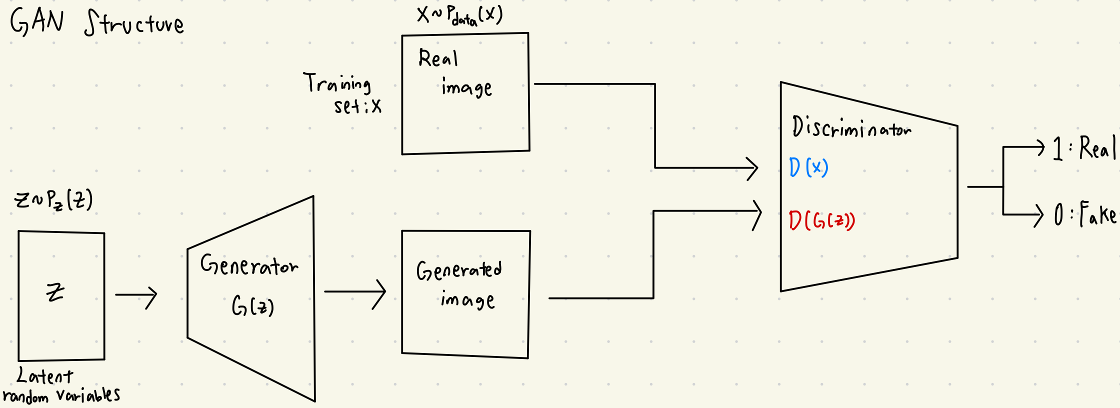
GAN Structure



GAN : Two player game

- Generator : try to fool the discriminator by generating real-looking images
- Discriminator : try to distinguish between real and fake images
- GAN eventually minimizes the distance between the real data distribution and the model distribution.

Objective function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim P_z(z)}[\log(1-D(G(z)))]$$

Objective function of GANs is actually equal to $\min_{G,D} \underbrace{JSD}_{\text{Jenson-Shannon Divergence}}(P_{data} \| P_{gen})$

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

where $M = \frac{1}{2}(P+Q)$

Proof:

1) Fix G, and obtain optimal D^*

$$D^*(x) = \underset{D}{\operatorname{argmax}}\, V(D) = \mathbb{E}_{x\sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim P_z(z)}[\log(1-D(G(z)))]$$

$*\, z \to G(z) \to x \to D(x)$

$$= \mathbb{E}_{x\sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{x\sim P_g(x)}[\log(1-D(x))]$$

$$= \int P_{data}(x)\log D(x)dx + \int P_g(x)\log(1-D(x))dx$$

$$= \int P_{data}(x)\log D(x) + P_g(x)\log(1-D(x))dx$$

$$D^*(x) = \underset{D}{\mathrm{argmax}}\, V(D) = \underset{D}{\mathrm{argmax}}\, P_{data}(x)\log(D(x)) + P_g(x)\log(1-D(x))$$

Substitute $a = P_{data}(x)$, $y = D(x)$, $b = P_g(x)$

Then, $a\log(y) + b\log(1-y)$

Differentiate w.r.t $D(x)$, then

$$\frac{a}{y} + \frac{-b}{1-y} = \frac{a-(a+b)y}{y(1-y)}$$

To get D^*, we need $\frac{a-(a+b)y}{y(1-y)} = 0$. And when $y = \frac{a}{a+b}$, $\frac{a-(a+b)y}{y(1-y)} = 0$

Hence, $D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_g(x)}$

$\log\left(1 - \frac{2}{2+3}\right) = \log\left(\frac{3}{2+3}\right)$?

$\log\left(1 - \frac{2}{5}\right) = \log\left(\frac{3}{5}\right)$

2) Fix D to D^*, train Generator.

$$\min_G \max_D V(D,G) = \min_G V(D^*, G)$$

$$\min_G V(D^*,G) = \mathbb{E}_{x\sim P_{data}(x)}\left[\log D^*(x)\right] + \mathbb{E}_{x\sim P_g(x)}\left[\log(1-D^*(x))\right]$$

$$= \int P_{data}(x) \cdot \log\left(\frac{P_{data}(x)}{P_{data}(x)+P_g(x)}\right)dx + \int P_g(x)\log\left(\frac{P_g(x)}{P_{data}(x)+P_g(x)}\right)dx$$

Since $1 - \frac{P_{data}(x)}{P_{data}(x)+P_g(x)} = \frac{P_g(x)}{P_{data}(x)+P_g(x)}$

$$= -\log 4 + \underbrace{\log 4}_{= 2\log 2 = \log 2 + \log 2} + \int P_{data}(x)\log\left(\frac{P_{data}(x)}{P_{data}(x)+P_g(x)}\right)dx + \int P_g(x)\log\left(\frac{P_g(x)}{P_{data}(x)+P_g(x)}\right)dx$$

$$= -\log 4 + \int P_{data}(x)\log\left(\frac{2\cdot P_{data}(x)}{P_{data}(x)+P_g(x)}\right)dx + \int P_g(x)\log\left(\frac{2\cdot P_g(x)}{P_{data}(x)+P_g(x)}\right)dx$$

$$= -\log 4 + KL\left(P_{data}(x) \,\Big\|\, \frac{P_{data}(x)+P_g(x)}{2}\right) + KL\left(P_g(x) \,\Big\|\, \frac{P_{data}(x)+P_g(x)}{2}\right),$$

$\star\; KL(P\|Q) = \sum_x P(x)\log\frac{P(x)}{Q(x)}$

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

$$= \cancel{-\log 4} + 2\,JSD(P_{data}\|P_g)$$

$\ni P_{data}(x) = P_g(x)$, then $D^*(x) = \frac{P_{data}(x)}{P_{data}(x)+P_g(x)} = \frac{1}{2}$ Hence, objective function of GAN = JSD

① Training Discriminator: $\min_{G} \max_{D} \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$

$G \Downarrow$ fixed

When $D(x) = 1$, $\Rightarrow \log(1)$: maximum

When $D(G(z)) = 0$, $\Rightarrow \log(1)$: maximum

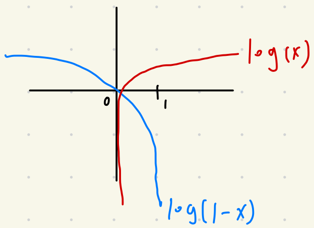
② Training Generator: $\min_{G} \max_{D} \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] = \max_{G} \mathbb{E}_{z \sim p_z(z)}[\log(D(G(z)))]$

$\min_G$

$D \Downarrow$ fixed

When $D(G(z)) = 0$ $\Rightarrow \log(0)$: minimum

When $D(G(z)) = 1$ $\Rightarrow \log(1)$: maximum



In most case, Discriminator > Generator at first, which lead $D(G(z)) = 0$ more.

∴ When $x = 0$, $\log(x)$ has steeper slope than $\log(1-x)$, which enable Generator to be trained efficiently.

☆ Binary Cross Entropy : Measuring the error of reconstruction in for example, an auto-encoder. Note that the targets 'y' should be numbers between 0 and 1.

$$\ell(x, y) = L = \{l_1, l_2, \ldots, l_N\}^T = -w_n[y_n \cdot \log x_n + (1 - y_n)\log(1 - x_n)]$$

↳ minimizing loss function

Then, for Discrimanator,

$$L_D = L_{D\_real} + L_{D\_fake} = \ell(D(x), \text{torch.ones\_like}(D(x))) + \ell(D(G(z)), \text{torch.zeros\_like}(D(G(z))))$$

For Generator,

$$L_{gen} = \ell(D(G(z)), \text{torch.ones\_like}(D(G(z))))$$