Adaptive Weighted Discriminator for Training Generative Adversarial Networks

Vasily Zadorozhnyy[1], Qiang Cheng[2,*], Qiang Ye[1,†]

[1] Department of Mathematics

[2] Institute for Biomedical Informatics, Departments of Computer Science and Internal Medicine

University of Kentucky, Lexington, Kentucky 40506-0027

{vasily.zadorozhnyy, Qiang.Cheng, qye3}@uky.edu

Abstract

Generative adversarial network (GAN) has become one of the most important neural network models for classical unsupervised machine learning. A variety of discriminator loss functions have been developed to train GAN's discriminators and they all have a common structure: a sum of real and fake losses that only depends on the actual and generated data respectively. One challenge associated with an equally weighted sum of two losses is that the training may benefit one loss but harm the other, which we show causes instability and mode collapse. In this paper, we introduce a new family of discriminator loss functions that adopts a weighted sum of real and fake parts, which we call adaptive weighted loss functions or aw-loss functions. Using the gradients of the real and fake parts of the loss, we can adaptively choose weights to train a discriminator in the direction that benefits the GAN's stability. Our method can be potentially applied to any discriminator model with a loss that is a sum of the real and fake parts. Experiments validated the effectiveness of our loss functions on unconditional and conditional image generation tasks, improving the baseline results by a significant margin on CIFAR-10, STL-10, and CIFAR-100 datasets in Inception Scores (IS) and Fréchet Inception Distance (FID) metrics.

1. Introduction

Generative Adversarial Network (GAN) [15] has become one of the most important neural network models for unsupervised machine learning. The origin of this idea lies in the combination of two neural networks, one generative and one discriminative, that work simultaneously. The task of the generator is to generate data of a given distribution, while the discriminator's purpose is to try to rec-

ognize which data are created by the generative model and which are the original ones. While a variety of GAN models have been developed, many of them are prone to issues with training such as instability where model parameters might destabilize and not converge, mode collapse where the generative model produces a limited number of different samples, diminishing gradients where the generator gradient vanishes and training does not occur, and high sensitivity to hyperparameters.

In this paper, we focus on the discriminative model to rectify the issues of instability and mode collapse in training GAN. In the GAN architecture, the discriminator model takes samples from the original dataset and the output from the generator as input and tries to classify whether a particular element in those samples is *real* or *fake data* [15]. The discriminator updates its parameters by maximizing a discriminator loss function via backpropagation through the discriminator network. In many of the proposed models [15, 16, 30, 28], the discriminator loss function consists of two equally weighted parts: the "real part" that purely relies on the original dataset and the "fake part" that depends on the generator network and its output; for simplicity we will call them $\mathcal{L}_r$ and $\mathcal{L}_f$ for *real* and *fake* losses, respectively. For example, in the original GAN paper [15], the discriminator loss function $\mathcal{L}_D$ is written as

$$\mathcal{L}_D = \mathcal{L}_r + \mathcal{L}_f, \tag{1}$$

with $\mathcal{L}_r = \mathbb{E}_{x \sim p_d}\big[\log D(x)\big]$ and $\mathcal{L}_f = \mathbb{E}_{z \sim p_z}\big[\log(1 - D(G(z)))\big]$, where D and G are the discriminative and generative models, respectively, p_d is the probability distribution of the real data, and p_z is the probability distribution of the generator parameter z.

The goal of the GAN discriminator training is to increase both $\mathcal{L}_r$ and $\mathcal{L}_f$ so that the discriminator $D(\cdot)$ assigns high scores to real data and low scores to fake data. This is done in (1) by placing equal weights on $\mathcal{L}_r$ and $\mathcal{L}_f$ [15]. However, the training with $\mathcal{L}_D$ is not performed equally on $\mathcal{L}_r$ and $\mathcal{L}_f$. Indeed, a gradient ascent training step along the $\nabla \mathcal{L}_D$ may decrease $\mathcal{L}_r$ (or $\mathcal{L}_f$), depending on the angle be-

tween $\nabla\mathcal{L}_D$ and $\nabla\mathcal{L}_r$ (or $\nabla\mathcal{L}_f$). For example, if we have a large obtuse angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$, which is the case in most training steps (see §5.1), training along the direction of $\nabla\mathcal{L}_D$ may potentially decrease either $\mathcal{L}_r$ or $\mathcal{L}_f$ by going in the opposite direction to $\nabla\mathcal{L}_r$ or $\nabla\mathcal{L}_f$ (see §3 and §5.2). We suggest that this reduction on the real loss may destabilize training and cause mode collapses. Specifically, if a generator is converging with its generated samples close to the data distribution (or a particular mode), a training step that increases the fake loss will reduce the discriminator scores on the fake data and, by the continuity of $D(\cdot)$, reduce the scores on the nearby real data as well. With the updated discriminator now assigning lower scores to the regions of data where the generator previously approximated well, the generator update is likely to move away from that region and to the regions with higher discriminator scores (possibly a different mode). Hence, we see instability or mode collapse. See §5.3 for experimental results.

We propose a new approach in training the discriminative model by modifying the discriminator loss function and introducing adaptive weights in the following way,

$$\mathcal{L}_D^{aw} = w_r \cdot \mathcal{L}_r + w_f \cdot \mathcal{L}_f. \tag{2}$$

We adaptively choose w_r and w_f weights to calibrate the training in the real and fake losses. Using the information of $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$, we can control the gradient direction, $\nabla\mathcal{L}_D^{aw}$, by either training in the direction that benefits both $\mathcal{L}_r$ and $\mathcal{L}_f$ or increasing one loss while not changing the other. This attempts to avoid a situation where training may benefit one loss but significantly harm the other. A more detailed mathematical approach is presented in §3.

Our proposed method can be applied to any GAN model with a discriminator loss function composed of two parts as in (1). For our experiments we have applied adaptive weights to the SN-GAN [34], AutoGAN [14], and BigGAN [5] models for unconditional as well as conditional image generating tasks. We have achieved significant improvements on them for CIFAR-10, STL-10 and CIFAR-100 datasets in both Inception Scores (IS) and Fréchet Inception Distance (FID) metrics, see §4. Our code is available at https://github.com/vasily789/adaptive-weighted-gans.

Notation: We use $\langle\cdot,\cdot\rangle_2$ to denote the Euclidean inner product, $\|x\|_2$ the Euclidean 2-norm, and $\angle_2(x,y) := \arccos\left(\frac{\langle x,y\rangle_2}{\|x\|_2\|y\|_2}\right)$ the angle between vectors x and y.

2. Related Work

GAN was first proposed in [15] for creating generative models via simultaneous optimization of a discriminative and a generative model. The original GAN may suffer from vanishing gradients during training, non-convergence of the model(s), and mode collapse; see [6, 31, 33, 38, 39]

for discussions. Several papers [1, 16, 28, 30] have addressed the issues of vanishing gradients by introducing new loss functions. The LSGAN proposed in [30] adopted the least squares loss function for the discriminator that relies on minimizing the Pearson χ^2 divergence, in contrast to the Jensen–Shannon divergence used in GAN. The WGAN model [1, 16] introduced another way to solve the problem of convergence and mode collapse by incorporating Wasserstein-1 distance into the loss function. As a result, WGAN has a loss function associated with image quality which improves learning stability and convergence. The hinge loss function introduced in [28, 43] achieved smaller error rates than cross-entropy, being stable against different regularization techniques, and having a low computational cost [12]. The models in [3, 26, 45] adopted a loss function called maximum mean discrepancy (MMD). A repulsive function to stabilize the MMD-GAN training was employed in [46], and the MMD loss function was weighted in [11] according to the contribution of data to the loss function. [37] presented a dual discriminator GAN that combines two discriminators in a weighted sum.

New loss functions are not the only way of improving GAN's framework. DCGAN [38], one of the first and more significant improvements in the GAN architecture, was the incorporation of deep convolutional networks. The Progressive Growing GAN [21] was created based on [1, 16] with the main idea of progressively adding new layers of higher resolution during training, which helps to create highly realistic images. [14, 13, 42] developed neural architecture search methods to find an optimal neural network architecture to train GAN for a particular task.

There are many works dedicated to the conditional GAN, for example BigGAN [5] which utilized a model with a large number of parameters and larger batch sizes showing a significant benefit of scaling.

There are many works devoted to improving or analyzing GAN training. [33] trained the generator by optimizing a loss function unrolled from several training iterations of the discriminator training. SN-GAN [34], normalized the spectral norm of each weight to stabilize the training. Recent work [40] introduced stable rank normalization that simultaneously controls the Lipschitz constant and the stable rank of a layer. [27] developed an analysis to suggest that first-order approximations of the discriminator lead to instability and mode collapse. [36] proved local stability under the model that both the generator and the discriminator are updated simultaneously via gradient descent. [9] analyzed the stability of GANs through stationarity of the generator. [32] points out that absolute continuity is necessary for GAN training to converge. Relativistic GAN [20] addressed the observation that with generator training increasing the probability that fake data is real, the probability of real data being real would decrease. [2] proposed a method

of re-weighting training samples to correct for mass shift between the transferred distributions in the domain transfer setup. [7] viewed GAN as an energy-based model and proposed an MCMC sampling based method.

3. Adaptive Weighted Discriminator

In GAN training, if we maximize $\mathcal{L}_D$ to convergence in training discriminator D, we should meet the goal to increase both $\mathcal{L}_r$ and $\mathcal{L}_f$. However, in practice, we train with a gradient ascent step along $\nabla \mathcal{L}_D = \nabla \mathcal{L}_r + \nabla \mathcal{L}_f$, which may be dominated by either $\nabla \mathcal{L}_r$ or $\nabla \mathcal{L}_f$. Then, the training may be done primarily on one of the losses, either $\mathcal{L}_r$ or $\mathcal{L}_f$. Consider a gradient ascent training iteration for $\mathcal{L}_D$,

$$\theta_1 \longleftarrow \theta_0 + \lambda \nabla \mathcal{L}_D, \qquad (3)$$

where λ is a learning rate. Then using the Taylor Theorem, we can expand both $\mathcal{L}_r$ and $\mathcal{L}_f$ about θ_0,

$$
\begin{aligned}
\mathcal{L}_r(\theta_1) &= \mathcal{L}_r(\theta_0) + \lambda \nabla \mathcal{L}_r^T \nabla \mathcal{L}_D + \mathcal{O}(\lambda^2) \qquad (4) \\
&= \mathcal{L}_r(\theta_0) \\
&\quad + \lambda \|\nabla \mathcal{L}_r\|_2 \|\nabla \mathcal{L}_D\|_2 \cos\left(\angle_2\left(\nabla \mathcal{L}_r, \nabla \mathcal{L}_D\right)\right) \\
&\quad + \mathcal{O}(\lambda^2) \qquad (5)
\end{aligned}
$$

and

$$
\begin{aligned}
\mathcal{L}_f(\theta_1) &= \mathcal{L}_f(\theta_0) \\
&\quad + \lambda \|\nabla \mathcal{L}_f\|_2 \|\nabla \mathcal{L}_D\|_2 \cos\left(\angle_2\left(\nabla \mathcal{L}_f, \nabla \mathcal{L}_D\right)\right) \\
&\quad + \mathcal{O}(\lambda^2), \qquad (6)
\end{aligned}
$$

where we have omitted the evaluation point θ_0 in all gradients (i.e. $\nabla \mathcal{L}_* = \nabla \mathcal{L}_*(\theta_0)$) to avoid cumbersome expressions. If one of $\angle_2\left(\nabla \mathcal{L}_r, \nabla \mathcal{L}_D\right)$ and $\angle_2\left(\nabla \mathcal{L}_f, \nabla \mathcal{L}_D\right)$ is obtuse, then to the first order approximation, the corresponding loss is decreased. This causes a decrease in the discriminator assigning a correct score $D(\cdot)$ to the real (or fake) data. Thus, a gradient ascent step with loss (1) may turn out to decrease one of the losses if the angle $\angle_2\left(\nabla \mathcal{L}_r, \nabla \mathcal{L}_f\right) > 90°$. This situation occurs quite often in GAN training; see §5.1 for some experimental results illustrating this.

This undesirable situation is expected to happen in GAN training when the generator has produced samples close to the data distribution or its certain modes. If a training step in the direction $\nabla \mathcal{L}_D$ results in an increase in the fake loss or equivalently a decrease in the discriminator scores $D(G(z))$ on the fake data, it will decrease the scores $D(x)$ on the real data as well by the continuity of $D(\cdot)$. Equivalently, this reduces the real loss. With the updated discriminator assigning lower scores to the regions of the data where the generator previously approximated well, the generator update using the new discriminator will likely move in the direction where the discriminator scores are higher and hence

leave the region it was converging to. We suggest that this is one of the causes of instability in GAN training. If the regions with high discriminator scores contain only a few modes of the data distribution, this leads to mode collapse; see the study in §5.3.

To remedy this situation, we propose to modify the training gradient $\nabla \mathcal{L}_D$ to encourage high discriminator scores for real data. We propose a new family of discriminator loss functions, which we call adaptive weighted loss function or aw-loss function; see equation (2).

We first show that the proposed weighted discriminator (2) with fixed weights carries the same theoretical properties of the original GAN as stated in [15, 32] for binary-cross-entropy loss function, i.e. when the min-max problem is solved exactly, we recover the data distribution.

Theorem 1. *Let $p_{\mathrm{d}}(x)$ and $p_{\mathrm{g}}(x)$ be the density functions for the data and model distributions, $\mathbb{P}_{\mathrm{d}}$ and $\mathbb{P}_{\mathrm{g}}$, respectively. Consider $\mathcal{L}^{aw}(D, p_{\mathrm{g}}) = w_r \mathbb{E}_{x \sim p_d}\left[\log D(x)\right] + w_f \mathbb{E}_{x \sim p_g}\left[\log(1 - D(x))\right]$ with fixed $w_r, w_f > 0$.*

1. *Given a fixed $p_{\mathrm{g}}(x)$, $\mathcal{L}^{aw}(D, p_{\mathrm{g}})$ is maximized by $D^*(x) = \frac{w_r p_{\mathrm{d}}(x)}{w_r p_{\mathrm{d}}(x) + w_f p_{\mathrm{g}}(x)}$ for $x \in \operatorname{supp}(p_{\mathrm{d}}) \cup \operatorname{supp}(p_{\mathrm{g}})$.*

2. $\min_{p_{\mathrm{g}}} \max_D \mathcal{L}^{aw}(D, p_{\mathrm{g}}) = w_r \log \frac{w_r}{w_r + w_f} + w_f \log \frac{w_f}{w_r + w_f}$ *with the minimum attained by $p_{\mathrm{g}}(x) = p_{\mathrm{d}}(x)$.*

See Appendix A for a proof of Theorem 1. To choose the weights w_r and w_f, we propose an adaptive scheme, where the weights w_r and w_f are determined using gradient information of both $\mathcal{L}_r$ and $\mathcal{L}_f$. This structure allows us to adjust the direction of the gradient of the discriminator loss function to achieve the goal of training to increase both $\mathcal{L}_r$ and $\mathcal{L}_f$, or at least not to decrease either loss. We propose Algorithm 1 based on the following gradient relations with various weight choices.

Theorem 2. *Consider $\mathcal{L}_D^{aw}$ in (2) and the gradient $\nabla \mathcal{L}_D^{aw}$.*

1. *If $w_r = \frac{1}{\|\nabla \mathcal{L}_r\|_2}$ and $w_f = \frac{1}{\|\nabla \mathcal{L}_f\|_2}$, then $\nabla \mathcal{L}_D^{aw}$ is the angle bisector of $\nabla \mathcal{L}_r$ and $\nabla \mathcal{L}_f$, i.e.*

$$
\begin{aligned}
\angle_2\left(\nabla \mathcal{L}_D^{aw}, \nabla \mathcal{L}_r\right) &= \angle_2\left(\nabla \mathcal{L}_D^{aw}, \nabla \mathcal{L}_f\right) \\
&= \angle_2\left(\nabla \mathcal{L}_r, \nabla \mathcal{L}_f\right)/2.
\end{aligned}
$$

2. *If $w_r = \frac{1}{\|\nabla \mathcal{L}_r\|_2}$ and $w_f = -\frac{\langle \nabla \mathcal{L}_r, \nabla \mathcal{L}_f \rangle_2}{\|\nabla \mathcal{L}_f\|_2^2 \cdot \|\nabla \mathcal{L}_r\|_2}$, then*

$$\angle_2\left(\nabla \mathcal{L}_D^{aw}, \nabla \mathcal{L}_f\right) = 90°, \ \angle_2\left(\nabla \mathcal{L}_D^{aw}, \nabla \mathcal{L}_r\right) \le 90°.$$

3. *If $w_r = -\frac{\langle \nabla \mathcal{L}_r, \nabla \mathcal{L}_f \rangle_2}{\|\nabla \mathcal{L}_r\|_2^2 \cdot \|\nabla \mathcal{L}_f\|_2}$ and $w_f = \frac{1}{\|\nabla \mathcal{L}_f\|_2}$, then*

$$\angle_2\left(\mathcal{L}_D^{aw}, \nabla \mathcal{L}_r\right) = 90°, \ \angle_2\left(\nabla \mathcal{L}_D^{aw}, \nabla \mathcal{L}_f\right) \le 90°.$$

See Appendix A for a proof of Theorem 2. The first case in the above theorem allows us to choose weights for (2) such that we can train $\mathcal{L}_r$ and $\mathcal{L}_f$ by going in the direction of the angle bisector. However, sometimes the direction of the angle bisector might not be optimal. For example, if the angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$ is close to $180°$, then the bisector direction will effectively not train either loss. During training, $\mathcal{L}_f$ is often easier to train than $\mathcal{L}_r$ meaning that the fake gradient has a larger magnitude. In this situation, we might want to train just on the real gradient direction by simply choosing $w_f = 0$, but if the angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$ is obtuse, we will increase $\mathcal{L}_r$ but significantly decrease $\mathcal{L}_f$ which is undesirable. The second case in Theorem 2 suggests a direction that forms an acute angle with $\nabla\mathcal{L}_r$ and orthogonal to $\nabla\mathcal{L}_f$ (see Figure 1); such a direction will increase $\mathcal{L}_r$ and to the first order approximation will leave $\mathcal{L}_f$ unchanged. When $\mathcal{L}_r$ is high, the third case in Theorem 2 would allow us to increase $\mathcal{L}_f$ while minimizing changes to $\mathcal{L}_r$.

Inspired by the Theorem 2 and observations that we have made, we can calibrate discriminator training in a way that produces and maintains high real loss to reduce fluctuations in the real loss (or real discriminator scores) to improve stability. Algorithm 1 describes the procedure for updating weights of the aw-loss function in (2) during training using the information of $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$.
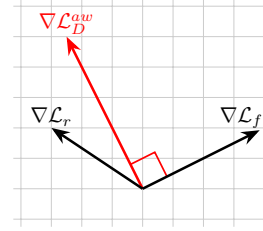

Algorithm 1: Adaptive weighted discriminator method for one step of discriminator training.

1: **Given:** $\mathbb{P}_d$ and $\mathbb{P}_g$ - data and model distributions;
2: **Given:** $\alpha_1 = 0.5$, $\alpha_2 = 0.75$, $\varepsilon = 0.05$, $\delta = 0.05$;
3: **Sample:** $x_1, \ldots, x_n \sim \mathbb{P}_d$ and $y_1, \ldots, y_n \sim \mathbb{P}_g$;
4: **Compute:** $\nabla\mathcal{L}_r, \nabla\mathcal{L}_f, s_r = \frac{1}{n}\sum_{i=1}^{n}\sigma(D(x_i))$,
$\quad s_f = \frac{1}{n}\sum_{j=1}^{n}\sigma(D(y_j))$;
5: **if** $s_r < s_f - \delta$ *or* $s_r < \alpha_1$ **then**
6: $\quad$ **if** $\angle_2\left(\nabla\mathcal{L}_r, \nabla\mathcal{L}_f\right) > 90°$ **then**
7: $\quad\quad$ $w_r = \frac{1}{\|\nabla\mathcal{L}_r\|_2} + \varepsilon$; $w_f = \frac{-\langle\nabla\mathcal{L}_r,\nabla\mathcal{L}_f\rangle_2}{\|\nabla\mathcal{L}_f\|_2^2\cdot\|\nabla\mathcal{L}_r\|_2} + \varepsilon$;
8: $\quad$ **else**
9: $\quad\quad$ $w_r = \frac{1}{\|\nabla\mathcal{L}_r\|_2} + \varepsilon$; $w_f = \varepsilon$;
10: $\quad$ **end**
11: **else if** $s_r > s_f - \delta$ *and* $s_r > \alpha_2$ **then**
12: $\quad$ **if** $\angle_2\left(\nabla\mathcal{L}_r, \nabla\mathcal{L}_f\right) > 90°$ **then**
13: $\quad\quad$ $w_r = \frac{-\langle\nabla\mathcal{L}_r,\nabla\mathcal{L}_f\rangle_2}{\|\nabla\mathcal{L}_r\|_2^2\cdot\|\nabla\mathcal{L}_f\|_2} + \varepsilon$; $w_f = \frac{1}{\|\nabla\mathcal{L}_f\|_2} + \varepsilon$;
14: $\quad$ **else**
15: $\quad\quad$ $w_r = \varepsilon$; $w_f = \frac{1}{\|\nabla\mathcal{L}_f\|_2} + \varepsilon$;
16: $\quad$ **end**
17: **else**
18: $\quad$ $w_r = \frac{1}{\|\nabla\mathcal{L}_r\|_2} + \varepsilon$; $w_f = \frac{1}{\|\nabla\mathcal{L}_f\|_2} + \varepsilon$;
19: **end**

Figure 1: Depiction of the second case of Theorem 2.

Algorithm 1 is designed to first avoid, up to the first order approximation, decreasing $\mathcal{L}_r$ or $\mathcal{L}_f$ during a gradient ascent iteration. Furthermore, it chooses to favor training real loss unless the mean real score is greater than the mean fake score (i.e. $s_f \leq s_r$) and the real mean score is at least $\alpha_1 = 0.5$ (i.e. $\alpha_1 \leq s_r$). Here the mean discriminator scores s_r and s_f represent the mean probability that the discriminator assigns to x_i's and y_j's respectively as real data. When s_r is highly satisfactory with $s_r \geq \alpha_2 = 0.75$ (the midpoint between the minimum probability 0.5 and the maximum probability 1 for correct classifications of real data), we favor training the fake loss; otherwise, we train both equally. By maintaining these training criteria, we will reduce the fluctuations in real and fake discriminator scores and hence avoid instability. See study in §5.3. Note that we impose a small gap $\delta = 0.05$ in $s_f - \delta > s_r$ to account for situations when s_r is nearly identical to s_f.

The way we favor training the real or fake loss depends on whether the angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$ is obtuse or not. In Algorithm 1, the first and the third cases are concerned with the more frequent situation (see §5.1 and Figure 4) where the angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$ is obtuse. These cases are the ones that are developed in Theorem 2. In the first case, we favor training real loss by going in the direction orthogonal to the $\nabla\mathcal{L}_f$, illustrated in Figure 1. In the third case, we favor the fake loss by going in the direction orthogonal to $\nabla\mathcal{L}_r$. In a similar manner, the second and the fourth cases are concerned with the situation when the angle between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$ is acute. We use the same criteria to decide if training should favor the real or fake directions, but in this case we favor training the real or fake loss by using the direction of the corresponding gradient. Lastly, in the fifth case, it is desirable to increase both s_r and s_f without either taking priority, so we choose to train in the direction of the angle bisector between $\nabla\mathcal{L}_r$ and $\nabla\mathcal{L}_f$.

The two threshold α_1 and α_2 in Algorithm 1 can be treated as hyperparameters. Our ablation studies show that the default $\alpha_1 = 0.5$ and $\alpha_2 = 0.75$ as discussed earlier are indeed good choices, see Appendix B.

All weights stated in Algorithm 1 normalize both the real and fake gradients for the purpose of avoiding differ-
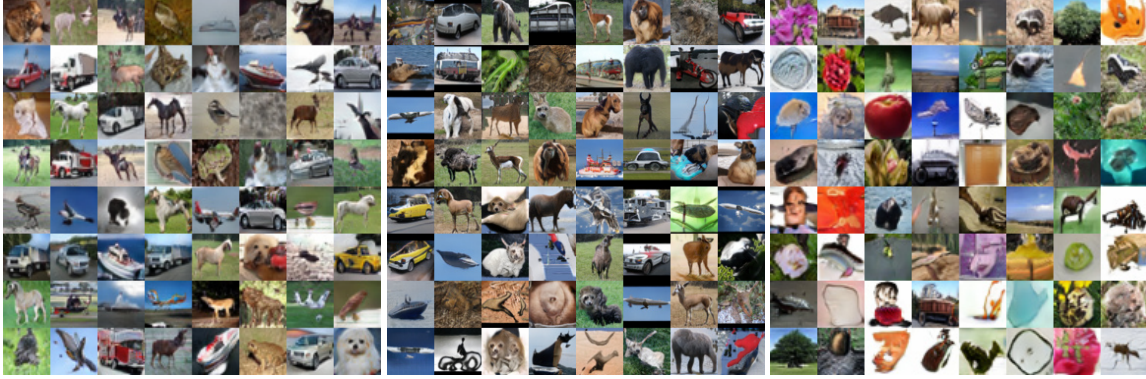
Figure 2: aw-AutoGAN: CIFAR-10 (left), STL-10 (center), CIFAR-100 (right); samples randomly generated.

ently sized gradients, which has the effect of preventing exploding gradients and speeds up training, i.e. achieves better IS and FID with fewer epochs, see Figure 3. With this implementation, we implement a linear learning rate decay to ensure convergence. However, aw-method performs well without normalization, and achieves comparable results. We list the detailed results in Appendix B.

A small constant ε is added to all the weights in Algorithm 1 to avoid numerical discrepancies in cases that would prevent the discriminator model from training/updating. As an example, there are cases when our algorithm would set $w_r = 0$ but at the same time $\nabla \mathcal{L}_f$ would be almost zero, which will result in $\nabla \mathcal{L}_D^{aw}$ being practically zero. We have set $\varepsilon = 0.05$ in all of our experiments.

Algorithm 1 has a small computational overhead. At each iteration we compute inner products and norms that are used for computing w_r and w_f, and then use these weight to update $\nabla \mathcal{L}_D^{aw}$. If we have k trainable parameters, then it takes an order of $6k$ operations to compute inner products between real–fake, real–real, and fake–fake gradients, and an order of $3k$ operations to form $\nabla \mathcal{L}_D^{aw}$, totalling to an order of $9k$ operations in Algorithm 1. This is a fraction of total computational complexity for one training iteration.

4. Experiments & Results

We implement our Adaptive Weighted Discriminator for SN-GAN [34] and AutoGAN [14] models, and for SN-GAN [34] and BigGAN [5] models, on unconditional and conditional image generating tasks, respectively (commonly referred to as unconditional and conditional GANs). AutoGAN is an architecture based on neural search. In our experiments; we do not invoke a neural search with our aw-loss, we have simply implemented the aw-method on the model and architecture exactly provided by [14].

We test our method on three datasets: *CIFAR-10* [25], *STL-10* [10], and *CIFAR-100* [25]. The datasets and implementation details are provided in Appendix B. We present

the implementation with normalized gradients using linear learning rate decay. We also give results of non-normalized version without learning rate decay in Appendix B.

All of the above mentioned models train the discriminator by minimizing the negative hinge loss [28, 43]. Our aw-loss also uses the negative hinge loss as follows:

$$\mathcal{L}_D^{aw} = - w_r \cdot \mathbb{E}_{x \sim p_d} \big[\min(0, D(x) - 1) \big] \\ - w_f \cdot \mathbb{E}_{z \sim p_z} \big[\min(0, -1 - D(G(z))) \big], \quad (7)$$

with w_r and w_f updated every iteration using Algorithm 1.

To evaluate the performance of the models, we employ the widely used Inception Score [39] (IS) and Fréchet Inception Distance [18] (FID) metrics; see [29] for more details. We compute these metrics every 5 epochs and we report the best IS and FID achieved by each model within the 320 (SN-GAN), 300 (AutoGAN), and BigGAN (1,000) training epochs as in the corresponding original works.

We first present the results for the unconditional GAN for the datasets CIFAR-10, STL-10 and CIFAR-100 in Tables 1, 2, and 3 respectively. In addition to baseline results, we have included top published results for each dataset for comparison purposes.

Method	IS ↑	FID ↓
Imp. MMD GAN [46]	8.29	16.21
MGAN [19]	$8.33_{\pm.10}$	26.7
MSGAN [44]	-	11.40
SRN-GAN [40]	$8.53_{\pm.04}$	19.57
StyleGAN2 [22]	$\mathbf{9.21_{\pm.09}}$	$\mathbf{8.32}$
SN-GAN [34]	$8.22_{\pm.05}$	21.7
aw-SN-GAN (Ours)	$8.53_{\pm.11}$	12.32
AutoGAN [14]	$8.55_{\pm.10}$	12.42
aw-AutoGAN (Ours)	$9.01_{\pm.03}$	11.82

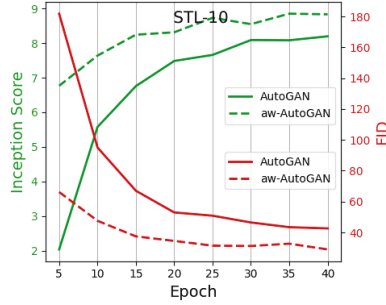Table 1: Unconditional GAN: CIFAR-10.

Figure 3: AutoGAN vs aw-AutoGAN IS and FID plots for the first 40 epochs.

For CIFAR-10 in Table 1, our methods significantly improve the baseline results. Indeed, our aw-AutoGAN achieves IS substantially above all comparisons other than StyleGAN2. StyleGAN2 outperforms aw-AutoGAN but uses 26.2M parameters vs. 5.4M for aw-AutoGAN.

Method	IS ↑	FID ↓
ProbGAN [17]	$8.87_{\pm.09}$	46.74
Imp. MMD GAN [46]	9.34	37.63
MGAN [19]	$9.22_{\pm.11}$	-
MSGAN [44]	-	27.10
SN-GAN [34]	$9.10_{\pm.04}$	40.10
aw-SN-GAN (Ours)	$\mathbf{9.53}_{\pm.\mathbf{10}}$	36.41
AutoGAN [14]	$9.16_{\pm.12}$	31.01
aw-AutoGAN (Ours)	$9.41_{\pm.09}$	**26.32**

Table 2: Unconditional GAN: STL-10.

For STL-10 in Table 2, our methods also significantly improve SN-GAN and AutoGAN baseline results. Our aw-SN-GAN achieved the highest IS and aw-AutoGAN achieved the lowest FID score among comparisons.

Method	IS ↑	FID ↓
SS-GAN [8]	-	21.02[†]
MSGAN [44]	-	19.74
SRN-GAN [40]	8.85	19.55
SN-GAN [34]	$8.18_{\pm.12}$*	22.40*
aw-SN-GAN (Ours)	$8.31_{\pm.02}$	19.08
AutoGAN [14]	$8.54_{\pm.10}$*	19.98*
aw-AutoGAN (Ours)	$\mathbf{8.90}_{\pm.\mathbf{06}}$	**19.00**

Table 3: Unconditional GAN: CIFAR-100; * - results from our test; [†] - quoted from [44].

For CIFAR-100 in Table 3, our methods improve IS sig-

nificantly for AutoGAN but modestly for SN-GAN. Our aw-Auto-GAN achieved the highest IS and the lowest FID score among comparisons.

We have also included some visual examples that were randomly generated by our aw-Auto-GAN model in Figure 2. We also consider the convergence of our method against training epochs by plotting in Figure 3 the IS and FID scores of 50,000 generated samples at every 5 epochs for Auto-GAN vs aw-AutoGAN. For all the datasets, our model consistently achieves faster convergence than the baseline.

We next consider our aw-method for a class conditional image generating task using two base models, SN-GAN [34] and BigGAN [5], on CIFAR-10 and CIFAR-100 datasets. Results are listed in Table 4.

	CIFAR-10		CIFAR-100	
Method	IS ↑	FID ↓	IS ↑	FID ↓
cGAN [35]	8.62	17.5	9.04	23.2
MHinge [23]	$\mathbf{9.58}_{\pm.09}$	7.50	$\mathbf{14.36}_{\pm.17}$	17.3
SNGAN [34]	$8.60_{\pm.08}$	17.5	$9.30^{†}$	$15.6^{†}$
aw-SNGAN	$9.03_{\pm.11}$	8.11	$9.48_{\pm.13}$	14.42
BigGAN [5]	9.22	14.73	$10.99_{\pm.14}$	11.73
aw-BigGAN	$9.52_{\pm.10}$	**7.03**	$11.22_{\pm.17}$	**10.23**

Table 4: Conditional GAN: CIFAR-10 and CIFAR-100; [†] - quoted from [41]; BigGAN [5] CIFAR-100 results based on our test using the code from [4].

Table 4 shows that our method works well for the conditional GAN too. The aw-method significantly improves the SN-GAN and BigGAN baselines. Indeed, our aw-BigGAN achieved the best FID for both CIFAR-10 and CIFAR-100 among comparisons.

5. Exploratory & Ablation Studies

In this section, we present three studies to illustrate potential problems of equally weighted GAN loss and advan-

tages of our adaptive weighted loss. The hinge loss is implemented in the first and second studies, and a binary cross-entropy loss function is used for the third.

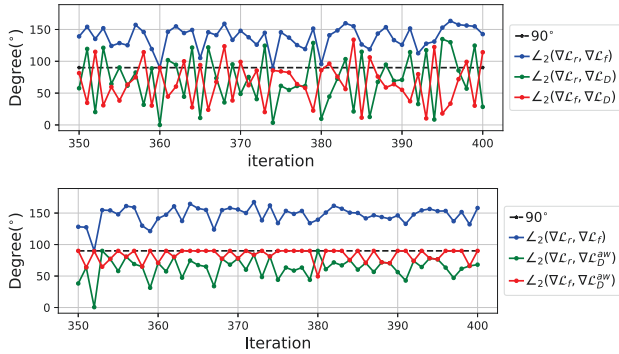## 5.1. Angles between Gradients

Figure 4: Angles between gradients at each iteration. Top: original loss; Bottom: aw-loss.

In the first study, we examine the angles between $\nabla\mathcal{L}_r$, $\nabla\mathcal{L}_f$, $\nabla\mathcal{L}_D$ (or $\nabla\mathcal{L}_D^{aw}$). We use the CIFAR-10 dataset with the DCGAN architecture [38] and we look at 50 iterations in the first epoch of training. In Figure 4, we plot the following 3 angles: $\angle_2(\nabla\mathcal{L}_r, \nabla\mathcal{L}_f)$, $\angle_2(\nabla\mathcal{L}_r, \nabla\mathcal{L}_D)$ and $\angle_2(\nabla\mathcal{L}_f, \nabla\mathcal{L}_D)$ against iterations for the original loss $\mathcal{L}_D$ (1) on the top and for the aw-loss $\mathcal{L}_D^{aw}$ on the bottom. For the original loss (Left), $\angle_2(\nabla\mathcal{L}_r, \nabla\mathcal{L}_f)$ (blue) stays greater then $90°$, closer to $180°$. $\angle_2(\nabla\mathcal{L}_r, \nabla\mathcal{L}_D)$ (green) often goes above $90°$ and so the training is often done to decrease the real loss. $\angle_2(\nabla\mathcal{L}_f, \nabla\mathcal{L}_D)$ also goes above $90°$, though to a lesser extent. With the aw-loss (Right), $\angle_2(\nabla\mathcal{L}_r, \nabla\mathcal{L}_D^{aw})$ and $\angle_2(\nabla\mathcal{L}_f, \nabla\mathcal{L}_D^{aw})$ stay below the $90°$ line and indicate that we train in the direction of $\nabla\mathcal{L}_r$ and orthogonal to $\nabla\mathcal{L}_f$ in most steps.

5.2. Real Discriminator Scores and Real-Fake Gap after Training

Our second experiment is an ablation study to show that aw-loss increases the discriminator scores for real data and increases the gap between real and fake discriminator scores. We again apply the DCGAN model with the original loss $\mathcal{L}_D$ to CIFAR-10 and at every iteration we examine the mean discriminator score for the mini-batch of the real set and the mean discriminator scores for the mini-batch of the fake dataset generated by the generator. We use the logit output of the discriminator network as the score. We plot these two mean scores against each iteration before training in the first row of Figure 5 and after training (with the original loss $\mathcal{L}_D$) in the second row. At each of the above training iterations, we replace $\mathcal{L}_D$ by the aw-loss $\mathcal{L}_D^{aw}$ (2) and
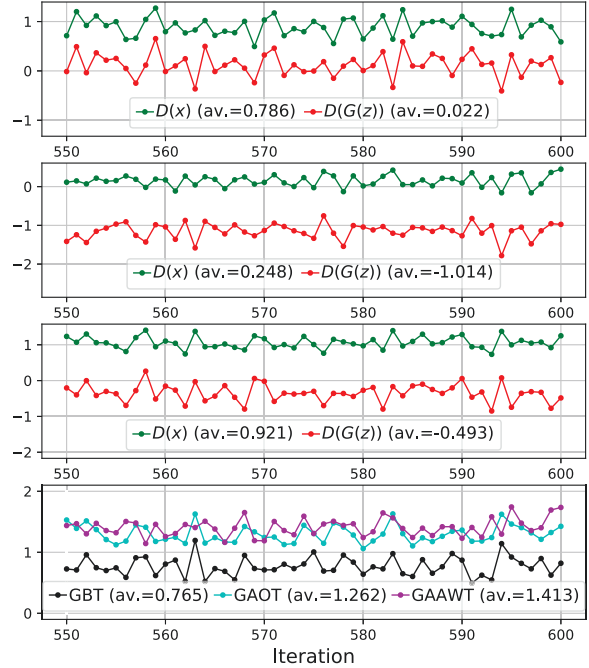
Figure 5: Mean discriminator scores for real data $D(x)$ and fake data $D(G(z))$ (Row 1: before training, Row 2: after original training with $\mathcal{L}_D$, Row 3: after training with aw-loss $\mathcal{L}_D^{aw}$) and their gap (Row 4: GBT - gap before training; GAOT - gap after original training; GAAWT - gap after aw-loss training)

train for one iteration with the same training mini-batch. We plot the mean discriminator scores for the mini-batches of the real and fake dataset after this training in the third row of Figure 5. We further present the gaps between the two scores before training and after training using the original loss and using aw-loss in the fourth row of Figure 5.

Figure 5 shows that training with aw-loss leads to higher real discriminator scores (0.921 epoch average) than training with the original loss (0.248 epoch average). The average gap between real and fake scores is also larger with the aw-loss at 1.413 vs. 1.262 of the original loss. Therefore, with the same model and the same training mini-batch, the aw-loss produces higher discriminator scores for the real dataset and larger gaps between real and fake scores. These are two important properties of a discriminator for the generator training.

5.3. Instability and Real Discriminator Scores

Our third study examines benefits of high discriminator scores for a real dataset with respect to instability and mode collapse of GAN training. We use a synthetic dataset with a mixture of Gaussian distributions used to test unrolled GAN
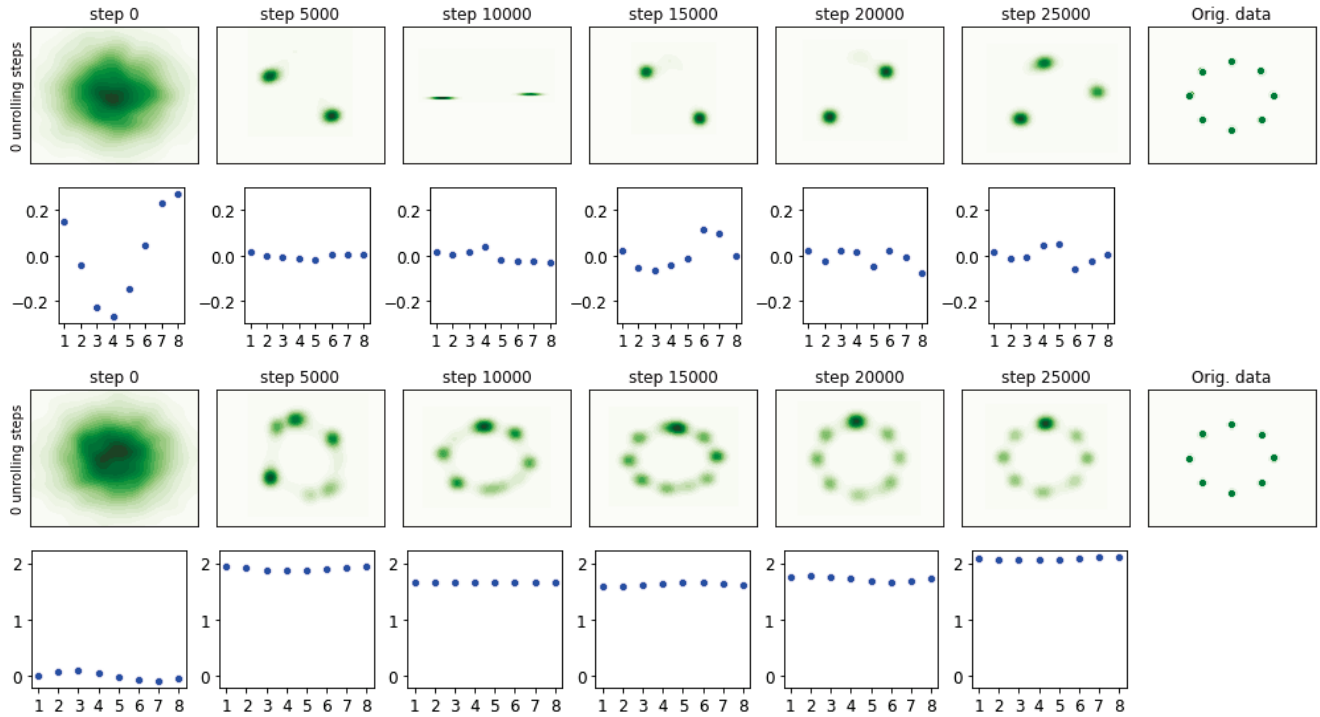
Figure 6: Mixture of eight 2D Gaussian distributions centered at 8 points (right-most column). Row 1: GAN sample points produced by generators; Row 2: GAN mean discriminator scores for each of 8 classes; Row 3: aw-GAN sample points produced by generators; Row 4: aw-GAN mean discriminator scores for each of eight classes;

in [33]. The dataset consists of eight 2D Gaussian distributions centered at eight equally distanced points on the unit circle. We train with a plain GAN as in [33] and we plot samples of (fake) data generated by the generator every 5,000 iterations on the first row of of Figure 6. We see that the generated data converging to two or three points but then moving off, demonstrating instability and mode collapse. To understand this phenomenon, at each of the iterations that we study in Figure 6, we generate 100 (real) data points from each of the eight classes and compute their mean discriminator scores (as the logit output of the discriminator). We plot the mean scores against the classes in the second row of Figure 6. We observe that the discriminator scores for the real data do not increase much during training, staying around 0, which corresponds to 0.5 probability after the logistic sigmoid function. The scores are also uneven among different classes. We believe these cause the instability in the generator training.

We compare the GAN results with aw-GAN that applies our adaptive weighted discriminator to the plain GAN and we present the corresponding plots of generated data points (fake) in the third row of Figure 6 and the corresponding discriminator scores on the eight classes in the bottom row. In this case, the generator gradually converges to all eight classes and the discriminator scores stay high for all eight classes. Even though the generator was starting to converge to a few classes (step 5,000), the discriminator scores remain high for all classes. Then the generator continues to converge while convergence to other classes occurs. We believe the high real discriminator scores maintains stability and prevents mode collapse in this case.

Conclusions

This paper pinpoints potential negative effects of the traditional GAN training on the real loss (and fake loss) and points out that this is a potential cause of instability and mode collapse. To remedy these issues, we have proposed the Adaptive Weighted Discriminator method to increase and maintain high real loss. Our experiments demonstrate the benefits and the competitiveness of this method.

Acknowledgements

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 2

[2] Mikolaj Binkowski, R. Devon Hjelm, and Aaron C. Courville. Batch weight for domain adaptation with mass shift. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1844–1853, 2019. 2

[3] Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 2

[4] Andy Brock. BigGAN-PyTorch. https://github.com/ajbrock/BigGAN-PyTorch. 6, 15, 16

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 2, 5, 6, 16

[6] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016. 2

[7] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. Your gan is secretly an energy-based model and you should use discriminator driven latent sampling, 2020. 3

[8] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby. Self-supervised gans via auxiliary rotation loss. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12146–12155, 2019. 6

[9] Casey Chu, Kentaro Minami, and Kenji Fukumizu. Smoothness and stability in gans. In *International Conference on Learning Representations*, 2020. 2

[10] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. 5, 14

[11] Maurice Diesendruck, Ethan R Elenberg, Rajat Sen, Guy W Cole, Sanjay Shakkottai, and Sinead A Williamson. Importance weighted generative networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 249–265. Springer, 2019. 2

[12] Hao-Wen Dong and Yi-Hsuan Yang. Towards a deeper understanding of adversarial losses under a discriminative adversarial network setting, 2019. 2

[13] Sivan Doveh and Raja Giryes. Degas: Differentiable efficient generator search, 2019. 2

[14] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. 2, 5, 6, 16

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 1, 2, 3

[16] 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*, NIPS'17, pages 5769–5779, Red Hook, NY, USA, 2017. Curran Associates Inc. 1, 2

[17] Hao He, Hao Wang, Guang-He Lee, and Yonglong Tian. Probgan: Towards probabilistic gan with theoretical guarantees. In *ICLR*, 2019. 6

[18] 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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6626–6637. Curran Associates, Inc., 2017. 5

[19] Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*, 2018. 5, 6

[20] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. In *International Conference on Learning Representations*, 2019. 2

[21] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 2

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020. 5

[23] Ilya Kavalerov, Wojciech Czaja, and Rama Chellappa. A multi-class hinge loss for conditional gans. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1290–1299, January 2021. 6

[24] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 15

[25] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 5, 14

[26] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2203–2213. Curran Associates, Inc., 2017. 2

[27] Jerry Li, Aleksander Madry, John Peebles, and Ludwig Schmidt. On the limitations of first-order approximation in GAN dynamics. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3005–3013, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. 2

[28] Jae Hyun Lim and Jong Chul Ye. Geometric gan, 2017. 1, 2, 5, 15

[29] Mario Lucic, Karol Kurach, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Are gans created equal? a large-scale study. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 698–707, Red Hook, NY, USA, 2018. Curran Associates Inc. 5

[30] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017. 1, 2

[31] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1825–1835. Curran Associates, Inc., 2017. 2

[32] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018. 2, 3

[33] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017*, 2017. 2, 8

[34] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 2, 5, 6, 16

[35] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *International Conference on Learning Representations*, 2018. 6

[36] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in neural information processing systems*, pages 5585–5595, 2017. 2

[37] Tu Dinh Nguyen, Trung Le, Hung Vu, and Dinh Q. Phung. Dual discriminator generative adversarial nets. *CoRR*, abs/1709.03831, 2017. 2

[38] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015. 2, 7

[39] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016. 2, 5

[40] Amartya Sanyal, Philip HS Torr, and Puneet K Dokania. Stable rank normalization for improved generalization in neural networks and gans. *arXiv preprint arXiv:1906.04659*, 2019. 2, 5, 6

[41] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan?, 2018. 6, 16

[42] 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, 2020. 2

[43] Dustin Tran, Rajesh Ranganath, and David M. Blei. Hierarchical implicit models and likelihood-free variational inference, 2017. 2, 5, 15

[44] Ngoc-Trung Tran, Viet-Hung Tran, Bao-Ngoc Nguyen, Linxiao Yang, and Ngai-Man (Man) Cheung. Self-supervised gan: Analysis and improvement with multi-class minimax game. In *Advances in Neural Information Processing Systems 32*, pages 13253–13264. Curran Associates, Inc., 2019. 5, 6

[45] Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, Martin Heusel, Hubert Ramsauer, and Sepp Hochreiter. Coulomb gans: Provably optimal nash equilibria via potential fields. *CoRR*, abs/1708.08819, 2017. 2

[46] Wei Wang, Yuan Sun, and Saman Halgamuge. Improving MMD-GAN training with repulsive loss function. In *ICLR*, 2019. 2, 5, 6