Generative Adversarial Networks (GANs)

Dr. Kamal Mammadov

September 29, 2024

Probability Space - Definition

A probability space is a mathematical framework for representing random events. It consists of three elements:

- **Sample space** Ω: The set of all possible outcomes.
- **Event space** $\mathcal{F}$: A set of events (subsets of Ω).
- **Probability function** P: A function assigning probabilities to events, $P : \mathcal{F} \to [0, 1]$.

$$(\Omega, \mathcal{F}, P)$$

The function P must satisfy:

$$P(\Omega) = 1, \quad P(A) \geq 0 \quad \text{for any event } A \in \mathcal{F}$$

Probability Space - Example

Example: Tossing a fair coin

- **Sample space**: $\Omega = \{\text{Heads, Tails}\}$
- **Event space**: $\mathcal{F} = \{\emptyset, \{\text{Heads}\}, \{\text{Tails}\}, \Omega\}$
- **Probability function**: $P(\{\text{Heads}\}) = 0.5, P(\{\text{Tails}\}) = 0.5$

$$(\Omega, \mathcal{F}, P) = (\{\text{Heads, Tails}\}, \{\emptyset, \{\text{Heads}\}, \{\text{Tails}\}, \Omega\}, P)$$

Generative Adversarial Networks (GANs)

Definition: A GAN consists of two players:

- **Generator** G: Tries to create data that looks like it came from the real data distribution.
- **Discriminator** D: Tries to distinguish between real data and the data generated by G.

The Original GAN Game

The original GAN is defined as the following game:

- Each probability space $(\Omega, \mu_{\text{ref}})$ defines a GAN game.
- There are 2 players: generator and discriminator.
- The generator's strategy set is $\mathcal{P}(\Omega)$, the set of all probability measures μ_G on Ω.
- The discriminator's strategy set is the set of Markov kernels $\mu_D : \Omega \to \mathcal{P}[0,1]$, where $\mathcal{P}[0,1]$ is the set of probability measures on $[0,1]$.

The GAN game is a zero-sum game, with objective function:

$$L(\mu_G, \mu_D) := \mathbb{E}_{x\sim\mu_{\text{ref}}, y\sim\mu_D(x)}[\ln y] + \mathbb{E}_{x\sim\mu_G, y\sim\mu_D(x)}[\ln(1-y)].$$

The generator aims to minimize the objective, and the discriminator aims to maximize the objective.

Conceptual Explanation: Painter and Critic

Painter and Critic analogy:

- The **painter** (generator) tries to create convincing fake paintings.
- The **critic** (discriminator) evaluates each painting, aiming to tell if it's real or fake.

Now, let's refine this idea with **strategy sets**, the different ways the painter and critic can operate.

Discriminator's Strategy Set

In the most general case, the critic's strategy can be highly complex, using **Markov kernels** $\mu_D : \Omega \to \mathcal{P}[0,1]$. A Markov kernel allows the critic to output a probability distribution between 0 and 1.

However, the **optimal critic strategy** is usually simple and deterministic. Instead of a random Markov kernel, we can use a function:

$$D : \Omega \to [0,1]$$

where $D(x)$ returns a score between 0 and 1 deterministically. In practice, $D(x)$ is a deep neural network.

Generator's Strategy Set

The painter could use any possible probability distribution μ_G on Ω, but this is too broad.
Instead, the painter starts with **random noise** $z \sim \mu_Z$ and uses a function G to transform it into a painting:

$$G(z) \to \text{Final painting}$$

G is applied to the random noise, typically from a Gaussian or uniform distribution, to create the final output.

Objective Function: Painter and Critic Game

The GAN game can be described by the following objective function:

$$L(G, D) = \mathbb{E}_{x \sim \mu_{\mathrm{ref}}}[\ln D(x)] + \mathbb{E}_{z \sim \mu_Z}[\ln(1 - D(G(z)))]$$

The first term ensures the critic maximizes correct classifications of real paintings.
The second term ensures the painter minimizes the chance of the critic identifying fake paintings.

Move Order and Strategic Equilibria

In the original GAN paper, it is usually assumed that the generator moves first and the discriminator moves second, giving the following minimax game:

$$\min_G \max_D L(G, D) = \min_G \max_D \left(\mathbb{E}_{x \sim \mu_{\text{ref}}}[\ln D(x)] + \mathbb{E}_{z \sim \mu_Z}[\ln(1 - D(G(z)))] \right)$$

If the strategy sets for both players are finite, by the minimax theorem:

$$\min_G \max_D L(G, D) = \max_D \min_G L(G, D)$$

meaning the move order does not matter.

Nash Equilibrium of GANs

Nash Equilibrium defined as a strategy $(\hat{G}, \hat{D})$ satisfying:

$$\hat{D} \in \arg\max_D L(\hat{G}, D) \quad \hat{G} \in \arg\min_G L(G, \hat{D})$$

Theorem: For the original GAN game, the unique equilibrium point is given by

$$\hat{D}(x) = \frac{1}{2}, \quad \hat{G} = \mu_{\text{ref}}$$

The generator perfectly mimics the reference distribution, and the discriminator outputs $\frac{1}{2}$ for all inputs.

Conditional GANs

Definition: Conditional GANs allow the generator to create samples based on additional information (e.g., class labels). The generator produces a distribution $\mu_G(c)$, conditioned on the class label c.

$$L(\mu_G, D) = \mathbb{E}_{c\sim\mu_C, x\sim\mu_{\text{ref}}(c)}[\ln D(x, c)] + \mathbb{E}_{c\sim\mu_C, x\sim\mu_G(c)}[\ln(1 - D(x, c))]$$

Example: Generate a picture of a cat given the class label "cat".

Conditional GAN - Example

Conceptual Example:

- Painter receives the instruction to paint a landscape.
- Critic knows the painter should create a landscape, and will check if the painting matches the "landscape" instruction.

Both the generator and discriminator receive the class label to condition their behavior.

CycleGAN: Horse-Zebra Example

CycleGAN aims to translate a photo of a horse into a photo of a zebra, and vice versa. The goal is to perform this translation without needing paired examples (i.e., without matching horse-zebra pairs).

The Two Domains:

- **Horse domain** Ω_X: Set of all horse photos, with probability distribution μ_X.
- **Zebra domain** Ω_Y: Set of all zebra photos, with probability distribution μ_Y.

CycleGAN: The Players in the Game

There are four players in CycleGAN, divided into two teams:

Generators:

- $G_X : \Omega_X \to \Omega_Y$: Transforms horse photos into zebra photos.
- $G_Y : \Omega_Y \to \Omega_X$: Transforms zebra photos into horse photos.

Discriminators:

- $D_X : \Omega_X \to [0, 1]$: Judges whether a photo is a real horse photo or a generated one.
- $D_Y : \Omega_Y \to [0, 1]$: Judges whether a photo is a real zebra photo or a generated one.

The GAN Loss

Each generator and discriminator pair plays a typical GAN game:

- For the horse-to-zebra generator G_X and zebra discriminator D_Y:

$$L_{GAN}(G_X, D_Y)$$

- Similarly, for the zebra-to-horse generator G_Y and horse discriminator D_X:

$$L_{GAN}(G_Y, D_X)$$

$$L_{GAN}(G_X, D_Y) + L_{GAN}(G_Y, D_X)$$

This ensures each generator creates realistic images according to the respective discriminator.

Cycle Consistency Loss

The key idea of CycleGAN is **cycle consistency**. If you start with a horse photo, convert it to a zebra using G_X, then back to a horse using G_Y, the result should be close to the original horse photo.

Cycle Consistency Loss:

$$L_{cycle}(G_X, G_Y) = \mathbb{E}_{x \sim \mu_X} \| G_Y(G_X(x)) - x \| + \mathbb{E}_{y \sim \mu_Y} \| G_X(G_Y(y)) - y \|$$

This encourages meaningful transformations that preserve core features:

- Horse to zebra $G_X(x)$, and back to horse $G_Y(G_X(x))$, should produce an image close to the original horse x.
- Zebra to horse $G_Y(y)$, and back to zebra $G_X(G_Y(y))$, should produce an image close to the original zebra y.

The Full CycleGAN Objective

The full objective combines the GAN losses and cycle consistency loss:

$$L(G_X, G_Y, D_X, D_Y) = L_{GAN}(G_X, D_Y) + L_{GAN}(G_Y, D_X) + \lambda L_{cycle}(G_X, G_Y)$$

Where λ is a parameter that controls the importance of cycle consistency relative to the GAN loss.

- **Generators** G_X (horse-to-zebra) and G_Y (zebra-to-horse) aim to minimize this objective by creating realistic translations and ensuring cycle consistency.
- **Discriminators** D_X (horse discriminator) and D_Y (zebra discriminator) aim to maximize the objective by distinguishing real from fake images.

Super-Resolution with GANs: Conceptual Overview

Super-resolution involves converting a low-resolution image into a high-resolution image by filling in missing details.

- The generator takes a low-resolution image and tries to produce a high-resolution image that looks realistic.
- The discriminatorevaluates whether the generated high-resolution image is real or fake by comparing it to actual high-resolution images.

Goal: The generator learns to add meaningful details to make the low-resolution input look like a high-resolution image that can fool the discriminator.

The Super-Resolution GAN Loss Function

The GAN objective for super-resolution is similar to standard GANs:

$$L(G, D) = \mathbb{E}_{x \sim \mu_{\text{HR}}}[\ln D(x)] + \mathbb{E}_{x_{\text{LR}} \sim \mu_{\text{LR}}}[\ln(1 - D(G(x_{\text{LR}})))]$$

- $x \sim \mu_{\text{HR}}$: High-resolution images from the real data.
- $x_{\text{LR}} \sim \mu_{\text{LR}}$: Low-resolution input images.
- $G(x_{\text{LR}})$: Generated high-resolution image from the low-resolution input.

Discriminator's Job: Correctly identify real high-resolution images.

Generator's Job: Produce convincing high-resolution images from low-resolution inputs.

Perceptual Loss in Super-Resolution GANs

Perceptual loss ensures that generated images are not only realistic but also close to the true high-resolution images in terms of visual details.

$$L_{\text{perceptual}}(G) = \mathbb{E}_{x_{\text{HR}}, x_{\text{LR}}} \left[\|\phi(G(x_{\text{LR}})) - \phi(x_{\text{HR}})\|^2 \right]$$

- ϕ is a feature extraction function (e.g., from a VGG network) capturing high-level details.
- Instead of pixel differences, perceptual loss measures the difference in **features** between the generated and real images.

Goal: Ensure that the generated high-resolution images look perceptually similar to real high-resolution images.

Full SRGAN Loss Function

The full loss function for Super-Resolution GANs (SRGAN) combines the adversarial loss and the perceptual loss:

$$L_{\text{SRGAN}}(G, D) = L(G, D) + \lambda_{\text{perceptual}} L_{\text{perceptual}}(G)$$

- $L(G, D)$: GAN loss, ensuring the generated image is realistic.
- $L_{\text{perceptual}}(G)$: Perceptual loss, ensuring perceptual similarity to the high-resolution image.
- $\lambda_{\text{perceptual}}$: Weight parameter controlling the balance between GAN and perceptual loss.