Introduction

The Wasserstein Generative Adversarial Network (WGAN) introduces a new approach to training GANs, aiming to improve stability and address vanishing gradients by minimizing the Wasserstein distance between the data and generated distributions.

Key Equations

The Wasserstein distance W can be expressed as:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|]$$

where P_r is the real data distribution, P_g is the generated data distribution, and $\Pi(P_r, P_g)$ is the set of all joint distributions with marginals P_r and P_g.

Algorithm 1: WGAN Training Procedure

Detailed Explanation

Critic Update (Lines 3–7)

- **Sample a batch of real data points:**

$$\{x^{(i)}\}_{i=1}^{m} \sim P_r.$$

- **Sample a batch of generated data points:**

$$z^{(i)} \sim P_z, \quad \{g_\theta(z^{(i)})\}_{i=1}^{m}.$$

- **Compute the gradient of the critic loss:**

$$g_w = \nabla_w \left[\frac{1}{m} \sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \right].$$

- **Update the critic's parameters:**

$$w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w).$$

- **Clip the critic's weights:**

$$w \leftarrow \text{clip}(w, -c, c).$$

- **Repeat the critic update $n_{\textbf{critic}}$ times:** This ensures the critic sufficiently approximates the Wasserstein distance before updating the generator.

Algorithm 1 WGAN Algorithm

Require: α: learning rate, c: clipping parameter, m: batch size, n_{critic}: number of critic updates per generator update.

Require: w_0: initial critic parameters, θ_0: initial generator parameters.

while θ has not converged **do**

  for $t = 0, \ldots, n_{\text{critic}} - 1$ **do**

    Sample $\{x^{(i)}\}_{i=1}^{m} \sim P_r$: a batch of real data points.

    Sample $\{z^{(i)}\}_{i=1}^{m} \sim P_z$: a batch of prior samples.

    Compute the gradient of the critic loss:

$$g_w = \nabla_w \left[\frac{1}{m} \sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \right].$$

    Update critic parameters:

$$w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w).$$

    Clip critic weights:

$$w \leftarrow \text{clip}(w, -c, c).$$

  end for

  Sample $\{z^{(i)}\}_{i=1}^{m} \sim P_z$: a batch of prior samples.

  Compute the gradient of the generator loss:

$$g_\theta = -\nabla_\theta \left[\frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \right].$$

  Update generator parameters:

$$\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta).$$

end while

Generator Update (Lines 8–11)

- **Sample a batch of prior samples:**

$$\{z^{(i)}\}_{i=1}^m \sim P_z, \quad \{g_\theta(z^{(i)})\}_{i=1}^m.$$

- **Compute the gradient of the generator loss:**

$$g_\theta = -\nabla_\theta \left[\frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \right].$$

- **Update the generator's parameters:**

$$\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta).$$

Key Concepts and Math

Wasserstein Distance

The WGAN minimizes the Wasserstein-1 distance between the real distribution P_r and the generated distribution P_g:

$$W(P_r, P_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)],$$

where $\|f\|_L \leq 1$ enforces that f (the critic) is 1-Lipschitz. The clipping of w approximates this constraint.

Critic Loss

The critic aims to maximize the difference between real and generated scores:

$$L_w = \mathbb{E}_{x\sim P_r}[f_w(x)] - \mathbb{E}_{z\sim P_z}[f_w(g_\theta(z))].$$

Generator Loss

The generator minimizes the critic's score for generated samples:

$$L_\theta = -\mathbb{E}_{z\sim P_z}[f_w(g_\theta(z))].$$

Optimization

The algorithm uses RMSProp for optimization. This adjusts the step size for each parameter based on the magnitude of recent gradients:

$$\text{RMSProp}(x, g) = \frac{g}{\sqrt{\mathbb{E}[g^2] + \epsilon}},$$

where ϵ is a small constant to prevent division by zero.

Clipping

The critic's weights are clipped to lie within $[-c, c]$ element-wise to ensure Lipschitz continuity:

$$\text{clip}(w, -c, c) = \max(-c, \min(c, w)).$$

Termination Condition

The algorithm continues until the generator parameters θ converge, i.e., when the generator produces high-quality samples indistinguishable from the real distribution.