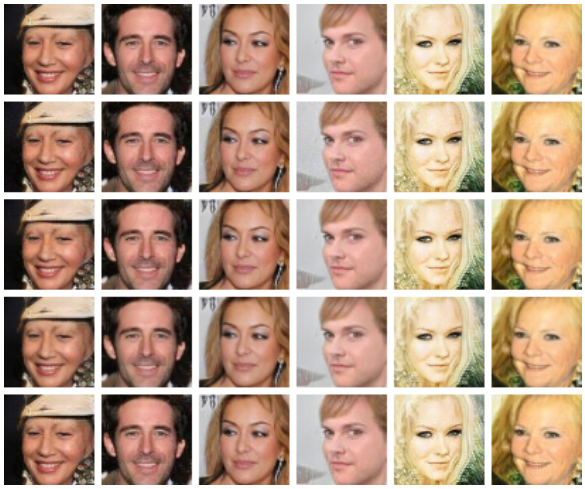


(a) Linear interpolation defense. These linear interpolated images are returned.

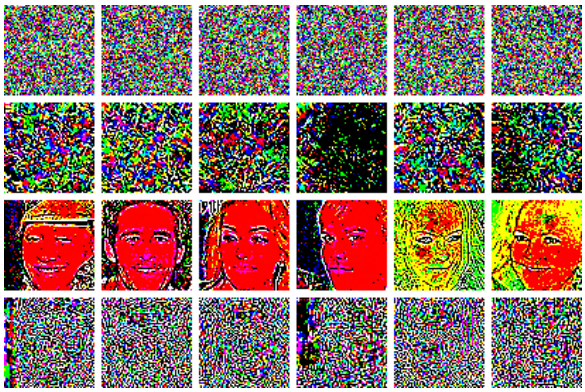


(b) Semantic interpolation defense. For one latent code, 12 latent codes containing semantic information are generated through semantic interpolation and corresponding images are shown above. These semantic interpolated images are returned.

Figure 14: Returned images after input perturbation-based defense techniques. Queried images and interpolated images both show good quality in visual comparison, and images generated by linear interpolation show more similarity than that by semantic interpolation.



(a) Output images. From top to bottom: generated images, Gaussian noise images, Adversarial noise images, Gaussian filter images and JPEG compression images.



(b) Noises. For the top two rows, they are Gaussian noises and adversarial noises, respectively. For the third row, it is the differences between Gaussian filter images and generated images. For the last row, it is the differences between JPEG compression images and generated images.

Figure 15: Returned images after output perturbation-based defense techniques.