22 The Minimax Principle II: Generative Adversarial Networks

Today we have powerful, modern generative frameworks like diffusion models (Chapter 17) and autoregressive transformers/LLMs (Chapter 20). These models are trained by optimizing a well-defined, **static** objective, such as a variational bound on the log-likelihood or a score-matching loss. Chapter 21 introduced adversarial examples, and the problem of making a classifier deep net robust to inputs that have been adversarially modified via optimization to confuse deep nets.

We now proceed to study a conceptually different and more volatile paradigm, one whose influence extends generative modeling into the core of multi-agent AI. This paradigm was introduced by Goodfellow et al. [1] in the form of *Generative Adversarial Networks (GANs)*. The core innovation was to replace a fixed, static objective with a dynamic, two-player game between:

[1]

1. A **Generator (**G**)**, a counterfeiter learning to produce synthetic data that looks real.
2. A **Discriminator (**D**)**, a detective learning to distinguish the generator's fakes from genuine data.

This framework is particularly elegant. It sidesteps the need to define an explicit likelihood function $p_{\text{data}}(x)$, which is often intractable. So long as a discriminator can learn to find *some* statistical difference between real and fake, it can provide a gradient to the generator. This fundamentally changes the learning problem. It is no longer a simple optimization task; it is a **two-player, zero-sum game**.

22.1 Rethinking Distributional Distance as an Adversary

Our goal is to train a generator G such that its output distribution, p_g, is close to the true data distribution, p_{data}. This immediately raises a question: how do we measure the distance between two distributions?

In statistics, there are many such measures. A fundamental one is the *Total Variation (TV) distance*, also known as the ℓ_1 distance, defined as:

$$d_{TV}(p_{\text{data}}, p_g) = \frac{1}{2}\int_x |p_{\text{data}}(x) - p_g(x)|\, dx$$

While simple to define, this distance is impossible to compute directly, as it requires knowing the density functions p_{data} and p_g everywhere. However, the TV distance has a beautiful alternative interpretation in terms of a classification problem. This provides our first link between statistical distance and an adversary.

Imagine a simple, binary classifier—a discriminator D—whose job is to distinguish between samples from p_{data} and p_g. The discriminator sees a sample x and outputs a guess, labeling it as either "real" (1) or "fake" (0). What is the best possible performance such a classifier could achieve? This is directly related to the TV distance.

Lemma. The total variation distance between p_{data} and p_g is equal to the maximum advantage that any discriminator $D : \mathcal{X} \to \{0,1\}$ can achieve over random guessing. Specifically,

$$d_{TV}(p_{\text{data}}, p_g) = \sup_{D:\mathcal{X}\to\{0,1\}} \left| \mathbb{E}_{x\sim p_{\text{data}}}[D(x)] - \mathbb{E}_{x\sim p_g}[D(x)] \right|$$

Proof Sketch. The supremum is over all possible ways to partition the data space into a "real" set $A = \{x | D(x) = 1\}$ and a "fake" set A^c. The term inside the supremum is simply $|p_{\text{data}}(A) - p_g(A)|$. To maximize this difference, the optimal discriminator should choose the set $A^* = \{x | p_{\text{data}}(x) > p_g(x)\}$. For this optimal choice, one can show that $p_{\text{data}}(A^*) - p_g(A^*)$ is precisely equal to the TV distance.

This lemma provides a powerful new perspective. Instead of thinking about an abstract statistical formula, we can think about a concrete game:

The distance between two distributions can be measured by the success of an optimal adversary trying to tell them apart.

If the distributions are very different, a good discriminator will exist and will succeed with high probability. If the distributions are identical, no discriminator can do better than chance, and the distance is zero.

Of course the above observation is of no practical use, because the optimal adversary involves actual likelihoods of the unknown distribution. But it is the conceptual starting point for GANs. The GAN

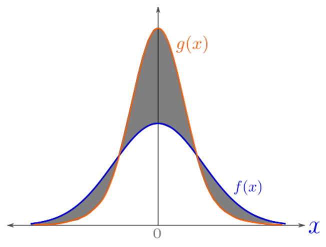

Figure 22.1: The optimal discriminator between two densities f and g corresponds to the probability mass of the shaded region.

framework takes this idea and generalizes it. Instead of a simple binary classifier and the TV distance, we use a more powerful, "soft" discriminator—a neural network that outputs a continuous probability $D(x) \in [0, 1]$. This more expressive adversary, when pitted against the generator in a minimax game, will define a different and more suitable divergence, leading us directly to the GAN value function.

22.2 *The GAN Value Function: A Two-Player, Zero-Sum Game*

The Total Variation distance lemma gives us a powerful starting point: the success of an optimal, "hard" binary classifier measures a valid distance between distributions. The GAN framework generalizes this principle. Instead of a classifier that makes a binary decision $D(x) \in \{0, 1\}$, we use a "soft" discriminator whose output is a probability, $D(x) \in [0, 1]$. This more expressive adversary, when pitted against a generator, defines a richer and more suitable objective for training.

The game is formalized by a single **value function**, $V(D, G)$, which the discriminator seeks to maximize and the generator seeks to minimize. The standard GAN value function is:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \quad (22.1)$$

Let's break down the roles of the two players in this minimax objective.

The Discriminator's Goal: Maximizing the Payoff The discriminator, D, wants to maximize the value function. For any fixed generator G, the expression $V(D, G)$ is simply the log-likelihood for a binary classifier trained on a balanced dataset of real and fake samples. To maximize this value, D must learn to:

- Output $D(x) \to 1$ for real samples from p_{data}. This makes the first term, $\log D(x)$, approach its maximum of 0.
- Output $D(G(z)) \to 0$ for fake samples from the generator. This makes the second term, $\log(1 - D(G(z)))$, also approach its maximum of 0.

A perfect discriminator would assign the correct label with certainty, achieving the maximum possible value of $V = 0$.

The Generator's Goal: Minimizing the Payoff The generator, G, wants to minimize the value function. It cannot control the first term, so its entire strategy revolves around the second. To minimize the value, G must produce samples $G(z)$ that fool the discriminator. Specifically, it wants to make $D(G(z)) \to 1$. As the discriminator becomes more convinced that the generator's fakes are real, the term $\log(1 - D(G(z)))$ plummets towards $-\infty$. The generator's victory is the discriminator's confusion.

This minimax formulation elegantly captures the adversarial dynamic. The training process involves alternating gradient-based updates: we take a step of gradient *ascent* on D's parameters to improve its classification, followed by a step of gradient *descent* on G's parameters to improve its ability to fool the updated D. The core theoretical question is: where does this process lead?

22.3 The Nash Equilibrium: When the Game is "Solved"

The minimax value function defines the rules of the game, but what does it mean to solve it? In a zero-sum game, a solution is a *Nash Equilibrium*, a state where neither player can improve their outcome by unilaterally changing their strategy. In the GAN setting this corresponds to a pair (G^*, D^*) where the generator produces samples that are indistinguishable from real data, and the discriminator is unable on average to detect a difference between synthetic and real samples.

To find this equilibrium, we first analyze the adversary's optimal move. We will show that for any given generator G, there is a single best possible discriminator, D^*_G. Then, we will substitute this optimal discriminator back into the value function to see what the generator's objective becomes when faced with a perfect opponent.

The Optimal Discriminator Let's fix the generator G. The distribution it produces, p_g, is therefore also fixed. The discriminator's goal is to find a function $D(x)$ that maximizes the value function $V(D, G)$:

$$V(D, G) = \int_x p_{\text{data}}(x) \log D(x)\, dx + \int_x p_g(x) \log(1 - D(x))\, dx$$

Since the integral is over all x, we can maximize the value function by maximizing the integrand point-wise for every value of x. For any given x, let's write $p_d = p_{\text{data}}(x)$, $p_g = p_g(x)$, and $y = D(x)$. We want to find the value of $y \in [0, 1]$ that maximizes the expression $p_d \log(y) + p_g \log(1 - y)$. Taking the derivative with respect to y and setting it to zero gives:

$$\frac{p_d}{y} - \frac{p_g}{1-y} = 0 \implies p_d(1-y) = p_g y \implies y = \frac{p_d}{p_d + p_g}$$

This yields the formula for the optimal discriminator, D^*_G, given a fixed generator G:

$$D^*_G(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \tag{22.2}$$

This is a remarkable result. It states that the theoretically perfect discriminator's output at a point x is simply the ratio of the true data density to the sum of the true and generated densities. If $p_g(x)$ is low and $p_{\text{data}}(x)$ is high, $D^*_G(x)$ approaches 1. If the reverse is true, it approaches 0.

The Generator's Objective Against a Perfect Adversary Now we perform the crucial step. We analyze the generator's objective under the assumption that it is playing against this perfect, omniscient discriminator. We substitute $D^*_G(x)$ back into the value function $V(G, D)$:

$$\begin{aligned} C(G) &= \max_D V(G, D) = V(G, D^*_G) \\ &= \mathbb{E}_{x\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\left[\log \frac{p_g(x)}{p_{\text{data}}(x) + p_g(x)}\right] \end{aligned}$$

This expression can be rewritten in terms of standard information-theoretic divergences. Let $M = \frac{p_{\text{data}}+p_g}{2}$ be the average distribution.

$$\begin{aligned} C(G) &= \mathbb{E}_{x\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(x)}{2M(x)}\right] + \mathbb{E}_{x\sim p_g}\left[\log \frac{p_g(x)}{2M(x)}\right] \\ &= \mathbb{E}_{p_{\text{data}}}\left[\log \frac{p_{\text{data}}}{M}\right] - \log 2 + \mathbb{E}_{p_g}\left[\log \frac{p_g}{M}\right] - \log 2 \\ &= KL(p_{\text{data}}||M) + KL(p_g||M) - 2\log 2 \end{aligned}$$

The Kullback-Leibler (KL) divergence terms define another, symmetric divergence.

Definition (Jensen-Shannon Divergence). The *Jensen-Shannon Divergence (JSD)* between two distributions P and Q is given by:

$$JSD(P||Q) = \frac{1}{2}KL\left(P||\frac{P+Q}{2}\right) + \frac{1}{2}KL\left(Q||\frac{P+Q}{2}\right)$$

Using this definition, we arrive at the final form of the generator's objective when playing against a perfect adversary:

$$C(G) = 2 \cdot JSD(p_{\text{data}}||p_g) - 2\log 2 \qquad (22.3)$$

This provides a profound interpretation of the GAN objective. The generator's task of fooling an optimal discriminator is mathematically equivalent to minimizing the Jensen-Shannon Divergence between the real data distribution and the one it generates.

The minimum possible value of the JSD is 0, which occurs if and only if $p_g = p_{\text{data}}$. At this point, the generator has perfectly learned the true data distribution. The optimal discriminator's output becomes $D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x)+p_{\text{data}}(x)} = \frac{1}{2}$ for all x. This is the unique **Nash Equilibrium** of the game: the generator perfectly mimics the data, and the discriminator is maximally confused, unable to perform better than a random coin flip. The value of the game at equilibrium is $C(G^*) = -2\log 2$.

22.3.1 A Zoo of Adversaries: Different Games, Different Distances

The analysis in the previous section is not unique to the Jensen-Shannon Divergence. The choice of the value function $V(D, G)$ is a design decision that implicitly defines the rules of the game. By changing these rules, we change the statistical divergence that the generator minimizes when playing against an optimal discriminator. This insight was formalized in a general framework known as *f-GAN* [2].

[2]

The starting point is the family of *f-divergences*, a general class of distances between distributions P and Q defined by a convex function f where $f(1) = 0$:

$$D_f(P||Q) = \int_x Q(x) f\left(\frac{P(x)}{Q(x)}\right) dx$$

The key result of the f-GAN framework is that for any choice of f, one can construct a value function $V(D, G)$ such that $\max_D V(D, G) = D_f(p_{\text{data}}||p_g)$. This turns the GAN into a general-purpose tool for divergence minimization. The table below shows several examples, connecting the choice of divergence to the structure of the discriminator's objective.

Div.	**Objective** $D_f(p_{\text{data}}\|\|p_g)$	**Discriminator Output** $D(x)$
JS	$JSD(p_{\text{data}}\|\|p_g)$	$\log(2) - \log(1 + e^{-V(x)})$
KL	$KL(p_{\text{data}}\|\|p_g)$	$1 + V(x)$
RKL	$KL(p_g\|\|p_{\text{data}})$	$-e^{-V(x)}$
TV	$TV(p_{\text{data}}, p_g)$	$0.5 \cdot \tanh(V(x))$

Table 22.1: Different choices of divergence and their corresponding discriminator objectives in the f-GAN framework. Acronyms: JS (Jensen-Shannon), KL (Kullback-Leibler), RKL (Reverse KL), TV (Total Variation).

A Different Kind of Game: Wasserstein GAN An alternative and highly influential approach is the Wasserstein GAN (WGAN) [3]. Instead of an f-divergence, it aims to minimize the *Earth-Mover (or Wasserstein-1) distance*. This distance has a dual form based on an optimization over a class of functions:

[3]

$$W(p_{\text{data}}, p_g) = \sup_{\|f\|_L \le 1} \left(\mathbb{E}_{x\sim p_{\text{data}}}[f(x)] - \mathbb{E}_{x\sim p_g}[f(x)]\right)$$

The supremum is taken over all 1-Lipschitz functions. This leads to a different and much simpler value function:

$$\min_G \max_{D\in\mathcal{D}} V(D, G) = \mathbb{E}_{x\sim p_{\text{data}}}[D(x)] - \mathbb{E}_{z\sim p_z}[D(G(z))]$$

Here, the discriminator D (called the "critic" in this context) is not a classifier outputting a probability. It is simply a function whose Lipschitz constant is constrained to be at most 1.

This change has profound practical implications. The critic's job is not to perfectly separate real from fake, but to find a function that has the largest possible "slope" between the two distributions. This provides smooth, non-saturating gradients for the generator, even when the critic is very accurate, mitigating the vanishing gradient problem and leading to much more stable training. This illustrates a key lesson: the theoretical properties of the game being played have direct consequences for the stability and performance of the algorithm.

22.4 *The Challenge of Finding Equilibrium: Why GANs Fail*

The theoretical derivation of the Nash Equilibrium is elegant, but it rests on a powerful, unrealistic assumption: that for any given generator G, we can instantly find and deploy the optimal discriminator D_G^*. In practice, we do not have access to the true probability densities p_{data} and p_g, so we cannot compute D_G^* analytically.

Instead, we are forced to approximate the game using an iterative, gradient-based algorithm. The training process consists of simultaneous, alternating updates:

1. Take a small step of gradient **ascent** to improve the discriminator: $\theta_d \leftarrow \theta_d + \eta \nabla_{\theta_d} V(D, G)$.
2. Take a small step of gradient **descent** to improve the generator: $\theta_g \leftarrow \theta_g - \eta \nabla_{\theta_g} V(D, G)$.

This process of alternating gradient updates is fundamentally different from standard optimization and not guaranteed to converge. The pathologies of GAN training arise from this dynamic.

The Pathology of Simultaneous Updates: A Toy Example To build intuition, consider the simplest possible non-trivial minimax game: the bilinear game on $\mathbb{R}$, with the value function $V(x, y) = xy$. The goal is to solve:

$$\min_x \max_y xy$$

The unique Nash Equilibrium is clearly at the saddle point $(x, y) = (0, 0)$. Let's see what simultaneous gradient updates do. The gradients are:

$$\nabla_x V = y \quad \text{and} \quad \nabla_y V = x$$

The update rules are therefore:

$$x_{t+1} = x_t - \eta \nabla_x V = x_t - \eta y_t$$

$$y_{t+1} = y_t + \eta \nabla_y V = y_t + \eta x_t$$

This system of updates is not a descent towards the equilibrium at $(0, 0)$. It is a rotation. The parameters (x_t, y_t) will orbit the origin in a circle (or spiral outwards if η is too large), never converging to the solution. The players are perpetually chasing each other around the saddle point.

Figure 22.2: The gradient field for the game $\min_x \max_y xy$. The vectors do not point towards the equilibrium at the origin; they flow in a circle around it. Simultaneous gradient updates will follow this rotation.

Reinterpreting GAN Failures as Strategic Failures This simple rotational dynamic is the key to understanding the practical difficulties of training GANs. The infamous failure modes are not bugs in the implementation, but symptoms of the failure to converge to the Nash Equilibrium.

Oscillations and Instability The most direct symptom is the unstable loss curve. Unlike in standard deep learning where the loss is expected to steadily decrease, the GAN value function often oscillates wildly during training. This is a high-dimensional echo of the rotational dynamics in our toy example. The generator and discriminator are locked in a chase, repeatedly finding temporary advantages over one another without ever settling at the equilibrium.

Mode Collapse This is a more subtle, but more damaging, strategic failure. *Mode collapse* occurs when the generator learns to produce only a few, or even just one, highly plausible sample. For example, a generator trained on a diverse dataset of animal faces might learn to produce only a single, perfect-looking cat. Why does this happen? The generator has discovered a strategy that decisively beats the *current* discriminator. By concentrating all its probability mass on one perfect cat, it presents an easy-to-learn target for the discriminator, which quickly learns to reject everything else. However, this is a strategic dead-end. It is a local victory for the generator that prevents it from ever achieving the global solution, which is to learn the entire data distribution. It has "won" a battle but lost the war, failing to converge to the game's true equilibrium.

Ultimately, the challenge of GANs is that gradient descent is designed to find the minimum of a function, but in a minimax game, the goal is to find an equilibrium. The gradient dynamics do not always point towards this equilibrium, leading to the instabilities that have motivated a rich and ongoing field of research into more stable training methods for adversarial learning.

22.5 "Mode Collapse" for GANs may be unavoidable

Section 16.2.1 discussed complications arising when we try to learn distributions from finite samples. In the GAN setting the training objective (**??**) was described using the full distribution but of course in practice the discriminator D is discriminating between finite samples of the two distributions.

Usually in machine learning good generalization means that the average loss function on test dataset is similar to that on the training dataset. However, we saw that for the usual loss functions like log-likelihood, this notion of generalization does not imply that the distribution has been learned well. After GANs were introduced there was extensive study of whether GAN approach could bypass these issues, and in this effort a large number of training objectives and algorithms were tried. It was noted that they were learning the distribution fairly imperfectly[4] but it was unclear whether this would go away with bigger training datasets or different objectives.

[4] A representative problem is *mode collapse*: the learnt distribution does not appear to have the same amount of diversity as the

Example 22.5.1. *Since the objective (**??**) allows maximization over all neural nets D of the allowed architecture, even two different sample sets from the same distribution can "look" very different to a neural net. For example if we take two sets of d^3 samples from the d-dimensional Gaussian $\mathcal{N}(0, I)$ they are distinguishable by a deep net that is somewhat larger than d^3. The reason is that the samples are two discrete sets in which all pairs of points are almost orthogonal (with high probability). While this may appear to be an artificial example, it is the case that in real-life GAN training, the objective does not usually drop to zero —the distinguisher net at the end is still able to somewhat distinguish the samples from the two distributions.*

We now describe theoretical analysis from [5] showing that the quality of the learnt distribution is inherently limited by the *representational capacity* of the discriminator *regardless of how large we make the the training dataset*.

[5] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and Equilibrium in Generative Adversarial Nets (GANs). *Proc. ICML*, 2017

As usual when discussing generalization, let us assume the net has some finite size, say N, and the sample size of the distributions is large enough for nets of size N to generalize. The following two problems.

Problem 22.5.2. *Suppose the loss is C-Lipschitz and the parameter vector is in $\Re^d$ and has ℓ_2-norm at most L. Suppose this discriminator trained on a training set of size N achieved training loss at most ϵ_1. Show that if $N > \Omega(\frac{L}{\epsilon_2^2} \log(C/\epsilon_2))$ then it has test loss at most $\epsilon_1 + \epsilon_2$ on the full distribution.* [6]

[6] Hint: Use the results of Chapter 5, eg Theorem 5.2.7.

Does this imply that GANs actually learn the distribution? No, just as in Example 16.2.1: generalization only implies training and test loss are close, not that the distributions are close.

Problem 22.5.3. *Under the same conditions as Problem 22.5.2 show that if the learned distribution P_{synth} has finite support and is a uniform distribution on $\Omega(\frac{L}{\epsilon_2^2}\log(C/\epsilon_2))$ random iid samples from P_{real} then no discriminator can achieve test loss more than $\epsilon_1 + \epsilon_2$ when comparing the distributions P_{synth} and P_{real}.*

In the previous problem, let's think of P_{real} as the distribution on all real-life images. Then P_{synth} is quite different from P_{real}: it is the uniform distribution on some small set of real-life images. Nevertheless no discriminator (whose size, norm and Lipschitz constant are suitably upper bounded) can distinguish between the two distributions. As we will see, such a P_{synth} is indeed observed in practice. Furthermore, changes to GANs architecture (e.g. Encoder-Decoder GANs) do not affect this basic result [7].

[7] Do GANs learn the Distribution? Some Theory and Empirics

The following exercise gives a (probably very loose) upper bound on the size of a generator that can produce such a P_{synth}.

Problem 22.5.4. *Show that the uniform distribution on M images where each image is in $\Re^d$ an be generated using a net with $O(Md^2)$ parameters.* [8]

[8] More precisely the distribution will be a mixture of gaussians of tiny variance centered at the M images.

Putting together the previous three problems we conclude that in the following scenario the GAN objective is insufficient to prevent the verifier from learning a distribution supported on a small finite set of images ("mode collapse"): The generator has somewhat larger "capacity" than the discriminator. [9]

[9] It is an open question whether mode collapse can be avoided when the discriminator is much larger, although in that case usually the training loss is hard to reduce, for reasons explored in Example 22.5.1.

22.5.1 *Experimental verification of Mode Collapse: Birthday Paradox Test*

The theory above suggests that GANs trained using discriminators of a certain "capacity"(in the sense of generalization theory) have solutions with low training and test error where the synthetic distribution P_{synth} is supported on a small set of images and thus is quite different from P_{real}). This phenomenon of the GAN ending up with a synthetic distribution consisting of a small set of images is called *mode collapse* and was earlier believed to be a result of either failed training or using a training set of real images that is too small. The results above suggested that mode collapse is not a surprising outcome with generators and discriminators of low-ish capacity (as opposed to capacity that scales with the number of distinct modes in P_{real}.)

This raised a question whether we can detect such model collapse in real-life GANs. In other words, estimate how many "distinct"images it can produce. At first glance, such an estimation seems very difficult. After all, automated/heuristic measures of image similarity can be easily fooled, and we humans surely don't have enough time to go through millions or billions of images, right?

Luckily, a crude estimate is possible using the simple birthday paradox, a staple of undergrad discrete math.

Problem 22.5.5 (Birthday paradox). [10] *Consider a uniform distribution on a set of size N. Show that a random sample of size* $2\sqrt{N}$ *contains a duplicate probability at least* $1 - 1/e$. *(The name for this paradox comes from its implication that if you put* 23 *random people in a room, then the odds are good that two of them have the same birthday is significant.)*

[10] Do GANs learn the Distribution? Some Theory and Empirics

Let's realize the implications for GANs. Imagine for argument's sake that P_{real} is the distribution on pictures of faces. What is the number of modes in this distribution? At a minimum it is the number of distinct human faces? This feels like a rather large set, because all of us know tens of thousands of faces (including those encountered in the news) and don't see any unrelated *doppelgangers*; only identical twins. More precisely, the birthday paradox says that if the number of distinct human faces is N then we would expect to have seen doppelgangers after having seen $\sqrt{N}$ faces.

In the GAN setting, the distribution is continuous, not discrete. Thus our proposed birthday paradox test for GANs [11] is as follows.

[11] Sanjeev Arora and Yi Zhang. Theoretical Limitations of Encoder-Decoder GANs Architectures. *Proc. ICLR*, 2018

(a) Pick a sample of size s from the generated distribution. (b) Use an automated measure of image similarity to flag the 20 (say) most similar pairs in the sample. (c) Visually inspect the flagged pairs and check for images that a human would consider near-duplicates. (d) Repeat.

If this test reveals that samples of size s have duplicate images with good probability, then suspect that the distribution has support size about s^2.

22.5.2 *Other notes on GANs and mode collapse*

While recent GANs (such as Progressive GAN) use very large discriminators and generators to produce images of better visual quality (as judged by humans) they still suffer from mode collapse, in line with the above theory.

On the other hand, Florian et al [12] argue that the above analysis takes assumes static near-equilibrium in training, whereas real-life training never arrives at an equilibrium, and the training dynamics resulting from non-equilibrium can act as a power shaper of the GAN's behavior.

[12] F Schaefer, H Zheng, and A Anandkumar. Implicit competitive regularization in GANs. *ICML*, 2020

A recent paper [13] shows that even though GANs suffer from mode collapse, they can be used to predict generalization. In other words, given a dataset S and a discriminative model trained on it, use the following predicter of generalization error: train a conditional GAN using S and use random samples from the trained GAN in lieu of

[13] Y Zhang, A Gupta, N Saunshi, and S Arora. On predicting generalization using gans. *ICLR*, 2022

held-out data to predict generalization. This is shown to work better than other predictors of generalization error.

The basic idea of GANs has been extended for other settings, most notably to learn good image to image maps (e.g., changing a photograph to a painting, or virtually trying on an article of clothing on the image of a person). This works very well and there is no analog of the mode collapse result.

22.6 *The Legacy of the Adversary: From GANs to Modern AI*

While diffusion models have largely superseded GANs for state-of-the-art image synthesis, the core principle pioneered by GANs—using a learned adversary to provide a rich, adaptive training signal—has become a fundamental tool in the modern AI toolkit. The minimax game, which we have analyzed in this chapter, is not just a historical curiosity; it is a live paradigm that appears in many high-stakes domains, often where the notion of a fixed loss function is insufficient.

AI Safety via Red Teaming. The process of safety-tuning large language models is a direct, large-scale implementation of adversarial training. An LLM provider (the **Learner**) seeks to minimize the model's propensity to generate harmful, biased, or unsafe content. To do this, they employ a "red team" (the **Adversary**), composed of human experts and other AI models, whose sole job is to find prompts and inputs that break the model's safety rules. This is an explicit minimax game:

$$\min_{\text{Model}} \max_{\text{Red Team}} \text{Harmfulness}(\text{Model}(\text{prompt}))$$

The data generated by the red team is then used to fine-tune the model, making it more robust. This is precisely the GAN dynamic, applied to the domain of model alignment rather than image generation.

Generator-Critic Architectures for Reasoning. The GAN architecture has been abstracted to tackle complex reasoning tasks. In such setups, one LLM agent acts as a **Generator**, proposing a solution to a problem (e.g., a mathematical proof or a block of code). A second agent acts as a **Critic** (the adversary), which analyzes the proposed solution for flaws, logical errors, or counterexamples. The feedback from the critic provides a powerful, targeted signal that the generator can use to improve its reasoning process. This adversarial loop allows models to refine their outputs in a way that is difficult to achieve with a static loss function alone.

Adversarial Self-Play for Skill Discovery. In reinforcement learning, an agent can become its own adversary to create a curriculum for learning. For example, in the framework of Asymmetric Self-Play, a "teacher" agent (the adversary) proposes goals for a "student" agent (the learner). The teacher's goal is to propose tasks that are at the perfect level of difficulty for the student—not too easy, not too hard. This can be framed as a minimax game where the student tries to maximize its reward, and the teacher tries to propose goals that keep the student's success rate within a certain range. This adversarial dynamic pushes the agent to continually improve and master a wide range of skills.

In all these cases, the core idea from GANs endures: when we do not know how to write down a perfect loss function, we can instead define an adversary whose job is to find the flaws in our current system. The struggle against this adaptive adversary provides a powerful and inexhaustible source of training data, driving the learner towards greater capability and robustness. This principle, moving from static optimization to dynamic games, is the true legacy of the GAN framework and a central theme of modern AI.

Exercises

Problem 22.6.1 (Mode Collapse with Linear Discriminators)**.** *This problem makes the theory of mode collapse concrete for a simple discriminator class by explicitly working out the sample complexity required for a generator to fool it.*

Let the data be in $\mathbb{R}^d$. Consider a class of linear discriminators $\mathcal{D} = \{D(x) = \sigma(w^T x) \mid \|w\|_2 \leq L\}$, where $\sigma(z) = 1/(1+e^{-z})$ is the sigmoid function.

(a) Generalization bounds for a function class often depend on its Lipschitz constant. Show that for any $D \in \mathcal{D}$, the function $f_D(x) = \log(1 - D(x))$ is $(L/2)$-Lipschitz. [14] *This implies the loss function in the GAN objective is Lipschitz with a constant C proportional to L.*

[14] Hint: Recall that $\sigma'(z) = \sigma(z)(1 - \sigma(z))$, which is maximized at $z = 0$.

(b) Based on the result from (a) and the generalization bounds presented in Chapter 5 (e.g., Theorem 5.2.7), the number of samples required to guarantee that the empirical loss is within ϵ of the true loss for all discriminators in $\mathcal{D}$ is $M = \Omega\left(\frac{L^2 d}{\epsilon^2}\right)$. Let us fix M to be this sample complexity.

(c) Now, consider a true data distribution p_{data} that is a uniform mixture of K well-separated modes, where $K \gg M$. For instance, the mixture of K gaussians $p_{data} = \frac{1}{k}\sum_{i=1}^{K} \mathcal{N}(c \cdot e_i, \sigma^2 I)$ for large c and small σ, where $\{e_i\}$ are standard basis vectors. Let the generator's distribution, p_g, be the uniform distribution over a set S of just M samples drawn i.i.d. from

p_{data}.

(d) (Mode Collapse!) *Show no discriminator in $\mathcal{D}$ can distinguish p_g from p_{data}. Specifically, use the law of large numbers (or Hoeffding's inequality for a more rigorous argument) to show that for any fixed w with $\|w\|_2 \leq L$, the difference in expectations, $|\mathbb{E}_{x \sim p_{data}}[w^T x] - \mathbb{E}_{x \sim p_g}[w^T x]|$, goes to zero as $M \to \infty$. Conclude by explaining why this implies that for a sufficiently large M (as defined in part b), the value of the GAN game will be close to the optimal value of $-2\log 2$, even though the generator has collapsed K modes onto a support of size M.*

Problem 22.6.2 (Vanishing Gradients in Saturating GANs)**.** *This problem illustrates the practical benefit of the WGAN objective over the standard GAN objective. Consider a simplified 1D setting where p_{data} is a point mass at $x = a$ and the generator produces a point mass at $x = \theta$, where θ is the single parameter of the generator G.*

(a) ***Standard GAN**: Assume the discriminator has enough capacity to perfectly separate the two point masses when $a \neq \theta$. What is the optimal discriminator $D^*(x)$? Substitute this into the generator's objective, $C(G) = \max_D V(G, D)$. Show that for any $\theta \neq a$, the gradient $\frac{dC(G)}{d\theta}$ is zero. Explain why this is a problem for training.*

(b) ***Wasserstein GAN**: The WGAN value function is $W(G, D) = \mathbb{E}_{x \sim p_{data}}[D(x)] - \mathbb{E}_{x \sim p_g}[D(x)]$, where D must be 1-Lipschitz. The optimal critic D^* will be a function that maximizes this difference. Find a simple 1-Lipschitz function D^* that is optimal for this problem.*

(c) *Using your D^* from part (b), compute the generator's objective $W(G) = \max_D W(G, D)$. Show that the gradient $\frac{dW(G)}{d\theta}$ is non-zero and constant for $\theta \neq a$. Explain why this gradient is more useful for training than the one from part (a).*

Problem 22.6.3 (Stabilizing Rotational Dynamics)**.** *The complex dynamics of GAN training near a saddle point can often be understood by studying a local, linear approximation of the game. This leads to the **bilinear game**, whose value function in one dimension is $V(x, y) = xy$. Here, $x \in \mathbb{R}$ can be thought of as the generator's parameter deviation from its equilibrium value, and $y \in \mathbb{R}$ as the discriminator's.*

As shown in Section 22.4, simultaneous gradient updates for $\min_x \max_y V(x, y)$ lead to rotational dynamics that do not converge. This problem explores how a simple form of regularization, equivalent to penalizing the discriminator's gradient, can stabilize these dynamics.

(a) *Consider the regularized game:*

$$\min_x \max_y V'(x, y) = xy - \frac{\alpha}{2} y^2$$

Derive the simultaneous gradient update rules for x and y under this new objective.

(b) *Show that for any $\alpha > 0$ and a small enough learning rate η, the updates now cause the parameters (x_t, y_t) to spiral in towards the equilibrium at $(0, 0)$ instead of orbiting it indefinitely. (Hint: Analyze the squared distance to the origin, $r_t^2 = x_t^2 + y_t^2$, and show that $r_{t+1}^2 < r_t^2$ for any non-zero state (x_t, y_t)).*