Wasserstein GAN

ECE57000: Artificial Intelligence
David I. Inouye
Motivation: **Vanishing gradients** for generator caused by a discriminator that is “too good”

From: https://developers.google.com/machine-learning/gan/problems

- Vanishing gradient means $\nabla_G V(D, G) \approx 0$.
  - Gradient updates do not improve $G$
  
- Theoretically, this is an issue of JSD

![Graphs showing distribution changes](image)

- Practically, careful balance during training required:
  - Optimizing $D$ too much leads to vanishing gradient
  - **But** training too little means it is not close to JSD

Wasserstein GAN: Better gradient values and better convergence (better stability)

- Better gradients even after significant training
- Convergent training even without batch normalization

Figure 6: Algorithms trained with a generator without batch normalization and constant number of filters at every layer (as opposed to duplicating them every time as in [18]). Aside from taking out batch normalization, the number of parameters is therefore reduced by a bit more than an order of magnitude. Left: WGAN algorithm. Right: standard GAN formulation. As we can see the standard GAN failed to learn while the WGAN still was able to produce samples.

Outline of Wasserstein GANs

▸ Preliminaries
  ▸ Optimal transport (OT) and Monge problem
  ▸ Wasserstein distribution distance based on OT
  ▸ Lipschitz continuous functions

▸ WGAN adversarial objective
  ▸ Wasserstein distance as maximization problem
  ▸ Comparison to standard GAN objective

▸ WGAN algorithms
  ▸ Clipping algorithm (original WGAN)
  ▸ Gradient penalty algorithm
What is Optimal Transport?

- The natural geometry for probability distributions.
- How close are two distributions?
- Which distribution is between two distributions?
- What is the shortest path between two distributions?

OT Monge map problem: Find the minimum (optimal) transportation plan between two distributions

- We want a map (i.e., a function) that moves the mass from the mountain to fill the hole (exactly)
**OT Monge map problem**: Find the minimum (optimal) transportation plan between two distributions

- One plan for showing movement of parts of the mass
**OT Monge map problem**: Find the minimum (optimal) transportation plan between two distributions

- Better transport plan if using squared Euclidean cost
**OT Monge map problem:** Find the optimal transportation plan between two distributions

- We denote the source distribution by $X \sim p_X$ and the target distribution by $Y \sim p_Y$.
  - $p_X$ is like the mound of dirt
  - $p_Y$ is like the hole in the ground
- The OT Monge problem can be formulated as the optimization problem
  $$\min_T \mathbb{E}_{p_X} [c(x, T(x))]$$
  subject to $p_{T(X)} = p_Y$
- Where the constraint makes sure that the transformed source distribution is aligned with target distribution (i.e., the moved dirt fills the hole exactly).
The Wasserstein-1 distribution distance can be derived from the solution to this OT problem

- The Wasserstein-1 distance is defined as:

$$W_1(p_X, p_Y) = \left( \min_T \mathbb{E}_{p_X} [\|x - T(x)\|_2] \right)$$

  s. t. \( p_{T(X)} = p_Y \)

- This is merely the optimal value of the Monge problem with \( c(x, T(x)) = \|x - T(x)\|_2 \)

- Other similar Wasserstein distances can be defined such as Wasserstein-2
Comparison of Wasserstein distance to JSD for disjoint uniform distributions in 1D

▸ Suppose both distributions are on a line segment in 2D
  ▸ JSD gives no information
  ▸ Wasserstein (also known as Earth Mover distance) gives nice information
  ▸ Wasserstein distance gives how far you need to move the line

\[ \theta = \text{distance between line distributions} \]

Figure from Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

Figure 1: These plots show \( \rho(\mathbb{P}_\theta, \mathbb{P}_0) \) as a function of \( \theta \) when \( \rho \) is the EM distance (left plot) or the JS divergence (right plot). The EM plot is continuous and provides a usable gradient everywhere. The JS plot is not continuous and does not provide a usable gradient.
Preliminaries for WGAN: What is a Lipschitz smooth function?

- Informally, a Lipschitz continuous function means that the function does not change too quickly.
- Intuitively, a double cone whose origin can be moved along the function so that the whole function always stays outside the double cone.

https://en.wikipedia.org/wiki/Lipschitz_continuity
Preliminaries for WGAN: What is a Lipschitz smooth function?

- Formally, the slope of the line connecting any two points on the function is bounded
  \[ \frac{\|f(x_2) - f(x_1)\|_2}{\|x_2 - x_1\|_2} \leq K \]
- If the function is continuously differentiable, then
  \[ \|\nabla_x f(x)\|_2 \leq K, \quad \forall x \]

Examples
- \( f(x) = ax \), with \( K = a \)
- \( f(x) = |x| \), with \( K = 1 \)
- \( f(x) = \sin(x) \), with \( K = 1 \)

Counterexamples
- \( f(x) = x^2 \)
- \( f(x) = \exp(x) \)
- \( f(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \)
Wasserstein GAN first reformulates the minimization over $T$ to maximization over $f$

- The original Wasserstein problem:
  \[
  W_1(p_X, p_Y) = \left( \min_T \mathbb{E}_{p_X} [\|x - T(x)\|_1] \right)
  \text{s.t. } p_{T(x)} = p_Y
  \]

- The equivalent dual problem (Kantorovich-Rubinstein duality):
  \[
  W_1(p_X, p_Y) = \left( \max_f \mathbb{E}_{p_X} [f(x)] - \mathbb{E}_{p_Y} [f(y)] \right)
  \text{s.t. } \|f\|_L \leq 1
  \]
  Where $\|f\|_L \leq 1$ means the Lipschitz constant of $f$ is less than 1

- Very informally, this switches the objective with the constraints and vice versa


Wasserstein GAN first reformulates the minimization over $T$ to maximization over $f$

- **Wasserstein-1 dual problem:**

$$W_1(p_X, p_Y) = \left( \max_f \mathbb{E}_{p_X}[f(x)] - \mathbb{E}_{p_Y}[f(y)] \right)$$

  s.t. $\|f\|_L \leq 1$

- **Compare with JSD maximization problem:**

$$JSD(p_X, p_Y) = \left( \max_D \mathbb{E}_{p_X}[\log D(x)] + \mathbb{E}_{p_Y}[\log(1 - D(y))] \right)$$

  s.t. $D: \mathbb{R}^d \to [0,1]$
Putting it all together for the final adversarial min-max problem of Wasserstein GAN

- **Wasserstein GAN objective**
  
  \[
  \min_G \max_f \mathbb{E}_{p_{data}}[f(x)] - \mathbb{E}_{p_z}[f(G(z))] \\
  \text{s.t. } \|f\|_L \leq 1
  \]

- **Original (JSD) GAN objective**
  
  \[
  \min_G \max_D \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log (1 - D(G(z)))] \\
  \text{s.t. } D: \mathbb{R}^d \to [0,1]
  \]
Comparison of Wasserstein distance to JSD in 1D

- This Lipschitz constraint rather than the classifier constraint produces better gradients

- No balancing of training objective required (in theory)
Key question: How do we enforce the Lipschitz constraint?

- Clip the parameter weights
- Why would this partially work?
  - If all weights are bounded, then the Lipschitz constant is bounded.
  - If the Lipschitz constant is bounded, it is equivalent to scaled Lipschitz constraint

---

**Algorithm 1** WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

**Require:** $\alpha$, the learning rate. $c$, the clipping parameter. $m$, the batch size. $n_{\text{critic}}$, the number of iterations of the critic per generator iteration.

**Require:** $w_0$, initial critic parameters. $\theta_0$, initial generator’s parameters.

1: while $\theta$ has not converged do
2:    for $t = 0, ..., n_{\text{critic}}$ do
3:        Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
4:        Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
5:        $g_w \leftarrow \nabla_w \left[ \frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) \right]$  
6:        $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$  
7:        $w \leftarrow \text{clip}(w, -c, c)$
8:    end for
9:    Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
10:   $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
11:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
12: end while

---

Better idea: Empirically encourage this constraint by adding a penalty term

- Original problem:
  \[
  \min_G \max_f \mathbb{E}_{p_{data}} [f(x)] - \mathbb{E}_{p_z} [f(G(z))] \\
  \text{for } f: \|f\|_{L^1} \leq 1
  \]

- Relaxed Lipschitz constraint via gradient penalty
  \[
  \min_G \max_f \mathbb{E}_{p_{data}} [f(x)] - \mathbb{E}_{p_z} [f(G(z))] \\
  - \lambda \mathbb{E}_{p_{\tilde{x}}} [(\|\nabla_x f(\tilde{x})\|_2 - 1)^2]
  \]

- For \( p_{\tilde{x}} \), they use interpolated samples between real and fake samples:
  \[
  \tilde{x} = \epsilon x + (1 - \epsilon) G(z)
  \]
  \[
  \text{Where } x \sim p_X, z \sim p_z, \epsilon \sim \text{Uniform}([0,1])
  \]

How do we implement the gradient penalty?

- Key problem: We don’t know gradients in closed-form so how do we compute the objective?
- First note that backprop itself is a computation!
- Solution: Use autograd to compute gradient and then backprop through that (gradient of gradient)

---

**Algorithm 1** WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

**Require:** The gradient penalty coefficient $\lambda$, the number of critic iterations per generator iteration $n_{\text{critic}}$, the batch size $m$, Adam hyperparameters $\alpha, \beta_1, \beta_2$.

**Require:** initial critic parameters $w_0$, initial generator parameters $\theta_0$.

1: while $\theta$ has not converged do
2:     for $t = 1, \ldots, n_{\text{critic}}$ do
3:         for $i = 1, \ldots, m$ do
4:             Sample real data $x \sim \mathbb{P}_r$, latent variable $z \sim p(z)$, a random number $\epsilon \sim U[0, 1]$.
5:             $\tilde{x} \leftarrow G_\theta(z)$
6:             $\hat{x} \leftarrow \epsilon x + (1 - \epsilon)\tilde{x}$
7:             $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda(\|\nabla_{\tilde{x}} D_w(\tilde{x})\|_2^2 - 1)^2$
8:         end for
9:     $w \leftarrow \text{Adam}(\nabla w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$
10:    end for
11:    Sample a batch of latent variables $\{z^{(i)}\}_{i=1}^m \sim p(z)$.
12:    $\theta \leftarrow \text{Adam}(\nabla \theta \frac{1}{m} \sum_{i=1}^m -D_w(G_\theta(z)), \theta, \alpha, \beta_1, \beta_2)$
13: end while

---

Outline of Wasserstein GANs

- Preliminaries
  - Optimal transport (OT) and Monge problem
  - Wasserstein distribution distance based on OT
  - Lipschitz continuous functions

- WGAN adversarial objective
  - Wasserstein distance as maximization problem
  - Comparison to standard GAN objective

- WGAN algorithms
  - Clipping algorithm (original WGAN)
  - Gradient penalty algorithm
Additional resources for optimal transport and Wasserstein distance

▸ A primer on Optimal Transport slides:
  ▸ Alternative link: [https://www.dropbox.com/s/55tb2cf3zipl6xu/aprimeronOT.pdf?dl=0](https://www.dropbox.com/s/55tb2cf3zipl6xu/aprimeronOT.pdf?dl=0)

▸ Optimal transport tutorial videos
  ▸ Video Part 1 - [https://www.youtube.com/watch?v=6iR1E6t1MMQ](https://www.youtube.com/watch?v=6iR1E6t1MMQ)
  ▸ Video Part 2 - [https://www.youtube.com/watch?v=R49Xb9eAUBA](https://www.youtube.com/watch?v=R49Xb9eAUBA)
  ▸ Video Part 3 - [https://www.youtube.com/watch?v=SZHumKEhgtA](https://www.youtube.com/watch?v=SZHumKEhgtA)

▸ Additional resources
  ▸ [https://optimaltransport.github.io/](https://optimaltransport.github.io/)