Generative Adversarial Nets (GAN)

2023.01.09 → Li Mu GAN

1. Generative Model ✓ v.s. Discriminative Model

2. Adversarial process
   - G → generative model to capture the data distribution
   - D → discriminative model
     - training data
     - G

   2a) train G, thru trying to cheat D

   2b) train D, thru trying to distinguish
   - training data
   - generated data

   → minimax 2-player game in Game Theory

3. Deep Learning works well in discriminative task

   behaves not well in generative task

   Issue: DL handles generative task through approximating intractable computation arising in MLE

   → very difficult

Actually, GAN proposes another approach which is more feasible.

Adversarial Nets Framework

- Generative Model → make fake money
- Discriminative Model → police / detector

we hope that the Generative Model can win!

造假者

4. In the paper setting, the general architecture is:

Gaussian Noise → Generative Model → Discriminative Model → { true / false }

input (Gaussian Noise)

MLP (Generative Model)

MLP (Discriminative Model)

output ({ true / false })

Rmk: ① [input → MLP G → MLP D → output] architecture

is so-called Adeversarial Nets

→ can be trained through BP Algo instead of MC samples

② Actually you can use other units to model Generator & Discriminator instead of MLP.

5. Related Work

a) Deep Generative Model

↓

(provide a parametric specification of prob. dist.) → most successful one ⇒ Deep Boltzmann Machine → Computationally infeasible

→ Issue: Intractable likelihood functions w.r.t GD → require approximation

Idea → 不去构造分布，去学一个Model来近似分布 → Strategy ⇒ b) Generative Stochastic Networks

Advantage/Trick
{ 1. not explicitly represent the likelihood
{ 2. can generate samples from desired distribution

No need for numerous approximation
Can exactly apply GD framework

→ Difference Between { Boltzmann Machine / Generative Stochastic Networks }

Boltzmann Machine → learn the $\boxed{\text{parametric distribution}}$
Generative Stochastic Nets → learn the $\boxed{\text{Approximate Generative Model}}$

$\boxed{\text{LIMITS}}$ → No parametric distribution

$\boxed{\text{EDGES}}$ → Computationally tractable

c) VAE → Variational Auto-Encoder

d) Predictability Minimization

6. GAN Algo

a) Idea:

Suppose that, we have a Game that generates images
Our aim is to generate more images from this Game.

Method 1 : write the code of this game

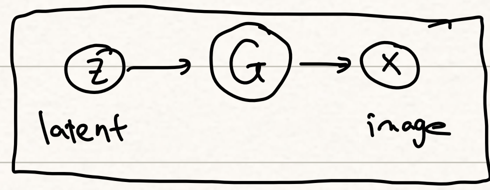
$\updownarrow$

learn the exact probability distribution

☆ Method 2 : focus on the images we have and learn to generate new images from our samples

samples

No exact distribution



Architecture

LIMITS (随机生成) $\Rightarrow$ 无法(很难)控制生成的内容

$\downarrow$

inverse problem : given an image, find its latent variable

b) Details :

$$\begin{cases} \text{Generator}: G(z;\theta_g): z \in \mathbb{R}^d \longrightarrow x \in \mathbb{R}^D \\ \text{Discriminator}: D(x;\theta_d): x \in \mathbb{R}^D \longrightarrow \text{scalar} \in [0,1] \subseteq \mathbb{R} \end{cases}$$

$\hookrightarrow$ interpretation : the prob. that x comes from data not generator

Objective : $$(\tilde{G}, \tilde{D}) = \underset{G}{\operatorname{argmin}}\ \underset{D}{\operatorname{argmax}}\ V(G,D)$$

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

Rmk : 1. $\boxed{\underset{G}{\operatorname{argmin}}}$ $\longrightarrow$ Generator wants to make Discriminator make mistakes

2. $\underset{D}{\boxed{\arg\max}}$ → Discriminator wants to Maximize Log-likelihood

$\boxed{\text{Binary Classification Likelihood like logistic regression}}$

3. Likelihood $= P(\text{Discriminator makes correct classification } x)$

$$= D(x)^{\mathbb{1}\{x \in \text{Data}\}} + (1 - D(x))^{\mathbb{1}\{x \in G\}}$$

$\boxed{\text{LABEL}}$

$$\Rightarrow \mathbb{E}_x[\log\text{-likelihood}]$$

$$\propto \mathbb{E}_{x \sim p_x(x)}[\log D(x)] + \mathbb{E}_{x \sim p_{fake}(x)}[\log(1 - D(x))]$$

$$= \mathbb{E}_{x \sim p_x(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

achieve from generation of fake images

7. Small issues that need to be modified

⇒ in the early stage of training, the generator G is very poor, then it is very easy for Discriminator D to perfectly distinguish Real Data & Fake Data.

⇒ Recap: $V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)]$

$$+ \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

when we need to update G, we actually do:

$$\tilde{G} = \arg\min_G \mathbb{E}_{z \sim p_z(z)}[\log(1 - \tilde{D}(G(z)))]$$

$\boxed{\text{Note}}$: $\tilde{D}(G(z)) \equiv 0$ if the Early-Stage Generator G is very poor $\Rightarrow$ Gradient $\nabla_{\theta_g} V(G, \tilde{D})$ saturates!!!

Modification : $\tilde{G} = \underset{G}{\arg\max}\ \mathbb{E}_{z \sim p_z(z)} [\log \tilde{D}(G(z))]$

(approximation for the tractability)

8. Theoretical Result

a) Fix Generator G, then Discriminator D is optimal

i.f.f $\quad D(x) = \dfrac{P_{data}(x)}{P_{data}(x) + P_G(x)}$

Rmk : A trivial result for Binary Classification task

when given the class conditional probability $\begin{cases} \mathbb{P}(Y=1 \mid x) \\ \mathbb{P}(Y=0 \mid x) \end{cases}$

Actually, $\begin{cases} \mathbb{P}(Y=1 \mid x) = P_{data}(x) \\ \mathbb{P}(Y=0 \mid x) = P_g(x) \end{cases}$

b) $\underset{D}{\max}\ V(G, D)$

$$= V(G, D_G^*)$$

$$= \mathbb{E}_{x \sim P_{data}(x)} \left[\log \frac{P_{data}(x)}{P_{data}(x) + P_g(x)} \right] + \mathbb{E}_{x \sim P_g} \left[\log \frac{P_g(x)}{P_{data}(x) + P_g(x)} \right]$$

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

$$\geq -\log 4$$

Note that the equality holds i.f.f $\begin{cases} P_{data} = \dfrac{P_{data}+P_g}{2} \\ P_g = \dfrac{P_{data}+P_g}{2} \end{cases}$

i.f.f $P_g = P_{data}$

$\Rightarrow$ the $\boxed{\text{global optimal}}$ for $\boxed{\min_G \max_D V(G, D)}$ is $-\log 4$

☆

and achieve when $\begin{cases} P_g(x) = P_{data}(x) & \text{for all } x \\ D(x) = 1/2 & \text{for all } x \end{cases}$