Gradient Normalization for Generative Adversarial Networks

Yi-Lun Wu   Hong-Han Shuai   Zhi-Rui Tam   Hong-Yu Chiu
National Yang Ming Chiao Tung University
{w86763777.eed08g, hhshuai, ray.eed08g, james77777778.eed07g}@nctu.edu.tw

Abstract

In this paper, we propose a novel normalization method called gradient normalization (GN) to tackle the training instability of Generative Adversarial Networks (GANs) caused by the sharp gradient space. Unlike existing work such as gradient penalty and spectral normalization, the proposed GN only imposes a hard 1-Lipschitz constraint on the discriminator function, which increases the capacity of the discriminator. Moreover, the proposed gradient normalization can be applied to different GAN architectures with little modification. Extensive experiments on four datasets show that GANs trained with gradient normalization outperform existing methods in terms of both Frechet Inception Distance and Inception Score.

1. Introduction

Generative Adversarial Networks (GANs) [9] have recently achieved great success for synthesizing new data from a given prior distribution, which facilitates a variety of applications, *e.g*., super resolution imaging [15], style transfer between domains [20, 32]. In the original definition, GANs consist of two networks: the *generator* aims to construct realistic samples to fool the discriminator while the *discriminator* learns to discriminate real samples from synthetic samples which are produced by the generator.

Although the state-of-the-art GANs generate high fidelity images that easily fool humans, the unstable training process remains a challenging problem. Therefore, a recent line of study focuses on overcoming the unstable training issue [2, 3, 10, 21, 23]. For example, one of the reasons for the unstable GAN training is caused by the sharp gradient space of the discriminator, which causes the mode collapse in the training process of the generator [23]. Although L2 normalization [14] and weight clipping [3] are simple but effective methods in stabilizing GANs, these additional constraints limit the model capacity of discriminator. As a result, the generator is inclined to fool the discriminator before learning to generate real images. Another kind of popular approaches is to formulate the discriminator as a Lip-

schitz continuous function bounded under a fixed Lipschitz constant K by applying regularization or normalization on discriminator [10, 18, 22, 23]. As such, the discriminator gradient space becomes smoother without significantly sacrificing the performance of the discriminator.

Imposing a Lipschitz constraint on the discriminator can be characterized by three properties. 1) *Model- or module-wise constraint.* If the constraint objective depends on full model instead of the summation of internal modules, we define such methods to be a model-wise constraint and the converse to be a module-wise constraint. For instance, the Gradient penalty (1-GP) [10] is a model-wise constraint, while Spectral Normalization (SN) [18] is a module-wise (layer-wise) constraint, *i.e.*. We argue that model-wise constraints are better since the module-wise constraints limit the layer capacities and thus reduce the multiplicative power of neural networks. 2) *Sampling-based or non-sampling-based constraint.* If the approach requires sampling data from a fixed pool, such method is defined as a sampling-based constraint. For example, 1-GP is a sampling-based constraint due to the regularization, while SN is a non-sampling-based constraint since the normalization only depends on the model architecture. The non-sampling-based methods are expected to perform better than sampling-based methods since the sampling-based constraints might not be effective for the data that are not sampled before. 3) *Hard or soft constraint.* If the Lipschitz constant of any function in the function space of the constrained discriminators is not greater than a fixed finite value, such approach is defined as a hard constraint and the converse as a soft constraint. For example, SN is a hard constraint and the fixed finite value is equal to 1, while 1-GP is a soft constraint since the tightness of the constraint is fluctuated by the regularization and thus the upper bound is not limited. The hard constraint is expected to perform better since the consistent Lipschitz constant can guarantee the gradient stability of unseen data.

To the best of our knowledge, none of the constraints in previous work is model-wise, non-sampling-based, and hard at the same time. In this paper, we propose a new normalization method, named gradient normalization (GN),

to enforce the Lipschitz constant to 1 for the discriminator model through dividing the outputs by the gradient norm of the discriminator. Unlike SN, the proposed Lipschitz constant does not decay from the multiplicative form of neural networks since we consider the discriminator as a general function approximator and the calculated normalization term is independent of internal layers. The proposed gradient normalization enjoys the following two favorable properties. 1) The normalization simultaneously satisfies three properties including model-wise, non-sampling-based and hard constraints, and does not introduce additional hyperparameters. 2) The implementation of GN is simple and compatible with different kinds of network architectures. The contributions of our paper are summarized as follows.

- In this paper, we propose a novel gradient normalization for GANs to strike a good balance between stabilizing the training process and increasing the ability of generation. To the best of our knowledge, this is the first work simultaneously satisfying the above-mentioned three properties.

- We theoretically prove that the proposed gradient normalization is 1-Lipschitz constrained. This property helps the generator to avoid the gradient explosion or vanishing, and thus stabilizes the training process.

- Experimental results show the proposed gradient normalization consistently outperforms the state-of-the-art methods with the same GAN architectures in terms of both Inception Score and Frechet Inception Distance. Our implementation is available at: `https://github.com/basiclab/GNGAN-PyTorch`.

2. Related Work

Previous works for addressing the issues of stabilizing GANs training mainly focus on two kinds of approaches: regularization and normalization of the discriminator. The idea behind these solutions is preventing the discriminator from generating sharp gradients during the training process. Specifically, regularization-based methods add regularization terms into the optimization process for stabilizing the training. For example, gradient penalty-based approaches [10, 23, 27, 28] compute the gradient norm from random samples as the penalty term. Lipschitz regularization [22] approximates maximum perturbation by power iteration. Moreover, consistency regularization [31] regularizes the discriminator outputs of two augmented samples. Orthogonal regularization [4] forces the square of layer weights to be an identity matrix.

Normalization-based methods normalize each layer by different kinds of layer norms. For example, spectral normalization-based approaches [18, 16, 12] normalize layer weights by the spectral norm. Furthermore, weight normalization [29] normalizes layer weights by L2 norm. It is worth noting that we observed that all the normalization-based approaches are non-sampling-based and thus are more general in stabilizing the training of GANs than regularization-based methods. For example, spectral normalization divides the discriminator weight matrix by the largest singular value so that the weight matrix satisfies the Lipschitz constant constraint around 1 for all inputs. However, due to the multiplicative nature of neural networks, the final Lipschitz constant of the discriminator is decayed to a small value if the Lipschitz constant of each layer is smaller than 1. This limitation restricts the capacity of the discriminator and thus deteriorates the performance of the generative model. It is worth noting that GANs are inclined to collapse when using spectral normalization with Wasserstein loss, which is also shown in the comments from the author of spectral normalization [17]. We believe that the failure is caused by the imprecise dual form solution of WGAN since the discriminator function has a limited Lipschitz constant.

3. Preliminaries

3.1. Generative Adversarial Networks

The game between generator $G : \mathbb{R}^{d_z} \to \mathbb{R}^n$ and discriminator $D : \mathbb{R}^n \to \mathbb{R}$ can be formulated as a minimax objective [9], *i.e.*,

$$\min_G \max_D \ \mathbb{E}_{x \sim p_r(x)} \big[log(D(x)) \big] + \\ \mathbb{E}_{\tilde{x} \sim p_g(x)} \big[log(1 - D(\tilde{x})) \big], \tag{1}$$

where $p_r(x)$ is the distribution of real data and $p_g(x)$ is the distribution defined by $p_g = G_*(p_z)$ ($*$ is the push-forward measure and p_z is the d_z-dimensional prior distribution). In this setting, the generator G is guaranteed to converge to real distribution $p_r(x)$ if discriminator D is always optimal [9]. However, training GANs suffers from many difficulties including but not limited to gradient vanishing and gradient explosion. There are two major reasons causing the unstable training issues. First, optimizing the objective function (1) is equivalent to minimizing Jensen–Shannon divergence (JS-divergence) between $p_g(x)$ and $p_r(x)$ [2, 3]. If $p_g(x)$ and $p_r(x)$ do not overlap each other, the JS-divergence would be a constant value which results in gradient vanishing. Second, finite real samples often make discriminator fall into overfitting [23], which indirectly causes gradient explosion around real samples.

3.2. Wasserstein GAN (WGAN)

WGAN is proposed to optimize GANs by minimizing the Wasserstein-1 distance between $p_g(x)$ and $p_r(x)$, *i.e.*,

$$\min_G \max_{D, L_D \leq 1} \ \mathbb{E}_{x \sim p_r(x)} \big[D(x) \big] - \mathbb{E}_{\tilde{x} \sim p_g(x)} \big[D(\tilde{x}) \big], \tag{2}$$

where L_D is the Lipschitz constant of discriminator D. L_D is defined as follows.

$$L_D := \inf \left\{ L \in \mathbb{R} : |D(x) - D(y)| \le L\|x - y\|, \right.$$
$$\left. \forall x, y \in \mathbb{R}^n \right\}. \quad (3)$$

In other words, L_D is the minimum real number such that:

$$|D(x) - D(y)| \le L_D\|x - y\|, \forall x, y \in \mathbb{R}^n. \quad (4)$$

It is worth noting that the metric $\| \cdot \|$ can be any norm of vector. The discriminator in WGAN aims to approximate Wasserstein distance by maximizing the objective function (2) under the Lipschitz constraint (4). Indeed, different choices of Lipschitz constant L_D do not affect the results since the Lipschitz constant of a function can be easily scaled by multiplying the function by a scaling factor. Moreover, the Wasserstein metric has been proved to be more sensible than KL metrics when the learning distributions are supported by low-dimensional manifolds [3]. Obviously, the approximation error rate is related to the size of discriminator capacity. If a discriminator can search in a larger function space, it can approximate the Wasserstein metric more precisely, and hence makes the generator better for modeling the real distribution. Meanwhile, the Lipschitz constraint limits the steepness of value surface and therefore alleviates the overfitting of the discriminator.

However, it is still a great challenge to achieve the Lipschitz constraint on neural networks since striking a good balance between the Lipschitz constraint and network capacity is a hard task. There are many approaches proposed to achieve this constraint. Some of the approaches directly limit the Lipschitz constant for each layer, but sacrifice the function space and get limited network capacity. On the contrary, weight clipping or regularization [3, 10] allows networks to search in a larger function space, but loosens the constraint. In the next section, we prove that the Lipschitz constant of a layer-wise Lipschitz constraint network is upper-bounded by any of its first k-layer subnetworks and propose a normalization method to solve this issue.

3.3. Notation

Definition 1. *Let $f_K : \mathbb{R}^n \to \mathbb{R}$ be a K-layer network, which can be formulated as a function composed of a bunch of affine transformations:*

$$f_K(x) = \phi_K(\mathbf{W}_K \cdot (\phi_{K-1}(\cdots \mathbf{W}_1 \cdot x + \mathbf{b}_1)) + \mathbf{b}_K)$$
$$= \phi_K(\mathbf{W}_K \cdot f_{K-1}(x) + \mathbf{b}_K), \quad (5)$$

where $\mathbf{W}_K \in \mathbb{R}^{d_K \times d_{K-1}}$ and $\mathbf{b}_K \in \mathbb{R}^{d_K}$ are the parameters of the K-th layer, d_K is the target dimension of the K-th layer and ϕ_K is the non-linear element-wise activation function at layer K. Let $f_k, \forall k \in \{1 \cdots K\}$ denote the first k-layer subnetwork.

In order to analyze the behavior of a layer-wise constrained network, we also define layer-wise Lipschitz networks as follows.

Definition 2. *Let $f_K : \mathbb{R}^n \to \mathbb{R}$ be a K-layer network. f_K is defined to be layer-wise L-Lipschitz constrained if $\exists L_k \le L, \forall k \in \{1 \cdots K\}$ s.t. L_k is the Lipschitz constant of the k-th layer:*

$$\|\mathbf{W}_k \cdot x - \mathbf{W}_k \cdot y\| \le L_k\|x - y\|, \forall x, y \in \mathbb{R}^{d_{k-1}}. \quad (6)$$

Since the definition of the Lipschitz constraint (Inequality (4)) is a pairwise relation which associates only two data points at a time, the sampling-based regularizations based on Inequality (4) cannot ensure that the Lipschitz constraint is tight enough everywhere for the discriminator. Such approaches are sampling-based and soft that may cause the gradient explosion due to the non-smooth decision boundary. Consequently, a non-pairwise condition of the Lipschitz constraint is needed for helping us to associate the Lipschitz constraints with gradients. Therefore, we propose the following lemma which associates the Lipschitz constant and the gradient norm of the discriminator.

Lemma 3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function and L_f be the Lipschitz constant of f. Then the Lipschitz constraint (4) is equivalent to*

$$\|\nabla_x f(x)\| \le L_f, \forall x \in \mathbb{R}^n. \quad (7)$$

Proof. See Appendix A in the supplementary material. $\square$

It is worth noting that the equivalence in Lemma 3 holds if and only if the underlying function f is continuous from the perspective of multivariate functions. Practically, almost all the neural networks have finite discontinuous points and therefore are continuous functions. We extend this observation and give a more useful assumption for characterizing such thorny points.

Assumption 4. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function which is modeled by a neural network, and all the activation functions of network f are piecewise linear. Then the function f is differentiable almost surely.*

This can be heuristically justified as the implementations of neural networks on a computer are subject to numerical error anyway. For complex arithmetic operations, *e.g.*, matrix multiplication, the numerical errors are accumulated and therefore the output values are perturbed to avoid the non-differentiable points.

4. Gradient Normalization

Lemma 3 motivates us to design a normalization technique by directly constraining the gradient norm. We first

Table 1: Summary of different $\zeta(x)$ used in gradient normalization (11). The derivation of the gradient norm is based on the assumption that the activation functions in the discriminator are piecewise linear. Furthermore, the sign ✗ means there is no potential issue of $\hat{f}(x)$ in this condition.

$\zeta(x)$	$|\nabla \hat{f}(x)|$	$f(x) \to \pm\infty$	$\|\nabla f(x)\| \to 0$
0	1	✗	$\hat{f}(x)$ explodes
1	$\frac{\|\nabla f(x)\|}{\|\nabla f(x)\|+1}$	$\hat{f}(x)$ explodes	✗
$|f(x)|$	$\frac{\|\nabla f(x)\|}{\|\nabla f(x)\|+|f(x)|}$	✗	✗

show that the Lipschitz constant of a layer-wise 1-Lipschitz constraint, *e.g.*, SN-GAN, may significantly decrease when the number of layers increases, which inspires the concept of the proposed gradient normalization. In the following, we assume that the activation functions are 1-Lipschitz functions* and prove that the Lipschitz constant of a deeper network is bounded by its shallow network.

Theorem 5. *Let $f_K : \mathbb{R}^n \to \mathbb{R}$ be a layer-wise 1-Lipschitz constrained K-layer network. The Lipschitz constant of the first k-layer network L_{f_k} is upper-bounded by $L_{f_{k-1}}$, i.e.,*

$$L_{f_k} \leq L_{f_{k-1}}, \forall k \in \{2 \cdots K\}. \tag{8}$$

Proof. See Appendix A in the supplementary material. □

In Theorem 5, the equality (8) holds if and only if $\exists x, y \in \mathbb{R}^n$ *s.t.*

$$\frac{\|f_k(x) - f_k(y)\|}{\|x - y\|} = \frac{\|f_{k-1}(x) - f_{k-1}(y)\|}{\|x - y\|} = L_{f_{k-1}}. \tag{9}$$

It is difficult to ensure that the equation holds, especially when the network is optimized by stochastic-gradient-based methods. If the equality does not hold at the k-th layer, the Lipschitz constant of the first k-layer subnetwork is less than the Lipschitz constant of the first $(k-1)$-layer subnetwork. By applying this rule iteratively, we have

$$L_{f_k} < L_{f_{k-1}} < \cdots < L_{f_1} \leq 1. \tag{10}$$

Therefore, the Lipschitz constant may be drastically reduced when the number of layers increases. It is worth noting that this is a potential reason which makes SN-GAN fail to integrate with Wasserstein distance when the gradient-based regularization is not applied. On the other hand, the Lipschitz constrained networks do not need to be layer-wise Lipschitz constrained, i.e., it is possible to build a network which is Lipschitz constrained by a model-wise characteristic of a network instead of a module-wise constraint.

---

*Activation functions commonly-used, *e.g.*, ReLU, Leaky ReLU, SoftPlus, Tanh, Sigmoid, ArcTan, Softsign are 1-Lipschitz functions [26].

Inspired by this observation, we propose a normalization method which strictly limits the Lipschitz constant but also maintains a high discriminator capacity. Specifically, as shown in Theorem 3, the Lipschitz constant is associated with the gradient norm. We then propose a new solution, Gradient Normalization (GN), to make the network search in a function space induced by constraint (7). Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function, the proposed GN normalizes the norm of the gradient $\|\nabla_x f(x)\|$ and bounds $f(x)$ simultaneously:

$$\hat{f}(x) = \frac{f(x)}{\|\nabla_x f(x)\| + \zeta(x)}, \tag{11}$$

where $\zeta(x) : \mathbb{R}^n \to \mathbb{R}$ is a universal term which can be associated with $f(x)$ or a constant to avoid the situation that $|\hat{f}(x)|$ approximates infinity or $\|\nabla_x \hat{f}(x)\|$ approximates 0. Here, we propose to set $\zeta(x)$ as $|f(x)|$ and prove that this gradient normalization is still 1-Lipschitz constrained. After that, the variants of $\zeta(x)$ are discussed for explanations.

Theorem 6. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function which is modeled by a neural network, and all the activation functions of network f are piecewise linear. The normalized function $\hat{f}(x) = f(x)/(\|\nabla_x f(x)\| + |f(x)|)$ is 1-Lipschitz constrained, i.e.,*

$$\|\nabla_x \hat{f}(x)\| = \left| \frac{\|\nabla f\|}{\|\nabla f\| + |f|} \right|^2 \leq 1. \tag{12}$$

Proof. See Appendix A in the supplementary material. □

The potential issues of two basic approaches for $\zeta(x)$, *i.e.*, $\zeta(x) = 0$ and $\zeta(x) = 1$, are summarized in Table 1. Considering the overfitting of a gradient normalized discriminator, the discriminator gives out confident predictions for both real and fake samples. From Eq.(11), the confident predictions are produced by $f(x) \to \pm\infty$ and $\|\nabla_x f(x)\| \to 0$, which are also the conditions shown in Table 1. Since the function norm $|f(x)|$ is not directly related to the gradient norm $\|\nabla_x f(x)\|$, the confident predictions may make normalized gradient norm $\|\nabla_x \hat{f}(x)\|$ and normalized function value $|\hat{f}(x)|$ explode†. To deal with this problem, therefore, we propose the formulation which sets $\zeta(x)$ to $|f(x)|$. In this setting, when the discriminator is saturated due to overfitting, the normalized gradient norm (12) is close to 0. This self-control mechanism prevents the generator from getting an exploded gradient and consequently stabilizes the training process of GANs. The pseudocode of the proposed GN is presented in Algorithm 1.

Gradient Analysis of Gradient Normalization. The gradient of $\hat{f}(x)$ with respect to W_k is derived as follows.

---

†The empirical results will be discussed later in Section 5.5

Algorithm 1 Gradient Normalized GAN (GN-GAN)

Input: generator and discriminator parameter θ_G, θ_D, learning rate α_G and α_D, batch size M, the number of discriminator updates per generator updates N_{dis}, total iterations N

1:    $\hat{D} := D(x)/(\|\nabla_x D(x)\| + |D(x)|)$
2:    **for** $t = 1$ to N **do**
3:      **for** $t_{dis} = 1$ to N_{dis} **do**
4:        **for** $i = 1$ to M **do**
5:          $z \sim p_z, x \sim p_r(x)$
6:          $\mathcal{L}_D^i \leftarrow \hat{D}(G(z)) - \hat{D}(x)$
7:        $\theta_D \leftarrow Adam\big(\frac{1}{2M}\sum_{i=1}^M \mathcal{L}_D^i, \alpha_D, \theta_D\big)$
8:      Sample a batch of latent variable $\{z_i\}_{i=1}^{2M} \sim p_z(z)$
9:      $\theta_G \leftarrow Adam\left(\frac{1}{2M}\sum_{i=1}^{2M}\left(-\hat{D}\big(G(z_i)\big)\right), \alpha_G, \theta_G\right)$

Please note that $\zeta(x)$ is set as $|f(x)|$ and function arguments are ignored here for simplicity.

$$\frac{\partial \hat{f}}{\partial \mathbf{W}_k} = \frac{\partial \hat{f}}{\partial f}\frac{\partial f}{\partial \mathbf{W}_k} + \frac{\partial \hat{f}}{\partial \|\nabla_x f\|}\frac{\partial \|\nabla_x f\|}{\partial \mathbf{W}_k} \tag{13a}$$

$$= \frac{\|\nabla_x f\|}{(\|\nabla_x f\| + |f|)^2}\frac{\partial f}{\partial \mathbf{W}_k} - \frac{f}{(\|\nabla_x f\| + |f|)^2}\frac{\partial \|\nabla_x f\|}{\partial \mathbf{W}_k} \tag{13b}$$

$$= \frac{1}{(\|\nabla_x f\| + |f|)^2}\left(\|\nabla_x f\|\frac{\partial f}{\partial \mathbf{W}_k} - f\frac{\partial \|\nabla_x f\|}{\partial \mathbf{W}_k}\right). \tag{13c}$$

Interestingly, according to Eq.(13c), the gradient normalization is a special form of an adaptive gradient regularization. More specifically, in Eq.(13a), the first term is the gradient of GAN objective which improves the discriminating power, while the second term is the regularization which penalizes the gradient norm of f with the adaptive regularization coefficient. Compared with 0-GP [23] and 1-GP [10], the gradient penalty in GN is more flexible and can automatically negotiate with GAN loss. Consequently, this self-balancing mechanism forces GN to become a hard Lipschitz constraint.

Comparisons of Different Approaches. Table 2 summarizes these three properties of several well-known methods. For the regularization-based methods [4, 10, 23, 31], the constraints are always soft due to the trade-off between the regularization and the objective of GANs. Furthermore, the non-sampling-based approaches [4, 18] consider the network as a function composed of multiple layers and impose the constraints on individual layers to achieve the Lipschitz constraint on the full model. By Theorem 5, these layer-wise constraints potentially reduce the capacity of the dis-

Table 2: Summary of different regularization and normalization techniques.

Method	Model-wise	Non-sampling-based	Hard
Orthogonal [4]		✓	
1-GP [10]	✓		
0-GP [23]	✓		
CR [31]	✓		
SN [18]		✓	✓
(our) GN	✓	✓	✓

criminator and therefore sacrifice the generation quality. In contrast, the gradient normalization does not depend on the specific subset of data and is applicable on the full model. Accordingly, by Theorem 6, such normalization is a model-wise, non-sampling-based and hard constraint.

5. Experiments

To evaluate gradient normalization, we first conduct the experiments of unconditional and conditional image generation on two standard datasets, namely CIFAR-10 dataset [6] and STL-10 dataset [25]. CIFAR-10 dataset contains 60k images of size $(32 \times 32 \times 3)$ which was partitioned into 50k training instances and 10K testing instances. STL-10 dataset is designed for developing unsupervised feature learning, containing 5k training images, 8k testing images and 100k unlabeled set of size $(48 \times 48 \times 3)$. Moreover, we also test the proposed method on two datasets with a higher resolution including CelebA-HQ [13] and LSUN [30] Church Outdoor. CelebA-HQ contains 30k human faces of size $(256 \times 256 \times 3)$ and LSUN Church Outdoor is a subset of the LSUN dataset containing 126k church outdoor scenes of size $(256 \times 256 \times 3)$.

Two popular evaluation metrics for generative models, *i.e.*, Inception Score [21] and Frechet Inception Distance (FID) [11], are used to quantitatively evaluate the proposed method. For a fair comparison, all evaluations are computed by the official implementation of the Inception Score and FID. In order to compare with previous works which did not carefully follow the standard evaluation protocol, our evaluation is designed according to several different configurations for calculating FID [8, 14]. Furthermore, we record the best model checkpoint in terms of FID throughout the training and report the averaged results. For more details of evaluation, see Appendix B in the supplementary material.

5.1. Unconditional Image Generation

Table 3 compares the proposed GN-GAN with several state-of-the art models, *e.g.*, SN-GAN [18], WGAN-GP [10] and CR-GAN [31]. The results manifest that the proposed GN-GAN outperforms existing normalization methods in terms of Inception Score and FID. We also com-

Table 3: Inception Score and FID with unconditional image generation on CIFAR-10 and STL-10. We report the average and standard deviation of the results trained with 5 different random seeds. Note that "-" denotes that result is not reported by the original paper. Moreover, † represents that the original paper does not provide an evaluation on STL-10, so we provide a implementation for reference. ‡ denotes that we provide a re-implementation result for reliable comparison.

Method	CIFAR-10			STL-10		
	Inception Score↑	FID(train)↓	FID(test)↓	Inception Score↑	FID(50k) ↓	FID(10k) ↓
Real data	11.24±.12	0	7.80	26.08±.26	0	0
Standard CNN						
SN-GAN [18]	7.58±.12	-	25.50	8.79±.14	-	43.20
SN-GAN‡	7.86±.09	18.52	22.67	8.87±.09	32.90	35.10
SN-GAN-CR [31]	7.93	-	18.72	8.69±.08†	32.11†	34.14†
(our) GN-GAN	7.71±.14	19.31±.76	23.52±.80	9.00±.15	30.18±.82	32.41±.73
(our) GN-GAN-CR	8.04±.19	18.59±1.5	22.89±1.5	9.00±.14	27.61±.69	29.53±.62
ResNet						
WGAN-GP [10]	7.86±.08	-	-	-	-	-
SN-GAN	8.22±.05	-	21.70±.21	9.10±.04	-	40.10±.50
SN-GAN‡	8.48±.11	12.35	16.59	9.18±.10	29.16	31.85
SN-GAN-CR	8.40	-	14.56	9.38±.07†	25.78†	28.4†
(our) GN-GAN	8.49±.11	11.13±.18	15.33±.16	9.60±.14	26.14±.7	28.12±0.61
(our) GN-GAN-CR	**8.72±.11**	**9.55±.47**	**13.71±.40**	**9.74±.15**	**23.62±.89**	**25.80±0.59**
Neural Architecture Search						
AutoGAN [8]	8.55±.10	12.42	-	9.16±.12	31.01	-
E²GAN [24]	8.51±.13	11.26	-	9.51±.09	25.53	-

bine GN-GAN with consistency regularization (CR) [31], named GN-GAN-CR, by directly replacing spectral normalization with GN. The result indicates that GN-GAN-CR can further improve SN-GAN-CR in both Inception Score and FID by simply replacing the normalization method. It is worth noting that the Wasserstein loss degenerates into hinge loss in the proposed GN-GAN since GN makes the output of network range within $[-1, 1]$. Therefore, we only test our model with two loss functions: hinge loss and non-saturating loss (NS) [9]. We derive the best performance by using hinge loss with ResNet and NS loss with Standard CNN. The architectures and most of optimization parameters in our experiments are the same as [18]. We stop training after 200k generator update steps for all the experiments shown in Table 3. The learning rate of the discriminator is slightly increased from 2×10^{-4} to 4×10^{-4} for ResNet architecture. In GN-GAN-CR, we test regularization weight $\lambda = 0.1, 1, 5, 10$ on CIFAR-10 with ResNet and find that $\lambda = 5$ performs better than the others.

5.2. Conditional Image Generation

We further conduct the experiments on the CIFAR-10 dataset with the same architecture proposed by Big-GAN [1]. Similarly, SN in the discriminator is replaced

Table 4: Inception Score and FID with conditional image generation on CIFAR-10.

Method	Inception Score↑	FID(test)↓
BigGAN	9.22	14.73
BigGAN-CR [31]	-	11.48
(our) GN-BigGAN	9.22±.13	**10.05±.23**

by GN.‡ Table 4 shows that GN can further improve Big-GAN by 31.8% in terms of FID. It is worth noting that we increase the discriminator learning rate from 1×10^{-4} to 2×10^{-4}, which is two times greater than that of the generator during training. This modification is motivated by the self-control mechanism of GN, which makes the outputs of the discriminator saturate thus needing more steps to give confident predictions.

5.3. Unconditional Large Scale Image Generation

To show that the proposed Gradient Normalization is able to generate high-resolution images, we leverage the architecture proposed by SN-GAN for generating 256×256

---

‡Following BigGAN, SN is still used in the generator.

images on CelebA-HQ and LSUN Church Outdoor. Similarly, SN is replaced by GN in the experiments. In Figures 1 and 2, generated results show the competitive quality. Due to the space constraint, more samples and quantitative results are provided in Appendix B of the supplementary materials.

5.4. Experimental Analysis for Theorem 5

According to Theorem 5, the Lipschitz constant decreases as the number of layers increases. To explain the results clear, we first introduce Corollary 1 in WGAN-GP [10], which states that if the GAN is trained with Wasserstein loss, there exists at least one optimal discriminator D^* satisfying $\|\nabla_x D^*(x)\| = 1$ almost everywhere under the support of p_r and p_g. Accordingly, by Lemma 3, the Lipschitz constant for such optimal discriminator is equal to 1 almost surely. We therefore design the experiments to test SN and GN with the Wasserstein loss. Figures 3a and 3b respectively show the Inception Scores and the Lipschitz constants of discriminators with regards to the training iterations for different approaches on CIFAR-10 dataset, where nL means both generator and discriminator are modeled by n convolutional layers and L_D is calculated by sampling 50k data from each of p_d and p_g. However, Figures 3a and 3b show that the generators of SN-GANs do not lead to a high IS and the Lipschitz constant of discriminators are much smaller than 1. Moreover, all the discriminators of SN-GANs cannot well approximate Wasserstein distance under the Lipschitz constraint $L_D \leq 1$ even

if the above corollary guarantees the existence of optimal discriminator. We believe the reason is that the SN over-restricts the Lipschitz constants of discriminators and hence discriminators cannot increase the Lipschitz constants of themselves to approach D^*. This situation becomes more worse when the number of layers is increased (from 3L to 9L). Conversely, the magnitude of Lipschitz constants of GN-GANs is invariant to the depth of model and leads to a better performance, which indeed matches the Theorem 5.

We further investigate the Lipschitz constant of internal layers. Figure 3c shows the Lipschitz constants of GN-9L and SN-9L in the layer level on CIFAR-10 dataset. It is worth noting that GN can achieve 1-Lipschitz constraint without constraining internal layers. Thus, the Lipschitz constant of each layer in GN-9L is unlimited and is more flexible than module-wise approach, *e.g.*, SN-9L. The multiplicative power of discriminators, therefore, is not limited, which is the potential superiority that makes GN converge well with Wasserstein loss.

5.5. Ablation Study

Activation Function. Theorem 6 only provides the upper bound of gradient norm with the assumption that the activation functions of the discriminator are piecewise linear. However, we hypothesize that GN works for most activation functions. Therefore, we reproduce the Standard CNN training reported by SN-GAN [18], WGAN-GP [10] and Vanilla GAN [9] with different activation functions. Figure 4 shows the performance of GAN, WGAN-GP, SN-

Figure 1: Generated samples on CelebA-HQ 256×256. FID=7.67.

Figure 2: Generated smaples on LSUN Church Outdoor 256×256. FID=5.405.
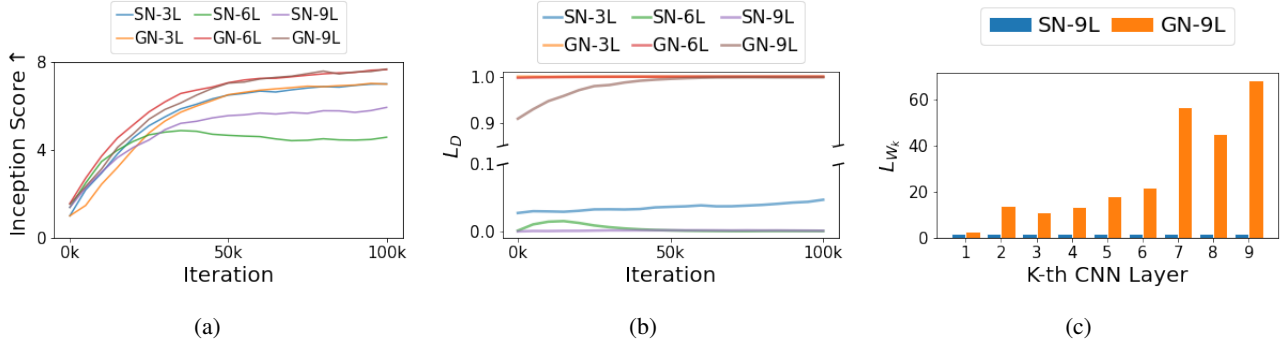
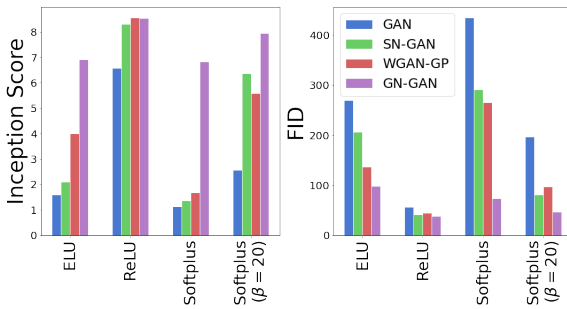Figure 3: Analysis of Theorem 5 on CIFAR-10 dataset. Best viewed in color.

Figure 4: Comparison of activation functions including ELU [5], ReLU [19] and Softplus [7] on STL-10.
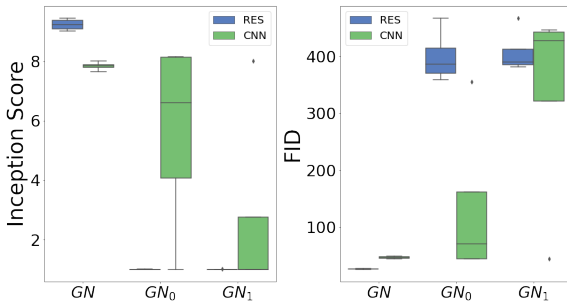

Figure 5: Comparison of variants of gradient normalization (11) on STL-10. The experiments include $\zeta(x) = |f(x)|$ (GN), $\zeta(x) = 1$ (GN_1) and $\zeta(x) = 0$ (GN_0).

GAN and GN-GAN with different activation functions in terms of IS and FID, which indicates that GN achieves the best scores on the ELU and ReLU, while gets competitive scores on Softplus ($\beta = 20$). It is worth noting that Softplus becomes similar to ReLU if the β increases. As such, GN performs better on Softplus($\beta = 20$) than Softplus.

Variants of Gradient Normalization. As discussed in Section 4, we set $\zeta(x)$ to $|f(x)|$ in all experiments for stability issues. Here, we compare three variants of gradient normalization, $\zeta(x) = |f(x)|$ (GN), $\zeta(x) = 1$ (GN_1) and

$\zeta(x) = 0$ (GN_0), by applying them for unconditional image generation. Figure 5 shows the results of three normalization approaches across different model architectures and datasets with the total iteration as 50k. The hyperparameter used for different $\zeta(x)$ is the same as 5.1. Moreover, we repeat the training process 4 times with different random seeds and report the average performance of the last checkpoints. The results manifest that the variance of Inception Score and FID for GN_0 is large for different architectures and datasets. Even if we add a constant to the denominator of (11) by setting $\zeta(x)$ to a non-zero constant 1, the training results of GN_1 are still inferior to the proposed GN. These results match our discussion in Section 4.

6. Conclusion and Future Work

In this paper, we propose a novel gradient normalization method for stabilizing the training of GANs, which can facilitate a variety of applications. The proposed GN is simple to implement and theoretically proven to satisfy the hard Lipschitz constraint and can effectively make any discriminator as a Lipschitz continuous function. Also, we apply GN to several different architectures on different datasets and most of them achieve state-of-the-art results. In the future, we plan to replace the gradient norm of the denominator term with a quasi formulation for further reducing the computation. Another interesting direction is to apply GN to other GAN-related tasks such as style transfer, super-resolution, and video generation.

References

[1] Karen Simonyan Andrew Brock, Jeff Donahue. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2019.

[2] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 214–223, 2017.

[4] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[5] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations (ICLR)*, 2016.

[6] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 215–223, 2011.

[7] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 315–323, 2011.

[8] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, M. Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014.

[10] Ishaan Gulrajani, F. Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, pages 5769–5779, 2017.

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS)*, pages 6629–6640, 2017.

[12] Haoming Jiang, Zhehui Chen, Minshuo Chen, Feng Liu, Dingding Wang, and Tuo Zhao. On computation and generalization of generative adversarial networks under spectrum control. In *International Conference on Learning Representations (ICLR)*, 2019.

[13] 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 (ICLR)*, 2017.

[14] Karol Kurach, M. Lucic, Xiaohua Zhai, M. Michalski, and S. Gelly. A large-scale study on regularization and normalization in gans. In *International Conference on Machine Learning (ICML)*, 2019.

[15] C. Ledig, L. Theis, Ferenc Huszár, J. Caballero, Andrew Aitken, Alykhan Tejani, J. Totz, Zehan Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017.

[16] Kanglin Liu, Wenming Tang, Fei Zhou, and Guoping Qiu. Spectral regularization for combating mode collapse in gans. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[17] Takeru Miyato. Miyato comment. `https://github.com/pfnet-research/sngan_projection/issues/15#issuecomment-392680419`, 2018.

[18] Takeru Miyato, T. Kataoka, Masanori Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[19] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML)*, pages 807–814, 2010.

[20] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1505–1514, 2019.

[21] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2234–2242, 2016.

[22] Dávid Terjék. Adversarial lipschitz regularization. In *International Conference on Learning Representations (ICLR)*, 2020.

[23] Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[24] Yuan Tian, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink. Off-policy reinforcement learning for efficient and effective gan architecture search. In *Proceedings of the 15th European Conference on Computer Vision (ECCV)*, 2020.

[25] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1958–1970, 2008.

[26] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS)*, 2018.

[27] Xiang Wei, Zixia Liu, Liqiang Wang, and Boqing Gong. Improving the improved training of wasserstein GANs. In *International Conference on Learning Representations (ICLR)*, 2019.

[28] Jiqing Wu, Zhiwu Huang, Janine Thoma, Dinesh Acharya, and Luc Van Gool. Wasserstein divergence for gans. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[29] Sitao Xiang and Hao Li. On the effects of batch and weight normalization in generative adversarial networks, 2017.

[30] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *ArXiv*, abs/1506.03365, 2015.

[31] H. Zhang, Zizhao Zhang, Augustus Odena, and H. Lee. Consistency regularization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2020.

[32] Jun-Yan Zhu, T. Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017.