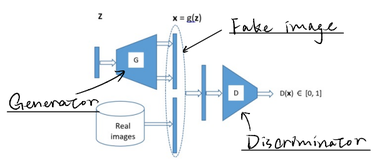
GAN: Generative Adversarial Network

$\{x_i\}_{i=1}^N \sim p(z) \longrightarrow x = g(z)$

⇒ Learning $g: z \to x$



Training G, D Alternatively:

training D ⇒ tell real & fake.

- D(x) = 1 or → 1 : real.
- D(x) = 0 or → 0 : fake

traning G ⇒ fool D

θ_g: parameter for G. θ_d: parameter for D

1 Improve the θ_d so that the discriminator becomes better at distinguishing between fake and real examples:Run the following k times:

- Sample m **real examples** $\mathbf{x}_1, \ldots, \mathbf{x}_m$ from training data.
- Generate m **fake examples** $g(\mathbf{z}_1), \ldots, g(\mathbf{z}_m)$ using the current generator g, where $\mathbf{z}_i \sim p(\mathbf{z})$.
- Improve θ_d so that the discriminator can better distinguish between **those** fake examples and **those** real examples.

2 Improve generator θ_g so as to generate examples the discriminator would find hard to tell fake or real:

- Generate m **new fake examples** $g(\mathbf{z}_1), \ldots, g(\mathbf{z}_m)$ using the current generator g. (Those are examples that the discriminator can tell as fake with high confidence because of the training in step 1.)
- Improve θ_g to generate examples that would be more like real images than **those** fake images.

Training D:

real: $\{x_i\}_{i=1}^N$ ⇒ Loss (Cross Entropy): $-\log(y=1 \mid x_i) = -\log D(x_i)$

fake: $\{g(z_i)\}_{i=1}^M$ ⇒ $-\log(y=0 \mid g(z)) = -\log D(g(z))$

⇒ D minimize: $-\frac{1}{m}\sum_{i=1}^m \log D(x_i) - \frac{1}{m}\sum_{i=1}^m \log D(1-g(z_i))$

⇔ maximize: $\frac{1}{m}\sum_{i=1}^m \log D(x_i) + \frac{1}{m}\sum_{i=1}^m \log D(1-g(z_i))$

$V(\theta_g, \theta_d) = \frac{1}{m}\sum_{i=1}^m \log D_{\theta_d}(x_i) + \frac{1}{m}\sum_{i=1}^m \log D(1-g_{\theta_g}(z_i))$

Obj ⟶ D: $\max_{\theta_d} V(\theta_g, \theta_d) = V(\theta_g)$ (Minimax)

G: $\min_{\theta_g} \max_{\theta_d} V(\theta_g, \theta_d) = \min_{\theta_g} V(\theta_g)$

Training G:

$\min_{\theta_g} V(\theta_g) = \min_{\theta_g} V(\theta_g, \theta_d^*)$

$= \frac{1}{m}\sum_{i=1}^m \log D^*(x_i) + \frac{1}{m}\sum_{i=1}^m \log(1 - D(g_{\theta_g}(z_i)))$ ← Does not depend on θ_g

$= \min_{\theta_g} \frac{1}{m}\sum_{i=1}^m \log(1 - D(g(z_i)))$

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do** (Train D)

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$. (→ $g(z^i)$ fake)
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$. (→ real)
  - Update the discriminator by ascending its stochastic gradient (max):

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  (Train G)

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$. (→ $g(z^i)$ fake)
  - Update the generator by descending its stochastic gradient (min):

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

- At each iteration, the discriminator is not trained to optimum. Instead, its parameters are improved only once by gradient ascent.
- Similarly, the parameters of the generators are also improved only once in each iteration by gradient descent.

FID

- FID compares how often the same high-level features are found within the real and generate.
- Lower FID score means that the generated images have better quality and diversity.
- FID is an improved of Inception Score.

Theory of GAN

D maximizes:

$V(\theta_g, \theta_d) = \frac{1}{m}\sum_{i=1}^m \log D(x_i) + \frac{1}{m}\sum_{i=1}^m \log(1 - D(g(z_i)))$

$V(G, D) = E_{x\sim p_r}[\log D(x)] + E_{x\sim p_g}[\log(1 - D(g(z_i)))]$

$= \int [p_r(x)\log D(x) + p_g(x)\log(1 - D(x))]\,dx.$

When V is optimized (fixed G)

$D^*(x) = \frac{p_r(x)}{p_r(x) + p_g(x)}$

$V(G, D^*) = \int p_r(x)\cdot\left(\log\frac{p_r(x)}{p_r(x)+p_g(x)} + p_g(x)\log\frac{p_r(x)}{p_r(x)+p_g(x)}\right)dx.$

$= \int p_r(x)\log\frac{p_r(x)}{p_a(x)}\,dx + \int p_g(x)\log\frac{p_g(x)}{p_a(x)}\,dx - 2\log 2.$

$= 2 \times \frac{1}{2} \times (D_{KL}(p_r \| p_a) + D_{KL}(p_g \| p_a)) - 2\log 2.$

$= 2\boxed{D_{JS}(p_r \| p_g)} - 2\log 2$

(Jensen-Shannon Divergence)

GAN: { Generator: minimize JS Divergence (↑); Discriminator: approximate JS Divergence }

Loss function for G:

$L = \frac{1}{m}\sum_{i=1}^m \log(1 - D(g(z_i)))$

Gradient $\nabla_{\theta_g} E_{x\sim p_g}[\log(1 - D(x))]$

$= \int \log(1 - D(x))\nabla_{\theta_g} p_g(x)\,dx.$

At Beginning: $D(x) \to 0$, $\log(1 - D(x)) \to 0$.

⇒ Small gradient ⇒ Slow training.

Alternative Loss for G

$L = \frac{1}{m}\sum_{i=1}^m -\log D(g(z_i))$

$\nabla_{\theta_g} L = \nabla_{\theta_g} E_{x\sim p_g(x)}[-\log D(x)]$

$= -\int \log D(x)\,\nabla_{\theta_g} p_g(x)\,dx.$

⇒ gradient is large at beginning ⇒ training is fast

⇒ when D is good ($D(x) \approx 0$ when $x \sim p_g(x)$), gradient ↑, unstable

- The original generator cost function leads to slow training, and the alternative cost function leads to unstable training.
- In practice, those difficulties are dealt with by not training the discriminator to optimum, not even close to optimum.

VAE:

(3) ■ Training is stable.

(2) ■ Does not suffer from mode dropping because it maximizes likelihood, ensuring coverage of the data distribution.

(1) ■ Outputs are often blurrier due to the reconstruction loss, which averages over possible outputs, sacrificing fine details.

min $KL(p_r \| p_g)$

GAN:

(1) ■ Generates sharper and more realistic outputs because the discriminator pushes the generator to produce highly detailed outputs.

(2) ■ Prone to mode dropping, where the discriminator does not force the generator to capture the full diversity of the data distribution.

(3) ■ Training is unstable and can suffer from issues like non-convergence or mode collapse.

{ VAE: Image Generation ⟶ Regression

GAN: Image Generation ⟶ Classification }