Wasserstein GAN

2022. 04. 20

Joonhun Lee

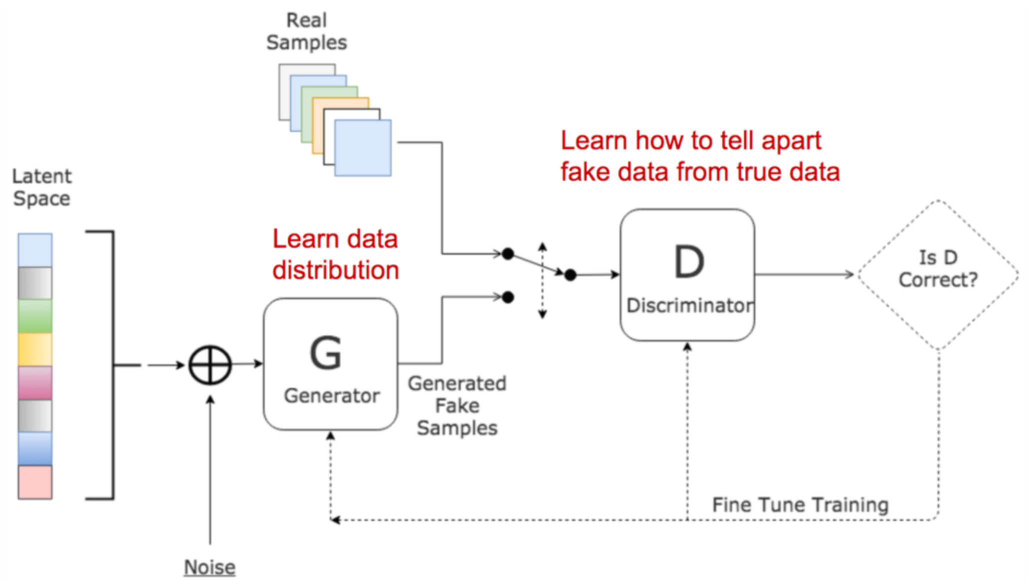
Agenda

Agenda

Generative Adversarial Networks

GANs are generative networks leveraging an adversarial system

- Discriminator D estimates the probability of a given sample coming from the real dataset
- Generator G synthesize samples given a noise



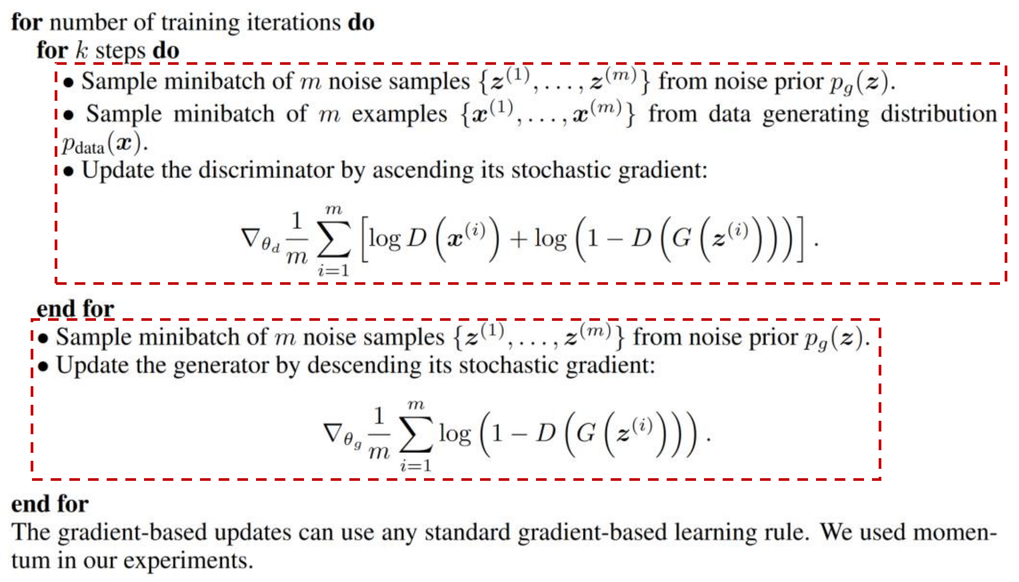
Weng, 2019

Generative Adversarial Networks

Vanilla GAN uses minimax algorithm to formulate loss function

- Minimax algorithm minimizes the maximum value of the cost function, i.e., $\min_G \max_D V(G, D)$
- The cost function of vanilla GAN $V(G, D) = \mathbb{E}_{x \sim \mathbb{P}_r(x)}[\log D(x)] + \mathbb{E}_{z \sim \mathbb{P}_z(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right]$

Sum of average values of the discriminator scores on real data and generated data



```
for number of training iterations do
  for $k$ steps do
    • Sample minibatch of $m$ noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
    • Sample minibatch of $m$ examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
    • Update the discriminator by ascending its stochastic gradient:
        $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right]$.
  end for
  • Sample minibatch of $m$ noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
  • Update the generator by descending its stochastic gradient:
      $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)$.
end for
```

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

SEOUL NATIONAL UNIVERSITY
NUMERICAL COMPUTING & IMAGE ANALYSIS LAB

Goodfellow et al., 2014

Problems of Vanilla GAN

Since the training of vanilla GAN is slow and unstable for five reasons,⋯

1.	Hard to achieve Nash equilibrium	Each model updates independently
2.	Vanishing gradient	G cannot learn if D is either perfect or completely bad → Feature matching, Historical averaging, ⋯
3.	Mode collapse	G always produces same outputs
4.	Low dimensional supports	The probability of $\mathbb{P}_r$ and $\mathbb{P}_\theta$ being disjoint is high due to their low dimensions
5.	Lack of proper evaluation metric	There is no good objective function for GAN

⋯ several improvements in GAN were emerged

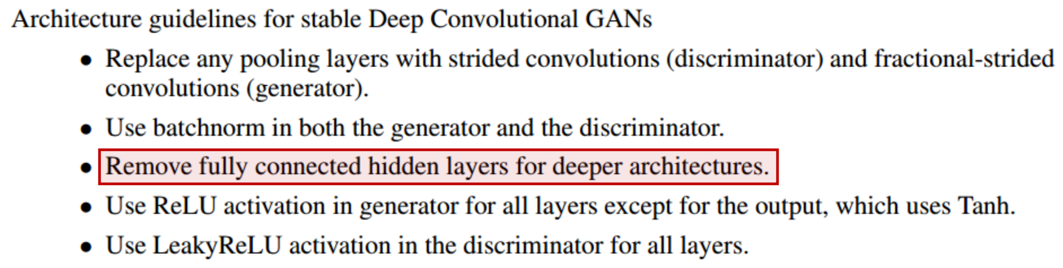
•	Least-squares GAN (LSGAN)	D outputs real-valued number
•	Wasserstein GAN (WGAN)	D is constrained to be Lipschitz-continuous
•	Gradient penalty	D is constrained to be continuous harder than WGAN
•	Spectral norm	D is constrained to be (mathematically) continuous
•	Instance noise	Noise is added to the data and generated samples
•	⋯	

Weng, 2019

Deep Convolutional GAN

DCGAN replaces fully-connected layers with convolutional layers

- DCGAN tries to solve problems on 1) resolution of generated samples, 2) interpretability by only using CNN, and 3) evaluation measure for samples by clarifying criteria



Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

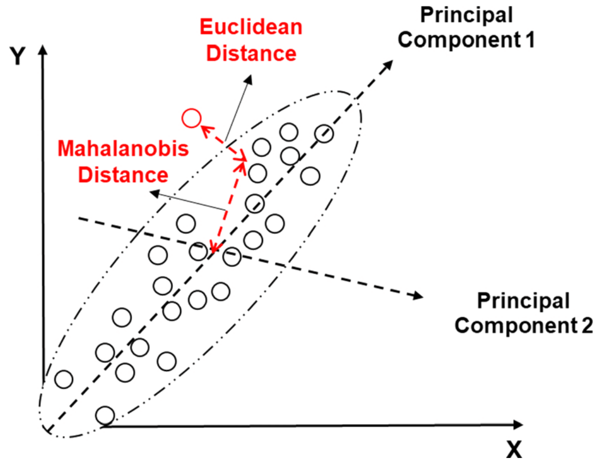
Radford et al., 2016

Agenda

Metric

Metric is a function $d: X \times X \to [0, \infty)$ such that

1) Non-negativity $d(x, y) = 0 \Leftrightarrow x = y \quad \forall x, y \in X$
2) Symmetry $d(x, y) = d(y, x) \quad \forall x, y \in X$
3) Triangle inequality $d(x, y) \leq d(x, z) + d(y, z) \quad \forall x, y, z \in X$



Lee et al., 2020

Metric

Metric for probability distributions can be defined

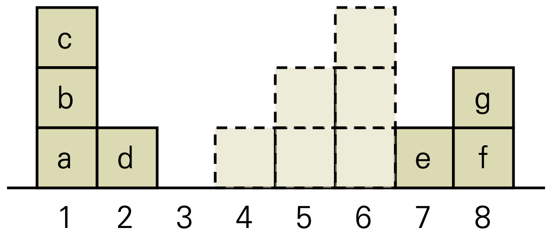
- Total variation distance $\delta(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{A \in \Sigma} |\mathbb{P}_r(A) - \mathbb{P}_\theta(A)|$
- Kullback–Leibler divergence $KL(\mathbb{P}_r || \mathbb{P}_\theta) = \int \log\left(P_r(x)/P_\theta(x)\right) P_r(x)\, d\mu(x)$
- Jensen–Shannon divergence $JS(\mathbb{P}_r, \mathbb{P}_\theta) = KL(\mathbb{P}_r || \mathbb{P}_m) + KL(\mathbb{P}_\theta || \mathbb{P}_m)$ where $\mathbb{P}_m = (\mathbb{P}_r + \mathbb{P}_\theta)/2$
- Wasserstein distance $W_p(\mathbb{P}_r, \mathbb{P}_\theta) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_\theta)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||^p]$ where $\Pi(\mathbb{P}_r, \mathbb{P}_\theta)$ is the set of joint distributions of $\mathbb{P}_r$ and $\mathbb{P}_\theta$

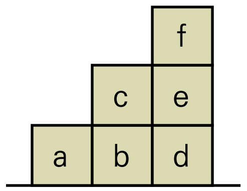
Arjovsky et al., 2017

Wasserstein Distance

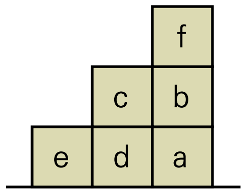
W_p can be interpreted as the minimum cost of moving mass from one to another distribution

- Therefore, W_1 is also called Earth Mover's distance (EM distance)

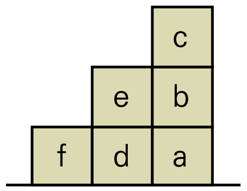




$$\|1-4\| + \|1-5\| + \|1-5\| + \|2-6\| + \|7-6\| + \|8-6\| = 18$$



$$\|1-6\| + \|1-6\| + \|1-5\| + \|2-5\| + \|7-4\| + \|8-6\| = 22$$



$$\|1-6\| + \|1-6\| + \|1-6\| + \|2-5\| + \|7-5\| + \|8-4\| = 24$$

Agenda

Relationship between Vanilla GAN and Jensen–Shannon Divergence

Mathematically, the loss function of GAN quantifies the similarity between $\mathbb{P}_r$ and $\mathbb{P}_g$ by Jensen–Shannon divergence

Since $V(G,D) = \int_x dx\big(P_r(x)\log D(x) + P_\theta(x)\log(1-D(x))\big)$,

$$D^* = \frac{P_r(x)}{P_r(x) + P_\theta(x)}$$

Therefore,

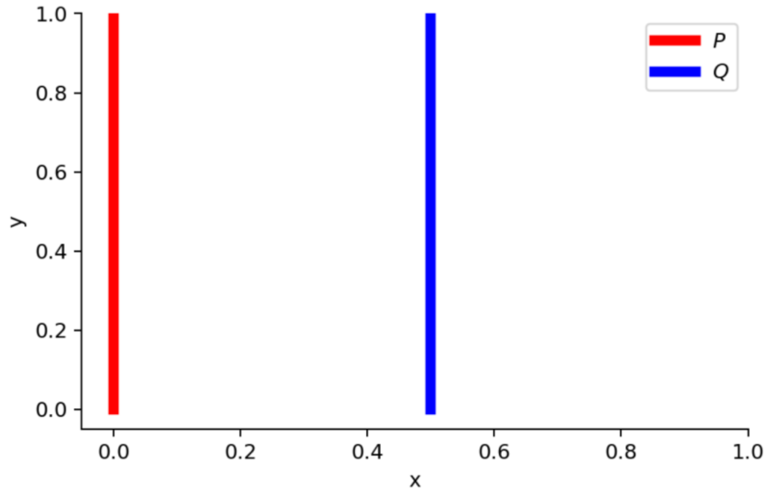
$$\begin{aligned} JS(P_r,P_\theta) &= \frac{1}{2}KL(P_r||\frac{P_r+P_\theta}{2}) + \frac{1}{2}KL(P_\theta||\frac{P_r+P_\theta}{2}) \\ &= \frac{1}{2}\left(\log 2 + \int_x dxP_r(x)\log\frac{P_r(x)}{P_r(x)+P_\theta(x)}\right) + \frac{1}{2}\left(\log 2 + \int_x dxP_\theta(x)\log\frac{P_\theta(x)}{P_r(x)+P_\theta(x)}\right) \\ &= \frac{1}{2}\big(2\log 2 + V(G,D^*)\big) \end{aligned}$$

Goodfellow et al., 2014

Wasserstein GAN

WGAN replaces JS with W_1 as the loss function

- W_1 is differentiable almost everywhere[1] and the minimization of W_1 is equivalent to convergence in distribution
- W_1 provide a smooth representation of the distance even when two distributions are disjoint



When $\theta \neq 0$,

$$\begin{cases} KL(P||Q) = +\infty \\ KL(Q||P) = +\infty \\ JS(P,Q) = \log 2 \\ W_1(P,Q) = |\theta| \end{cases}$$

And when $\theta = 0$,

$$\begin{cases} KL(P||Q) = KL(Q||P) = JS(P,Q) = 0 \\ W_1(P,Q) = |\theta| = 0 \end{cases}$$

1) Arjovsky et al., 2017
2) Weng, 2019

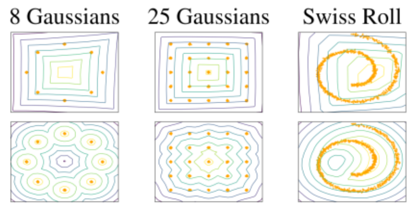
Wasserstein GAN

WGAN replaces JS with W_1 as the loss function

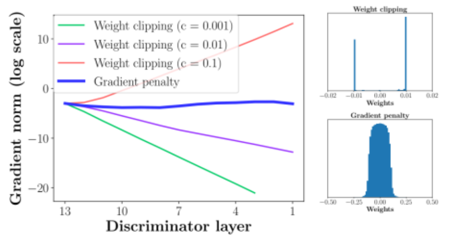
- W_1 is transformed based on the Kantorovich–Rubinstein duality since it is intractable to exhaust all the possible joint distributions[2]

$$W_1 = \frac{1}{K} \sup_{\|f\|_L \le K} \left(\mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_\theta}[f(x)] \right)$$

- D now learns w to find good f_w for configuring the loss function
- Additionally, weight clipping is used to maintain 1–Lipschitz continuity



(a) Value surfaces of WGAN critics trained to optimality on toy datasets using (top) weight clipping and (bottom) gradient penalty. Critics trained with weight clipping fail to capture higher moments of the data distribution. The ‘generator’ is held fixed at the real data plus Gaussian noise.



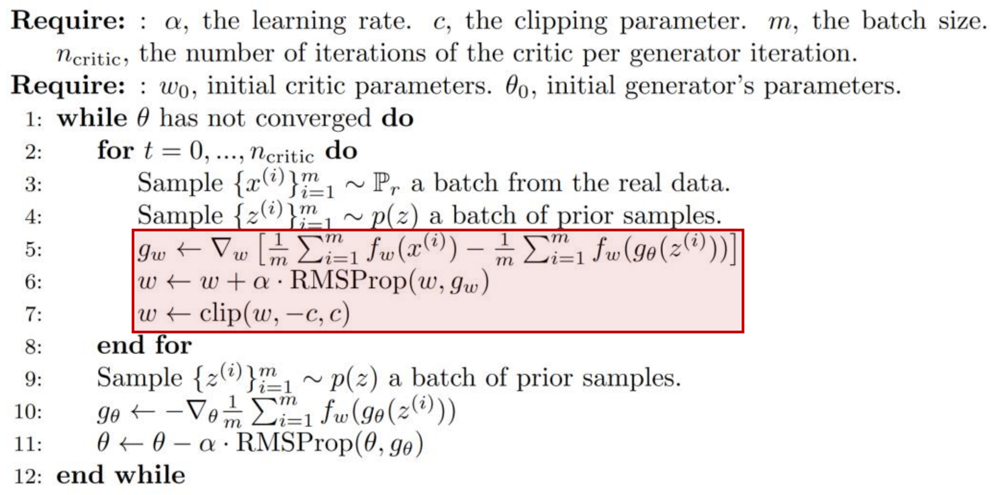
(b) (left) Gradient norms of deep WGAN critics during training on toy datasets either explode or vanish when using weight clipping, but not when using a gradient penalty. (right) Weight clipping (top) pushes weights towards two values (the extremes of the clipping range), unlike gradient penalty (bottom).

1) Arjovsky et al., 2017
2) Gulrajani et al., 2017

Wasserstein GAN

WGAN is differ from vanilla GAN in three ways

1. Use loss function derived from W_1
2. Use RMSProp as optimizer rather than a momentum−based optimizer
3. Clamp the weights to $[-c, c]$



```
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
```

SEOUL NATIONAL UNIVERSITY
NUMERICAL COMPUTING & IMAGE ANALYSIS LAB

1) Arjovsky et al., 2017
2) Weng, 2019

Agenda

Results

There are two practical benefits using WGAN

- Meaningfulness — Loss function correlates with quality of samples and convergence of G
- Stability — Optimization process becomes more stable

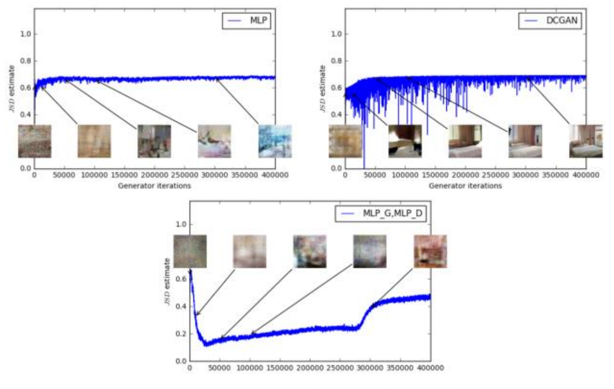


Figure 4: JS estimates for an MLP generator (upper left) and a DCGAN generator (upper right) trained with the standard GAN procedure. Both had a DCGAN discriminator. Both curves have increasing error. *Samples get better for the DCGAN but the JS estimate increases or stays constant, pointing towards no significant correlation between sample quality and loss. Bottom: MLP with both generator and discriminator. The curve goes up and down regardless of sample quality. All training curves were passed through the same median filter as in Figure 3.*

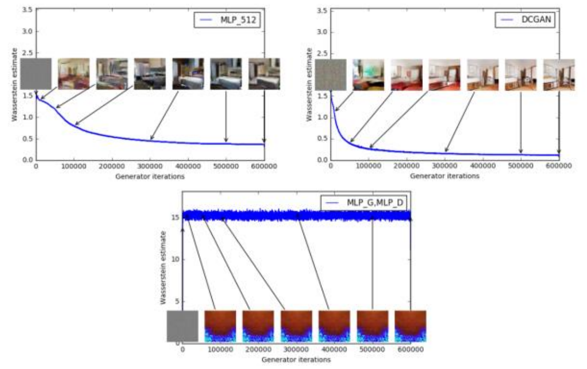


Figure 3: Training curves and samples at different stages of training. We can see a clear correlation between lower error and better sample quality. Upper left: the generator is an MLP with 4 hidden layers and 512 units at each layer. The loss decreases constistently as training progresses and sample quality increases. Upper right: the generator is a standard DCGAN. The loss decreases quickly and sample quality increases as well. In both upper plots the critic is a DCGAN without the sigmoid so losses can be subjected to comparison. Lower half: both the generator and the discriminator are MLPs with substantially high learning rates (so training failed). Loss is constant and samples are constant as well. The training curves were passed through a median filter for visualization purposes.

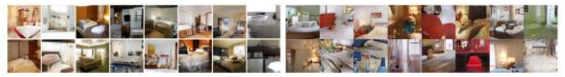


Figure 5: Algorithms trained with a DCGAN generator. Left: WGAN algorithm. Right: standard GAN formulation. Both algorithms produce high quality samples.

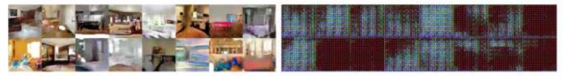


Figure 6: Algorithms trained with a generator without batch normalization and constant number of filters at every layer (as opposed to duplicating them every time as in [18]). Aside from taking out batch normalization, the number of parameters is therefore reduced by a bit more than an order of magnitude. 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.

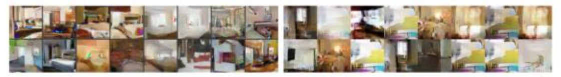


Figure 7: Algorithms trained with an MLP generator with 4 layers and 512 units with ReLU nonlinearities. The number of parameters is similar to that of a DCGAN, but it lacks a strong inductive bias for image generation. Left: WGAN algorithm. Right: standard GAN formulation. The WGAN method still was able to produce samples, lower quality than the DCGAN, and of higher quality than the MLP of the standard GAN. Note the significant degree of mode collapse in the GAN MLP.

Arjovsky et al., 2017

E.O.D

Q & A

Appendix

1. Justification of W_1 loss
2. Transformation of W_1

Theorem1 (Continuity and Differentiability of W)

$$W(\mathbb{P}_{\theta_1},\mathbb{P}_{\theta_2}) = \inf_{\gamma\in\Pi(\mathbb{P}_{\theta_1},\mathbb{P}_{\theta_2})} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|] \le \int_{\mathcal{X}\times\mathcal{X}} \|x-y\| d\gamma = \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|] = \mathbb{E}_{(x,y)\sim\gamma}\left[\left\|g_{\theta_1}(z) - g_{\theta_2}(z)\right\|\right]$$

If g is continuous in θ, then $g_{\theta_1}(z) \rightarrow_{\theta_1\rightarrow\theta_2} g_{\theta_2}(z)$ so that

$$\left\|g_{\theta_1} - g_{\theta_2}\right\| \rightarrow 0$$

Since $\mathcal{X}$ is compact, let M be constant, then

$$\therefore\ \left\|g_{\theta_1}(z) - g_{\theta_2}(z)\right\| \le M$$

By the bounded convergence theorem,

$$W(\mathbb{P}_{\theta_1},\mathbb{P}_{\theta_2}) \le \mathbb{E}_z\left[\left\|g_{\theta_1}(z) - g_{\theta_2}(z)\right\|\right] \rightarrow_{\theta_1\rightarrow\theta_2} 0$$

Since W is metric,

$$\left|W(\mathbb{P}_r,\mathbb{P}_{\theta_1}) - W(\mathbb{P}_r,\mathbb{P}_{\theta_2})\right| \le W(\mathbb{P}_{\theta_1},\mathbb{P}_{\theta_2}) \rightarrow_{\theta_1\rightarrow\theta_2} 0$$

Arjovsky et al., 2017

Theorem1 (Continuity and Differentiability of $\boldsymbol{W}$)

$W(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere and differentiable almost everywhere

Let g be locally Lipschitz and $L(\theta, z)$ be a constant for a given pair $(\theta, z) \in U$ where U is an open set, then

$$\left\|g_{\theta_1}(z_1) - g_{\theta_2}(z_2)\right\| \leq L(\theta_1, z_1)(\|\theta_1 - \theta_2\| + \|z_1 - z_2\|)$$

By taking expectation and let $z_1 = z_2 = z$, then

$$\mathbb{E}_z\left[\left\|g_{\theta_1}(z) - g_{\theta_2}(z)\right\|\right] \leq \|\theta_1 - \theta_2\| \mathbb{E}_z[L(\theta, z)]$$

Therefore, there is an open set $U_\theta = \{\theta' | (\theta', z') \in U\}$

Define $L(\theta) = \mathbb{E}_z[L(\theta, z)]$, then

$$\left|W\left(\mathbb{P}_r, \mathbb{P}_{\theta_1}\right) - W\left(\mathbb{P}_r, \mathbb{P}_{\theta_2}\right)\right| \leq W\left(\mathbb{P}_{\theta_1}, \mathbb{P}_{\theta_2}\right) \leq L(\theta)\|\theta_1 - \theta_2\|$$

which means $W(\mathbb{P}_r, \mathbb{P}_\theta)$ is locally Lipschitz

Arjovsky et al., 2017

Wasserstein Distance

1. Write W_1 in matrix form

Let $\Gamma = \gamma(x, y) \in \mathbb{R}^{l\times l}$ and $D = \|x - y\| \in \mathbb{R}^{l\times l}$, then

$$\begin{aligned} W_1(P_r, P_g) &= \inf_{\gamma\in\Pi} \mathbb{E}_{(x,y)\sim\gamma} \|x - y\| \\ &= \inf_{\gamma\in\Pi} \langle D, \Gamma\rangle_F \end{aligned}$$

where $\langle\cdot,\cdot\rangle_F$ is the Frobenius inner product

Basso, 2015

Linear Programming

2. Transform W_1 in standard form of LP

Let $b = [P_r \quad P_g]^T$, then by vectorizing Γ and D to x and c respectively,

$$W_1 = \min_{Ax=b, x\geq 0} z = c^T x$$

where

$$\begin{array}{cc}
 & \left[\begin{array}{cccc|cccc} P_r(x_1) & P_r(x_2) & \ldots & P_r(x_n) & P_\theta(y_1) & P_\theta(y_2) & \ldots & P_\theta(y_n) \end{array}\right] \Big\} \mathbf{b}^T \\
\mathbf{x} \left\{ \left[\begin{array}{c} \gamma(x_1, y_1) \\ \gamma(x_1, y_2) \\ \vdots \\ \hline \gamma(x_2, y_1) \\ \gamma(x_2, y_2) \\ \vdots \\ \hline \vdots \\ \hline \gamma(x_n, y_1) \\ \gamma(x_n, y_2) \\ \vdots \end{array}\right] \right. & \left. \left[\begin{array}{cccc|cccc}
1 & 0 & \ldots & 0 & 1 & 0 & \ldots & 0 \\
1 & 0 & \ldots & 0 & 0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ldots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \hline
0 & 1 & \ldots & 0 & 1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 & 0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ldots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline
0 & 0 & \ldots & 1 & 1 & 0 & \ldots & 0 \\
0 & 0 & \ldots & 1 & 0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ldots & \vdots & \vdots & \vdots & \ddots & \vdots
\end{array}\right] \right\} \mathbf{A}^T
\end{array}$$

Basso, 2015

Dual LP

3. Transform to dual form

Let $y = [f \quad g]^T$, then by duality theorems,

$$W_1 = \max_{A^T y \le c} \tilde{z} = b^T y$$

where

$$\left.\left[\begin{array}{ccc|ccc|c|ccc} \mathbf{D}_{1,1} & \mathbf{D}_{1,2} & \dots & \mathbf{D}_{2,1} & \mathbf{D}_{2,2} & \dots & \dots & \mathbf{D}_{n,1} & \mathbf{D}_{n,2} & \dots \end{array}\right]\right\} \mathbf{c}^T$$

$$\mathbf{y}\left\{\begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \\ \hline g(x_1) \\ g(x_2) \\ \vdots \\ g(x_n) \end{bmatrix}\right. \qquad \left.\left[\begin{array}{ccc|ccc|c|ccc} 1 & 1 & \dots & 0 & 0 & \dots & \dots & 0 & 0 & \dots \\ 0 & 0 & \dots & 1 & 1 & \dots & \dots & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & \dots & 1 & 1 & \dots \\ \hline 1 & 0 & \dots & 1 & 0 & \dots & \dots & 1 & 0 & \dots \\ 0 & 1 & \dots & 0 & 1 & \dots & \dots & 0 & 1 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \dots & \vdots & \vdots & \ddots \\ 0 & 0 & \dots & 0 & 0 & \dots & \dots & 0 & 0 & \dots \end{array}\right]\right\} \mathbf{A}$$

Basso, 2015

Dual Implementation

4. Solve dual LP

Since the constraints are $f(x_i) + g(x_j) \le D_{i,j}$, if $i = j$, then

$$g(x_i) \le -f(x_i) \quad (\because D_{i,i} = 0)$$

And to maximize the objective function,

$$g(x_i) = -f(x_i)$$

Therefore,

$$f(x_i) - f(x_j) \le D_{i,j} \text{ and } g(x_i) - g(x_j) \le D_{i,j} \text{ (1-Lipschitz continuous function)}$$