DIVIDE AND CONQUER POLICY FOR EFFICIENT GAN TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in Generative Adversarial Networks (GANs) have achieved impressive results for the purpose of generating high-quality synthetic imagery. While capable of synthesizing high-fidelity images, these models often generate unsatisfactory images which fall outside of the data manifold. A considerable research effort has investigated the data manifold, either by simply discarding images having a lower probability according to the discriminator output, or by filtering real images which are within the sparse regions of the data manifold. While effective, these methods fail to get access to either the fake distribution or the real distribution. In this paper, we propose a divide and conquer policy for GAN training. We first introduce a new local data-manifold detector (LDMD), which estimates whether the generated images are inside or outside of the data manifold. With the proposed LDMD, we further introduce a noise replay mode if it is outside the manifold, and a fake sample reuse mode if it is inside the manifold. Extensive experimental results on a number of GANs variants (e.g., SAGAN,SNGAN,BigGAN and StyleGAN) demonstrate qualitatively and quantitatively that our method improves the GAN's performance, resulting in more realistic images than previous methods as confirmed by a significant drop in the FID.

1 INTRODUCTION

GANs achieve state-of-the-art synthesis results on image data and beyond. They have been shown to generate high-quality realistic images (Karras et al., 2018; 2019; Brock et al., 2019; Esser et al., 2021; Sauer et al., 2022; Epstein et al., 2022). GANs, however, suffer from mode-collapse and training instabilities. To combat these challenges, a significant number of studies have focused on improving the architectures (Karras et al., 2018; Esser et al., 2021; Sauer et al., 2022; Epstein et al., 2022), and the optimization of training (Arjovsky et al., 2017; Gulrajani et al., 2017; Miyato et al., 2018; Zhang et al., 2018; Liu et al., 2020; Sauer et al., 2021). In this paper, we push the envelope further: how to reduce mode-collapse and training instabilities by investigating the generated data-manifold.

Recent work by (Wu et al., 2019; Sinha et al., 2020; DeVries et al., 2020; Casanova et al., 2021) suggests that exploring the training data manifold can be useful when the GAN network is trained on samples closer to data-manifold. That is, leveraging more realistic generated samples helps the generator converge better. Wu et al. (2019) updates the generator and discriminator parameters using the sampling noise as input of the generator which outputs more realistic images. However, they fail to use 'less realistic' generated samples. Top-k (Sinha et al., 2020) discards the gradient contributions from the elements of the batch that the discriminator scores as 'least realistic'. Yet, it finally only learn the main distribution of the real data, and suffers from challenging of using small training batch size. DeVries et al. (2020); Casanova et al. (2021) train a GAN network by considering the local data-manifold which is expressed with the nearest neighbor. DeVries et al. (2020) simply zero out the training sample which is within the low data dense, resulting in using all training samples. Casanova et al. (2021) requires extra input images when using the trained generator at test time. In this paper, we explore how to detect and utilize the generated sample closer to the real data-manifold. Furthermore, we propose a new method to improve the quality of synthesized samples which are far away from the data-manifold.

To further improve the training of GANs: we investigate both the fake data-manifold and the real data-manifold, and propose a *divide and conquer* policy for GAN training. As shown in Fig. 1 (left),

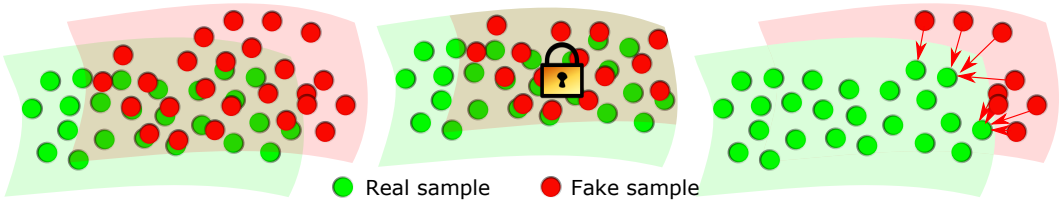

Figure 1: (Left) the manifolds of both the real (green ball) and fake (red ball) samples overlap for already trained GAN model. (Middle) we intensify the fake images which are within the real data-manifold. (Right) we push the generated images which are not within the real data-manifold.

we hypothesize the manifolds of both the real and fake samples overlap for already trained GAN model,which we think it holds (Theis et al., 2015). We introduce a local data-manifold detector (LDMD) which deploys the generated image. That is, LDMD is used to estimate whether the generated sample is within the real data-manifold or not. With the introduced LDMD, we label the generated image as reliable fake image (RFI) or non-reliable fake image (nRFI). The former (Fig. 1 (middle)) is within the real data-manifold, and provides more useful gradients when updating the generator (Wu et al., 2019; Casanova et al., 2021), which encourages the generator to preserve strong connection with the local training manifold. The latter (i.e., nRFI) is not within the real data-manifold, for which we propose a new method to push the generator output to the side of the true real data-manifold (Fig. 1 (right)). After distinguishing RFI and nRFI, we propose two methods by using both the input noise and the generated fake image. Furthermore, we consider the density around each RFI, which can weight the importance of RFI which is high density area, improving GANs performance. We evaluate the proposed method on multiple datasets, including complex datasets which have significant differences such as Cifar10 (Krizhevsky et al., 2009), AFHQ (Choi et al., 2020) and ImageNet (Deng et al., 2009).

In sum, our work makes the following **contributions**:

- We propose a local data-manifold detector (LDMD), which contributes to estimate whether the generated images are inside or outside of the data manifold.
- With the proposed local data-manifold detector(LDMD), we further propose a new resampling method of both the input noise and the generated image, which are utilized to improve GAN optimization.
- Extensive experimental results in a number of GANs variants demonstrate qualitatively and quantitatively that our method improves GANs performance, resulting in more realistic images than previous methods as confirmed by a significant drop in the FID.

2 Related Works

GANs. The original GANs (Goodfellow et al., 2014) is defined as containing two components: generator and discriminator, and subsequent related works also follow this structure. Mode collapse and training instability are two main challenges in the training process of GANs. Many approaches focus on solving these above-mentioned problems (Gulrajani et al., 2017; Salimans et al., 2016; Mao et al., 2017; Arjovsky et al., 2017; Miyato et al., 2018; Bang & Shim, 2021; Liu et al., 2022). SNGAN (Miyato et al., 2018) proposed a novel weight normalization technique to stabilize the training of the discriminator. StyleGAN (Karras et al., 2019) improves the architecture of the generator to control the high-level attributes and stochastic variation of synthesis images. BigGAN (Brock et al., 2019) successfully generates conditional high-resolution and various images using ImageNet (Deng et al., 2009). StyleGAN-XL (Sauer et al., 2022) inserts conditional information for StyleGAN, and manages to generate the high-fidelity image. Some methods (Schwarz et al., 2020; Nguyen-Phuoc et al., 2019; Chan et al., 2021; Niemeyer & Geiger, 2021b;a; Nguyen-Phuoc et al., 2020; Lunz et al., 2020; Henderson et al., 2020; Gadelha et al., 2017; Jimenez Rezende et al., 2016; Gu et al., 2021; Zhao et al., 2022) have already combined 3D scene representations with GANs, which can be trained using only single-view images.

Data manifold for GANs. Recent methods (DeVries et al., 2020; Casanova et al., 2021; Sinha et al., 2020; Wu et al., 2019) consider the data-manifold for GANs training, since exploring the training data manifold could improve model performance when the GAN network is trained on samples closer to data-manifold. That is, more realistic samples could be useful for generating more realistic images. (Wu et al., 2019) focuses on the local noise distribution which is corresponding to more realistic image. Top-k (Sinha et al., 2020) indicates that (Wu et al., 2019) ignores 'less realistic' generated samples, and proposed to only leverage the elements of the batch, which has the maximum probability scores of the discriminator outputs. Top-k, however, fails to get access to the fake distribution as well as the challenge usage of small training batch size. (DeVries et al., 2020; Casanova et al., 2021) explore the impact of reducing the size of the training set when training GANs. Specially they use the nearest neighbour to present the data manifold, and strengthen the usage of the high-dense manifold. Although (DeVries et al., 2020) got better metric scores, it fails to use all training samples. (Casanova et al., 2021) needs extra input image when using the trained generator at test time. In this paper, the proposed method avoids this disadvantages, and improve GANs performance.

GANs Evaluation. Better correlation with human perception has been found in the widely used Inception Score (Salimans et al., 2016), but recent works have also shown its limitations (Zhou et al., 2018). Latest quantitative metrics (Heusel et al., 2017; Sajjadi et al., 2018; Kynkäänniemi et al., 2019; Naeem et al., 2020) have achieved convincing performance. For image generation tasks, FID score (Heusel et al., 2017), the most popular metric, has empirically exhibited good agreements with human perceptual scores. FID, however, fails to separate two important properties of the quality of generative models: fidelity and diversity. To address this shortcoming, (Sajjadi et al., 2018; Kynkäänniemi et al., 2019; Naeem et al., 2020) proposed two-values metric: *Precision/Recall*, and *density/coverage*. These methods, however, only evaluate the trained model instead of guiding directly GANs training. Inspired by (Kynkäänniemi et al., 2019) we use the nearest neighbour to estimate the data-manifold, and optimize GANs training.

3 METHOD

Here we briefly introduce GANs. A GAN consists of a deep generative model G and a discriminative model D, both of which play a mini-max game. The aim of the generator is to generate a distribution p_g that is similar to the real data distribution p_{data}, such that the discriminative network cannot distinguish between the images from the real data distribution and the generated ones (the model distribution).

Let $\boldsymbol{z} \in \mathbb{R}^Z$ be random noise, and $\boldsymbol{x}$ (where $\boldsymbol{x} \in \mathcal{X}$) be a real image drawn from the real data distribution p_{data}. The generator G takes the noise variable $\boldsymbol{z}$ as input, aiming to synthesize samples obeying the distribution p_g. The discriminative model $D(\boldsymbol{x})$ computes the probability of which input data $\boldsymbol{x}$ is from p_{data} rather than the generated model distribution p_g. Ideally, $D(\boldsymbol{x}) = 0$ if $\boldsymbol{x} \sim p_g$ and $D(\boldsymbol{x}) = 1$ if $\boldsymbol{x} \sim p_{data}$. More formally, the generative model and discriminative model are trained by solving:

$$\min_G \max_D V(D, G) = E_{\boldsymbol{x} \sim p_{data}} [\log D(\boldsymbol{x})] + E_{\boldsymbol{z} \sim p(\boldsymbol{z})} [\log (1 - D(G(\boldsymbol{z})))] \tag{1}$$

where $p(\boldsymbol{z})$ follows the normal distribution.

3.1 A DIVIDE AND CONQUER POLICY FOR GAN EFFICIENT TRAINING

Given a well-trained generator (Karras et al., 2019; Brock et al., 2019), we observe that some synthesized samples well follow the real data manifold and have high-fidelity quality, while others do not. This phenomenon inspires us to explore processing the two groups of generated samples differently. Therefore, we propose a divide and conquer policy for GAN training. For the sample within the real data manifold, we can reuse it to further improve GAN performance as suggested (Casanova et al., 2021; DeVries et al., 2020; Sinha et al., 2020; Wu et al., 2019), aiming to reinforce the real data manifold. The noise, i.e., the samples that are outside the real data manifold, are replayed. In this paper, we present the local data manifold detector (LDMD), studying the implicit data local manifold to split the generated images during training. Finally, we propose a more efficient training scheme that handles the two groups of generated images differently.

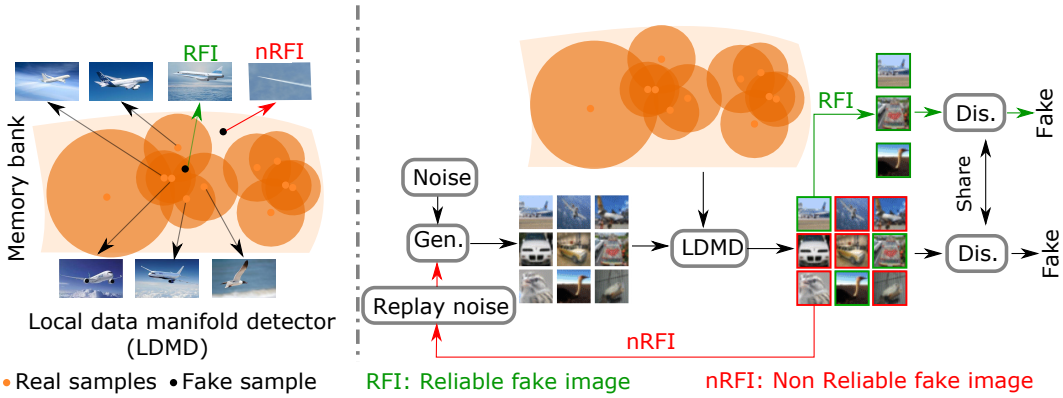

Figure 2: (Left) Overview of *local data manifold detector* (LDMD). We expect the generator to generate realistic images (i.e., black solid dots) similar to the neighbors (i.e., orange solid dots), defined circus with orange color. We define the generated image as *reliable fake image* (RFI) if it follows E.q 5. If not, it is a non *reliable fake image* (nRFI). (Right) The proposed local data manifold detector (LDMD) finds both RFI and nRFI. RFI is taken as input for the discriminator as well as the batch fake images, to push the generator to connect strongly to the local data manifold, aiming to improve GAN performance. For nRFI, we replay the noise which is corresponding to nRFI, and feed it into the generator, aiming to close the local data manifold.

Local data manifold detector. Inspired by the GAN evaluation method (Kynkäänniemi et al., 2019), we build a local data manifold detector based on the k-nearest neighbour algorithm. Generated images are considered to be within the local manifold if the generated image is within the manifold (estimated by of the nearest neighbours) of the real image, and the real image also is within the nearest neighbours of the generated image [1]. Using k-nearest neighbors has proven to be very successful in a myriad of computer vision tasks such as categorization (Zhang et al., 2006), detection (Cohen et al., 2020), and retrieval (Dwibedi et al., 2021). Thus, we use k-nearest neighbors to achieve the goal of finding the local data manifold. Specially, we first extract feature $\boldsymbol{f}$ of each real image $\boldsymbol{x}$ with the feature extractor $\mathcal{F}$ (i.e., InceptionV3 (Szegedy et al., 2016)). Then we build the memory bank $\mathcal{S}$ to store all real image features:

$$\boldsymbol{f} = \mathcal{F}(\boldsymbol{x}) \tag{2}$$

$$\mathcal{S} = [\boldsymbol{f}_0, \boldsymbol{f}_1, \ldots, \boldsymbol{f}_{n_t-1}] \tag{3}$$

where n_t is the number of the real images $\mathcal{X}$. Similarly, we also extract the feature representation $\hat{f}$ of fake images $\hat{\boldsymbol{x}}_j$ from the current batch $G(\boldsymbol{z}_j)$:

$$\hat{\boldsymbol{f}} = \mathcal{F}(\hat{\boldsymbol{x}}_j) = \mathcal{F}(G(\boldsymbol{z}_j)) \tag{4}$$

The index j is 0,1,...,$n_b - 1$, and n_b is the batch size. We build the nearest neighbour set $\mathcal{N}_k^i(i$=0,1,...,$n_t - 1)$ of the training data, here $\mathcal{N}_k^i$ is the index set of the k-nearest neighbors of feature $\boldsymbol{f}_i$ for the training sample $\boldsymbol{x}_i$. We define $\mathcal{D}_{max}(\mathcal{N}_k^i)$ to be the maximum distance of the k-nearest neighbors. We also construct the nearest neighbour set $\mathcal{N}_k^j(j$=0,1,...,$n_b - 1)$ between the features of the generated image and the real image, here the index j is 0,1,...,$n_b - 1$, and n_b is the batch size. We define $\mathcal{D}_{max}(\mathcal{N}_k^j)$ to be the maximum distance of the k-nearest neighbor.

Inspired by recent work (Kynkäänniemi et al., 2019; Naeem et al., 2020), we use the Euclidean distance for nearest neighbor retrieval. Constructing the nearest neighbor aims to find local manifold of the training data in the feature space (Kynkäänniemi et al., 2019; Naeem et al., 2020). As Kynkäänniemi et al. (2019) mentions, it is possible to estimate the true training manifold by computing pairwise Euclidean distances between all feature vectors in the training data. For each training feature embedding, we construct a hypersphere with a radius equal to the distance to its kth nearest neighbor (i.e., $\mathcal{D}_{max}(\mathcal{N}_k^i)$).

[1]The nearest neighbour of the real image is selected among the real images, while the nearest neighbour of the generated image is computed between the generated fake image and all real images.

Based on both the nearest neighbour index set $\mathcal{N}_k^i$ and $\mathcal{N}_k^j$, we propose to further annotate the generated images separately. We label each fake image as being either a *reliable fake image* (RFI) or a *non-reliable fake image* (nRFI). As shown in Fig. 2 (left), a reliable fake image (RFI) is defined by the following constraints:

$$\begin{cases} \mathcal{D}(\hat{\boldsymbol{x}}_j, \boldsymbol{x}_i) < \mathcal{D}_{max}(\mathcal{N}_k^i) \\ \mathcal{D}(\boldsymbol{x}_i, \hat{\boldsymbol{x}}_j) < \mathcal{D}_{max}(\mathcal{N}_k^j), \end{cases} \tag{5}$$

where $\mathcal{D}(\hat{\boldsymbol{x}}_j, \boldsymbol{x}_i)$ is the distance between $\hat{\boldsymbol{x}}_j$ and $\boldsymbol{x}_i$ ($i \in \mathcal{N}_k^j$). Eq. 5 indicates that the generated image is within the estimated nearest neighbour of the real image (i.e., $\mathcal{N}_k^i$), and the real image also is within the estimated nearest neighbour of the generated image (i.e., $\mathcal{N}_k^j$). We assume that the generated image is reliable if the generated image meets Eq. 5, which we believe is reasonable since it has been used to evaluate GANs model (Kynkäänniemi et al., 2019) at test time. If the generated image $G(\boldsymbol{z}_j)$ does not meet Eq. 5, we label it as a non-reliable fake image (nRFI), which indicates that the generated image $G(\boldsymbol{z}_j)$ is not within the training manifold.

The definition of a reliable fake image (RFI) indicates that the fake image $\hat{\boldsymbol{x}}_{\boldsymbol{j}}$ is strongly connected with the local training data manifold, since the given fake image feature is within the local training manifold. (Kynkäänniemi et al., 2019) proposed this mechanism to match the distributions of both the real data and the fake data when evaluating GAN models. In this paper, we adapt it to train GANs. In the following section, we introduce how to use both RFI and nRFI to efficiently train GANs.

Reliable fake image (RFI). Recent work by (Casanova et al., 2021; DeVries et al., 2020; Sinha et al., 2020; Wu et al., 2019) indicate that exploring the training data manifold can be more useful when the GAN network is trained on samples closer to data-manifold. Motivated by the same insight, we propose to reuse RFI. To be specific, as shown on Fig. 2 (right) we re-enter RFI into the discriminator, as well as using current batch samples.

$$\begin{aligned} \min_G \max_D V(D,G) = & E_{\boldsymbol{x}\sim p_{data}}\left[\log D(\boldsymbol{x})\right] + E_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log\left(1 - D\left(G(\boldsymbol{z})\right)\right)\right] \\ & + E_{\boldsymbol{z}\sim p_{rfi}}\left[\log\left(1 - w(m(G(\boldsymbol{z}))) * D\left(G(\boldsymbol{z})\right)\right)\right] \end{aligned} \tag{6}$$

$$w(\cdot) = 1 + sigmoid(\cdot) \tag{7}$$

where p_{rfi} follows the normal distribution, from which noise $\boldsymbol{z}$ is sampled to generate the correspondence to RFI. m is the number of the real image which meets Eq. 5. We hypothesize that the more real images $\boldsymbol{x}_i$ meet Eq. 5, the higher the probability that the generated sample $\boldsymbol{x}_j = G(\boldsymbol{z})$ is within the real data-manifold, which indicates $\boldsymbol{x}_j = G(\boldsymbol{z})$ is more useful when computing gradient, as also suggested (Casanova et al., 2021; Wu et al., 2019). This inspires us to introduce a weight $w(\cdot)$. In this paper, we only utilize this mechanism when updating the generator. Specially, when updating the generator RFI is reused to compute the gradient, we do not re-enter RFI when optimizing the discriminator, since the discriminator suffers from the challenge of balancing both the real and fake image distribution.

Non-Reliable fake image (nRFI). We re-enter the same noise into the generator when the corresponding output of the generator is nRFI. Since nRFI is not within the real sample manifold, we sample the same noise to push the generator output to the side of the true real sample manifold. We define the loss as:

$$\begin{aligned} \min_G \max_D V(D,G) = & E_{\boldsymbol{x}\sim p_{data}}\left[\log D(\boldsymbol{x})\right] + E_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log\left(1 - D\left(G(\boldsymbol{z})\right)\right)\right] \\ & + E_{\boldsymbol{z}\sim p_{nrfi}}\left[\log\left(1 - D\left(G(\boldsymbol{z}+\delta)\right)\right)\right] \end{aligned} \tag{8}$$

where p_{nrfi} follows the normal distribution which is corresponding to nRFI. δ is a noise perturbation. Prior work (Yang et al., 2021) has mentioned the continuity of images synthesized from the latent codes within a neighbourhood are positive with the continuity of the latent codes. Thus we hypothesize that the small area of the noise devoted to nRFI is still corresponding to the fake image distribution outside of the real data-manifold. Therefore we introduce a noise perturbation δ strategy into the noise which is corresponding to nRFI. In our paper we sample δ from a Gaussian distribution whose variance (i.e., 0.5) is sufficiently smaller than that of $\boldsymbol{z}$. (Yang et al., 2021) uses this technique for contrastive learning.

Figure 3: Results on CIFAIR-10(Top-left), Tiny-ImageNet(Top-right) and AFHQ-cat(bottom).

Full Objective. The full objective function of our model is:

$$\begin{aligned}\min_G \max_D V(D,G) = & E_{\boldsymbol{x}\sim p_{data}}\left[\log D(\boldsymbol{x})\right] + E_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log\left(1 - D\left(G(\boldsymbol{z})\right)\right)\right] \\ & + E_{\boldsymbol{z}\sim p_{rfi}}\left[\log\left(1 - w(G(\boldsymbol{z})) * D\left(G(\boldsymbol{z})\right)\right)\right] \\ & + E_{\boldsymbol{z}\sim p_{nrfi}}\left[\log\left(1 - D\left(G(\boldsymbol{z}+\delta)\right)\right)\right]\end{aligned} \tag{9}$$

4 Experiments

In this section, we first introduce the used evaluation measures, datasets and architectures. Then, we explore a wide variety of configurations for our approach, and evaluate our method on different GAN architectures and datasets.

4.1 Experiment setting

Datasets. We evaluate our model on varying kinds of datasets, including CIFAR-10 (Krizhevsky et al., 2009), AFHQ (Choi et al., 2020) and ImageNet (Deng et al., 2009). CIFAR-10 dataset contains 60000 32×32 colour images and 10 classes in total. There are 50000 training images and 10000 test images. AFHQ contains 3 classes (i.e., *cat*, *dog* and *wild*), each one has about 5000 training images and 500 test images. We evaluate our method on each category with resized image 256×256. We use 1000 images to train, and the remaining images for test. Here we only use *AFHQ-cat*. ImageNet (Deng et al., 2009) contains 1,281,167 training images with 128×128 considered, and 1000 object classes. We also consider *Tiny ImageNet*, which contains 100,000 images of 200 classes (500 for each class) downsized to 64×64 colored images. We leverage validation set to test our model.

Figure 4: The nearest neighbour of the generated image on Tiny-ImageNet. The left images are the generated samples, while the remaining ones are the nearest neighbour samples of the training dataset

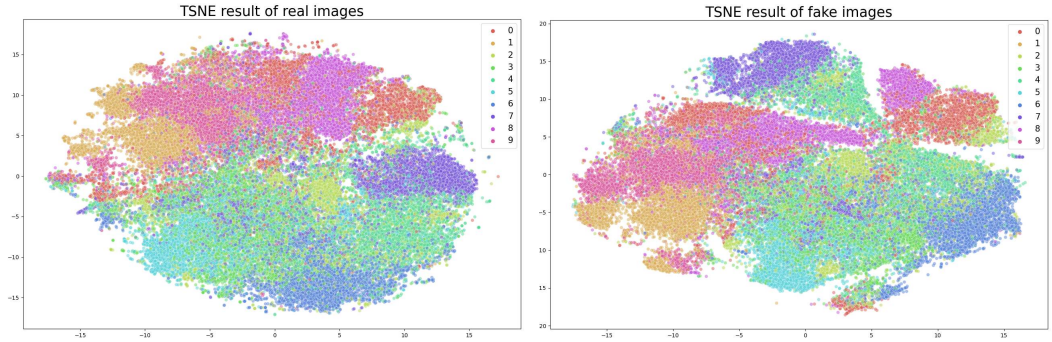

Figure 5: T-SNE of both the training data and the fake data on CIFAR-10. The proposed method manages to learn the class-specific distribution.

Baselines. We compare to DCGAN (Radford et al., 2015), SAGAN (Zhang et al., 2018), SNGAN (Miyato et al., 2018),Top-k (Sinha et al., 2020), ISGAN (DeVries et al., 2020) and StyleGAN (Karras et al., 2019). Top-k (Sinha et al., 2020) discards the fake images which have the lowest discriminator outputs, and preserve the ones which have the highest discriminator outputs. ISGAN (DeVries et al., 2020) focuses on the areas of the high data-manifold. For StyleGAN (Karras et al., 2019) we leverage the pretrained model [2] to compare both baselines and the proposed method due to the limited computing resources. All of aforementioned methods perform unconditional GANs. We also consider BigGAN (Brock et al., 2019) to conduct conditional image generation. Our code highly relies on StudioGAN project [3]. The training details for all models are introduced in StudioGAN.

Evaluation Measures. We employ two widely used Inception Score (IS) (Salimans et al., 2016) and Fréchet Inception Distance (FID) (Heusel et al., 2017) for evaluation. We also use two-value metrics: Precision and Recall (PR) (Kynkäänniemi et al., 2019) and Density and Coverage (DC) (Naeem et al., 2020). Both PR and DC evaluate the quality and the diversity.

[2]https://github.com/rosinality/stylegan2-pytorch.

[3]https://github.com/POSTECH-CVLab/PyTorch-StudioGAN.

	Architecture	Method	FID	IS	Precision	Recall	Density	Coverage
CIFAR-10	DCGAN	Naive	49.02	6.63	0.56	0.31	0.44	0.35
		Top-k	45.62	6.65	0.60	**0.33**	0.52	0.35
		ISGAN	56.73	5.89	0.52	0.26	0.38	0.29
		Ours	**42.77**	**6.67**	**0.64**	0.32	**0.58**	**0.42**
	SNGAN	Naive	23.65	7.86	0.65	0.64	0.69	0.63
		Top-k	22.54	7.97	0.65	0.63	0.71	0.68
		ISGAN	29.43	6.67	0.58	0.55	0.64	0.59
		Ours	**19.87**	**8.01**	**0.69**	**0.66**	**0.73**	**0.70**
	SAGAN	Naive	22.31	7.92	0.63	0.64	0.66	0.61
		Top-k	18.21	**8.70**	0.66	0.66	0.73	0.65
		ISGAN	30.15	6.75	0.61	0.52	0.62	0.55
		Ours	**17.86**	8.65	**0.70**	**0.67**	**0.75**	**0.73**
	BigGAN	Naive	7.24	9.43	0.65	0.74	0.92	0.85
		Top-k	7.21	9.89	0.68	0.72	0.99	0.91
		ISGAN	7.11	10.0	0.67	0.73	0.97	0.92
		Ours	**7.09**	**10.10**	**0.69**	**0.75**	**1.01**	**0.93**
Tiny-ImageNet	SNGAN	Naive	69.57	6.26	0.38	0.26	0.24	0.19
		Top-k	66.29	6.75	0.41	0.27	0.31	0.22
		ISGAN	65.62	6.89	0.46	0.29	0.24	0.23
		Ours	**63.76**	**7.05**	**0.50**	**0.31**	**0.29**	**0.24**
	SAGAN	Naive	65.12	6.89	0.44	0.28	0.27	0.22
		Top-k	63.90	7.24	0.47	0.31	0.32	0.26
		ISGAN	62.81	7.67	0.50	0.30	0.29	**0.28**
		Ours	**60.84**	**7.84**	**0.51**	**0.33**	**0.33**	0.27
	BigGAN	Naive	23.65	7.12	0.63	0.59	0.66	0.62
		Top-k	22.19	7.64	0.68	0.62	0.69	0.67
		ISGAN	22.02	7.34	0.65	0.60	0.64	0.66
		Ours	**19.87**	**8.01**	**0.69**	**0.66**	**0.73**	**0.70**

Table 1: Reporting the metric values on both CIFAR-10 and Tiny-ImageNet dataset for various GAN architectures. The GAN variants considered are SNGAN, SAGAN and BigGAN. *Naive* means that we directly use the corresponding architecture method.

AFHQ-cat							ImageNet						
	IS	FID	Precision	Recall	Density	Coversity		IS	FID	Precision	Recall	Density	Coversity
StyleGAN	1.91	7.23	0.27	0.64	0.13	0.09	BigGAN	98.51	8.54	0.75	0.60	0.96	0.82
Top-K	2.02	6.42	0.31	0.65	0.13	0.10	Top-K	105.62	8.50	0.76	0.62	0.96	0.83
Ours	2.34	6.29	0.38	0.71	0.17	0.15	Ours	109.32	8.32	0.78	0.63	0.98	0.83

Table 2: Quantitative results of the proposed method on AFHQ-cat(left) and ImageNet(right).

4.2 RESULTS

Quantitative results. Tables 1 (up) reports the quantitative results of the baselines and the proposed method on the CIFAR-10 dataset to assess both conditional and unconditional GANs. Both the baselines and our method use the same architecture for each of the GAN variants. We can see that the proposed method almost everywhere achieves the best performance in terms of six evaluation metrics, except for the IS metric using SAGAN. This indicates that our model produces the most realistic and correct class-specific images among all the methods compared. Top-k using both DCGAN and SAGAN has the best Recall and IS scores. As reported on Table 1 (bottom), our method wins in all metrics on Tiny-ImageNet dataset, clearly verifying the importance of the proposed method(LDMD, RFI and nRFI) to improve the GAN's performance.

We further compare our method on different architectures, including StyleGANan d BigGAN. We evaluate our method on AFHQ-cat and ImagNet. As reported on Table 2, we obtain the best score for all metrics. Here we use the pretrained StyleGAN model from FFHQ (Karras et al., 2019), and finetune it on AFHQ-cat.

Qualitative results. Fig. 3 exhibits the generated images on three datasets: CIFAR-10, Tiny-ImageNet and AFHQ-cat. We observe that our method provides a high visual quality. Fig. 4 shows the nearest neighbor images of RFI (here k=7). This indicates that our found RFI, based on LDMD, are close to local data-manifold. The T-SNE plot (Fig. 5) also shows that our method successfully generates class-specific images.

4.3 ABLATION STUDY

We conduct ablation studies to isolate the validity of the key components of our method: RFI and nRFI. We analyze the behavior of the proposed method with different nearest neighbor factors. We also ablate the impact of both the weight w (Eq. 7) and the noise perturbation δ(Eq. 8).

RFI and nRFI. We explore a wide variety of configurations for our approach, including: *DCGAN+RFI*, *DCGAN+nRFI*, and *DCGAN+RFI+nRFI*. Fig. 6 presents a comparison between several

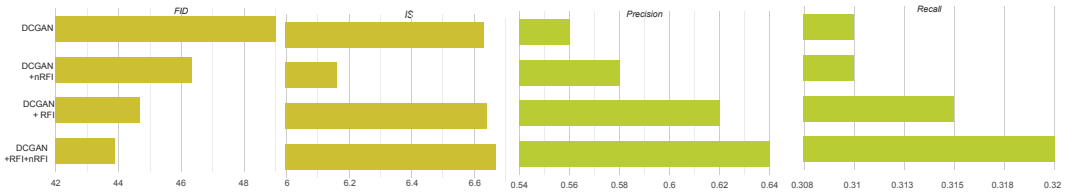

Figure 6: Ablation study of the variants of our method with DCGAN architecture on CIFAR-10. For FID score the lower is better, while the remaining metrics the higher is better.

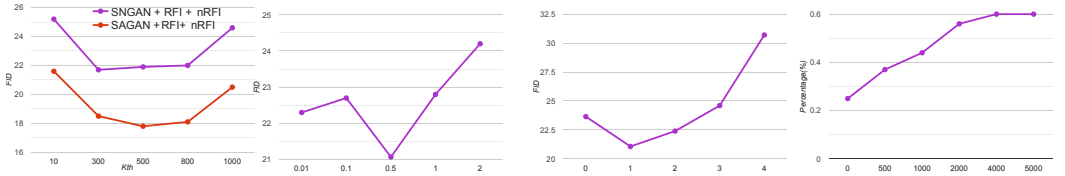

Figure 7: (Left 1) ablation study on the impact of varying kth nearest neighbour on CIFAR-10. (Left 2) ablation study of the varying δ of our method. (Left 3) the impact of varying times when re-entering RFI and replaying noise which is corresponding to nRFI. (Right 1) evolution of the percentage of RFI, which is conducted on AFHQ-cat with StyleGAN architecture.

variants of our method in terms of FID, IS, Percision and Recall with DCGAN architecture on CIFAR-10. As shown in Fig. 6, the proposed method using both RFI and nRFI is able to achieve a better score than DCGAN on all metrics. Using either RFI or nRFI contributes to improve the model performance except for *DCGAN+nRFI* which has lower IS value than DCGAN.

***k*th nearest neighbor.** We ablate the impact of varying kth nearest neighbor for both SAGAN and SNGAN. Fig. 7 (left 1) presents qualitative results on CIFAR-10. Changing this value has a significant effect on the GAN's performance. Using either small or larger k values leads to negative performance. We are able to get better results when leveraging the value from 300 to 800. We found that k value depend on the dataset, since each category of different datasets has a different number of images. We set 300 for CIFAR-10, 800 for ImageNet (also Tiny-ImageNet) and 100 for AFHQ-cat.

Noise perturbation. We evaluate the impact of varying Gaussian mean. As shown on Fig. 7 (left 2) with SNGAN on CIFAR-10, the proposed method exhibits the best performance when mean $\delta = 0.5$. Without Gaussian noise (i.e., $\delta = 0$), our method has sub-optimal result.

Varying times of reusing the noise and the fake image. We ablate the influence of the times for reusing the noise and the fake image. As shown on Fig. 7 (left 3) with SNGAN on CIFAR-10, the FID value decreases when reusing 1 time. This is followed by a stable rise, which indicates that increasing the reuse does not correspond to improved GAN performance. Thus, in this paper, we only replay once.

Change of parentage of RFI Fig. 7 (right 1) shows the evolution of the percentage of RFI. We evaluate our method using the pretrained StyleGAN on AFHQ-cat. We observe the percentage gets larger during training, which indicates that more fake images are similar to the real images, and the area of overlapping of both the fake and real data distributions get larger.

5 Conclusion

We have described a divide and conquer policy for GAN training which is effective for improving GAN performance. We propose the nearest neighbour aware local data manifold detector, which accurately and efficiently annotates the generated images into RFI and nRFI. we further introduce a noise replay mode if it is nRFI, and a fake sample reuse mode if it is RFI. In the experiment, we show that our method achieves good performance on varying kinds of GAN architecture variants and datasets, which proves the effectiveness of our method.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *ICLR*, 2017.

Duhyeon Bang and Hyunjung Shim. Mggan: Solving mode collapse using manifold-guided training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2347–2356, 2021.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

Arantxa Casanova, Marlène Careil, Jakob Verbeek, Michal Drozdzal, and Adriana Romero Soriano. Instance-conditioned gan. *Advances in Neural Information Processing Systems*, 34, 2021.

Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5799–5809, 2021.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020.

Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14453–14462, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pp. 248–255. Ieee, 2009.

Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *Advances in Neural Information Processing Systems*, 33:13285–13296, 2020.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9588–9597, 2021.

Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A Efros. Blobgan: Spatially disentangled scene representations. *arXiv preprint arXiv:2205.02837*, 2022.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *2017 International Conference on 3D Vision (3DV)*, pp. 402–411. IEEE, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pp. 2672–2680, 2014.

Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, pp. 5767–5777, 2017.

Paul Henderson, Vagia Tsiminaki, and Christoph H Lampert. Leveraging 2d data to learn textured 3d mesh generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7498–7507, 2020.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, pp. 6626–6637, 2017.

Danilo Jimenez Rezende, SM Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. *Advances in neural information processing systems*, 29, 2016.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pp. 4401–4410, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019.

Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized gan training for high-fidelity few-shot image synthesis. In *International Conference on Learning Representations*, 2020.

Haozhe Liu, Bing Li, Haoqian Wu, Hanbang Liang, Yawen Huang, Yuexiang Li, Bernard Ghanem, and Yefeng Zheng. Combating mode collapse in gans via manifold entropy estimation. *arXiv preprint arXiv:2208.12055*, 2022.

Sebastian Lunz, Yingzhen Li, Andrew Fitzgibbon, and Nate Kushman. Inverse graphics gan: Learning to generate 3d shapes from unstructured 2d data. *arXiv preprint arXiv:2002.12674*, 2020.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, pp. 2794–2802, 2017.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *International Conference on Machine Learning*, pp. 7176–7185. PMLR, 2020.

Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7588–7597, 2019.

Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Advances in Neural Information Processing Systems*, 33:6767–6778, 2020.

Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *2021 International Conference on 3D Vision (3DV)*, pp. 951–961. IEEE, 2021a.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021b.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2015.

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. *Advances in Neural Information Processing Systems*, 31, 2018.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, pp. 2234–2242, 2016.

Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. *Advances in Neural Information Processing Systems*, 34:17480–17492, 2021.

Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, pp. 1–10, 2022.

Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33: 20154–20166, 2020.

Samarth Sinha, Zhengli Zhao, Anirudh Goyal ALIAS PARTH GOYAL, Colin A Raffel, and Augustus Odena. Top-k training of gans: Improving gan performance by throwing away bad samples. *Advances in Neural Information Processing Systems*, 33:14638–14649, 2020.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In *ICLR*, 2015.

Yan Wu, Jeff Donahue, David Balduzzi, Karen Simonyan, and Timothy Lillicrap. Logan: Latent optimisation for generative adversarial networks. *arXiv preprint arXiv:1912.00953*, 2019.

Ceyuan Yang, Yujun Shen, Yinghao Xu, and Bolei Zhou. Data-efficient instance generation from instance discrimination. *Advances in Neural Information Processing Systems*, 34, 2021.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

Hao Zhang, Alexander C Berg, Michael Maire, and Jitendra Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 2126–2136. IEEE, 2006.

Xiaoming Zhao, Fangchang Ma, David Güera, Zhile Ren, Alexander G Schwing, and Alex Colburn. Generative multiplane images: Making a 2d gan 3d-aware. *arXiv preprint arXiv:2207.10642*, 2022.

Zhiming Zhou, Han Cai, Shu Rong, Yuxuan Song, Kan Ren, Weinan Zhang, Yong Yu, and Jun Wang. Activation maximization generative adversarial nets. In *ICLR*, 2018.