FAKE CAN BE REAL IN GANS

Anonymous authors
Paper under double-blind review

ABSTRACT

In order to alleviate the notorious mode collapse phenomenon in generative adversarial networks (GANs), we propose a novel training method of GANs in which certain fake samples can be reconsidered as real ones during the training process. This strategy can reduce the gradient value that generator receives in the region where gradient exploding happens. We show that the theoretical equilibrium between the generators and discriminations actually can be seldom realized in practice. And this results in an unbalanced generated distribution that deviates from the target one, when fake datepoints overfit to real ones, which explains the non-stability of GANs. We also prove that, by penalizing the difference between discriminator outputs and considering certain fake datapoints as real for adjacent real and fake sample pairs, gradient exploding can be alleviated. Accordingly, a modified GAN training method is proposed with a more stable training process and a better generalization. Experiments on different datasets verify our theoretical analysis.

1 INTRODUCTION

In the past few years, Generative Adversarial Networks (GANs) Goodfellow et al. (2014) have been one of the most popular topics in generative models and achieved great success in generating diverse and high-quality images recently (Brock et al. (2019);Karras et al. (2019);Donahue & Simonyan (2019)). GANs are powerful tools for learning generative models, which can be expressed as a zero-sum game between two neural networks. The generator network produces samples from the arbitrary given distribution, while the adversarial discriminator tries to distinguish between real data and generated data. Meanwhile, the generator network tries to fool the discriminator network by producing plausible samples which are close to real samples. When a final theoretical equilibrium is achieved, discriminator can never distinguish between real and fake data. However, we show that a theoretical equilibrium often can not be achieved with discrete finite samples in datasets during the training process in practice.

Although GANs have achieved remarkable progress, numerous researchers have tried to improve the performance of GANs from various aspects (Arjovsky et al. (2017);Nowozin et al. (2016);Gulrajani et al. (2017); Miyato et al. (2018)) because of the inherent problem in GAN training, such as unstability and mode collapse. Arora et al. (2017) showed that a theoretical generalization guarantee does not be provided with the original GAN objective and analyzed the generalization capacity of neural network distance. The author argued that for a low capacity discriminator, it can not provide generator enough information to fit the target distribution owing to lack of ability to detect mode collapse. Thanh-Tung et al. (2019) argued that poor generation capacity in GANs comes from discriminators trained on discrete finite datasets resulting in overfitting to real data samples and gradient exploding when generated datapoints approach real ones. As a result, Thanh-Tung et al. (2019) proposed a zero-centered gradient penalty on linear interpolations between real and fake samples (GAN-0GP-interpolation) to improve generalization capability and prevent mode collapse resulted from gradient exploding. Recent work Wu et al. (2019) further studied generalization from a new perspective of privacy protection.

In this paper, we focus on mode collapse resulted from gradient exploding studied in Thanh-Tung et al. (2019) and achieve a better generalization with a much more stable training process. Our contributions are as follows:

Table 1: NOTATIONS

Symbol	Meaning
p_r	the target dsitribution
p_g	the model distribution
d_x	the dimensionality of a data sample
D	discriminator with sigmoid function in the last layer
D_0	discriminator with sigmoid function in the last layer removed
$D_r = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$	the set of n real samples
$D_g = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\}$	the set of m generated samples
$D_f = \{\boldsymbol{f}_1, \cdots, \boldsymbol{f}_m\}$	the candidate set of M_1 generated samples to be selected as real
$D_{FAR} \subset \{\boldsymbol{f}_1, \cdots, \boldsymbol{f}_m\}$	the set of M_0 generated samples considered as real

1. We show that a theoretical equilibrium, when optimal discriminator outputs a constant for both real and generated data, is unachievable for an empirical discriminator during the training process. Due to this fact, it is possible that gradient exploding happens when fake datapoints approach real ones, resulting in an unbalanced generated distribution that deviates from the target one.
2. We show that when generated datapoints are very close to real ones in distance, penalizing the difference between discriminator outputs and considering fake as real can alleviate gradient exploding to prevent overfitting to certain real datapoints.
3. We show that when more fake datapoints are moved towards a single real datapoint, gradients of the generator on fake datapoints very close to the real one can not be reduced, which partly explains the reason of a more serious overfitting phenomenon and an increasingly unbalanced generated distribution.
4. Based on the zero-centered gradient penalty on data samples (GAN-0GP-sample) proposed in Mescheder et al. (2018), we propose a novel GAN training method by considering some fake samples as real ones according to the discriminator outputs in a training batch to effectively prevent mode collapse. Experiments on synthetic and real world datasets verify that our method can stabilize the training process and achieve a more faithful generated distribution.

In the sequel, we use the terminologies of generated samples (datapoints) and fake samples (datapoints) indiscriminately. Tab. 1 lists some key notations used in the rest of the paper.

2 Related works

Unstability. GANs have been considered difficult to train and often play an unstable role in training process Salimans et al. (2016). Various methods have been proposed to improve the stability of training. A lot of works stabilized training with well-designed structures (Radford et al. (2015);Karras et al. (2018); Zhang et al. (2019);Chen et al. (2019)) and utilizing better objectives (Nowozin et al. (2016);Zhao et al. (2016);Arjovsky et al. (2017);Mao et al. (2017)). Gradient penalty to enforce Lipschitz continuity is also a popular direction to improve the stability including Gulrajani et al. (2017),Petzka et al. (2018),Roth et al. (2017),Qi (2017). From the theoretical aspect, Nagarajan & Kolter (2017) showed that GAN optimization based on gradient descent is locally stable and Mescheder et al. (2018) proved local convergence for simplified zero-centered gradient penalties under suitable assumptions. For a better convergence, a two time-scale update rule (TTUR) (Heusel et al. (2017)) and exponential moving averaging (EMA) (Yazıcı et al. (2019)) have also been studied.

Mode collapse. Mode collapse is another persistent essential problem for the training of GANs, which means lack of diversity in the generated samples. The generator may sometimes fool the discriminator by producing a very small set of high-probability samples from the data distribution. Recent work (Arora et al. (2017);Arora et al. (2018)) studied the generalization capacity of GANs and showed that the model distributions learned by GANs do miss a significant number of modes.

A large number of ideas have been proposed to prevent mode collapse. Multiple generators are applied in Arora et al. (2017),Ghosh et al. (2018),Hoang et al. (2018) to achieve a more faithful distribution. Mixed samples are considered as the inputs of discriminator in Lin et al. (2018),Lucas et al. (2018) to convey information on diversity. Recent work He et al. (2019) studied mode collapse from probabilistic treatment and Yamaguchi & Koyama (2019) from entropy of distribution.

3 BACKGROUND

In the original GAN Goodfellow et al. (2014), the discriminator D maximizes the following objective:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}\sim p_r}[\log(D(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{y}\sim p_g}[\log(1 - D(\boldsymbol{y}))]. \tag{1}$$

The logistic sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ is usually used in practice, leading to

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}\sim p_r}[\log(\sigma(D_0(\boldsymbol{x})))] + \mathbb{E}_{\boldsymbol{y}\sim p_g}[\log(1 - \sigma(D_0(\boldsymbol{y})))], \tag{2}$$

and to prevent gradient collapse, the generator G maximizes

$$\mathcal{L}_G = \mathbb{E}_{\boldsymbol{y}\sim p_g}[\log(\sigma(D_0(\boldsymbol{y})))], \tag{3}$$

where D_0 is usually represented by a neural network. Goodfellow et al. (2014) showed that the optimal discriminator D in Eqn.1 is $D^*(\boldsymbol{v}) = \frac{p_r(\boldsymbol{v})}{p_r(\boldsymbol{v})+p_g(\boldsymbol{v})}$ for any $\boldsymbol{v} \in supp(p_r) \cup supp(p_g)$. As the training progresses, p_g will be pushed closer to p_r. If G and D are given enough capacity, a global equilibrium is reached when $p_r = p_g$, in which case the best strategy for D on $supp(p_r) \cup supp(p_g)$ is just to output $\frac{1}{2}$ and the optimal value for Eqn.1 is $2\log(\frac{1}{2})$.

With finite training examples in training dataset D_r in practice, an empirical version is applied to approximate Eqn.1, using $\frac{1}{n}\sum_{i=1}^{n}\log(D(\boldsymbol{x_i}))$ to estimate $\mathbb{E}_{\boldsymbol{x}\sim p_r}[\log(D(\boldsymbol{x}))]$ and $\frac{1}{m}\sum_{i=1}^{m}[1 - \log(D(\boldsymbol{y_i}))]$ to estimate $\mathbb{E}_{\boldsymbol{y}\sim p_g}[\log(1 - D(\boldsymbol{y}))]$, where $\boldsymbol{x_i}$ is from the set D_r of n real samples and $\boldsymbol{y_i}$ is from the set D_g of m generated samples, respectively.

Mode collapse in the generator is attributed to gradient exploding in discriminator, according to Thanh-Tung et al. (2019). When a fake datapoint $\boldsymbol{y_0}$ is pushed to a real datapoint $\boldsymbol{x_0}$ and if $|D(\boldsymbol{x_0}) - D(\boldsymbol{y_0})| \geq \epsilon$, is satisfied, the absolute value of directional derivative of D in the direction $\boldsymbol{\mu} = \boldsymbol{x_0} - \boldsymbol{y_0}$ will approach infinity leading to gradient exploding:

$$|(\nabla_{\boldsymbol{\mu}} D)_{\boldsymbol{x_0}}| = \lim_{\boldsymbol{y_0}\overset{\mu}{\to}\boldsymbol{x_0}} \frac{|D(\boldsymbol{x_0}) - D(\boldsymbol{y_0})|}{||\boldsymbol{x_0} - \boldsymbol{y_0}||} \geq \lim_{\boldsymbol{y_0}\overset{\mu}{\to}\boldsymbol{x_0}} \frac{\epsilon}{||\boldsymbol{x_0} - \boldsymbol{y_0}||} = \infty. \tag{4}$$

Since the gradient $\nabla_{\boldsymbol{y_0}} D(\boldsymbol{y_0})$ at $\boldsymbol{y_0}$ outweights gradients towards other modes in a mini-batch, gradient exploding at datapoint $\boldsymbol{y_0}$ will move multiple fake datapoints towards $\boldsymbol{x_0}$ resulting in mode collapse.

4 OPTIMAL DISCRIMINATOR IN EMPIRICAL VERSION

4.1 OPTIMAL DISCRIMINATOR EMPIRICALLY DOES NOT HAVE A THEORETICAL EQUILIBRIUM

Theoretically discriminator outputs a constant $\frac{1}{2}$ when a global equilibrium is reached. However in practice, discriminator can often easily distinguish between real samples and fake samples (Goodfellow et al. (2014);Arjovsky et al. (2017)), making a theoretical equilibrium unachievable. Because the distribution p_r of real data is unknown for discriminator, discriminator will always consider datapoints in the set D_r of real samples as real while D_g of generated samples as fake. Even when generated distribution p_g is equivalent to the target distribution p_r, D_r and D_g is disjoint with probability 1 when they are sampled from two continuous distributions respectively (proposition 1 in Thanh-Tung et al. (2019)). In this case, actually p_g is pushed towards samples in D_r instead of the target distribution. However, we show next because of the fact of an unachievable theoretical equilibrium for empirical discriminator during the training process, an unbalanced distribution would be generated that deviates from the target distribution.

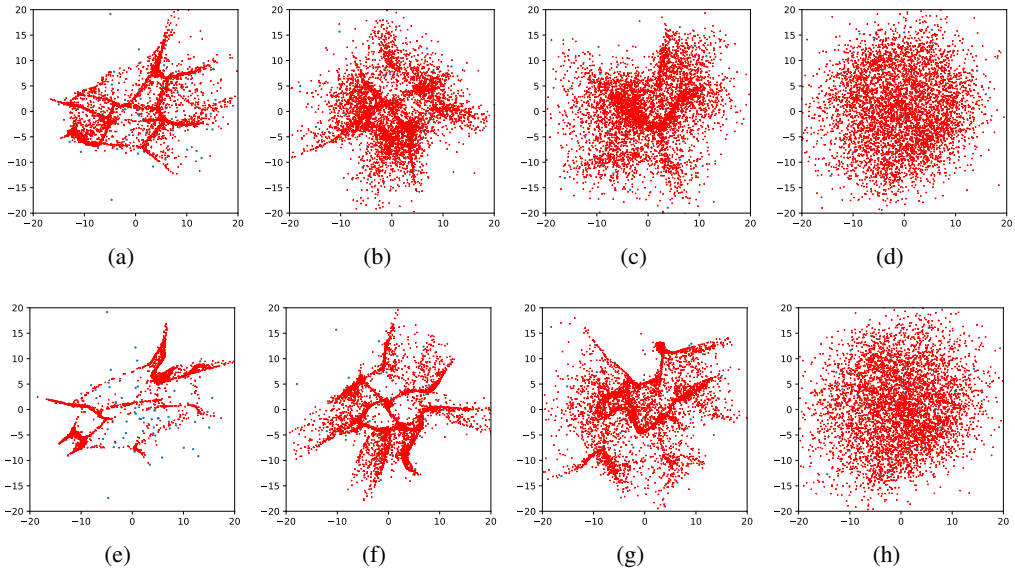


Figure 1: Results on finite samples from a Gaussian distribution of GANs trained with different gradient penalties and our method. (blue datapoints represent real samples and red datapoints represent generated samples) (a).(e) GAN with no GP, iter. 100,000 and 200,000. (b).(f) GAN-0Gp-sample, iter. 100,000 and 200,000. (c).(g) GAN-0Gp-interpolation, iter. 100,000 and 200,000. (d).(h) GAN-0Gp-sample with our method, iter. 100,000 and 200,000.

Proposition 1. *For empirical discriminator in original GAN, unless p_g is a discrete uniform distribution on D_r, the optimal discriminator D output on D_r and D_g is not a constant $\frac{1}{2}$, since there exists a more optimal discriminator which can be constructed as a MLP with $\mathcal{O}(2d_x)$ parameters.*

See Appendix A for the detailed proof. If all the samples in D_r can be remembered by discriminator and generator, and only if generated samples can cover D_r uniformly, which means $D_g = D_r$, a theoretical equilibrium in discriminator can be achieved. However, before generator covers all the samples in D_r uniformly during the training process, the fact of an unachievable theoretical equilibrium makes it possible that there exists a real datapoint $\boldsymbol{x_0}$ with a higher discriminator output than that of a generated datapoint $\boldsymbol{y_0}$. When $\boldsymbol{y_0}$ approaches $\boldsymbol{x_0}$ very closely, gradient exploding and overfitting to a single datapoint happen, resulting an unbalanced distribution and visible mode collapse. See the generated results on a Gaussian dataset of original GAN in Fig. 1a and 1e. The generated distribution neither covers the target Gaussian distribution nor fits all the real samples in D_r, making an unbalanced distribution visible. Furthermore, in practice discriminator and generator are represented by a neural network with finite capacity and dataset D_r is relatively huge, generator can never memorize every discrete sample resulting in a theoretical equilibrium unachievable. In the following subsections, we are interested in the way of stabilizing the output of discriminator to alleviate gradient exploding to achieve a more faithful generated distribution.

4.2 Penalizing the difference between discriminator outputs on close real and fake pairs

Let's first consider a simplified scenario where a fake datapoint $\boldsymbol{y_0}$ is close to a real datapoint $\boldsymbol{x_0}$. Generator updates $\boldsymbol{y_0}$ according to the gradient that the generator receives from the discriminator with respect to the fake datapoint $\boldsymbol{y_0}$, which can be computed as:

$$\nabla_{\boldsymbol{y_0}} \mathcal{L}_G(\boldsymbol{y_0}) \quad = \quad \frac{\partial \log(\sigma(D_0(\boldsymbol{y_0})))}{\partial D_0(\boldsymbol{y_0})} \nabla_{\boldsymbol{y_0}} D_0(\boldsymbol{y_0}) = \frac{\nabla_{\boldsymbol{y_0}} D_0(\boldsymbol{y_0})}{1 + e^{D_0(\boldsymbol{y_0})}}. \tag{5}$$

When $\boldsymbol{y_0}$ approaches $\boldsymbol{x_0}$ very closely and a theoretical discriminator equilibrium is not achieved here, namely $D_0(\boldsymbol{x_0}) - D_0(\boldsymbol{y_0}) \geq \epsilon$, the absolute value of directional derivative $(\nabla_{\boldsymbol{\mu}} D_0)_{\boldsymbol{y_0}}$ in the direction $\boldsymbol{\mu} = \boldsymbol{x_0} - \boldsymbol{y_0}$ at $\boldsymbol{y_0}$ tends to explode and will outweigh directional derivatives in other

Table 2: Changes of different optimal values with variables discussed in Section 4. ($\nearrow$ means increasing and $\searrow$ means decreasing)

	D_0 output on real ϵ_0^*	D_0 output on fake ϵ^*	$\epsilon_0^* - \epsilon^*$	gradient value ∇
$k_0 \nearrow$	$\searrow$	$\nearrow$	$\searrow$	$\searrow$
$m_0 \nearrow$	$\searrow$	$\searrow$	$\nearrow$	$\nearrow$
$p_0 \nearrow$	$\nearrow$	$\nearrow$	$\searrow$	$\searrow$

directions. Hence, the gradient $\nabla_{\boldsymbol{y_0}} D_0(\boldsymbol{y_0})$ is equivalent to directional derivative $(\nabla_{\boldsymbol{\mu}} D_0)_{\boldsymbol{y_0}}$ here. When $\boldsymbol{y_0}$ is very close to $\boldsymbol{x_0}$, the norm of the gradient generator receives from the discriminator with respect to $\boldsymbol{y_0}$ can be computed as:

$$||\nabla_{\boldsymbol{y_0}} \mathcal{L}_G(\boldsymbol{y_0})|| \approx \frac{|D_0(\boldsymbol{x_0}) - D_0(\boldsymbol{y_0})|}{(1 + e^{D_0(\boldsymbol{y_0})})||\boldsymbol{x_0} - \boldsymbol{y_0}||}. \tag{6}$$

If $\boldsymbol{y_0}$ is in the neighborhood of $\boldsymbol{x_0}$, i.e., $\boldsymbol{y_0} \in \mathcal{N}^\delta(\boldsymbol{x_0}) = \{\boldsymbol{y_0} : d(\boldsymbol{x_0}, \boldsymbol{y_0}) \leq \delta, \delta > 0\}$, where δ is a small positive value, we call $\{\boldsymbol{x_0}, \boldsymbol{y_0}\}$ a close real and fake pair. We are interested in reducing the approximated value of the gradient for a fixed pair $\{\boldsymbol{x_0}, \boldsymbol{y_0}\}$ to prevent multiple fake datapoints overfitting to a single real one. Note that the output of D_0 for real datapoint $\boldsymbol{x_0}$ has a larger value than that of fake datapoint $\boldsymbol{y_0}$. So for a fixed pair $\{\boldsymbol{x_0}, \boldsymbol{y_0}\}$, when $D_0(\boldsymbol{y_0})$ increases and $D_0(\boldsymbol{x_0}) - D_0(\boldsymbol{y_0})$ decreases, the target value decreases. And, when $D_0(\boldsymbol{y_0})$ decreases and $D_0(\boldsymbol{x_0}) - D_0(\boldsymbol{y_0})$ increases, the target value increases, according to Eqn. 6.

Now we consider a more general scenario where for a real datapoint $\boldsymbol{x_0}$, in a set of n real samples, there are m_0 generated datapoints $\{\boldsymbol{y_1}, \boldsymbol{y_2}, \cdots, \boldsymbol{y_{m_0}}\}$ very close to $\boldsymbol{x_0}$ in the set of m generated samples. We are specially interested in the optimal discriminator output at $\boldsymbol{x_0}$ and $\{\boldsymbol{y_1}, \boldsymbol{y_2}, \cdots, \boldsymbol{y_{m_0}}\}$. For simplicity, we make the assumption that discriminator outputs at these interested points are not affected by other datapoints in D_r and D_g. We also assume discriminator has enough capacity to achieve optimum in this local region.

However, without any constraint, discriminator will consistently enlarge the gap between outputs for real datapoints and that for generated ones. Thus, an extra constraint is needed to alleviate the difference between discriminator outputs on real and fake datapoints. It comes naturally to penalize the L_2 norm of $D_0(\boldsymbol{x_0}) - D_0(\boldsymbol{y_i})$. Denoting the discriminator output for $\boldsymbol{x_0}$, $D_0(\boldsymbol{x_0})$ as ξ_0 and $D_0(\boldsymbol{y_i})$ as ξ_i, $i = 1, \cdots, m_0$, we have the following empirical discriminator objective:

$$\begin{aligned}
\mathcal{L} &= \frac{1}{n}(\log \sigma(\xi_0) + \sum_{i=2}^{n} \log \sigma(D_0(\boldsymbol{x_i}))) \\
&\quad + \frac{1}{m}(\sum_{i=1}^{m_0} \log(1 - \sigma(\xi_i)) + \sum_{i=m_0+1}^{m} \log(1 - \sigma(D_0(\boldsymbol{y_i})))) \\
&\quad - \frac{k_0}{m_0} \sum_{i=1}^{m_0} (\xi_0 - \xi_i)^2 \\
&= C_1 + \frac{1}{n} f(\xi_0, \xi_0, \cdots, \xi_{m_0}),
\end{aligned} \tag{7}$$

where the interested term $f(\xi_0, \xi_1, \cdots, \xi_{m_0})$ is

$$f(\xi_0, \xi_1, \cdots, \xi_{m_0}) = \log \sigma(\xi_0) + \frac{n}{m} \sum_{i=1}^{m_0} \log(1 - \sigma(\xi_i)) - \frac{n k_0}{m_0} \sum_{i=1}^{m_0} (\xi_0 - \xi_i)^2. \tag{8}$$

Proposition 2. *Assume $\{\xi_0^*, \cdots, \xi_{m_0}^*\}$ is an optimizer that achieves the maximum value of $f(\xi_0, \xi_1, \cdots, \xi_{m_0})$. Then with k_0 increasing, ξ_0^* decreases, ξ_i^* increases and $\xi_0^* - \xi_i^*$ decreases, and, with m_0 increasing, ξ_0^* decreases, ξ_i^* decreases and $\xi_0^* - \xi_i^*$ increases, where $i = 1, \cdots, m_0$.*

Proof. See Appendix B. □

Based on Proposition 2, penalizing the difference between discriminator outputs on close real and fake pairs $\{\boldsymbol{x_0}, \boldsymbol{y_i}\}$ can reduce the norm of $\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})$ from Eqn.6, making it possible to move fake

datapoints to other real datapoints instead of only being trapped at $\boldsymbol{x_0}$. However in practice, it is hard to find the close real and fake pairs to penalize the corresponding difference between discriminator outputs. If we directly penalize the L_2 norm of $D_0(\boldsymbol{x_i}) - D_0(\boldsymbol{y_i})$ when $\{\boldsymbol{x_i}, \boldsymbol{y_i}\}$ are not a pair of close datapoints, $||\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})||$ for $\boldsymbol{y_i}$ may even get larger. Consider $D_0(\boldsymbol{y_i})$ has a higher value than $D_0(\boldsymbol{x_i})$, which could happen when $\boldsymbol{x_i}$ has more corresponding close fake datapoints than the real datapoint $\boldsymbol{x}_{\boldsymbol{y_i}}$ corresponding to $\boldsymbol{y_i}$ from Proposition 2. Direct penalization will make $D_0(\boldsymbol{y_i})$ lower, then $D_0(\boldsymbol{x}_{\boldsymbol{y_i}}) - D_0(\boldsymbol{y_i})$ gets higher and $||\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})||$ higher. Thus in practice we could enforce a zero-centered gradient penalty of the form $||(\nabla D_0)_{\boldsymbol{v}}||^2$ to stabilize discriminator output, where $\boldsymbol{v}$ can be real datapoints or fake datapoints. Although Thanh-Tung et al. (2019) thought that discriminator can have zero gradient on the training dataset and may still have gradient exploding outside the training dataset, we believe a zero-centered gradient penalty can make it harder for discriminator to distinguish between real and fake datapoints and fill the gap between discriminator outputs on close real and fake pairs to prevent overfitting to some extent. Fig. 1b and 1f alleviate overfitting phenomenon compared with no gradient penalty in Fig. 1a and 1e.

Thanh-Tung et al. (2019) proposed another zero-centered gradient penalty of the form $||(\nabla D_0)_{\boldsymbol{v}}||^2$, where $\boldsymbol{v}$ is a linear interpolation between real and fake datapoints, to prevent gradient exploding. However, we consider it's not a very efficient method to fill the gap between discriminator outputs on close real and fake pairs. To begin with, the results of direct linear interpolation between real and fake datapoints may not lie in $supp(p_r) \cup supp(p_g)$. Although the author also considered the interpolation on latent codes, it needs an extra encoder which increases operational complexity. Furthermore, for arbitrary pair of real and fake datapoints, the probability that linear interpolation between them lie where gradient exploding happens is close to 0, because large gradient happens when a fake datapoint approaches closely a real datapoint, resulting in the gap between discriminator outputs on close real and fake pairs hard to fill.

Based on Proposition 2, we also find that when more fake datapoints are moved to the corresponding real datapoint, $||\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})||$ for a fake datpoint $\boldsymbol{y_i}$ only to increase from Eqn.6. It means with the training process going on, more fake datapoints tend to be attracted to one single real datapoint and it gets easier to attract much more fake datapoints to the real one. It partly explains the unstability of GAN training process that especially during the later stage of training, similar generated samples are seen. Compared with Fig. 1a, 1b and 1c at iter.100,000, Fig. 1e, 1f and 1g at iter.200,000 have a worse generalization and much more similar samples are generated with the training process going on.

4.3 Considering fake as real on close real and fake pairs

In this subsection, we aim to make $||\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})||, i = 1, \cdots, m_0$ smaller for optimal empirical discriminator by considering some fake as real on close real and fake pairs based on the above discussions. Suppose for each fake datapoint, it's considered as real datapoint with probability p_0 when training real datapoints, resulting in the following empirical discriminator objective:

$$\begin{aligned}
\mathcal{L} &= \frac{1}{n}(\log \sigma(\xi_0) + \mathbb{E}[A \sum_{i=1}^{m_0} \log \sigma(\xi_i)] + \sum_{i=2}^{n} \log \sigma(D_0(\boldsymbol{x_i}))) \\
&\quad + \frac{1}{m}(\sum_{i=1}^{m_0} \log(1 - \sigma(\xi_i)) + \sum_{i=m_0+1}^{m} \log(1 - \sigma(D_0(\boldsymbol{y_i})))) \\
&\quad - \frac{k_0}{m_0} \sum_{i=1}^{m_0} (\xi_0 - \xi_i)^2 \\
&= C_2 + \frac{1}{n} h(\xi_0, \xi_0, \cdots, \xi_{m_0}),
\end{aligned} \tag{9}$$

where A is a binary random variable taking values in $\{0, 1\}$ with $\Pr(A = 1) = p_0$ and the interested term $h(\xi_0, \xi_1, \cdots, \xi_{m_0})$ is

$$h(\xi_0, \xi_1, \cdots, \xi_{m_0}) = \log \sigma(\xi_0) + p_0 \sum_{i=1}^{m_0} \log \sigma(\xi_i) + \frac{n}{m} \sum_{i=1}^{m_0} \log(1 - \sigma(\xi_i)) - \frac{n k_0}{m_0} \sum_{i=1}^{m_0} (\xi_0 - \xi_i)^2. \tag{10}$$

Proposition 3. *Assume* $\{\xi_0^*, \cdots, \xi_{m_0}^*\}$ *is an optimizer that achieves the maximum value of* $h(\xi_0, \xi_1, \cdots, \xi_{m_0})$. *Then with* p_0 *increasing,* ξ_0^* *increases,* ξ_i^* *increases and* $\xi_0^* - \xi_i^*$ *decreases, where* $i = 1, \cdots, m_0$.

Proof. See Appendix C. □

Note that only penalizing the difference between discriminator outputs on close real and fake pairs in Subsection 4.2 is just a special case of considering fake as real here when $p_0 = 0$. Based on Proposition 3, considering fake datapoints as real with increasing probability p_0 for real datapoints training part can reduce the norm of $\nabla_{\boldsymbol{y_i}} \mathcal{L}_G(\boldsymbol{y_i})$ from Eqn.6. It means when we consider more fake as real where large gradient happens for real training part, the attraction to the real datapoint for fake ones can be alleviate to make it easier to be moved to other real datapoints and prevent the overfitting to one single real datapoint. Note that for a fixed p_0, when the number m_0 of fake datapoints very close to the real one increases, more fake datapoints will be considered as real to alleviate the influences of increasing m_0 discussed in Subsection 4.2.

5 METHOD

To overcome the problem of overfitting to some single datapoints and achieve a better generalization, we propose that fake samples generated by generator in real time can be trained as real samples in discriminator. For original N real samples in a training batch in training process, we substitute them with N_0 real samples in D_r and M_0 generated samples in D_g, where $N = N_0 + M_0$. Our approach is mainly aimed at preventing large gradient in the region where many generated samples overfit one single real sample. To find generated samples in these regions, we choose the generated ones with low discriminator output, owing to the reason that discriminator tends to have a lower output for the region with more generated datapoints approaching one real datapoint from Proposition 2. Therefore, we choose needed M_0 generated samples denoted as set D_{FAR} as real samples from a larger generated set D_f containing M_1 generated samples $\{\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_{M_1}\}$ according to corresponding discriminator output:

$$D_{FAR} = \{\boldsymbol{f}_i\}, i \in \text{index of top } M_0 \text{ in} \{-D_0(\boldsymbol{f}_1), -D_0(\boldsymbol{f}_2), \cdots, -D_0(\boldsymbol{f}_{M_1})\}. \tag{11}$$

We also add a zero-centered gradient penalty on real datapoints Mescheder et al. (2018) based on the discussions in Subsection 4.2, resulting in the following empirical discriminator objective in a batch containing N real samples and M fake samples:

$$\begin{aligned} \mathcal{L}_{FAR} \quad = \quad & \frac{1}{N}[\sum_{i=1}^{N_0} log(\sigma(D_0(\boldsymbol{x}_i))) + \sum_{i=1}^{M_0} log(\sigma(D_0(\boldsymbol{f}_i)))] \\ & + \frac{1}{M} \sum_{i=1}^{M} log(1 - \sigma(D_0(\boldsymbol{y}_i)) + \frac{\lambda}{N} \sum_{i=1}^{N} ||(\nabla D_0)_{\boldsymbol{c}_i}||^2, \end{aligned} \tag{12}$$

where $\boldsymbol{x}_i \in D_r$,$\boldsymbol{f}_i \in D_{FAR}$, $\boldsymbol{y}_i \in D_g$ and $\boldsymbol{c}_i \in D_r \cup D_{FAR}$. In practice, we usually let $N = M$. Because some fake datapoints are trained as real ones, the zero-centered gradient penalty are actually enforced on the mixture of real and fake datapoints.

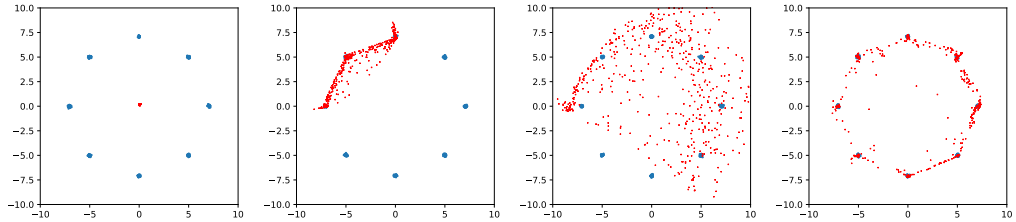
When we sample more generated datapoints for D_f to decide the needed M_0 datapoints as real, the probability of finding the overfitting region with large gradient is higher. When more fake datapoints in D_{FAR} that are close to corresponding real ones are considered as real for training, it is equivalent to increase the value of p_0 in Subsection 4.3. For a larger D_{FAR}, the number of real samples N_0 will decrease for a fixed batchsize N and the speed to cover real ones may be slowed at the beginning of training owning to the reason that some fake datapoints are considered as real and discriminator will be not so confident to give fake ones a large gradient to move them to real ones. Our method can stabilize discriminator output and prevent mode collapse caused by gradient exploding efficiently based on our theoretical analysis. A more faithful generated distribution will be achieved in practice.

6 EXPERIMENTS

6.1 SYNTHETIC DATA

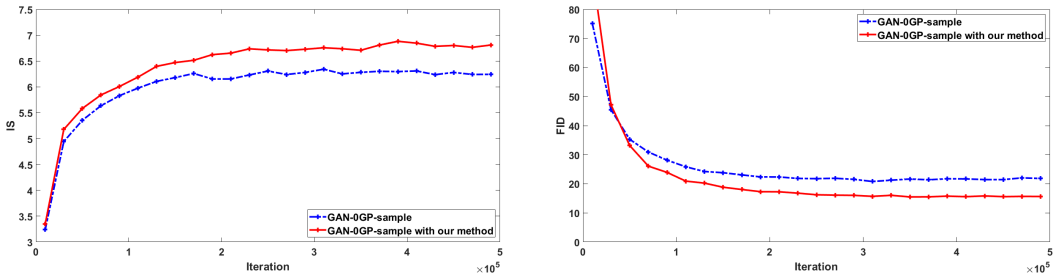
To test the effectiveness of our method in preventing an unbalanced distribution resulted from overfitting to only some real datapoints, we designed a dataset with finite real samples coming from a Gaussian distributions and trained MLP based GANs with different gradient penalties and our method on that dataset. For gradient penalties in all GANs, the weight λ is set 10. Training batch is set 64 and one quarter of the real training batch are generated samples picked from 256 generated samples according to discriminator output, namely $M_0 = 16$ and $M_1 = 256$ in Eqn. 11. Learning rate is set 0.003 for both G and D. The result is shown in Fig.1. It can be observed that original GAN, GAN-0GP-sample and GAN-0GP-interpolation all have serious overfitting problem leading to a biased generated distribution with training process going on, while our method can generate much better samples with good generalization.

We also test our method on a mixture of 8 Gaussians dataset where random samples in different modes are far from each other. The evolution of our method is depicted in Fig.2. We observe that although our method only covers 3 modes at the beginning, it can cover other modes gradually because our method alleviates the gradient exploding on close real and fake datapoints. It is possible that fake datapoints are moved to other Gaussian modes when the attraction to other modes is larger than to the overfitted datapoints. Hence, our method has the ability to find the uncovered modes to achieve a faithful distribution even when samples in high dimensional space are far from each other. More synthetic experiments can be found in Appendix D.



(a) (b) (c) (d)

Figure 2: Evolution of our method on a mixture of 8 Gaussians dataset. (a) iter. 0. (b) iter. 100,000. (c) iter. 335,000. (d) iter. 500,000.

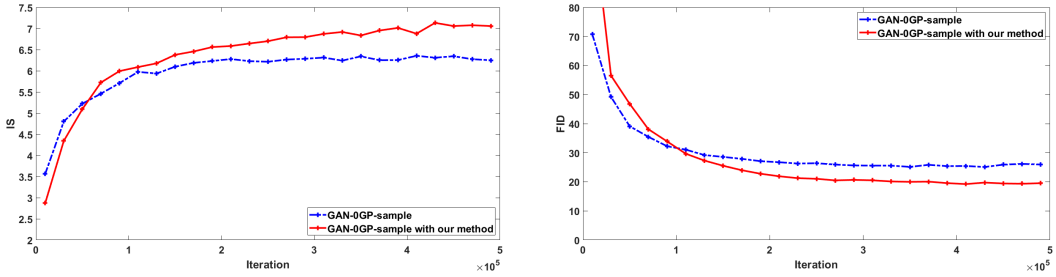


(a) (b)

Figure 3: Inception score and FID on CIFAR-10 of GAN-0GP-sample and GAN-0GP-sample with our method

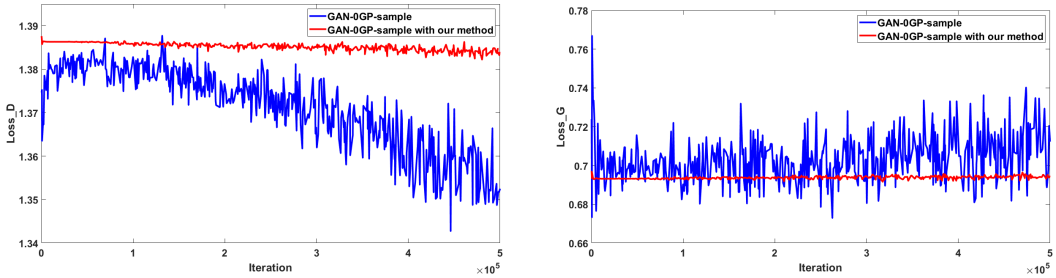
6.2 REAL WORLD DATA

To test our method on real world data, we compare our method with GAN-0GP-sample on CIFAR-10 (Antonio et al. (2008)), CIFAR-100 (Antonio et al. (2008)) and a more challenging dataset ImageNet (Russakovsky et al. (2015)) with ResNet-architectures similar with that in Mescheder et al. (2018). Inception score (Salimans et al. (2016)) and FID (Heusel et al. (2017)) are used as quantitative



(a) (b)

Figure 4: Inception score and FID on CIFAR-100 of GAN-0GP-sample and GAN-0GP-sample with our method



(a) (b)

Figure 5: Losses of discriminator (not including regularization term) and generator on CIFAR-10 of GAN-0GP-sample and GAN-0GP-sample with our method

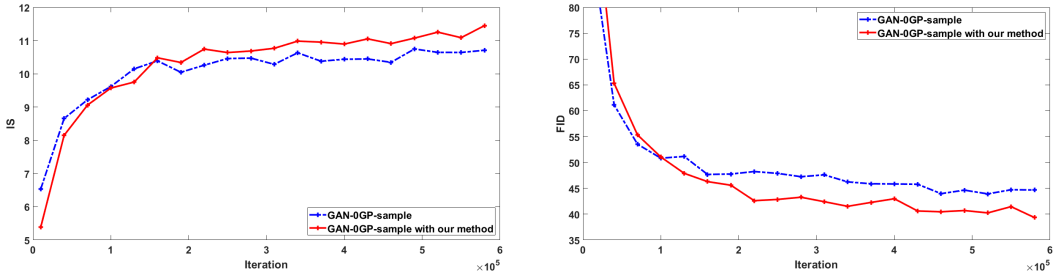
measures. For Inception score, we follow the guideline from Salimans et al. (2016). The FID score is evaluated on $10k$ generated images and statistics of data are calculated at the same scale of generation. Better generation can be achieved with higher inception score and lower FID value. The maximum number of iterations for CIFAR experiment is $500k$, while for ImageNet is $600k$ because of training difficulty with much more modes.We use the code from Mescheder et al. (2018).

The weight λ for gradient penalty is also set 10. Training batch is set 64 and for a better gradient alleviation on close real and fake datapoints, half of the real training batch are generated samples with $M_0 = 32$ and $M_1 = 256$ in Eqn. 11. For CIFAR experiments, we use the RMSProp optimizer with $\alpha = 0.99$ and a learning rate of 10^{-4}. For ImageNet experiments, we use the Adam optimizer with $\alpha = 0$, $\beta = 0.9$ and TTUR with learning rates of 10^{-4} and 3×10^{-4} for the generator and discriminator respectively. We use an exponential moving average with decay 0.999 over the weights to produce the final model.

6.2.1 CIFAR-10 AND CIFAR-100

The results on Inception score and FID are shown in Fig. 3 and 4. Our method outperforms GAN-0GP-sample by a large margin. As predicted in Section 5, the speed of our method to cover real ones could be slowed at the beginning of training with some fake considered as real. However, our method can cover more modes and have a much better balanced generation than the baseline.

The losses of discriminator and generator during the process of CIFAR-10 training are shown in Fig.5. It can be observed that our method has a much more stable training process. Owing to the overfitting to some single datapoints and an unbalanced generated distribution missing modes, the losses of discriminator and generator for GAN-0GP-sample gradually deviate from the optimal theoretical value, namely $2 \log 2 \approx 1.386$ for discriminator and $\log 2 \approx 0.693$ for generator respectively. However, our method has a much more stable output of discriminator to achieve the losses for discriminator and generator very close to theoretical case. It proves practically that our method can stabilize discriminator output on close real and fake datapoints to prevent more datapoints from



(a) (b)

Figure 6: Inception score and FID on ImageNet of GAN-0GP-sample and GAN-0GP-sample with our method

trapped in a local region and has a better generalization. The losses of discriminator and generator on CIFAR-100 and image samples can be found in Appendix D.

6.2.2 IMAGENET

For the challenging ImageNet task, we train GANs to learn a generative model of all 1000 classes at resolution 64×64 with the limitation of our hardware. However, our models are completely unsupervised learning models with no labels used instead of another 256 dimensions being used in latent code z as labels in Mescheder et al. (2018).

The results in Fig. 6 show that our methods still outperforms GAN-0GP-sample on ImageNet. Our method can produce samples of state of the art quality without using any category labels and stabilize the training process. Random selected samples and losses of discriminator and generator during the training process can be found in Appendix D.

7 CONCLUSION

In this paper, we explain the reason that an unbalanced distribution is often generated in GANs training. We show that a theoretical equilibrium for empirical discriminator is unachievable during the training process. We analyze the affection on the gradient that generator receives from discriminator with respect to restriction on difference between discriminator outputs on close real and fake pairs and trick of considering fake as real. Based on the theoretical analysis, we propose a novel GAN training method by considering some fake samples as real ones according to the discriminator outputs in a training batch. Experiments on diverse datasets verify that our method can stabilize the training process and improve the performance by a large margin.

REFERENCES

Torralba Antonio, Fergus Rob, and William T Freeman. 80 million tiny images: a large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(11):1958–1970, 2008.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh (eds.), Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pp. 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In ICML, pp. 224–232, 2017.

Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do GANs learn the distribution? some theory and empirics. In International Conference on Learning Representations, 2018.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In International Conference on Learning Representations, 2019.

Ting Chen, Mario Lucic, Neil Houlsby, and Sylvain Gelly. On self modulation for generative adversarial networks. In International Conference on Learning Representations, 2019.

Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. arXiv preprint arXiv:1907.02544, 2019.

Arnab Ghosh, Viveka Kulharia, Vinay P Namboodiri, Philip HS Torr, and Puneet K Dokania. Multi-agent diverse generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8513–8521, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pp. 2672–2680, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In Advances in neural information processing systems, pp. 5767–5777, 2017.

Hao He, Hao Wang, Guang-He Lee, and Yonglong Tian. Bayesian modelling and monte carlo inference for GAN. In International Conference on Learning Representations, 2019.

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 Advances in Neural Information Processing Systems, pp. 6626–6637, 2017.

Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. In International Conference on Learning Representations, 2018.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In International Conference on Learning Representations, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4401–4410, 2019.

Zinan Lin, Ashish Khetan, Giulia C. Fanti, and Sewoong Oh. Pacgan: The power of two samples in generative adversarial networks. In NeurIPS, pp. 1505–1514, 2018.

Thomas Lucas, Corentin Tallec, Yann Ollivier, and Jakob Verbeek. Mixed batches and symmetric discriminators for gan training. In ICML, pp. 2850–2859, 2018.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, pp. 2794–2802, 2017.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? In Jennifer Dy and Andreas Krause (eds.), Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pp. 3481–3490, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In International Conference on Learning Representations, 2018.

Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In Advances in Neural Information Processing Systems, pp. 5585–5595, 2017.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In Advances in neural information processing systems, pp. 271–279, 2016.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Henning Petzka, Asja Fischer, and Denis Lukovnikov. On the regularization of wasserstein GANs. In International Conference on Learning Representations, 2018.

Guo-Jun Qi. Loss-sensitive generative adversarial networks on lipschitz densities. arXiv preprint arXiv:1701.06264, 2017.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. In Advances in neural information processing systems, pp. 2018–2028, 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3):211–252, 2015.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In NIPS, pp. 2226–2234, 2016.

Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. In International Conference on Learning Representations, 2019.

Bingzhe Wu, Shiwan Zhao, Haoyang Xu, ChaoChao Chen, Li Wang, Xiaolu Zhang, Guangyu Sun, and Jun Zhou. Generalization in generative adversarial networks: A novel perspective from privacy protection. arXiv preprint arXiv:1908.07882, 2019.

Shoichiro Yamaguchi and Masanori Koyama. DISTRIBUTIONAL CONCAVITY REGULARIZATION FOR GANS. In International Conference on Learning Representations, 2019.

Yasin Yazıcı, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, and Vijay Chandrasekhar. The unusual effectiveness of averaging in GAN training. In International Conference on Learning Representations, 2019.

Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In ICML, pp. 7354–7363, 2019.

Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. arXiv preprint arXiv:1609.03126, 2016.

A PROOF FOR PROPOSITION 1

For empirical discriminator, it maximizes the following objective:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}\in D_r}[\log(D(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{y}\in D_g}[\log(1 - D(\boldsymbol{y}))]. \tag{13}$$

When p_g is a discrete uniform distribution on D_r, and generated samples in D_g are the same with real samples in D_r. It is obvious that the discriminator outputs $\frac{1}{2}$ to achieve the optimal value when it cannot distinguish fake samples from real ones.

For continues distribution p_g, Thanh-Tung et al. (2019) has proved that an ϵ-optimal discriminator can be constructed as a one hidden layer MLP with $\mathcal{O}(d_x(m+n))$ parameters, namely $D(\boldsymbol{x}) \geq \frac{1}{2} + \frac{\epsilon}{2}, \forall \boldsymbol{x} \in D_r$ and $D(\boldsymbol{y}) \leq \frac{1}{2} - \frac{\epsilon}{2}, \forall \boldsymbol{y} \in D_g$, where D_r and D_g are disjoint with probability 1. In this case, discriminator objective has a larger value than the theoretical optimal version:

$$\begin{aligned} \mathcal{L} &\geq \mathbb{E}_{\boldsymbol{x}\in D_r}[\log(\frac{1}{2} + \frac{\epsilon}{2})] + \mathbb{E}_{\boldsymbol{y}\in D_g}[\log(\frac{1}{2} + \frac{\epsilon}{2})] \\ &= 2\log(\frac{1}{2} + \frac{\epsilon}{2}) > 2\log\frac{1}{2}. \end{aligned} \tag{14}$$

So the optimal discriminator output on D_r and D_g is not a constant $\frac{1}{2}$ in this case.

Even discriminator has much less parameters than $\mathcal{O}(d_x(m+n))$, there exists a real datapoint $\boldsymbol{x_0}$ and a generated datapoint $\boldsymbol{y_0}$ satisfying $D(\boldsymbol{x_0}) \geq \frac{1}{2} + \frac{\epsilon}{2}$ and $D(\boldsymbol{y_0}) \leq \frac{1}{2} - \frac{\epsilon}{2}$. Whether p_g is a discrete distribution only cover part samples in D_r or a continues distribution, there exists a generated datapoint $\boldsymbol{y_0}$ satisfying $\boldsymbol{y_0} \notin D_r$. Assume that samples are normalized:

$$||\boldsymbol{x_i}|| = ||\boldsymbol{y_i}|| = 1, \forall \boldsymbol{x} \in D_r, \boldsymbol{y} \in D_g. \tag{15}$$

Let $\boldsymbol{W_1} \in \mathbb{R}^{2\times d_x}$, $\boldsymbol{W_2} \in \mathbb{R}^{2\times 2}$ and $\boldsymbol{W_3} \in \mathbb{R}^2$ be the weight matrices, $\boldsymbol{b} \in \mathbb{R}^2$ offset vector and k_1,k_2 a constant, We can construct needed discriminator as a MLP with two hidden layer containing $\mathcal{O}(2d_x)$ parameters. We set weight matrices

$$\boldsymbol{W_1} = \begin{bmatrix} \boldsymbol{x_0}^T \\ \boldsymbol{y_0}^T \end{bmatrix}, \boldsymbol{W_2} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \boldsymbol{W_3} = \begin{bmatrix} \frac{1}{2} + \frac{\epsilon}{2} \\ \frac{1}{2} - \frac{\epsilon}{2} \end{bmatrix}. \tag{16}$$

For any input $\boldsymbol{v} \in D_r \cup D_g$, the discriminator output is computed as:

$$D(\boldsymbol{v}) = \boldsymbol{W_3}^T \sigma(k_2 \boldsymbol{W_2} \sigma(k_1(\boldsymbol{W_1}\boldsymbol{v} - \boldsymbol{b}))), \tag{17}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function. Let $\alpha = \boldsymbol{W_1}\boldsymbol{v} - \boldsymbol{b}$, we have

$$\alpha_1 = \begin{cases} 1 - b_1, \text{if } \boldsymbol{v} = \boldsymbol{x_0} \\ l - b_1, \text{if } \boldsymbol{v} \neq \boldsymbol{x_0} \end{cases}, \alpha_2 = \begin{cases} 1 - b_2, \text{if } \boldsymbol{v} = \boldsymbol{y_0} \\ l - b_2, \text{if } \boldsymbol{v} \neq \boldsymbol{y_0} \end{cases}, \tag{18}$$

where $l < 1$. Let $\beta = \sigma(k_1\alpha)$, we have

$$\beta_1 = \begin{cases} 1, \text{if } \boldsymbol{v} = \boldsymbol{x_0} \\ 0, \text{if } \boldsymbol{v} \neq \boldsymbol{x_0} \end{cases}, \beta_2 = \begin{cases} 1, \text{if } \boldsymbol{v} = \boldsymbol{y_0} \\ 0, \text{if } \boldsymbol{v} \neq \boldsymbol{y_0} \end{cases} \tag{19}$$

as $k_1 \to \infty$ and $b \to 1^-$. Let $\gamma = \sigma(k_2 \boldsymbol{W_2} \beta)$, we have

$$\gamma_1 = \begin{cases} 1, & \text{if } \boldsymbol{v} = \boldsymbol{x_0} \\ 0, & \text{if } \boldsymbol{v} = \boldsymbol{y_0} \\ \frac{1}{2}, & \text{if } \boldsymbol{v} \neq \boldsymbol{x_0}, \boldsymbol{y_0} \end{cases}, \gamma_2 = \begin{cases} 0, & \text{if } \boldsymbol{v} = \boldsymbol{x_0} \\ 1, & \text{if } \boldsymbol{v} = \boldsymbol{y_0} \\ \frac{1}{2}, & \text{if } \boldsymbol{v} \neq \boldsymbol{x_0}, \boldsymbol{y_0} \end{cases} \tag{20}$$

as $k_2 \to \infty$. Hence, for any input $\boldsymbol{v} \in D_r \cup D_g$, discriminator outputs

$$D(\boldsymbol{v}) = \boldsymbol{W_3}^T \gamma = \begin{cases} \frac{1}{2} + \frac{\epsilon}{2}, \text{if } \boldsymbol{v} = \boldsymbol{x_0} \\ \frac{1}{2} - \frac{\epsilon}{2}, \text{if } \boldsymbol{v} = \boldsymbol{y_0} \\ \frac{1}{2}, \text{else} \end{cases}. \tag{21}$$

In this case, discriminator objective also has a more optimal value than the theoretical optimal version:

$$\begin{aligned} \mathcal{L} &= \frac{1}{n}((n-1)\log\frac{1}{2} + \log(\frac{1}{2} + \frac{\epsilon}{2})) + \frac{1}{m}((m-1)\log\frac{1}{2} + \log(\frac{1}{2} + \frac{\epsilon}{2})) \\ &> 2\log\frac{1}{2}. \end{aligned} \tag{22}$$

So the optimal discriminator output on D_r and D_g is also not a constant $\frac{1}{2}$ in this case.

B PROOF FOR PROPOSITION 2

We rewrite $f(\xi_0, \xi_1, \cdots, \xi_{m_0})$ here

$$f(\xi_0, \xi_1, \cdots, \xi_{m_0}) = \log\sigma(\xi_0) + \frac{n}{m}\sum_{i=1}^{m_0}\log(1 - \sigma(\xi_i)) - \frac{nk_0}{m_0}\sum_{i=1}^{m_0}(\xi_0 - \xi_i)^2. \tag{23}$$

To achieve the optimal value, let $f'(\xi_i) = 0, i = 0, \cdots, m_0$ and we have

$$f'(\xi_0) = 1 - \sigma(\xi_0) - \frac{2nk_0}{m_0}\sum_{i=1}^{m_0}(\xi_0 - \xi_i) = 0, \tag{24}$$

$$f'(\xi_i) = -\frac{n}{m}\sigma(\xi_i) + \frac{2nk_0}{m_0}(\xi_0 - \xi_i) = 0, i = 1, \cdots, m_0. \tag{25}$$

It is obvious that $\xi_1 = \xi_2 = \cdots = \xi_{m_0} = \xi$. Hence we have

$$1 - \sigma(\xi_0) - 2nk_0(\xi_0 - \xi) = 0, \tag{26}$$

$$-\frac{n}{m}\sigma(\xi) + \frac{2nk_0}{m_0}(\xi_0 - \xi) = 0. \tag{27}$$

We can solve

$$\xi = -\ln(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1). \tag{28}$$

Substitute Eqn. 28 into Eqn. 26 and we get

$$f'(\xi_0) = \frac{1}{1 + e^{\xi_0}} - 2nk_0(\xi_0 + \ln(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1)) = 0. \tag{29}$$

We can also have from Eqn. 28 and Eqn. 26 respectively

$$\xi_0 - \xi = \xi_0 + \ln(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1), \tag{30}$$

$$= \frac{1}{2nk_0(1 + e^{\xi_0})}. \tag{31}$$

Note that there must exist an optimal ξ_0 satisfying $f'(\xi_0) = 0$ in Eqn. 29, so $\xi_0 + \ln(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1) > 0$ and $f'(\xi_0)$ in Eqn. 29 decreases with k_0 increasing. Also considering that $f'(\xi_0)$ is a monotonically decreasing function on ξ_0, ξ_0^* decreases with k_0 increasing. From Eqn. 28 and Eqn. 30, we know ξ^* increases and $\xi_0^* - \xi^*$ decreases with k_0 increasing. Similarly, note that $f'(\xi_0)$ in Eqn. 29 decreases with m_0 increasing and $f'(\xi_0)$ is a monotonically decreasing function on ξ_0, we have that ξ_0^* decreases with m_0 increasing. From Eqn. 31, we further know that ξ^* decreases and $\xi_0^* - \xi^*$ increases with m_0 increasing.

C PROOF FOR PROPOSITION 3

We rewrite $h(\xi_0, \xi_1, \cdots, \xi_{m_0})$ here

$$h(\xi_0, \xi_1, \cdots, \xi_{m_0}) = \log\sigma(\xi_0) + p_0\sum_{i=1}^{m_0}\log\sigma(\xi_i) + \frac{n}{m}\sum_{i=1}^{m_0}\log(1-\sigma(\xi_i)) - \frac{nk_0}{m_0}\sum_{i=1}^{m_0}(\xi_0 - \xi_i)^2. \tag{32}$$

Let $f'(\xi_i) = 0, i = 0, \cdots, m_0$ and we have $\xi_1 = \xi_2 = \cdots = \xi_{m_0} = \xi$,

$$1 - \sigma(\xi_0) - 2nk_0(\xi_0 - \xi) = 0, \tag{33}$$

$$p_0(1 - \sigma(\xi_0)) - \frac{n}{m}\sigma(\xi) + \frac{2nk_0}{m_0}(\xi_0 - \xi) = 0. \tag{34}$$

We can solve

$$\xi = -\ln\frac{\frac{nm_0}{m}(1 + e^{\xi_0}) - 1}{1 + p_0(1 + e^{\xi_0})}. \tag{35}$$

Substitute Eqn. 35 into Eqn. 33 and we get

$$p_0 = \frac{1}{1 + e^{\xi_0}}[e^{2nk_0\xi_0 - \frac{1}{1+e^{\xi_0}}}(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1)^{2nk_0} - 1] = g(\xi_0). \tag{36}$$

The derivative of $g(\xi_0)$ with respect to ξ_0 is computed as

$$\begin{aligned}(1 + e^{\xi_0})^2 g'(\xi_0) = {} & [e^{2nk_0\xi_0 - \frac{1}{1+e^{\xi_0}}}(2nk_0 + \frac{e^{\xi_0}}{(1 + e^{\xi_0})^2})(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1)^{2nk_0} \\ & + e^{2nk_0\xi_0 - \frac{1}{1+e^{\xi_0}}}2nk_0(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1)^{2nk_0 - 1}\frac{nm_0}{m}e^{\xi_0}]\frac{1}{1 + e^{\xi_0}} \\ & + [e^{2nk_0\xi_0 - \frac{1}{1+e^{\xi_0}}}(\frac{nm_0}{m}(1 + e^{\xi_0}) - 1)^{2nk_0} - 1]\frac{e^{\xi_0}}{(1 + e^{\xi_0})^2}.\end{aligned} \tag{37}$$

Because

$$\frac{nm_0}{m}(1+e^{\xi_0})-1=e^{-\xi}+(1+e^{\xi_0})p_0e^{-\xi}>0 \tag{38}$$

and

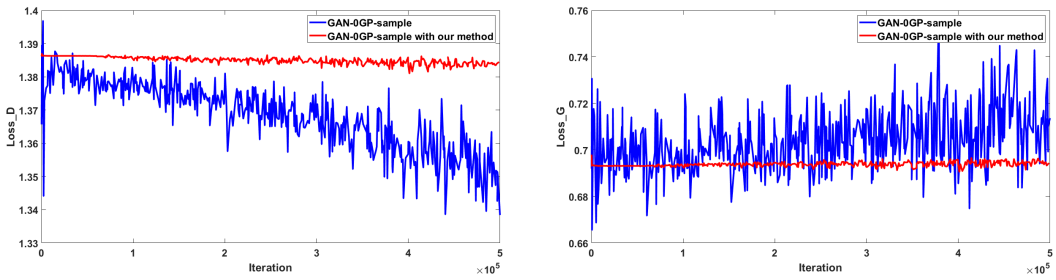
$$\frac{1}{1+e^{\xi_0}}[e^{2nk_0\xi_0-\frac{1}{1+e^{\xi_0}}}(\frac{nm_0}{m}(1+e^{\xi_0})-1)^{2nk_0}-1]=p_0\geq 0, \tag{39}$$

$g'(\xi_0)>0$. Hence ξ_0^* increases with p_0 increasing. From Eqn. 33, we also have

$$\xi_0-\xi=\frac{1}{2nk_0(1+e^{\xi_0})}. \tag{40}$$

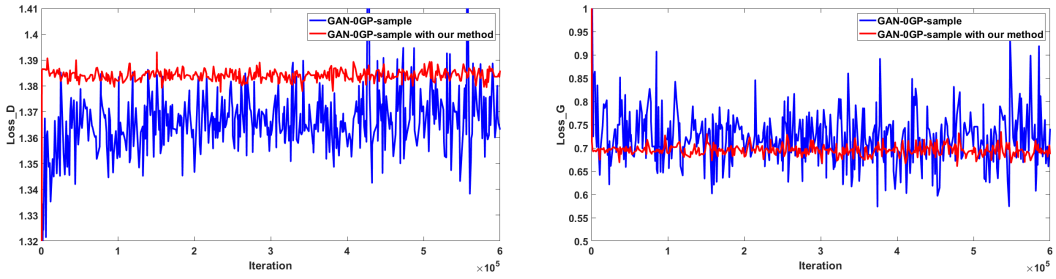
we further know that ξ^* increases and $\xi_0^*-\xi^*$ decreases with p_0 increasing.

D FURTHER RESULTS



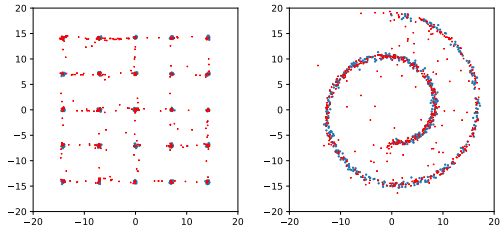
(a) (b)

Figure 7: Losses of discriminator (not including regularization term) and generator on CIFAR-100 of GAN-0GP-sample and GAN-0GP-sample with our method



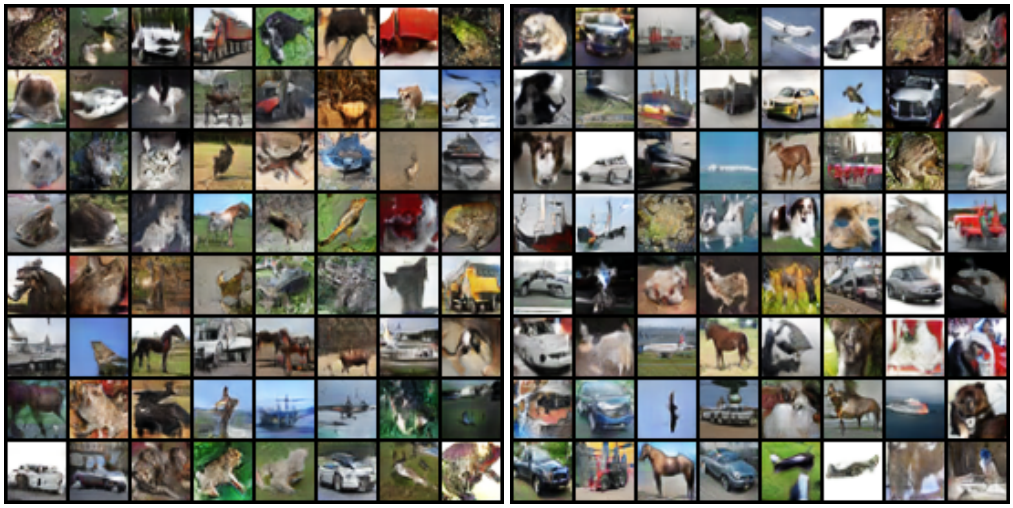
(a) (b)

Figure 8: Losses of discriminator (not including regularization term) and generator on ImageNet of GAN-0GP-sample and GAN-0GP-sample with our method



(a) (b)

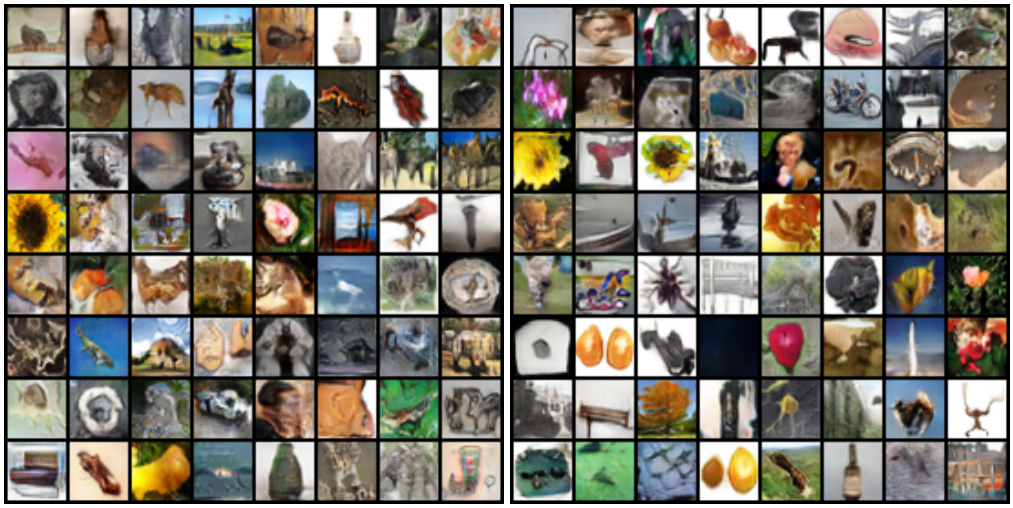
Figure 9: Generation of our method on a mixture of 25 Gaussians dataset and swissroll datatset.



(a) image generation of GAN-0GP-sample

(b) image generation of GAN-0GP-sample with our method

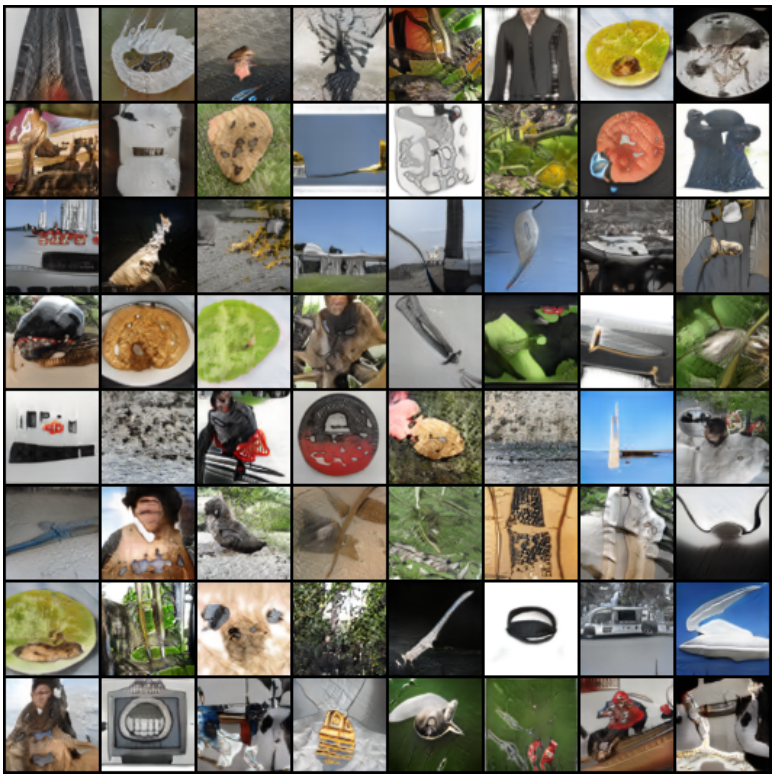
Figure 10: Image generation of CIFAR-10.



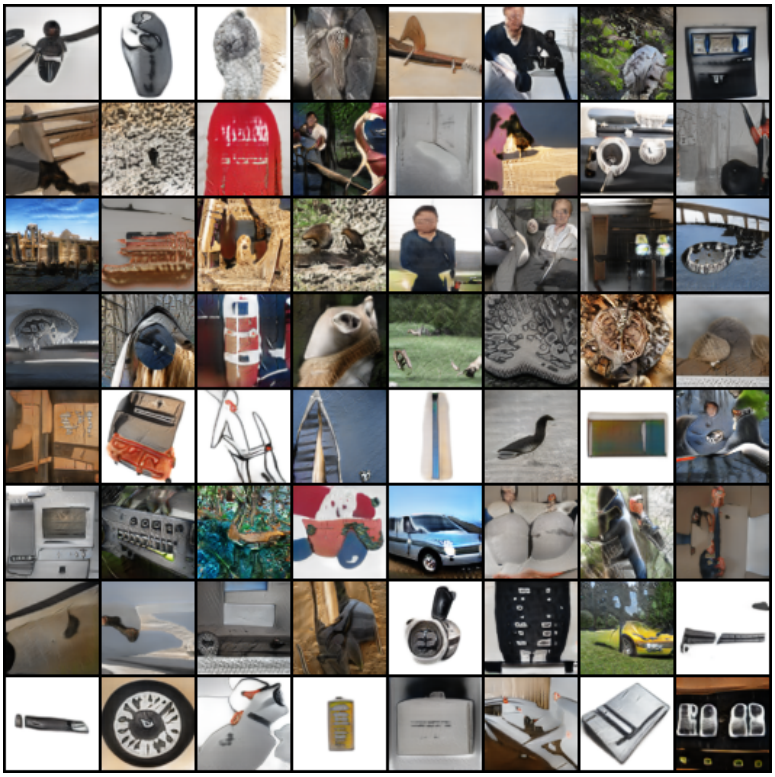
(a) image generation of GAN-0GP-sample

(b) image generation of GAN-0GP-sample with our method

Figure 11: Image generation of CIFAR-100.



(a) image generation of GAN-0GP-sample



(b) image generation of GAN-0GP-sample with our method

Figure 12: Image generation of ImageNet.

E NETWORK ARCHITECTURES

For synthetic experiment, the network architectures are the same with that in Thanh-Tung et al. (2019). While for real world data experiment, we use the similar architectures in Mescheder et al. (2018). We use Pytorch (Paszke et al. (2017)) for development.

Table 3: Generator architecture in synthetic experiment

Layer	output size	filter
Fully connected	64	$2 \rightarrow 64$
RELU	64	-
Fully connected	64	$64 \rightarrow 64$
RELU	64	-
Fully connected	64	$64 \rightarrow 64$
RELU	64	-
Fully connected	2	$64 \rightarrow 2$

Table 4: Discriminator architecture in synthetic experiment

Layer	output size	filter
Fully connected	64	$2 \rightarrow 64$
RELU	64	-
Fully connected	64	$64 \rightarrow 64$
RELU	64	-
Fully connected	64	$64 \rightarrow 64$
RELU	64	-
Fully connected	1	$64 \rightarrow 1$

Table 5: Generator architecture in CIFAR experiment

Layer	output size	filter
Fully connected	$512 \cdot 4 \cdot 4$	$128 \rightarrow 512 \cdot 4 \cdot 4$
Reshape	$512 \times 4 \times 4$	-
Resnet-Block	$256 \times 4 \times 4$	$512 \rightarrow 256 \rightarrow 256$
NN-Upsampling	$256 \times 8 \times 8$	-
Resnet-Block	$128 \times 8 \times 8$	$256 \rightarrow 128 \rightarrow 128$
NN-Upsampling	$128 \times 16 \times 16$	-
Resnet-Block	$64 \times 16 \times 16$	$128 \rightarrow 64 \rightarrow 64$
NN-Upsampling	$64 \times 32 \times 32$	-
Resnet-Block	$64 \times 32 \times 32$	$64 \rightarrow 64 \rightarrow 64$
Conv2D	$3 \times 32 \times 32$	$64 \rightarrow 3$

Table 6: Discriminator architecture in CIFAR experiment

Layer	output size	filter
Conv2D	$64 \times 32 \times 32$	$3 \to 64$
Resnet-Block	$128 \times 32 \times 32$	$64 \to 64 \to 128$
Avh-Pool2D	$128 \times 16 \times 16$	-
Resnet-Block	$256 \times 16 \times 16$	$128 \to 128 \to 256$
Avh-Pool2D	$256 \times 8 \times 8$	-
Resnet-Block	$512 \times 8 \times 8$	$256 \to 256 \to 512$
Avh-Pool2D	$512 \times 4 \times 4$	-
Reshape	$512 \cdot 4 \cdot 4$	-
Fully Connected	1	$512 \cdot 4 \cdot 4 \to 1$

Table 7: Generator architecture in ImageNet experiment

Layer	output size	filter
Fully connected	$1024 \cdot 4 \cdot 4$	$256 \to 1024 \cdot 4 \cdot 4$
Reshape	$1024 \times 4 \times 4$	-
Resnet-Block	$1024 \times 4 \times 4$	$1024 \to 1024 \to 1024$
Resnet-Block	$1024 \times 4 \times 4$	$1024 \to 1024 \to 1024$
NN-Upsampling	$1024 \times 8 \times 8$	-
Resnet-Block	$512 \times 8 \times 8$	$1024 \to 512 \to 512$
Resnet-Block	$512 \times 8 \times 8$	$512 \to 512 \to 512$
NN-Upsampling	$512 \times 16 \times 16$	-
Resnet-Block	$256 \times 16 \times 16$	$512 \to 256 \to 256$
Resnet-Block	$256 \times 16 \times 16$	$256 \to 256 \to 256$
NN-Upsampling	$256 \times 32 \times 32$	-
Resnet-Block	$128 \times 32 \times 32$	$256 \to 128 \to 128$
Resnet-Block	$128 \times 32 \times 32$	$128 \to 128 \to 128$
NN-Upsampling	$128 \times 64 \times 64$	-
Resnet-Block	$64 \times 64 \times 64$	$128 \to 64 \to 64$
Resnet-Block	$64 \times 64 \times 64$	$64 \to 64 \to 64$
Conv2D	$3 \times 64 \times 64$	$64 \to 3$

Table 8: Discriminator architecture in ImageNet experiment

Layer	output size	filter
Conv2D	$64 \times 64 \times 64$	$3 \to 64$
Resnet-Block	$64 \times 64 \times 64$	$64 \to 64 \to 64$
Resnet-Block	$128 \times 64 \times 64$	$64 \to 64 \to 128$
Avh-Pool2D	$128 \times 32 \times 32$	-
Resnet-Block	$128 \times 32 \times 32$	$128 \to 128 \to 128$
Resnet-Block	$256 \times 32 \times 32$	$128 \to 128 \to 256$
Avh-Pool2D	$256 \times 16 \times 16$	-
Resnet-Block	$256 \times 16 \times 16$	$256 \to 256 \to 256$
Resnet-Block	$512 \times 16 \times 16$	$256 \to 256 \to 512$
Avh-Pool2D	$512 \times 8 \times 8$	-
Resnet-Block	$512 \times 8 \times 8$	$512 \to 512 \to 512$
Resnet-Block	$1024 \times 8 \times 8$	$512 \to 512 \to 1024$
Avh-Pool2D	$1024 \times 4 \times 4$	-
Resnet-Block	$1024 \times 4 \times 4$	$1024 \to 1024 \to 1024$
Resnet-Block	$1024 \times 4 \times 4$	$1024 \to 1024 \to 1024$
Fully Connected	1	$1024 \cdot 4 \cdot 4 \to 1$