GraN-GAN: Piecewise Gradient Normalization for Generative Adversarial Networks

Vineeth S. Bhaskara*[1]   Tristan Aumentado-Armstrong*[1,2,3]   Allan Jepson[1]   Alex Levinshtein[1]

[1]Samsung AI Centre Toronto   [2]University of Toronto   [3]Vector Institute for AI

{s.bhaskara,allan.jepson,alex.lev}@samsung.com, tristan.a@partner.samsung.com

Abstract

Modern generative adversarial networks (GANs) predominantly use piecewise linear activation functions in discriminators (or critics), including ReLU and LeakyReLU. Such models learn piecewise linear mappings, where each piece handles a subset of the input space, and the gradients per subset are piecewise constant. Under such a class of discriminator (or critic) functions, we present Gradient Normalization (GraN), a novel input-dependent normalization method, which guarantees a piecewise $\mathcal{K}$-Lipschitz constraint in the input space. In contrast to spectral normalization, GraN does not constrain processing at the individual network layers, and, unlike gradient penalties, strictly enforces a piecewise Lipschitz constraint almost everywhere. Empirically, we demonstrate improved image generation performance across multiple datasets (incl. CIFAR-10/100, STL-10, LSUN bedrooms, and CelebA), GAN loss functions, and metrics. Further, we analyze altering the often untuned Lipschitz constant $\mathcal{K}$ in several standard GANs, not only attaining significant performance gains, but also finding connections between $\mathcal{K}$ and training dynamics, particularly in low-gradient loss plateaus, with the common Adam optimizer.

1. Introduction

Generative adversarial networks (GANs) [12] are a class of generative models that have been shown to be very effective, especially for unsupervised high-resolution image generation [27, 13, 16, 17]. GANs usually consist of two networks, a generator $G(z)$ that generates synthetic data conditioned on a noise vector z (sampled from a known noise distribution, usually standard normal) and a discriminator (or critic) $D(x)$ that classifies real data from the generated synthetic data. G and D are generally parametrized as deep neural networks and optimize a mini-max objective. A Nash equilibrium of the zero-sum game is attained when G models the real data distribution and D is maximally uncertain in
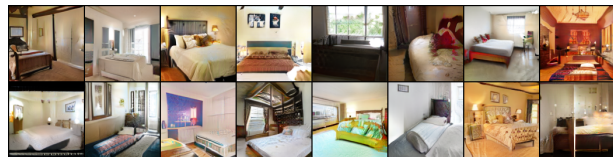
*Equal contribution.
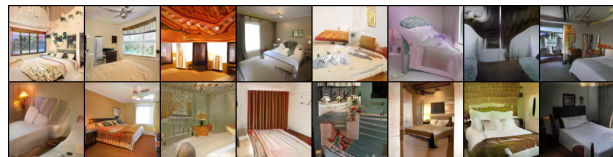
(a) WGAN-GP (FID: 13.6)
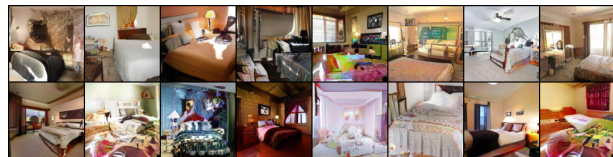
(b) SNGAN (FID: 13.2)

(c) GraND-GAN (FID: 10.8)

Figure 1: Images generated by WGAN-GP, SNGAN, and our method (GraND-GAN) on 128×128 LSUN-Bedrooms (zoom in for better viewing). Lower FID is better.

discriminating the real from synthetic samples.

Despite the effectiveness of GANs in modeling high-dimensional data distributions, they are hard to train. The quality of the images output by G is dependent on the magnitude of the *input gradients* of the generator loss $\mathcal{L}_G$, written

$$\nabla_x \mathcal{L}_G(D(x)) = \nabla_D \mathcal{L}_G(D(x)) \nabla_x D(x), \qquad (1)$$

where $x = G(z)$. The characteristics of the input gradient of $\mathcal{L}_G$, in general, are determined both by the architecture of the discriminator $D(x)$ as well as the loss function $\mathcal{L}_G$ used in formulating the mini-max objective. In turn, $\nabla_x D(x)$ is a function of the parameters θ_D of the discriminator (or critic) that is trained to minimize a loss $\mathcal{L}_D$ to separate real samples x_r from the synthetically generated ones $x_f = G(z)$ for a given G. Therefore, in contrast to image classification, the

role of D is not only to accurately discriminate real data from fake or synthetic data, but also to have a well-behaved input gradient $\nabla_x D(x)$, which is the primary signal that G relies on for learning. Designing such a discriminator is a major objective for GAN research.

Towards this goal, in this paper, we present a novel input-dependent normalization method called *Gradient Normalization* or *GraN*, which guarantees bounded gradients and a piecewise Lipschitz constraint almost everywhere. GraN can be applied to neural networks with piecewise linear activation functions, a prominent class of function approximators within deep learning, and we use the normalized output for discriminators (or critics). Fig. 1 shows a qualitative comparison of our method on the popular dataset LSUN-Bedrooms.

Other works have considered constraining D for stabilizing GAN training. In contrast to spectral normalization [27], our method does not restrict processing at the individual layers and achieves a tight, piecewise Lipschitz bound on the class of functions that D can model. Unlike gradient penalties [13], which softly penalize gradient norms at samples from the input space, our method strictly enforces piecewise constant gradients over the entire input space and ensures the gradient norms within each piece are strictly bounded by a single Lipschitz constant. However, our normalization introduces discontinuities in the discriminator itself, and it is only piecewise continuous and piecewise Lipschitz. Nevertheless, we show that our method achieves empirical results that are competitive with or better than existing methods.

Our main contributions are as follows:

- We present Gradient Normalization (GraN) for piecewise linear networks $f(x)$ that strictly constrains f to be piecewise $\mathcal{K}$-Lipschitz, where the input space x is partitioned into convex polytopes in each of which $f(x)$ is linear with $\|\nabla_x f(x)\| = \mathcal{K}$.

- We show that normalizing discriminators (or critics) with GraN bounds the gradients received by the generator G almost everywhere, stabilizing GAN training.

- Unlike spectral normalization (SN), GraN does not restrict processing at the individual layers and does not suffer from gradient attenuation. Further, in contrast to both SN and gradient penalties, GraN enforces the piecewise Lipschitz property as a hard constraint.

- Empirically, GraN performs better or competitive to existing methods on multiple datasets (CIFAR-10, CIFAR-100, STL-10, LSUN bedrooms, and CelebA), and two loss functions (discriminators with a non-saturating cross-entropy loss and critics with a soft hinge loss).

- While GraN only enforces a local, piecewise Lipschitz constraint, we find the finite-difference gradient norm is empirically well-behaved across large step sizes, likely including jumps across the polytopes modeled by f.

- We also investigate the effect of the Lipschitz constant $\mathcal{K}$ on standard baseline models, finding (a) constrained discriminators (trained with cross-entropy loss) outperform constrained critics, (b) $\mathcal{K}$ influences training dynamics when using Adam, especially on loss plateaus, and (c) tuning $\mathcal{K}$ can significantly improve performance.

2. Related Work

2.1. Stabilizing GANs

Most previous works on stabilizing GANs take one of the following approaches: (1) proposing novel loss functions (e.g., [24, 4]), (2) devising improved architectures (e.g., [17, 39, 40, 28, 11, 9, 15]) or (3) introducing new constraints on D (e.g., [2, 3, 31, 35]). Herein, we continue along the latter line of research, constructing an architectural regularization on D that improves training without sacrificing network capacity.

One of the earliest works on constraining D to stabilize GAN training is the Wasserstein GAN (WGAN) [3]. They propose a novel loss function and present weight-clipping as a way to regularize $D(x)$ to be 1-Lipschitz in x. However, subsequent work [13] showed gradient penalties to be more effective, as they do not impede optimization or severely reduce network capacity.

2.2. Gradient Penalties

Recent research has found gradient penalties (GPs), of the form $P_\delta(x) = (\|\nabla_x f(x)\|_2 - \delta)^2$, to be useful for GANs. Building on WGAN, among the most popular GAN variants is WGAN-GP [13], which forgoes weight clipping by applying P_1 along random convex combinations of reals and fakes, since any C^0 function with unit-length gradient is necessarily 1-Lipschitz. Later research considered *one-sided* GPs [30], as well as alternate sampling methods [36].

Part of the motivation for turning to optimal transport distances is the reduction of gradient uninformativeness, which is a problem for most f-divergence-based GANs when the real and fake distributions do not sufficiently overlap [31]. However, Zhou *et al*. [41] show that Lipschitz continuity can combat gradient uninformativeness in GANs more generally. Separately, Roth *et al*. [31] showed that increasing distributional overlap via noise is approximately equivalent to a zero-centered GP. This was simplified to the popular "R1" GP (defined as P_0 on real data) [25], used in recent state-of-the-art GANs (e.g., [17, 7]).

Clearly, gradient regularization has seen empirical success in improving GANs. Yet, a downside of soft GPs is that they may not enforce exactly the desired value at a given position; furthermore, they are only applied to a subset of the input domain, which may shift over time. In contrast, GraN enforces unit gradients almost everywhere by construction.

2.3. Weight Normalization

Weight normalization (WN) [33] reparametrizes layers in a manner conducive to better conditioned optimization, used in early GAN work [32]. In WN, for each individual layer i of the network, the corresponding weight vectors are rewritten as $\widetilde{w_i} = w_i \rho_i / ||w_i|| \, \forall \, i$, where the learned scalar ρ_i controls the norm and $w_i / ||w_i||$ represents the direction.

For piecewise linear networks, each x is linearly mapped to $f(x) = w \cdot x + b$, where w is locally constant around x, is implicitly input-dependent, and may be interpreted as the *effective weight vector* of a (local) linear model (see §3.2 for details). In a manner reminiscent of WN, GraN essentially normalizes f by the norm of its gradient, i.e., $\nabla_x f = ||w||$. In contrast, however, GraN acts on the full network, rather than a single layer at a time, and enforces piecewise constant gradients, rather than reparametrizing the network.

2.4. Spectral Normalization and Gradient Attenuation under Global Lipschitz Constraints

Building on prior regularizations, such as GPs and weight clipping, Miyato *et al.* [27] present spectral normalization (SN) as an alternative method of ensuring D is 1-Lipschitz, without an additional penalty in the objective for D. This is enforced by a layer-wise weight normalization technique, dividing by an estimate of the maximal singular value from each weight matrix. Empirically, SN is an effective stabilizer for GANs during training, independent of the loss function employed. As a result, it is a major component in recent large-scale GANs [6, 34]. We discuss SN further in §5.

At an architectural level, balancing network capacity and regularization is difficult for globally Lipschitz-constrained networks. Indeed, Anil *et al.* [1] showed that standard networks struggle to solve simple tasks when globally Lipschitz constrained, and that smooth SNed ReLU networks with unit gradients become globally linear, which they solve by enforcing gradient norm preservation and weight matrix orthonormality. We avoid this in GraN by permitting discontinuities in D with respect to the input space and by only constraining the Lipschitz constant $\mathcal{K}$ locally in a piecewise manner. Later work [21] combated the gradient attenuation induced by Lipschitz constraints with a novel orthogonal convolution operator. In contrast, GraN can be applied on top of any piecewise linear network without globally constraining $\mathcal{K}$, while doing so locally in a piecewise sense.

3. Background

3.1. Generative adversarial networks

Let $\mathbb{P}_r$ represent the distribution of real data and $\mathbb{P}_g$ be the distribution of generated data at a given state of the generator G. Let $\mathcal{L}_G$ and $\mathcal{L}_D$ represent the loss functions for the generator G and the discriminator (or critic) D, respectively. Let $z \sim \mathcal{N}(0, 1)$ be a $|\mathcal{Z}|$-dimensional noise vector sampled from an i.i.d. standard normal distribution. Let $f : \mathbb{R}^d \to \mathbb{R}$ represent a deep neural network encoding a scalar field. For image generation, $\mathbb{R}^d = \mathbb{R}^{3 \times H \times W}$.

Goodfellow *et al.* [12] originally propose a cross-entropy (CE) loss-based objective for training G and D as follows:

$$\mathcal{L}_D = \mathbb{E}_{x \sim \mathbb{P}_r} \left[-\log D(x) \right] + \mathbb{E}_{x \sim \mathbb{P}_g} \left[-\log(1 - D(x)) \right], \tag{2}$$

$$\mathcal{L}_G = \mathbb{E}_{x \sim \mathbb{P}_g} \left[\log(1 - D(x)) \right], \tag{3}$$

where the discriminator $D(x) = \sigma(f(x)) \in [0, 1]$ represents the probability of a sample x coming from the real distribution $\mathbb{P}_r$ and $\sigma(\cdot)$ is the sigmoid function $1/(1 + e^{-(\cdot)})$. Hence, f computes a logit mapped to a probability by $\sigma(\cdot)$. The gradient of $\mathcal{L}_G$ with respect to the inputs x is then

$$\nabla_x \mathcal{L}_G = \mathbb{E}_{x \sim \mathbb{P}_g} \left[-D(x) \nabla_x f(x) \right]. \tag{4}$$

Notice that, when $D(x) \to 0$ for some generated $x \sim \mathbb{P}_g$, the contribution to $\nabla_x \mathcal{L}_G$ from such points will necessarily be small and have little influence on updating G. This is particularly probable early in training, when $\mathbb{P}_g$ and $\mathbb{P}_r$ are easily separated. To overcome this problem, in the same work, the authors suggest using an alternative objective for G that is non-saturating, given as

$$\mathcal{L}_G = \mathbb{E}_{x \sim \mathbb{P}_g} \left[-\log D(x) \right], \tag{5}$$

for which the gradient is

$$\nabla_x \mathcal{L}_G = \mathbb{E}_{x \sim \mathbb{P}_g} \left[-(1 - D(x)) \nabla_x f(x) \right]. \tag{6}$$

In this case, note that when D confidently rejects fakes (i.e., $D(G(z)) \approx 0$), they can still contribute to $\nabla_x \mathcal{L}_G$. For future convenience of notation, we use "NSGAN" to refer to the non-saturating (NS) version of training a discriminator via a GAN, where $\mathcal{L}_D$ is given by Eq. (2) and $\mathcal{L}_G$ is given by Eq. (5). Note that although $\mathcal{L}_G$ in Eq. (5) is relatively non-saturating early in the training compared to Eq. (3), it may still saturate later in training as $\mathbb{P}_g$ gets closer to $\mathbb{P}_r$.

Subsequently, the WGAN model [3] attempted to address two issues with the original GAN formulation. First, due to the use of the Jensen-Shannon divergence, disjoint support for $\mathbb{P}_g$ and $\mathbb{P}_r$ leads to poor gradients (as a confident D rapidly saturates, leading to vanishing gradients). Second, for fixed D, the optimal G samples a sum of Dirac deltas at points with the *highest* value of $D(x)$, leading to gradients that encourage mode collapse [26]. Using an approximate Wasserstein distance in WGANs mitigates these issues.

This WGAN objective for G and critic D may be written

$$\mathcal{L}_D = -\mathbb{E}_{x \sim \mathbb{P}_r} \left[D(x) \right] + \mathbb{E}_{x \sim \mathbb{P}_g} \left[D(x) \right], \tag{7}$$

$$\mathcal{L}_G = -\mathbb{E}_{x \sim \mathbb{P}_g} \left[D(x) \right], \tag{8}$$

where $D(x) = f(x)$, and $f(x)$ is constrained to be 1-Lipschitz in x, denoted by $\|f\|_{\text{Lip}} = 1$. We now have

$$\nabla_x \mathcal{L}_G = \mathbb{E}_{x \sim \mathbb{P}_g} \left[-\nabla_x f(x) \right], \qquad (9)$$

which does not saturate, unlike NSGAN (Eq. (6)).

Another commonly used variant of the critic loss $\mathcal{L}_D$ is the hinge loss [22, 27]:

$$\mathcal{L}_D = \mathbb{E}_{x \sim \mathbb{P}_r} \left[\text{ReLU} \left(1 - D(x) \right) \right] +$$
$$\mathbb{E}_{x \sim \mathbb{P}_g} \left[\text{ReLU} \left(1 + D(x) \right) \right], \qquad (10)$$

where $\text{ReLU}(x) = \max\{0, x\}$ is the rectified linear unit. It is straightforward to define a smooth analogue of the hinge loss in Eq. (10) by using a smooth approximation of ReLU defined as $\text{SoftPlus}(x) = \log(1 + e^x)$, which we refer to as the *soft hinge loss*. For convenience of notation, we call D a *critic* with $D(x) = f(x)$, when $\mathcal{L}_D$ is any of the Wasserstein, hinge, or soft hinge loss, and a *discriminator* with $D(x) = \sigma(f(x))$, when $\mathcal{L}_D$ is a variant of the cross-entropy loss (as in NSGANs).

For GANs, several methods have been proposed to effectively constrain D, generally to 1-Lipschitz function spaces. WGAN uses weight clipping, which compromises optimization ease and capacity, while WGAN-GP imposes a soft gradient penalty to improve upon this (see §2.1 and §2.2). More recently, SNGAN tries to impose a 1-Lipschitz constraint by enforcing unit spectral norm of every layer. In §5, we show that a composition of such constrained functions typically results in the overall Lipschitz constant being bounded loosely by 1, and discuss empirical results in §7.

While constraining D to be 1-Lipschitz is necessary for training WGANs, Miyato *et al.* [27] empirically show that such a constraint is also beneficial when training NSGANs. One may immediately see why such smoothness constraints might help. Consider the norm of $\nabla_x \mathcal{L}_G$ for NSGANs (Eq. (6)) and WGANs (Eq. (9)) as follows

$$\|\nabla_x \mathcal{L}_G\|_{\text{NSGAN}} = \|\mathbb{E}_{x \sim \mathbb{P}_g} \left[-(1 - D(x))\nabla_x f(x) \right] \|$$
$$\leq \mathbb{E}_{x \sim \mathbb{P}_g} \left[\|(1 - D(x))\nabla_x f(x)\| \right]$$
$$\leq \mathbb{E}_{x \sim \mathbb{P}_g} \left[\|\nabla_x f(x)\| \right] \qquad (11)$$
$$\|\nabla_x \mathcal{L}_G\|_{\text{WGAN}} = \|\mathbb{E}_{x \sim \mathbb{P}_g} \left[-\nabla_x f(x) \right] \|$$
$$\leq \mathbb{E}_{x \sim \mathbb{P}_g} \left[\|\nabla_x f(x)\| \right]. \qquad (12)$$

When f is constrained to be $\mathcal{K}$-Lipschitz, clearly, $\|\nabla_x \mathcal{L}_G\|_{\text{NSGAN}}$ and $\|\nabla_x \mathcal{L}_G\|_{\text{WGAN}}$ are both bounded above by $\mathcal{K}$. This ensures that the gradients received by G throughout the training are well-behaved and do not explode, thereby improving training stability.

In this paper, we introduce an alternative way to strictly bound the gradients by enforcing a piecewise $\mathcal{K}$-Lipschitz continuity almost everywhere (as opposed to a global $\mathcal{K}$-Lipschitz constraint). In a similar spirit to Miyato *et al.*

[27], we empirically show that our method benefits both discriminators and critics.

In the next section, we introduce a notation for the class of functions implemented by deep neural networks with piecewise linear activations, and subsequently, in §4, we present gradient normalization for discriminators and critics.

3.2. Deep piecewise linear networks

Modern deep neural networks predominantly use piecewise linear activation functions such as ReLU and LeakyReLU. Such activation functions do not admit regions with saturation, and, hence, allow training very deep networks effectively without vanishing gradients.

Let $f(x) : \mathbb{R}^d \to \mathbb{R}$ represent a (deterministic) deep neural network with piecewise linear activation functions. Denote the network parameters by θ. Then one may write

$$f(x, \theta) = w(x, \theta) \cdot x + b(x, \theta), \qquad (13)$$

where $w(x, \theta)$ and $b(x, \theta)$ are piecewise constant in x. We denote $w(x, \theta) \cdot x$ as the scalar dot-product of flattened tensors. Thus, $w(x, \theta)$ and $b(x, \theta)$ being piecewise constant in x means that $\exists$ input sub-sets $S_k \in \{S_j\}_j \subseteq \mathbb{R}^d$, where

$$w(x \in S_k, \theta) = w_k(\theta) \text{ and } b(x \in S_k, \theta) = b_k(\theta),$$

such that w_k and b_k are independent of x within the sub-set S_k. Hence, $\forall x \in S_k$, we have

$$f(x \in S_k, \theta) = w_k(\theta) \cdot x + b_k(\theta), \qquad (14)$$

which is linear in $x \in S_k$. Then $f(x, \theta)$ is the composition of continuous, piecewise linear functions, and is therefore itself a continuous and piecewise linear function of x. That is, there exist disjoint open input subsets S_k, such that $\cup_{k=1}^K S_k = \mathbb{R}^d$, where Eq. 14 holds. I.e., $f(x, \theta)$ is a linear function of $x \in S_k$, with coefficients w_k and b_k that only depend on θ. One can interpret $w_k(\theta)$ and $b_k(\theta)$ as the *effective* weights and bias of a linear functional (given by Eq. (14)) that equals the predictions of the deep neural network, $f(x)$, $\forall x \in S_k$. Note that, unlike a linear hyperplane in logistic regression, the effective weights $w_k(\theta)$ and bias $b_k(\theta)$ are only applicable for $x \in S_k$.

Given this structure, for points off of ∂S_k (the boundary of S_k), the gradient takes a simple form:

$$\nabla_x f(x, \theta) = w_k(\theta), \qquad (15)$$

which is a constant vector $\forall x \in S_k$.

4. Gradient Normalization

In this section, we present *gradient normalization* (GraN), which strictly constrains piecewise linear networks to be 1-Lipschitz almost everywhere.

As in §3.2, let $f(x) : \mathbb{R}^d \to \mathbb{R}$ represent a piecewise linear neural network with parameters θ. We then define the gradient normalized function $g(x)$ as

$$g(x) = f(x)\, \mathcal{R}_\epsilon(\|\nabla_x f(x)\|) = \frac{f(x)\,\|\nabla_x f(x)\|}{\|\nabla_x f(x)\|^2 + \epsilon}, \quad (16)$$

where $\epsilon > 0$ is a fixed constant for numerical stability and $\mathcal{R}_\epsilon(n) = n/(n^2 + \epsilon)$ is the normalization factor, with $n = \|\nabla_x f(x)\|$. Any bounded $R(n)$, with $R(n) = (1/n)(1 + o(1))$ as $n \to \infty$, could be tried. We briefly experimented with $R(n) = 1/(n + \epsilon)$, which produced similar although slightly worse results than $\mathcal{R}_\epsilon$. The choice of $R(n)$ could benefit from further study. We remark that a concurrent work [37] to ours independently explores a similar technique to regularize D, but with a different normalization factor.

Given this normalization, if f is an arbitrary piecewise linear function then g is piecewise linear such that $\|\nabla_x g(x)\| \leq 1$ analytically almost everywhere in $x \in \mathbb{R}^d$. While the gradient $\nabla_x g(x)$ is bounded, $g(x)$ itself can still take real values with no bounds, i.e., $g(x) \in \mathbb{R}$.

One can better describe this result with the notation developed in §3.2. Consider an arbitrary input $x \in S_k$ (without any loss of generality) that belongs to the input open subset S_k and is mapped by the network to a linear piece given by $f(x) = w_k(\theta) \cdot x + b_k(\theta)\ \forall\ x \in S_k$. Then one has $\nabla_x f(x, \theta) = w_k(\theta)\ \forall\ x \in S_k$ (Eq. (15)). Therefore, the GraNed function $g(x)$, given $f(x)$, for $x \in S_k$ becomes

$$g(x \in S_k) = [w_k(\theta) \cdot x + b_k(\theta)]\ \frac{\|w_k(\theta)\|}{\|w_k(\theta)\|^2 + \epsilon}. \quad (17)$$

Consequently, $\nabla_x g$ and its norm can be written as

$$\nabla_x g(x \in S_k) = w_k(\theta)\ \frac{\|w_k(\theta)\|}{\|w_k(\theta)\|^2 + \epsilon}, \quad (18)$$

$$\implies \|\nabla_x g(x \in S_k)\| = \frac{\|w_k(\theta)\|^2}{\|w_k(\theta)\|^2 + \epsilon} < 1. \quad (19)$$

Since x and S_k were arbitrarily chosen, it follows that $\|\nabla_x g(x)\| < 1$ except at the boundaries, say $x \in \partial S_k$, where the gradient does not exist. Since $\cup_k \partial S_k$ is measure zero, we have $\|\nabla_x g(x)\| < 1$ almost everywhere in $\mathbb{R}^d$. Further, for $\|w_k(\theta)\| \gg \epsilon$, we have $\|\nabla_x g\| \approx 1$ in S_k.

Note that, since the original piecewise linear function $f(x)$ does not have a smooth gradient $\nabla_x f(x)$, the normalization factor $\mathcal{R}_\epsilon(\|\nabla_x f(x)\|)$ will have discontinuities for $x \in \cup_k \partial S_k$. Therefore, g is typically discontinuous and not guaranteed to be globally 1-Lipschitz. However, we empirically find that $g(x)$ has bounded finite-differences over substantial perturbations (see §7).

Due to this piecewise constant gradient property (with unit-bounded norm), we remark that g is *piecewise 1-Lipschitz*, since g is 1-Lipschitz with respect to any pair of points within each subset S_k. It is also *locally* 1-Lipschitz

continuous almost everywhere, since there exists an open ball around every point $x \in \cup_k S_k$, within which 1-Lipschitz continuity holds. Empirically, we find that the tight bound on $\|\nabla_x g(x)\|$ almost everywhere assists with GAN training.

4.1. Gradient Normalized GANs

Given a deep neural network represented by $f(x)$, we write $D(x) = \sigma(f(x))$ when D represents discriminators, and $D(x) = f(x)$, when D represents critics, respectively, where $\sigma(\cdot)$ is the sigmoid function (see §3.1).

When $f(x)$ is piecewise linear in x, we define the gradient normalized discriminator (GraND) and critic (GraNC) as $D(x) = \sigma(g(x))$ and $D(x) = g(x)$, respectively, where

$$g(x) = \frac{f(x)}{\tau}\ \frac{\|\nabla_x f(x)\|}{\|\nabla_x f(x)\|^2 + \epsilon}, \quad (20)$$

and τ is a positive constant that constrains g to be $\mathcal{K}$-Lipschitz, with $\mathcal{K} = 1/\tau$. For GraNDs, τ takes a role analogous to the temperature of a sigmoid, and, hence, we term it as the "temperature" hyperparameter.

Like spectral normalization [27], the Lipschitz constant $\mathcal{K} = 1/\tau$ is the only additional hyperparameter that needs to be tuned for our method. Moreover, in practice, our method achieves a tight bound on the Lipschitz constant, unlike spectral normalization, which imposes a loose upper bound with $\|g\|_{\text{Lip}} \leq \mathcal{K}$. See §5 for details.

Finally, note that since the gradients in Eqs. (11) and (12) are computed by back-propagation, the step discontinuities in $g(x)$ are ignored. Therefore, for GANs with GraNDs and GraNCs, the generator G receives gradients that are always bounded, i.e., $\|\nabla_x \mathcal{L}_G\| \leq \mathcal{K}$.

5. Comparison to Layer-wise Norms

Spectral Normalization As noted in §2.2, GPs softly encourage Lipschitz continuity in a data-dependent manner. Improving on this, the spectrally normalized GAN (SNGAN) [27] achieves it at the architectural level. In particular, a linear layer $\ell(x) = Wx$ (ignoring the bias term) with weights W is normalized via $\widetilde{W} = W/\|W\|_2$, before being applied to an input ($\widetilde{\ell}(x) = \widetilde{W}x$), where $\|W\|_2 = \sigma_1(W)$ is the spectral norm of W, equal to its largest singular value (SV) σ_1. This is a form of WN, but acts on the whole matrix rather than its individual rows. Notice that $\|\ell\|_{\text{Lip}} \leq \sigma_1(W)$ and $\sigma_1(\widetilde{W}) = 1$, so $\|\widetilde{\ell}\|_{\text{Lip}} \leq 1$. As such, SN ensures layer-wise 1-Lipschitz continuity, and thus enforces it across the whole network, where SVs are estimated via power iteration.[1]

One downside of SN is the tendency to over-constrain the network, reducing capacity and attenuating gradients, due to the layer-wise enforcement mechanism [1, 21]. SN guarantees a function is globally 1-Lipschitz by bounding the

---

[1]Although note that, at each step, power iteration provides a *lower bound* on $\sigma_1(W)$, and thus in practice it is possible that $\|\ell\|_{\text{Lip}} > 1$.

Lipschitz constant (LC) of every layer, as this then bounds their composition: $||f \circ g||_{\text{Lip}} \leq ||f||_{\text{Lip}}||g||_{\text{Lip}}$. However, this upper-bound is often loose, over-constraining the network and attenuating gradients (expanded upon below). In comparison, GraN acts upon the network as a whole, leaving weights per layer free to vary, and ensuring that gradients with respect to the input are always of unit norm.

The Looseness of Layerwise Constraints For illustration, consider the simple case of two SNed linear layers without biases, ignoring non-linear activations (though, for instance, this occurs in the positive domain of ReLU): $z = f(g(x))$, where $f(y) = By$ and $g(x) = Ax$. Assuming sufficient power iterations under SN, the largest singular values (SVs) of A and B are one, assigning each layer an LC of one. We next examine the conditions for which the composition of the layers, $f \circ g$, also has a LC of one.

In this case, $z = f(g(x)) = BAx$, and so $f \circ g$ has a sharp LC of one if and only if the maximal SV of BA, namely $\sigma_1(BA)$, is also one. Let $\Gamma_\sigma(A)$ denote the span of the right singular vectors of A with corresponding SVs equal to σ. We show in supplementary material, §D, that $\sigma_1(BA) = 1$ if and only if the *first principal angle* [10, 42] between the subspaces $\Gamma_1(A^T)$ and $\Gamma_1(B)$ is zero. This only occurs if they intersect in at least one dimension. However, if *even one* SV of A and B is less than one, then $\Gamma_1(A^T)$ and $\Gamma_1(B)$ will be measure zero, meaning the network must solve a high dimensional "alignment" problem of two measure zero sets. The only scenario avoiding this is when every SV of either A or B are one, which is also a measure zero event.

Importantly, since the SN framework does not directly encourage these subspaces to align, or all the SVs of the weight matrices to be one, in practice it is likely that $\sigma_1(BA) < 1$. This issue is exacerbated for deeper networks, as the overall LC equals the product of $\sigma_1(BA)$ for every adjacent pair of layers. In addition, though training may encourage the network to utilize its capacity by avoiding small SVs, empirically it struggles to do so [1]. In contrast, GraN not only guarantees the function is locally 1-Lipschitz, but does so without constraining individual layers (avoiding subspace alignment issues) and enforces *exactly* unit gradient almost everywhere as well (attaining the sharp LC bound within every S_k). Fig. 3 displays this exactness for GraN; note that SNGAN, due its residual architecture, actually has an LC of 1024, showcasing the looseness of the bound.

6. Experiments

Model Architectures and Training We evaluate our method on unconditional image generation across datasets of various sizes: CIFAR-10/100 (32×32) [19], STL-10 (48×48) [8], LSUN bedrooms (128×128) [38], and CelebA (128×128) [23]. We use Mimicry [20] with PyTorch [29] on a single NVIDIA V100 GPU for training our models. The generator G and discriminator (or critic) D

Table 1: Hyperparameters tested in Fig. 2 for GraND-GAN, SNGAN, and WGAN-GP on CIFAR-10, where α is the learning rate, β_1 and β_2 parametrize Adam in Eq. (21), and n_{dis} is the number of discriminator steps per generator step.

Setting	α (LR)	β_1	β_2	n_{dis}
A	0.0001	0.5	0.9	5
B	0.0002	0.5	0.999	1
C	0.001	0.5	0.999	5
D	0.001	0.9	0.999	5
E (default)	0.0002	0.0	0.9	5

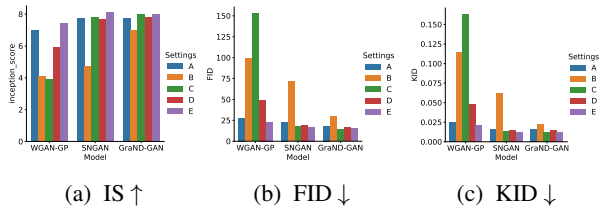(a) IS ↑   (b) FID ↓   (c) KID ↓

Figure 2: Inception scores (IS), FIDs, and KIDs on CIFAR-10 image generation across different hyperparameters listed in Table 1 for GraND-GAN, WGAN-GP, and SNGAN. Results show superior robustness for GraND.
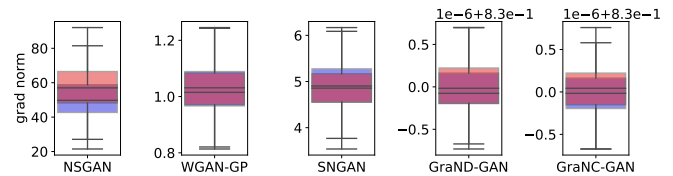

Figure 3: Boxplots of gradient norms at real (blue) and fake (red) samples for different methods at 50K iterations (out of 100K) on CIFAR-10 during training. Gradient norms for GraND/C have a very narrow distribution (spanning less than $\pm 10^{-6}$) around the piecewise Lipschitz constant $\mathcal{K}{=}0.83$.

architectures are identical across methods for a given dataset with identical number of learnable parameters for fair comparison. We use the Adam [18] optimizer with $\beta_1 = 0.0$, $\beta_2 = 0.9$, and a batch size of 64 for 100K iterations, with a dataset-dependent learning rate (LR) α and number of D steps per G step n_{dis}. NSGANs and GraND-GAN use the cross-entropy loss for D in Eq. (2) and the non-saturating loss for G in Eq. (5). GraNC-GAN uses a soft version of the hinge loss for D in Eq. (10), replacing ReLU with softplus, and Eq. (8) for G (see §3.1). The soft hinge loss was found to be more performant and stable than the (hard) hinge loss with GraNC-GANs: on LSUN, GraNC diverged with the hard hinge loss, while on CelebA, a lower FID was obtained. See supplementary material for further model details, ablation experiments, and hyperparameter choices.

Lipschitz Constant Analysis We find that tuning the Lipschitz constant $\mathcal{K}$ significantly affects the performance and stability of models when using the Adam optimizer. This is due to the interdependence of ϵ_{Adam} in the update with $\mathcal{K}$:

$$\delta\theta = -\frac{\langle \mathbf{g} \rangle_{\beta_1}}{\sqrt{\langle \mathbf{g}^2 \rangle_{\beta_2}} + \epsilon_{\text{Adam}}}. \tag{21}$$

Changing $\mathcal{K}$ from its default value of one has an effect of scaling the gradients of the loss function **g** by $\mathcal{K}$. This in turn has an effect of scaling $\epsilon_{\text{Adam}} \longrightarrow \epsilon_{\text{Adam}}/\mathcal{K}$. Empirically, we find that the Adam update for individual parameters can go $\ll 10^{-7}$ in magnitude on the plateaus of the loss landscape where ϵ_{Adam} becomes significant. We find it helps having smaller $\mathcal{K}$ generally when training on larger image resolutions which has an effect of increasing ϵ_{Adam} from its default value of 1×10^{-8}. Intuitively, a larger ϵ_{Adam} suppresses the Adam update when the loss gradient magnitudes are $\lesssim \epsilon_{\text{Adam}}$. Therefore, ϵ_{Adam} determines the extent of noisy Adam updates at plateaus of the loss landscape, and so may need tuning. More details on the choice of $\mathcal{K}$ are provided in the supplementary material.

Baselines and Evaluation To directly compare our normalization method GraN with gradient penalty (GP) and spectral normalization (SN), independent of the loss function, we train NSGAN with a gradient penalty P_1 loss (NSGAN-GP), and with SNed layers (NSGAN-SN). We also train NSGAN-GP† and NSGAN-SN†, which correspond to models constrained with a tuned Lipschitz constant (instead of 1). WGAN-GP† and SNGAN† are defined analogously.

We quantitatively evaluate the methods by Inception Score (IS) [32], FID [14], and KID [5] with 50K synthetic images randomly sampled from G and 50K real images from the dataset. IS is not used for LSUN and CelebA, as these comprise a single class, for which IS performs poorly [20]. See supplemental material §A for additional details.

Unconditional Image Generation Table 2 presents the comparative results on CIFAR-10, CIFAR-100, and STL-10 image generation. Our methods rank among the top two across every metric (KID, FID, IS) on CIFAR-10 and CIFAR-100. On STL-10, our two methods rank the highest in IS and FID, however, we fall behind on KID by a small margin compared to SNGAN and SNGAN†.

Table 3 summarizes our results on LSUN bedrooms and CelebA image generation. The NSGAN model did not converge (i.e., FID > 70) in two random restarts of training for both LSUN and CelebA. Similarly, WGAN-GP failed to converge on CelebA in two runs. GraND-GAN achieves the best results on CelebA, and the second-best on LSUN bedrooms, falling slightly behind NSGAN-GP†. Among critics, GraNC-GAN performs best by FID as well.

Figure 2 presents a comparison of GraND-GAN with WGAN-GP and SNGAN on CIFAR-10 image generation across various training settings listed in Table 1. For setting
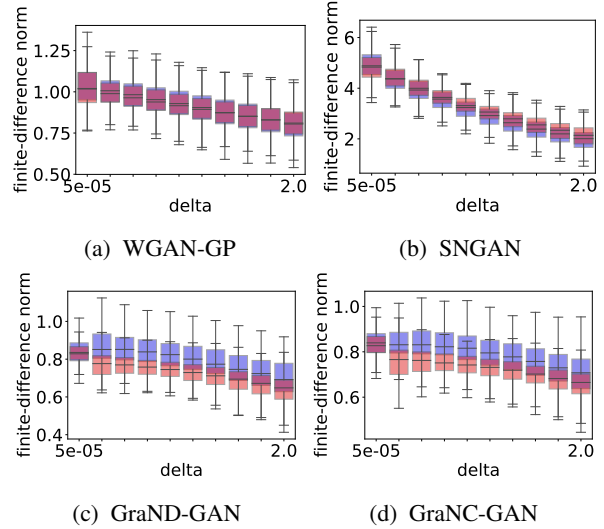

(a) WGAN-GP   (b) SNGAN

(c) GraND-GAN   (d) GraNC-GAN

Figure 4: Boxplots of estimated finite-difference gradient norm with increasing $L2$ perturbation strengths δ around the real (blue) and fake samples (red) on CIFAR-10 at 50K iterations (out of 100K) for different methods. Empirically, GraND/C is fairly Lipschitz bounded across polytopes globally, close to the piecewise Lipschitz constant of $\mathcal{K} = 0.83$.

B ($n_{\text{dis}} = 1$), our method retains a respectable FID score (and other metrics) compared to SNGAN and WGAN-GP. For settings C and D with larger learning rates and momentum hyperparameters, the performance of WGAN-GP degrades while our method and SNGAN remain quite robust.

7. Gradient Normalization Empirical Analysis

In this section, we empirically analyze the effect of gradient normalization on (a) the gradient norms and (b) the finite-difference approximation to the gradient norm at increasing levels of perturbation δ, and compare it with spectral normalization and gradient penalty.

Figure 3 shows a boxplot of $\|\nabla_x f(x)\|$ for the baselines, and $\|\nabla_x g(x)\|$ for our methods, on a CIFAR-10 image generation task at 50K iterations (out of 100K), where $f(x)$ is the piecewise linear discriminator (or critic) network and $g(x)$ is its gradient normalized version. The gradient norms $\|\nabla_x f(x)\|$ for NSGAN with an unconstrained discriminator are substantially larger. For WGAN-GP and SNGAN, $\|\nabla_x f(x)\|$ are bounded within a reasonable range. Gradient normalized discriminators and critics with a piecewise Lipschitz constant of $\mathcal{K} = 0.83$, have a narrow distribution with a gradient norm that is $\approx \mathcal{K} \, (\pm 10^{-6})$ across samples.

Unlike spectral normalization or gradient penalty, our method does not constrain the discriminator or critic to be globally $\mathcal{K}$-Lipschitz. We investigate this on a CIFAR-10 image generation task by first sampling fake data from G

Table 2: Inception scores (IS), FIDs, and KIDs with unsupervised image generation on CIFAR-10, CIFAR-100, and STL-10. The best and the second best models per evaluation metric and GAN family (i.e., with discriminators or critics) are indicated by **bold red** and **bold blue** fonts. † indicates modified baselines with an altered Lipschitz constant $\mathcal{K}$. The table is split comparing discriminators (top) and critics (bottom). We write "**–**" for cases where a model did not achieve a FID < 70.

Method	IS ↑			FID ↓			KID (×1000) ↓		
	CIFAR-10	CIFAR-100	STL-10	CIFAR-10	CIFAR-100	STL-10	CIFAR-10	CIFAR-100	STL-10
NSGAN	7.655	6.611	7.920	23.750	30.842	44.179	14.5	20.5	40.0
NSGAN-GP	8.016	–	8.568	**15.813**	–	38.848	**12.9**	–	38.9
NSGAN-SN	7.792	7.258	8.167	20.998	25.564	**38.669**	15.7	18.4	**35.7**
NSGAN-GP†	**8.019**	**7.892**	**8.623**	15.911	**20.894**	40.110	13.1	**17.0**	39.8
NSGAN-SN†	7.814	7.526	8.135	20.323	24.200	39.013	15.3	17.7	36.7
GraND-GAN (Ours)	**8.031**	**8.314**	**8.743**	**14.965**	**18.978**	**35.226**	**12.3**	**13.7**	**35.0**
WGAN-GP	7.442	7.520	8.492	22.927	27.231	42.170	21.1	23.5	43.0
SNGAN	**8.112**	7.778	8.385	17.107	20.739	38.218	**12.6**	**14.3**	**34.3**
WGAN-GP†	7.344	7.684	8.466	22.705	25.211	42.595	20.6	21.2	44.7
SNGAN†	**7.991**	**7.959**	**8.552**	**16.740**	**20.104**	**36.203**	**12.0**	**14.3**	**33.3**
GraNC-GAN (Ours)	7.966	**8.208**	**8.957**	**16.361**	**19.131**	**35.770**	13.7	**14.8**	35.4

Table 3: Unsupervised 128×128 image generation on LSUN-Bedrooms and CelebA. We write "**–**" to indicate the cases where a model did not achieve a FID < 70 in two random training restarts. The best and the second best models per evaluation metric and GAN family (i.e., with discriminators or critics) are indicated by **bold red** and **bold blue** fonts. † indicates modified baselines with an altered Lipschitz constant $\mathcal{K}$. We split the table into discriminators (top) and critics (bottom), to better highlight the differences per loss function. GraN performs best or second-best across all datasets, losses, and performance metrics; in the case of discriminators, GraND-GAN and NSGAN-GP† (our Lipschitz-tuned GP-based approach) are the top two performers.

Method	FID ↓		KID (×1000) ↓	
	LSUN	CelebA	LSUN	CelebA
NSGAN	–	–	–	–
NSGAN-GP	–	–	–	–
NSGAN-SN	74.926	14.33	44.8	21.2
NSGAN-GP†	**10.483**	**9.385**	**7.2**	**5.8**
NSGAN-SN†	12.635	9.644	8.3	**5.8**
GraND-GAN (Ours)	**10.795**	**9.377**	**7.3**	**5.2**
WGAN-GP	13.562	–	9.8	–
SNGAN	**13.237**	**13.466**	**8.0**	8.9
WGAN-GP†	16.884	–	12.0	–
SNGAN†	67.346	15.874	32.0	**8.7**
GraNC-GAN (Ours)	**12.533**	**12.000**	**9.1**	**8.1**

at a given training iteration and real data from the dataset. We estimate $\Delta_i = \|h(x_i + \delta n_i) - h(x_i)\|/\delta$ for each of the samples, which is the finite difference along a step of magnitude $\delta > 0$ along the local gradient direction $n_i = \nabla h(x_i)/\|\nabla h(x_i)\|$, where h denotes f for baselines and g for our methods, respectively. This provides a probe for the LC (albeit a lower bound) similar in spirit to prior work [43].

Figure 4 shows a boxplot of the resulting Δ_i for increasing perturbation magnitudes δ. Evidently, gradient normalized discriminators and critics have well-behaved finite-differences, even for fairly large neighborhoods δ, despite being only piecewise Lipschitz in theory. Moreover, the variance of the computed Δ_i's across δ for our methods is comparable to WGAN-GP.

8. Discussion

Limitations We observed instabilities when training gradient normalized critics with the Wasserstein loss (Eq. (7)). The hinge loss (Eq. (10)) improved this, but still struggled for larger images; the soft hinge approach was found to work better. However, training GraND with the NS loss (Eq. (2)) was found to be more stable, especially on larger images. Also, on such images, our method periodically diverged late in training, an issue present for the baselines as well.

Future work While GraN does not guarantee a global Lipschitz constraint due to discontinuities, it does enforce constant bounded-norm gradients almost everywhere (and thus piecewise Lipschitz continuity). Moreover, empirically, it is competitive with, or better than, existing baselines. Investigating global versus local Lipschitz continuity, as well as gradient regularization, is thus an enticing future direction.

Conclusion We introduced a novel input-dependent normalization for piecewise linear critics and discriminators. Our method guarantees a bounded input gradient norm almost everywhere and is piecewise $\mathcal{K}$-Lipschitz. We empirically showed that our method improves unconditional image generation using GANs across a range of datasets. Finally, though our method does not explicitly impose a global $\mathcal{K}$-Lipschitz constraint, empirically, the finite-difference gradient norm is well-behaved in a large local neighbourhood.

References

[1] Cem Anil, James Lucas, and Roger Grosse. Sorting out lipschitz function approximation. In *International Conference on Machine Learning*, pages 291–301. PMLR, 2019.

[2] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[4] David Berthelot, Thomas Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[5] Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[7] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. *arXiv preprint arXiv:2012.00926*, 2020.

[8] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[9] 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*, pages 8513–8521, 2018.

[10] Gene H Golub and Charles F Van Loan. *Matrix computations (fourth edition)*. JHU press, 2013.

[11] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3224–3234, 2019.

[12] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of Wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5769–5779, 2017.

[14] 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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6629–6640, Red Hook, NY, USA, 2017. Curran Associates Inc.

[15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

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

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

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[20] Kwot Sin Lee and Christopher Town. Mimicry: Towards the reproducibility of gan research. *CVPR Workshop on AI for Content Creation*, 2020.

[21] Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Joern-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 15390–15402. Curran Associates, Inc., 2019.

[22] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[23] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[24] 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*, pages 2794–2802, 2017.

[25] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[26] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

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

[28] Jeeseung Park and Younggeun Kim. Styleformer: Transformer based generative adversarial networks with style vector. *arXiv preprint*, arXiv:2106.07023, 2021.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu

Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[30] Henning Petzka, Asja Fischer, and Denis Lukovnicov. On the regularization of wasserstein gans. *arXiv preprint arXiv:1709.08894*, 2017.

[31] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *arXiv preprint arXiv:1705.09367*, 2017.

[32] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

[33] Tim Salimans and Diederik P Kingma. Weight normalization: a simple reparameterization to accelerate training of deep neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 901–909, 2016.

[34] Edgar Schonfeld, Bernt Schiele, and Anna Khoreva. A u-net based discriminator for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8207–8216, 2020.

[35] Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularizing generative adversarial networks under limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7921–7931, 2021.

[36] Xiang Wei, Boqing Gong, Zixia Liu, Wei Lu, and Liqiang Wang. Improving the improved training of wasserstein gans: A consistency term and its dual effect. In *International Conference on Learning Representations (ICLR)*, 2018.

[37] Yi-Lun Wu, Hong-Han Shuai, Zhi-Rui Tam, and Hong-Yu Chiu. Gradient normalization for generative adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6373–6382, October 2021.

[38] 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 preprint arXiv:1506.03365*, 2015.

[39] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019.

[40] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[41] Zhiming Zhou, Jiadong Liang, Yuxuan Song, Lantao Yu, Hongwei Wang, Weinan Zhang, Yong Yu, and Zhihua Zhang. Lipschitz generative adversarial nets. In *International Conference on Machine Learning*, pages 7584–7593. PMLR, 2019.

[42] Peizhen Zhu and Andrew V Knyazev. Angles between subspaces and their tangents. *arXiv preprint arXiv:1209.0523*, 2012.

[43] Dongmian Zou, Radu Balan, and Maneesh Singh. On lipschitz bounds of general convolutional neural networks. *IEEE Transactions on Information Theory*, 66(3):1738–1759, 2019.