Reducing Differences Between Real and Realistic Samples to Improve GANs

Shen Zhang, Huaxiong Li*, Yaohui Li, Xianzhong Zhou, Chunlin Chen
Department of Control and Systems Engineering,
School of Management and Engineering, Nanjing University, Nanjing, China
shenzhang@smail.nju.edu.cn, huaxiongli@nju.edu.cn, yaohuili@smail.nju.edu.cn, zhouxz@nju.edu.cn, clchen@nju.edu.cn

2021 IEEE International Conference on Networking, Sensing and Control (ICNSC) | 978-1-6654-4048-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICNSC52481.2021.9702260

***Abstract*—Generative Adversarial Nets (GANs) receive much attention and show great superiority in generating realistic images. However, GANs suffer from mode collapse. To address this problem, we introduce sample differences penalization (SDP) as a regularization term to the objective function of GANs. SDP is an easy-to-implement method that aims to reduce the score differences and the feature differences between the realistic generated samples and their nearest real samples. By introducing SDP, the discriminator presents reasonable outputs to the close pairs. The theoretical analyses demonstrate that SDP can help mitigate the gradient at real samples to some extent, which contributes to a more stable training process. Extensive experiments on real-world datasets including CIFAR-10, CIFAR-100, and Tiny ImageNet demonstrate that our GAN-SDP has a more stable training process and leads to a better performance than existing related methods in Frechet Inception Distance (FID) metric.**

***Index Terms*—Generative adversarial nets, mode collapse, sample differences penalization**

I. Introduction

Generative Adversarial Nets (GANs) [1] have been widely adopted in generative models and shown great superiority in generating various realistic images [2]–[4]. The innovative applications are also increasing, see [5]–[8] . GANs can be regarded as a kind of nonconvex min-max games between generator and discriminator [9]. They can learn to generate complex data distribution without defining data distribution explicitly.

Despite the remarkable performance in generating realistic objects, GANs suffer from several deficiencies, e.g., mode collapse. Therefore, many approaches have been proposed to tackle these limitations and improve the performance of GANs [10]–[13]. [14] introduced an adaptive weighted sum of the two loss parts of the discriminator to benefit the GANs' training stability.

We concentrate on improving the performance and mitigating the mode collapse of GANs. Generally, the discriminator takes samples from the real dataset and the generated dataset as input and tries to judge whether the input is real or fake. Typically, when the output value of the discriminator is larger, the discriminator considers that the input sample is more real. As the training progresses, the generator can gradually generate more and more realistic samples. But the goal of the discriminator is only to tell the fake samples from the real samples no matter how realistic the fake samples are. [15], [16] demonstrated this problem could lead to gradient exploding and mode collapse. [17] considered that the quality of generated samples improved and the traditional positive-negative classification criterion should be changed by regarding some generated samples as real samples according to their quality. [16] regarded certain fake samples with the lowest scores as real ones during the training process and proved that gradient exploding can be mitigated.

*Corresponding author.

In this paper, we aim to mitigate the mode collapse based on the assumption that the realistic generated samples are close to their nearest real samples and the discriminator should present similar outputs to the two samples. To this end, we introduce sample differences penalization (SDP) to the objective function of both the discriminator and the generator as a regularization term. SDP is to penalize the score differences and the feature differences between a realistic generated sample and its nearest real sample. We demonstrate that the gradient exploding and mode collapse can be mitigated by adding SDP so that a better sample quality and a more stable training process can be reached.

We conduct extensive experiments to demonstrate the effectiveness of SDP. Results on various real-world datasets including CIFAR-10, CIFAR-100 and Tiny ImageNet validate that the sample quality and the training stability of GAN-SDP improves compared to existing related methods in Frechet Inception Distance (FID) metric.

II. Related Works

A variety of approaches have been introduced to improve the performance of GANs. Those developments can be divided into three parts: new network architectures, new training strategies, and new loss functions. [10] replaced the net architecture of SGANs (standard GANs) with deep convolutional networks to make the training more stable. [4] proposed a progressive network that GANs generated low-resolution images in the beginning of the training. As the training goes on, the layers of the generator and the discriminator go deeper and deeper so that GANs can achieve the purpose of generating high-resolution images. BigGAN [18] trained a model with a large number of parameters and larger batch sizes and apply orthogonal regularization to the generator, showing a remarkable improvement of sample quality. [19] introduced self-attention into GANs to model long range, multi-level dependencies across image. In order to disentangle the attributes of the images, StyleGAN [2] designed two networks for the generator: Mapping network that controls the style of the generated samples and synthesis network that generates samples given a constant tensor. [20] introduced AutoGAN that applied the neural architecture search (NAS) algorithm to find the proper architecture of GANs.

[9] proposed various techniques to mitigate the non-convergence problem of GANs, which improved both sample generation and semi-supervised learning performance [21] introduced a framework Multi-Agent Diverse GAN that adopted multiple generators and diversity enforcing terms to capture the diverse modes of the real data. [22] proposed a two time-scale update rule (TTUR) that used separate learning rates for the generator and the discriminator in the training. Stationary local Nash equilibrium can be reached based on mild assumptions by adopting TTUR. SN-GAN [23] proposed spectral normalization that normalizes the spectral norm of the weight matrix to meet the Lipschitz constraint and stabilize the training. [19] applied spectral normalization on the generator and TTUR to update the generator and the discriminator.

LSGAN [11] substituted the original loss function of the discriminator with the least square loss function, aiming to minimize the Pearson χ^2 divergence. Hinge loss [24], [25] was introduced to make training more stable. [12] proposed WGAN to minimizes the Wasserstein distance instead of the Jensen-Shannon divergence. [13] introduced WGAN-GP that replaced weight clipping with a one-centered gradient penalty to the discriminator, while [15], [26] proposed a zero-centered gradient penalty to improve the performance of GANs. [27] found that enforcing gradient penalty on points near real samples to avoid mode collapse does make sense. [28] argued that the scores of the real samples should be decreased and introduced RaGAN that estimates the probability that a real sample is more realistic than a generated sample to induce this property. [29] introduced two exponential moving average variables to penalize the output of the discriminator to build more robust GANs under limited data situation.

III. Methodology

A. Sample Differences Penalization

GANs consist of a discriminator D and a generator G, the general training framework of GANs is

$$\begin{aligned}\min_G \max_D V(D,G) =& \mathbb{E}_{\boldsymbol{x}\sim p_d}[f_D(D(\boldsymbol{x}))] \\ &+ \mathbb{E}_{\boldsymbol{z}\sim p_z}[f_G(D(G(\boldsymbol{z})))]. \end{aligned} \tag{1}$$

In practice, the nominal distributions $\hat{p}_d = \sum_{i=1}^m \delta_{\boldsymbol{x}_i}$ and $\hat{p}_z = \sum_{i=1}^m \delta_{\boldsymbol{z}_i}$ are used to estimate p_d and p_z, respectively. The empirical formulation is

$$\begin{aligned}\min_G \max_D \hat{V}(D,G) =& \sum_{i=1}^m [f_D(D(\boldsymbol{x}_i))] \\ &+ \sum_{i=1}^m [f_G(D(G(\boldsymbol{z}_i)))], \end{aligned} \tag{2}$$

where $\boldsymbol{x}_i$ denotes a real sample and $G(\boldsymbol{z}_i)$ denotes a generated sample. $D(\boldsymbol{x}_i)$, $D(G(\boldsymbol{z}_i))$ are the score of $\boldsymbol{x}_i$, $G(\boldsymbol{z}_i)$, respectively. The larger $D(\boldsymbol{x}_i)$ is, the more the discriminator considers $\boldsymbol{x}_i$ is real. The discriminator tries to present a high score to the real sample $\boldsymbol{x}_i$ and present a low score to the fake sample $G(\boldsymbol{z}_i)$ during the training, while the generator tries to improve the score $D(G(\boldsymbol{z}_i))$ by learning to generating more realistic samples.

With the training progressing, the generator learns how to imitate the target distribution and some of the generated samples are realistic [17]. For the discriminator, It still stubbornly tries to distinguish the generated samples despite their high quality, which is not reasonable and may cause gradient exploding and mode collapse, which we will discuss in III-B.

We argue that the discriminator should output similar results when given a realistic sample and a real sample close to it. But the close sample is not easy to find. We regard the nearest real sample of the realistic generated one as the replacement of its close sample. We consider two kinds of differences between a realistic generated sample and its nearest real sample should be reduced: score differences and feature differences. Reducing score differences is to decrease the differences of the outputs of the discriminator between the two samples. By doing this, the discriminator will make similar judgments on the two close samples, which is more reasonable. Reducing feature differences is to decrease the differences of the convolutional features of the discriminator between the two samples. For a close pair, they may share similar patterns, such as geometry and color. When inputted into the discriminator, the convolutional features of the close pair should not be too far apart.

The first thing needed to do is defining what a realistic sample is. We assume that there exists realistic samples after t training iterations. This is reasonable as the generator is not capable of generating realistic samples in the beginning of the training. After t training iterations, we sample a batch of real samples $D_r = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ from the dataset and generate a batch of generated samples $D_g = \{G(\boldsymbol{z}_1), G(\boldsymbol{z}_2), \ldots, G(\boldsymbol{z}_m)\}$. The scores of the generated samples are $S_g = \{D(G(\boldsymbol{z}_1)), D(G(\boldsymbol{z}_2)), \ldots, D(G(\boldsymbol{z}_m))\}$. We consider $\{G(\boldsymbol{z}'_1), G(\boldsymbol{z}'_2), \ldots, G(\boldsymbol{z}'_k)\}$ as realistic generated samples if their scores are the top k of S_g, i.e.

$$\begin{aligned}D_{rg} =& \{G(\boldsymbol{z}'_1), G(\boldsymbol{z}'_2), \ldots, G(\boldsymbol{z}'_k)\} \\ =& \{G(\boldsymbol{z}_i) | D(G(\boldsymbol{z}_i)) \in \text{the top } k \text{ of } S_g\}. \end{aligned} \tag{3}$$

The nearest sample of $G(\boldsymbol{z}_i)'$ among D_r can be found by KNN based on the Euclidean distance

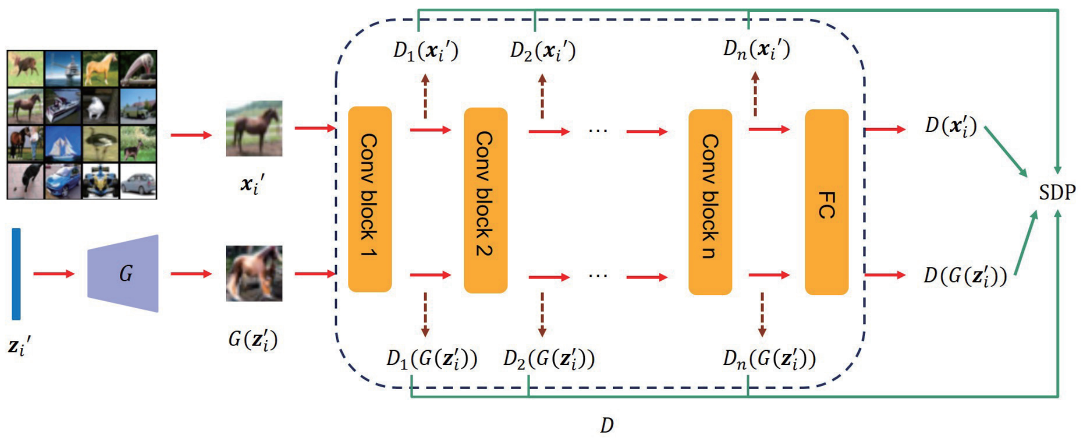


Fig. 1. The framework of GAN-SDP. Given a realistic generated sample and its nearest sample, we extract the feature of the convolutional block and compute the score differences and the feature differences between the two samples to get SDP and add it to the objective function to stabilize the training.

$$\begin{aligned} D_{rn} =& \{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \dots, \boldsymbol{x}'_k\} \\ =& \{\boldsymbol{x}'_i | \boldsymbol{x}'_i = \arg\min_{\boldsymbol{x}\in D_r} ||x - G(\boldsymbol{z}_i)'||_2^2\}. \end{aligned} \tag{4}$$

We reduce the score differences between the realistic generated samples and their nearest real samples by adding the following constraints

$$(D(G(\boldsymbol{z}'_i)) - D(\boldsymbol{x}'_i))^2 \leq \epsilon, \quad i = 1, 2, \dots, k. \tag{5}$$

The constraint (5) ensures the score differences between the realistic samples and the real samples are bounded. Besides, the feature differences should be reduced as well. Suppose the convolutional neural network of the discriminator consists of n convolutional blocks and $D_j(\boldsymbol{x}) \in \mathbb{R}^{C_j \times H_j \times W_j}$ represents the feature map of the j-th convolutional block when given the input $\boldsymbol{x}$. We aggregate the feature maps spatially to get the attention of each channel to the input and reduce the computation cost as well. The feature after aggregation is defined as

$$\hat{D}_j(\boldsymbol{x}) = \frac{f_{ag}(D_j(\boldsymbol{x}))}{||f_{ag}(D_j(\boldsymbol{x}))||_2^2}, \tag{6}$$

where f_{ag} represents an aggregator. We choose global max pooling (GMP) as the aggregator. For the i-th realistic sample and the j-th convolutional block, the feature differences can be reduced by adding the following constraints

$$||\hat{D}_j(G(\boldsymbol{z}'_i)) - \hat{D}_j(\boldsymbol{x}'_i))||_2^2 \leq \tau. \tag{7}$$

(5) and (7) can be considered as a penalty to the GAN training. We define sample differences penalization (SDP) as

$$\begin{aligned} L_{SDP} =& \sum_{i=1}^{k} (D(G(\boldsymbol{z}'_i)) - D(\boldsymbol{x}'_i))^2 \\ &+ \sum_{i=1}^{k}\sum_{j=1}^{n} ||\hat{D}_j(G(\boldsymbol{z}'_i)) - \hat{D}_j(\boldsymbol{x}'_i))||_2^2. \end{aligned} \tag{8}$$

The framework of computing SDP is illustrated in Fig. 1 By adding SDP as a regularization term into (2), the training framework of GANs with can be formulated as

$$\begin{aligned} \max \hat{V}_1(D) =& \sum_{i=1}^{m} [f_D(D(\boldsymbol{x}_i))] + \sum_{i=1}^{m} [f_G(D(G(\boldsymbol{z}_i)))] \\ =& \hat{V}(D, G) - \frac{\lambda}{k} L_{SDP}, \end{aligned} \tag{9}$$

$$\begin{aligned} \min \hat{V}_2(G) =& \sum_{i=1}^{m} [f_D(D(\boldsymbol{x}_i))] + \sum_{i=1}^{m} [f_G(D(G(\boldsymbol{z}_i)))] \\ =& \hat{V}(D, G) + \frac{\lambda}{k} L_{SDP}. \end{aligned} \tag{10}$$

When applying SDP to penalize (2), the discriminator will not strictly distinguish the realistic generated samples from its nearest real samples but distinguishes the generated samples with low quality from the real samples. At the same time, the generator will focus on making the realistic generated samples more realistic. The algorithm of training GANs with SDP is presented in Algorithm 1. Note that k increases periodically since we consider that there will be more realistic generated samples as the training goes on.

B. Theoretical Analysis

In this part, we theoretically demonstrate SDP can mitigate the gradient exploding and mode collapse problem.

Assume $G(\boldsymbol{z}'_i)$ is a realistic generated sample and a real sample $\boldsymbol{x}'_i$ is close to it. The absolute value of directional derivative of D in the direction $\boldsymbol{l} = \boldsymbol{x}'_i - G(\boldsymbol{z}'_i)$ at $\boldsymbol{x}'_i$ is

$$|(\nabla_{\boldsymbol{l}} D)_{\boldsymbol{x}'_i}| = \lim_{G(\boldsymbol{z}'_i) \xrightarrow{\boldsymbol{l}} \boldsymbol{x}'_i} \frac{|D(\boldsymbol{x}'_i) - D(G(\boldsymbol{z}'_i))|}{||\boldsymbol{x}'_i - G(\boldsymbol{z}_i)'||_2}. \tag{11}$$

If D strictly distinguishes the generated samples from real samples, there will be a score gap ϵ between the generated samples and the real samples, which exists in practical settings [30], i.e.

$$(D(G(\boldsymbol{z}'_i)) - D(\boldsymbol{x}'_i))^2 \geq \epsilon. \tag{12}$$

Algorithm 1 Training Process of GAN-SDP

Input: Dataset, Initialized discriminator D and generator G, maximum training iteration number $ITER$, threshold t, the initialized number of realistic samples k, the period of increasing the number of realistic samples I_p, parameter λ.
Output: Generator G.

```
1:  Let $iter = 0$.
2:  while $iter \leq ITER$ do
3:      Sample $m$ real samples $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ from the dataset and $m$ genrated samples $\{G(\boldsymbol{z}_1), G(\boldsymbol{z}_2), \ldots, G(\boldsymbol{z}_m)\}$ from the generator.
4:      if $iter \leq t$ then
5:          Update $D$ and $G$ following the empirical training process (2).
6:      else
7:          Select the top $k$ generated samples $\{G(\boldsymbol{z}'_1), G(\boldsymbol{z}'_2), \ldots, G(\boldsymbol{z}'_k)\}\}$ with the highest score;
8:          Search for the nearest samples $\{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_k\}$ of $\{G(\boldsymbol{z}'_1), G(\boldsymbol{z}'_2), \ldots, G(\boldsymbol{z}'_k)\}$ by KNN based on the Euclidean distance;
9:          Update $D$ by ascending stochastic gradient of (9);
10:         Update $G$ by descending stochastic gradient (10).
11:     end if
12:     $iter = iter + 1$.
13:     if $iter \,\%\, I_p$ == 0 then
14:         $k = k + 1$.
15:     end if
16: end while
17: return $G$.
```

When the realistic sample approaches the real sample, (11) becomes

$$\begin{aligned} |(\nabla_{\boldsymbol{l}} D)_{\boldsymbol{x}'_i}| &= \lim_{G(\boldsymbol{z}'_i) \xrightarrow{\boldsymbol{l}} \boldsymbol{x}'_i} \frac{|D(\boldsymbol{x}'_i) - D(G(\boldsymbol{z}'_i))|}{||\boldsymbol{x}'_i - G(\boldsymbol{z}_i)'||_2} \\ &\geq \frac{\epsilon}{||\boldsymbol{x}'_i - G(\boldsymbol{z}_i)'||_2} = \infty. \end{aligned} \tag{13}$$

The gradient at $\boldsymbol{x}'_i$ will explode because

$$|(\nabla_{\boldsymbol{l}} D)_{\boldsymbol{x}'_i}| = |(\nabla_{\boldsymbol{x}} D)_{\boldsymbol{x}'_i}| \cos\theta \Rightarrow |(\nabla_{\boldsymbol{x}} D)_{\boldsymbol{x}'_i}| = \infty. \tag{14}$$

The gradient exploding means that when the generator pushes the generated samples to $\boldsymbol{x}'_i$, the scores of them will improve more significantly than pushing them to other real samples. The generator will only generate samples that are close to $\boldsymbol{x}'_i$, which leads to mode collapse. SDP contains the penalization of the score differences between the realistic generated samples and their nearest real samples. It can smooth the gradient at $\boldsymbol{x}'_i$ and prevent (12) from happening. The feature differences can guarantee the similarity of the features between the realistic generated samples and their nearest real samples. It plays an auxiliary role in reducing the score differences.

TABLE I
FID SCORE ON CIFAR-10, CIFAR-100.

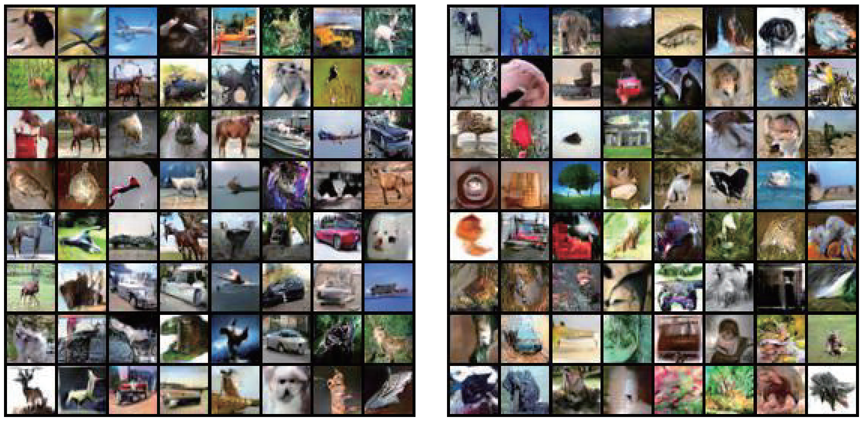
	CIFAR-10	CIFAR-100
Method	FID (↓)	
LSGAN [11]	37.28	39.14
LSGAN-SDP (ours)	**32.58**	38.50
WGAN-GP [13]	38.06	37.26
WGAN-GP-SDP (ours)	**31.52**	**32.05**
WGAN-GP (lr=0.001) [13]	116.69	/
WGAN-GP-SDP (lr=0.001) (ours)	**38.55**	
WGAN-GP (No Norm) [13]	49.74	/
WGAN-GP-SDP (No Norm) (ours)	**34.76**	
HingeGAN [24], [25]	38.20	38.54
HingeGAN-SDP (ours)	**34.96**	**36.39**
SN-GAN [23]	17.45	21.10
SN-GAN-SDP (ours)	**16.79**	**20.74**
AutoGAN [20]	21.62	25.70
AutoGAN-SDP (ours)	**20.13**	**23.46**

IV. EXPERIMENTS

We evaluate our GAN-SDP on three real-world datasets including CIFAR-10 [31], CIFAR-100 [31], and Tiny ImageNet [32]. We choose standard CNN [10] and ResNet [33] as the network architecture. The noise vector $\boldsymbol{z} \in \mathbb{R}^{128}$ follows normal distribution. The size of mini-batch $m = 64$. We iterate 100,000 times to train D and G. Adam is used for optimization, the parameters are set to be $\beta_1 = 0.5$, $\beta_2 = 0.999$ in standard CNN and $\beta_1 = 0$, $\beta_2 = 0.9$ in ResNet. We set threshold $t = 3000$, the initialized number of realistic samples $k = 5$, the period of increasing the number of realistic samples $I_p = 5000$, parameter $\lambda = 0.01$. We choose Frechet Inception Distance (FID) metric [22] to evaluate the quality of generated samples. $10,000$ images are randomly selected from both the dataset and the generator to compute the FID score. Note that a lower FID score means higher quality and diversity. We combine SDP with five GANs: LSGAN [11], WGAN-GP [13], HingeGAN [24], [25], SN-GAN [23], and AutoGAN [20]. For AutoGan, the learning rate for the discriminator is 0.0004 and the learning rate for the generator is 0.0001. For the others, we set the learning rate to be 0.0002. We update discriminator 5 steps per generator update in the training of WGAN-GP and SN-GAN. All experiments are implemented under the framework of Pytorch [34].

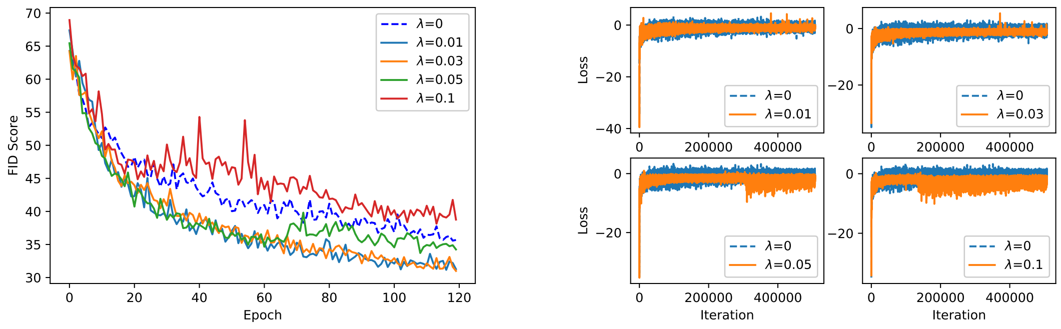
A. *Evaluating Generated Samples and Training*

In this part, we train D and G on three datasets: CIFAR-10, CIFAR-100, and Tiny ImageNet. CIFAR-10 contains 10 classes, $50,000$ training samples and $10,000$ test samples. CIFAR-100 includes 100 classes containing 500 training samples and 100 test samples each. The resolution of each image of CIFAR-10 and CIFAR-100 is 32×32. Tiny ImageNet consists of 200 classes with 500 samples in each class. There are 10,000 images in the test set. The resolution of each image is 64×64. Training difficulty increases on these three datasets due to larger size samples and increasing class numbers. In this experiment, we use training samples to train D and G.



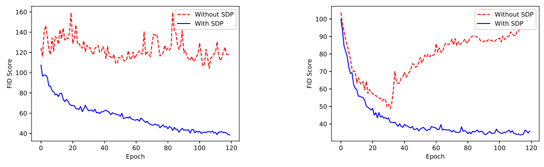
(a) (b)

Fig. 2. Images randomly generated by SN-GAN-SDP on CIFAR-10, CIFAR-100.



(a) (b)

Fig. 3. (a) The FID score of WGAN-GP-SDP with differne λ; (b) The discriminator loss of WGAN-GP-SDP with differne λ.



(a) (b)

Fig. 4. The FID score of WGAN-GP and WGAN-GP-SDP with aggressive seetings on CIFAR-10. (a) lr=0.001; (b) Without normalization layer.

TABLE II
FID SCORE ON TINY IMAGENET.

	Tiny ImageNet
Method	FID (↓)
LSGAN [11]	61.45
LSGAN-SDP (ours)	**53.91**
WGAN-GP [13]	47.79
WGAN-GP-SDP (ours)	**44.96**
HingeGAN [24], [25]	64.10
HingeGAN-SDP (ours)	**56.95**
SN-GAN [23]	51.47
SN-GAN-SDP (ours)	52.48

Table I shows FID metric on CIFAR-10 and CIFAR-100. SDP significantly improves almost all baseline results. Our SN-GAN-SDP achieves the best score on both CIFAR-10 and CIFAR-100. Table II shows the results on Tiny ImageNet. SDP can improve GANs or can help generate images that are of equal quality relative to the basic GAN.

It is worth noting that SDP can effectively improve baseline in aggressive settings. We process two experiments in aggressive settings and Fig. 4 shows the results. When training WGAN-GP with lr=0.001, it falls into mode collapse and cannot converge, while WGAN-GP-SDP can converge and generate higher-quality images. When training WGAN-GP without normalization layer, The training is stable at first, but after about 30 epochs, FID score began to diverge, which means that mode collapse happens. Our WGAN-GP-SDP can still be well trained and converge. Therefore, SDP can effectively mitigate mode collapse and improve the performance of the baseline.

B. The Impact of λ

In this part, we conduct several experiments to illustrate the significance of different λ. From (9), (10), it can be seen that the larger the λ is, the more confidently we regard the top k samples as realistic samples. We conduct five experiments on WGAN-GP setting $\lambda = \{0, 0.01, 0.03, 0.05, 0.1\}$, respectively. The FID metric and the discriminator loss are shown in Fig. IV. It can be seen that when $\lambda = 0.01, 0.03$, WGAN-GP-SDP significantly outperforms WGAN-GP. The performance of WGAN-GP-SDP is the same as WGAN-GP when $\lambda = 0.05$, while WGAN-GP-SDP is surpassed by WGAN-GP when $\lambda = 0.1$. The discriminator losses show that SDP can stabilize the training when $\lambda = 0.01, 0.03$ while destabilize the training when $\lambda = 0.05, 0.1$. Therefore, a proper choice of λ is 0.01 and 0.03.

V. Conclusion

In this paper, we propose a novel regularization term called sample differences penalization (SDP) for GANs. By reducing the score differences and the feature differences between a realistic generated sample and its nearest sample, mode collapse could be alleviated. Extensive experiments on three real-world datasets verify that GAN-SDP has a more stable training process and leads to a better performance than existing related methods.

VI. ACKNOWLEDGMENT

This work was partially supported by National Natural Science Foundation of China (Nos. 62176116, 71732003, 62073160) and the National Key Research and Development Program of China (No. 2018YFB1402600).

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[2] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[3] J. Donahue and K. Simonyan, "Large scale adversarial representation learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 10 542–10 552.

[4] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[5] H. Han, W. Ma, M. Zhou, Q. Guo, and A. Abusorrah, "A novel semi-supervised learning approach to pedestrian reidentification," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 3042–3052, 2021.

[6] J. Pan, C. Li, Y. Tang, W. Li, and X. Li, "Energy consumption prediction of a cnc machining process with incomplete data," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 5, pp. 987–1000, 2021.

[7] L. Chen, X. Hu, W. Tian, H. Wang, D. Cao, and F.-Y. Wang, "Parallel planning: a new motion planning framework for autonomous driving," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 1, pp. 236–246, 2019.

[8] P. Xiang, L. Wang, F. Wu, J. Cheng, and M. Zhou, "Single-image deraining with feature-supervised generative adversarial network," *IEEE Signal Processing Letters*, vol. 26, no. 5, pp. 650–654, 2019.

[9] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *arXiv preprint arXiv:1606.03498*, 2016.

[10] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[11] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2794–2802.

[12] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[14] V. Zadorozhnyy, Q. Cheng, and Q. Ye, "Adaptive weighted discriminator for training generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4781–4790.

[15] H. Thanh-Tung, T. Tran, and S. Venkatesh, "Improving generalization and stability of generative adversarial networks," *arXiv preprint arXiv:1902.03984*, 2019.

[16] S. Tao and J. Wang, "Alleviation of gradient exploding in gans: Fake can be real," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1191–1200.

[17] T. Guo, C. Xu, J. Huang, Y. Wang, B. Shi, C. Xu, and D. Tao, "On positive-unlabeled classification in gan," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8385–8393.

[18] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[19] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7354–7363.

[20] X. Gong, S. Chang, Y. Jiang, and Z. Wang, "Autogan: Neural architecture search for generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3224–3234.

[21] A. Ghosh, V. Kulharia, V. P. Namboodiri, P. H. Torr, and P. K. Dokania, "Multi-agent diverse generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8513–8521.

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.

[23] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[24] D. Tran, R. Ranganath, and D. M. Blei, "Hierarchical implicit models and likelihood-free variational inference," *arXiv preprint arXiv:1702.08896*, 2017.

[25] J. H. Lim and J. C. Ye, "Geometric gan," *arXiv preprint arXiv:1705.02894*, 2017.

[26] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for gans do actually converge?" *arXiv preprint arXiv:1801.04406*, 2018.

[27] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017.

[28] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard gan," *arXiv preprint arXiv:1807.00734*, 2018.

[29] H.-Y. Tseng, L. Jiang, C. Liu, M.-H. Yang, and W. Yang, "Regularizing generative adversarial networks under limited data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7921–7931.

[30] H. Thanh-Tung, T. Tran, and S. Venkatesh, "Improving generalization and stability of generative adversarial networks." in *International Conference on Learning Representations*, 2019.

[31] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[32] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[34] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8026–8037.