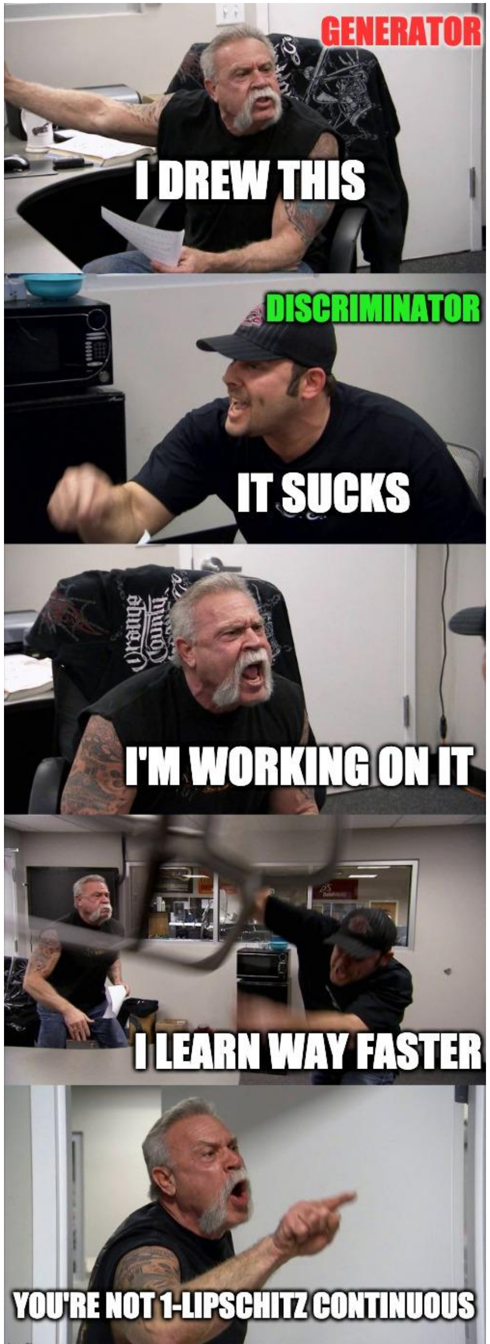


https://cs236g.stanford.edu/memes/1/

Wasserstein GAN

ECE324, Winter 2023

Michael Guerzhoy

Key points

- The Wasserstein GAN training algorithm is similar to GAN, but it constrains the discriminator
  - Making the discriminator more constrained helps the generator train
- To invent WGAN, Martin Arjovski reformulated the cost function of the GAN as minimizing the distance between the generator distribution and the data distribution
  - He then used a different notion of distance to derive WGAN

Pre-requisites

- Understand what “data distribution” and “generator distribution” are
  - The probability distributions over the samples implied by the training data and the data generated by the generator
- Understand the Min-Max formulation of the GAN cost function
- Understand the operation of marginalizing distributions
- Gradient descent + moment in neural networks

Connections to what you have seen

- The KL divergence is very similar to the cross-entropy cost function
  - The cross-entropy cost function measures the difference between the target labels and the outputs of the network

Heavy math ahead, optional math in green

The objective function of the original GAN

- For a fixed generator *G*, the optimal discriminator is

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

- *Proof:* we are maximizing

$$\begin{aligned} V(G, D) &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x}))dx + \int_{z} p_{\boldsymbol{z}}(\boldsymbol{z}) \log(1 - D(g(\boldsymbol{z})))dz \\ &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x})) + p_g(\boldsymbol{x}) \log(1 - D(\boldsymbol{x}))dx \end{aligned}$$

For every x, the integrand is maximized at $\frac{p_{data}(x)}{p_{data}(x)+p_{g(x)}}$

(alog(y)+blog(1-y) is maximized at a/(a+b) using calculus)

Reformulating the cost function

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}[\log(1 - D_G^*(G(\boldsymbol{z})))] \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x} \sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right]
\end{aligned}
$$

Divide each of the terms by two in order to get an expression in terms of KL divergences

$$
C(G) = -\log(4) + KL\left(p_{\text{data}} \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right) + KL\left(p_g \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right)
$$

$$
= -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \| p_g\right)
$$

(Jensen-Shannon divergence)

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

- Another measure of how similar P and Q are
- Symmetric, unlike the KL divergence

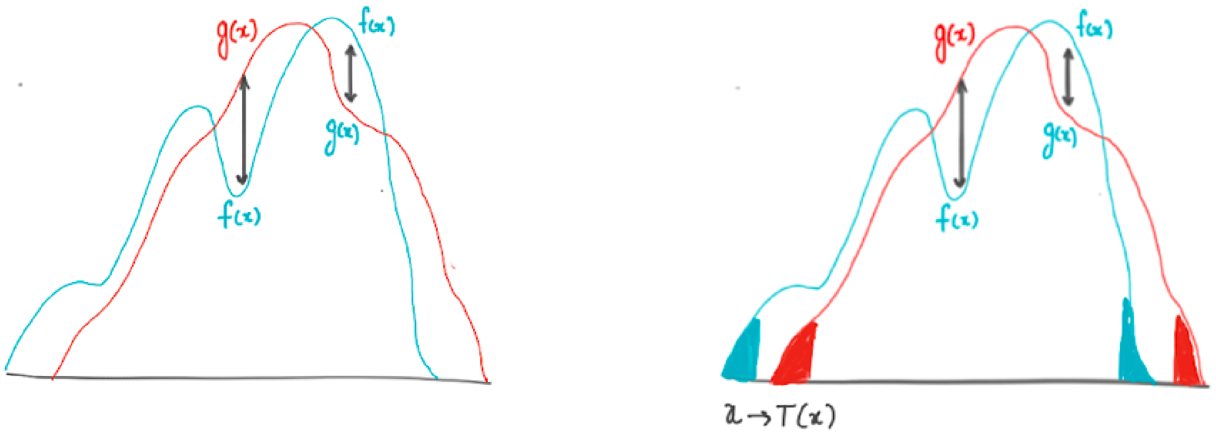
Wasserstein Distance

- $W(P,Q) = \inf_{\gamma \in \Pi(P,Q)} E_{(x,y)\sim\gamma}[|x-y|]$
- $\Pi(P,Q)$ is the set of distributions over $R^{\dim(P)+\dim(Q)}$ whose marginal are P and Q. For $\gamma \in \Pi(P,Q)$, and densities p and Q for P, Q

$$\int \gamma(x,y)\,dx = q(y), \int \gamma(x,y)dy = p(x)$$

Wasserstein Distance intuition

- $W(P,Q) = \inf_{\gamma \in \Pi(P,Q)} E_{(x,y)\sim\gamma}[\|x-y\|] = \inf_{\gamma \in \Pi(P,Q)} \int |x-y|\gamma(x,y)dxdy$



https://kowshikchilamkurthy.medium.com/wasserstein-distance-contraction-mapping-and-modern-rl-theory-93ef740ae867

Wasserstein Distance intuition

$$\inf_{\gamma \in \Pi(P,Q)} \int |x-y|\gamma(x,y)dxdy$$

- Want to move $\gamma(x,y)$ from x to y
- In total, move $\int \gamma(x,y)dy = p(x)$ from x

  move $\int \gamma(x,y)dx = q(y)$ to y

so we don't run out of mass

Wasserstein Distance dual formulation

- Theorem (Kantorovich-Rubinestein):
$$W(P,Q) = \sup_{||f||_L \leq 1} |E_{x\sim P}[f(x)] - E_{y\sim Q}[f(y)]|$$
- A function is K-Lipschitz if for all x, y
$$|f(x) - f(y)| \leq K|x - y|$$

we write this as $||f||_L \leq K$

- Proof: https://drive.google.com/file/d/0B6JeBUquZ5BwVlV1dEpsTHVVbTA/view?usp=sharing&resourcekey=0-5xbvKhDXZjrYRLfqppsHuQ p. 121

Motivating example

- Let $Z \sim Unif([0, 1])$
- Let P_θ be the distribution $(\theta,\ Z) \in R^2$
- $W(P_0, P_\theta) = |\theta|$
  - Using the original definition, need to move the probability mass by $|\theta|$
- $JS(P_0, P_\theta) = \frac{1}{2}(KL(P_0 || \frac{P_0+P_\theta}{2}) + KL(P_\theta || \frac{P_0+P_\theta}{2}))$

  $= \log 2$ if $\theta \neq 0$, and 0 otherwise

Derivation for $KL(P_0 || \frac{P_0+P_\theta}{2})$

$KL(P_0 || \frac{P_0+P_\theta}{2})$=$\int p_0(x,y) \log \frac{p_0(x,y)}{((p_0+p_\theta)/2)(x,y)} dxdy$

$p_0(x,y)$ is only non-zero x=0, $p_\theta(x,y)$ is only non zero at $x = \theta$ so the integral equals

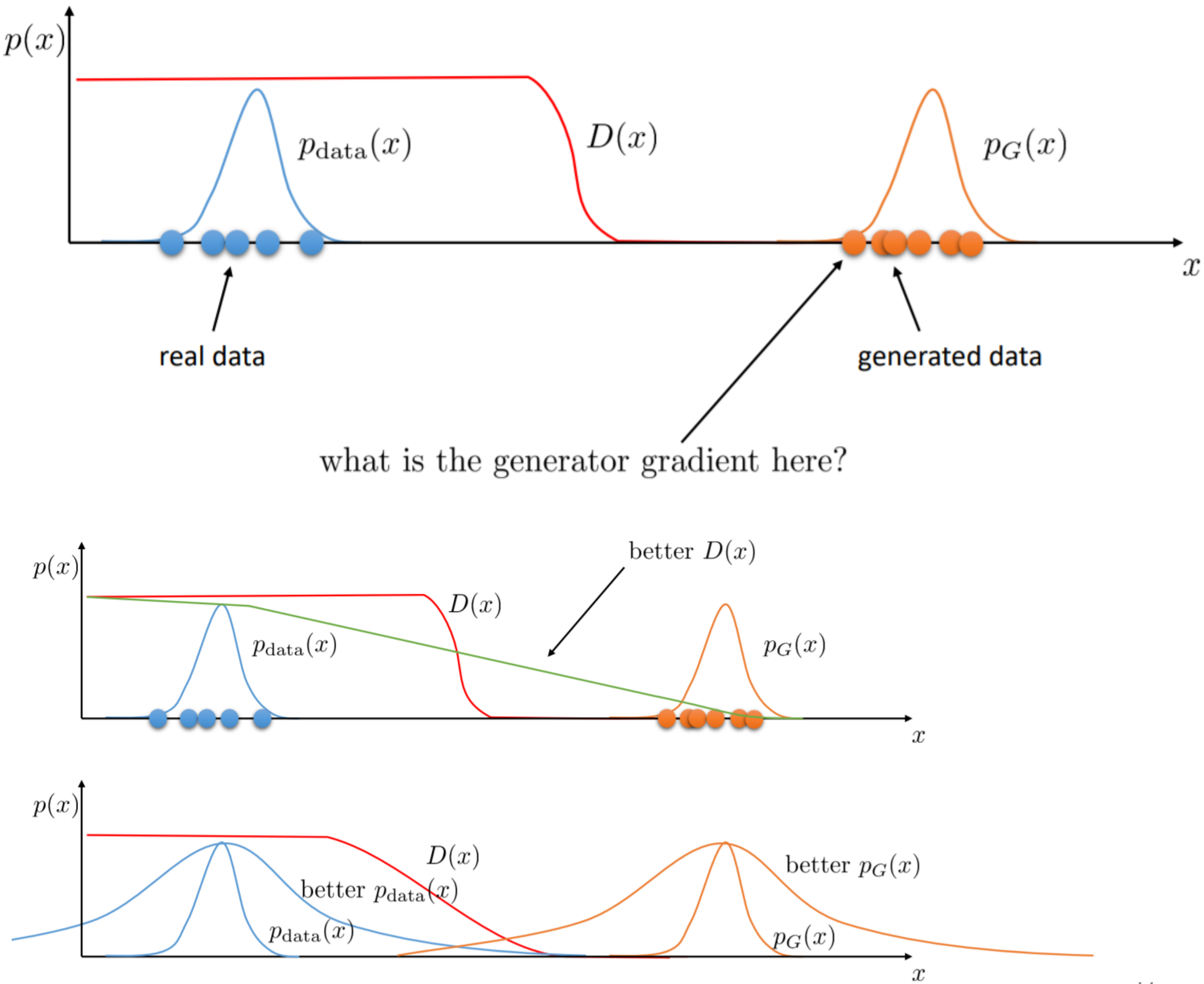
$$\int p_0(x,y) \log \frac{p_0(x,y)}{(p_0/2)(x,y)} dxdy$$

$$= \int p_0(x,y) \log 2 \, dxdy = \log 2$$

Motivating example

- The JS distance does not provide gradient signal in the motivating example
- The Wasserstein distance does
- In the motivating example, the support for the two distribution is disjoint
  - That may be unusual
  - But it is not unusual for the distribution to be supported by different low-dimensional manifolds that intersect but otherwise don't overlap

Gradient visualization



https://cs182sp21.github.io/static/slides/lec-19.pdf

The Wasserstein GAN

- $W(P,Q) = \sup_{||f||_L \leq 1} |E_{x\sim P}[f(x)] - E_{y\sim Q}[f(y)]|$
- Make P the data distribution, and Q the generator distribution
- Make f_w be K-Lipschitz
  - Can do that by clipping all the weights to be in e.g [-0.01, 01]
  - Sketch of argument: the set of all functions is a closed set that way, so a function with a maximum K is somewhere in that closed set
  - Another argument: the first layer transforms the input by $W^{(1)}$, the second by $W^{(2)}, etc$. This is *at most* a small linear transformation for clipped weights, so the function is K-Lipschitz
  - If we find the sup for a K-Lipschitz function, a sup is actually attained for $f = f/|K|$

The Wasserstein GAN

- $\max_{w \in W} E_{x \sim p_{data}} f_w(x) - E_{z \sim p(z)} f_w\left(g_\theta(z)\right)$
- Alternately optimize the objective and the critic f_w

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: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

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

Intuition: Wasserstein GAN

- Theory says that we get a more informative gradient w.r.t θ
- The critic is not allowed to overfit because of clipping

Reminder: RMSProp

- rmsprop: Keep a moving average of the squared gradient for each weight

$$MeanSquare(w,t) = 0.9\ MeanSquare(w,\ t-1) + 0.1\left(\partial E / \partial w\,(t)\right)^2$$

- Dividing the gradient by $\sqrt{MeanSquare(w,\ t)}$ makes the learning work much better (Tijmen Tieleman, unpublished).

Better ways of ensuring f_θ is K-Lipschitz

- Penalize the gradient of f_θ directly: optimize

$$E_{x\sim p_{\text{data}}}[f_\theta(x) - \underbrace{\lambda(||\nabla_x f_\theta(x)||_2 - 1)^2}_{\text{make norm of gradient close to 1}}] - E_{z\sim p(z)}[f_\theta(G(z))]$$

- Normalize the weights matrices by the matrix's largest singular value

$$\sigma(W) = \max_{h:h\neq 0} \frac{||Wh||}{||h||} = \max_{||h||\leq 1} ||Wh||$$

largest singular value of W

$$W_\ell \leftarrow \frac{W_\ell}{\sigma(W_\ell)}$$

https://cs182sp21.github.io/static/slides/lec-19.pdf