

# GAN

## Maximum Likelihood Estimate

- Given  $P_{\text{data}}(x)$
- Have  $P_G(x; \theta)$
- To find  $\theta$  such that  $P_G(x; \theta)$  close to  $P_{\text{data}}(x)$

Sample  $\{x^1, x^2 \dots x^m\}$  from  $P_{\text{data}}$ .

Compute  $P_G(x^i; \theta)$

Likelihood of the generating samples.

$$L = \prod_{i=1}^m P_G(x^i; \theta)$$

Find  $\theta^*$  maximizing the likelihood.

$$\theta^* = \arg \max_{\theta} \prod_{i=1}^m P_G(x^i; \theta)$$

$$= \arg \max_{\theta} \sum_{i=1}^m \log P_G(x^i; \theta)$$

$$\approx \arg \max_{\theta} \mathbb{E}_{x \sim P_{\text{data}}} [\log P_G(x; \theta)]$$

$$\begin{aligned} & (\text{minimize KL-divergence}) \\ & \leftarrow \arg \max_{\theta} \int_x P_{\text{data}}(x) \log P_G(x; \theta) dx - \int_x P_{\text{data}}(x) \log P_{\text{data}}(x) dx \\ & = \arg \min_{\theta} KL(P_{\text{data}}(x) || P_G(x; \theta)) \end{aligned}$$

A strong  $P_G(x; \theta)$ : Neural Network

$$P_G(x) = \int_z P_{\text{prior}}(z) I_{[G(z)=x]} dz$$

(It's difficult to compute likelihood).

# Basic Idea of GAN

- Generator  $G$

- $G$  is a function, input  $z$ , output  $x$

- Given  $P_{\text{prior}}(z)$ ,  $P_G(x)$

- Discriminator  $D$

- $D$  is a function, input  $x$ , output scalar

- Evaluate the "difference" between  $P_G(x)$  and  $P_{\text{data}}(x)$

- Function  $V(G, D)$

$$G^* = \underset{G}{\operatorname{argmin}} \underset{D}{\max} V(G, D)$$

$$V = E_{x \sim P_{\text{data}}} [\log D(x)] + E_{x \sim P_G} [\log (1 - D(x))]$$

Given a  $\setminus G$ , max  $V(G, D)$  evaluate the "difference"

$D^*$  maximizing

$$\therefore V = \int [P_{\text{data}}(x) \log D(x) + P_G(x) \log (1 - D(x))] dx$$

$\therefore$  separately Given  $x$ , the optimal  $D^*$  maximizing

$$P_{\text{data}}(x) \log D(x) + P_G(x) \log (1 - D(x))$$

$$\Rightarrow D^* \text{ maximizing } f(D) = a \log D + b \log (1 - D)$$

$$\Rightarrow D^* = \frac{a}{a+b} = \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_G(x)}$$

$$0 < \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_G(x)} < 1$$

$\hookrightarrow$  "Difference" Between  $P_{\text{data}}$  and  $P_G$

$$\max V(G, D) = V(G, D^*)$$

$$= -2 \log 2 + KL(P_{\text{data}}(x) \parallel \frac{P_{\text{data}}(x) + P_G(x)}{2})$$

$$+ KL(P_G(x) \parallel \frac{P_{\text{data}}(x) + P_G(x)}{2})$$

$$0 < \quad < \log 2$$

$$= -2 \log 2 + 2 JSD(P_{\text{data}}(x) \parallel P_G(x))$$

Jensen-Shannon divergence

$$JSD(P \parallel Q)$$

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

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

## Algorithm

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

$L(G)$

- To find the best  $G$  minimizing the loss function  $L(G)$

$$\theta_G \leftarrow \theta_G - \eta \frac{\partial L(G)}{\partial \theta_G}$$

$$f(x) = \max \{D_1(x), D_2(x), D_3(x)\}$$

$$\frac{df(x)}{dx} = \frac{dD_i(x)}{dx}$$

if  $D_i(x)$  is the max one

Given  $G_0$

Find  $D_0^*$  maximizing  $V(G_0, D)$

$$\theta_G \leftarrow \theta_G - \eta \frac{\partial L(G)}{\partial \theta_G} \rightarrow \text{obtain } G_1$$

Find  $D_1^*$  maximizing  $V(G_1, D)$

$$\theta_G \leftarrow \theta_G - \eta \frac{\partial V(G, D_1^*)}{\partial \theta_G} \rightarrow \text{obtain } G_2$$

In Practice ...

$$V = E_{x \sim P_{\text{data}}} [\log D(x)] + E_{z \sim P_G} [\log (1 - D(z))]$$

- Given  $G$ , compute  $\max_D V(G, D)$

Sample  $x, \tilde{x}$  from  $P_{\text{data}}$  and  $P_G$

$$\text{Maximize } \tilde{V} = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(\tilde{x}^i)) \text{ like cross-entropy}$$

Do binary classification

$P_{\text{data}}(x) \rightarrow \text{Positive example}$

$P_G(x) \rightarrow \text{Negative example}$

Initialize  $\theta_d, \theta_g$ .

from  $P_{\text{prior}}(z)$

- In each training iteration:

- Sample  $X_m$  from  $P_{\text{data}}(x)$  and sample  $m$  noise samples  $Z_m$
- Obtain  $\tilde{X}_m = G(Z_m)$
- Update discriminator  $\theta_d$  to maximize
  - $\hat{V} = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(\tilde{x}^i))$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \hat{V}(\theta_d)$

Learning D

(Repeat k times)

Learning G

(Only Once)

- Sample another  $m$  noise samples  $\{z^1, z^2, \dots, z^m\}$  from  $P_{\text{prior}}(z)$
- Update generator  $\theta_g$  to minimize
  - $\tilde{V} = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(z^i)))$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

## Objective Function for Generator in Real Implementation

$$V = \cancel{E_{x \sim P_{\text{data}}} [\log D(x)]} + E_{x \sim P_G} [\log (1 - D(x))]$$

slow at beginning.

$$\rightarrow V = E_{x \sim P_G} [-\log (D(x))]$$

Discriminator

loss = 0.

Reason 1. Approximate by sampling

2. the nature of data

Both  $P_{\text{data}}$  and  $P_G$  are lowdim manifold in high-dim space.

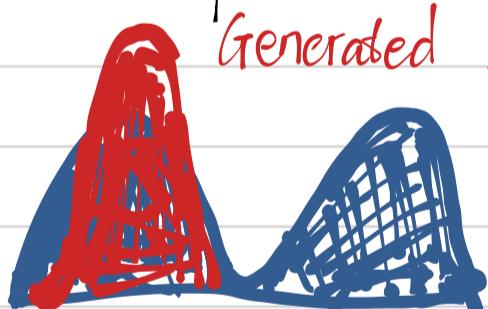
To solve ...

Add Noise

- Add some artificial noise to the inputs of  $D$
- Make the labels noisy for discriminator

to let  $P_{\text{data}}$  and  $P_G$  have some overlap so that the JS divergence will be less than  $\log 2$   
 & Noise decay over time

Mode Collapse

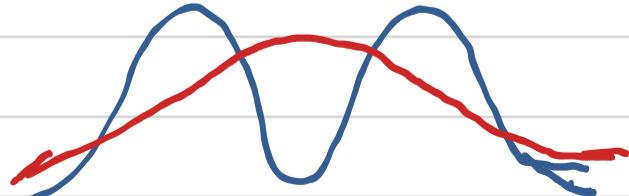


Generated Distribution

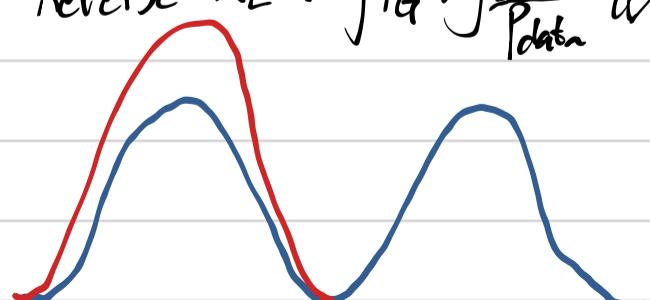
Data Distribution

Flaw in Optimization?

$$KL = \int P_{\text{data}} \log \frac{P_{\text{data}}}{P_G} dx$$



$$\text{Reverse KL} = \int P_G \log \frac{P_G}{P_{\text{data}}} dx$$



# Conditional GAN

challenge.

Text  $\rightarrow$  NN

(a point, Not a distribution)

