Generative Model

1. Supervised vs Unsupervised

	Supervised	Unsupervised
Data	(x, y)	x ; no label
Goal	learn function to map x → y	learn underlying hidden structure of data
ex	classification, regression, …	clustering, dim reduction, …
		k-means clustering, PCA, Auto encoder : Generative model로 feature 배우는

2. Generative Models : train data의 분포로부터 new sample을 generate하는 모델

Let $p_{model}(x)$ to be similar to $p_{data}(x)$

→ address density estimation, a core problem in unsupervised learning

① 모델을 명확하게 정의해두고 추정해 나가며 $p_{model}(x)$를 구하는 방법

② 명확한 정의 없이 $p_{model}(x)$를 구하는 방법

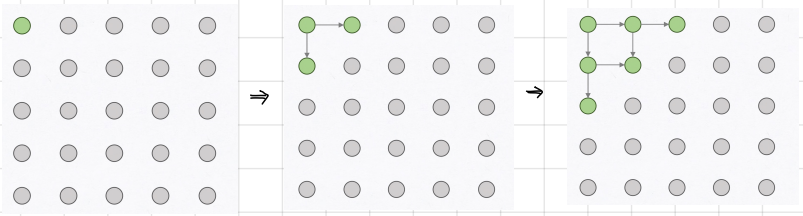
⇒ 장점 : latent factor inference → general feature 찾아낼 수 있음.

3. Pixel RNN / CNN : explicit density model, tractable (다루기 쉬운) tractable한 density function을 정의하고, training data의 likelihood를 optimize (maximize)

① $p(x) = \prod_{i=1}^{n} p(x_i | x_1, \cdots, x_{i-1})$ i : 픽셀 위치 x : training data

이미지 x에 대한 likelihood → likelihood maximize가 목표 ⇒ pixel value에 따른 distribution이 complex ⇝ neural network 사용

② Pixel RNN : 이전 픽셀들에 대한 dependency를 RNN(LSTM)을 이용해 모델링



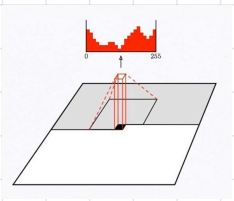
Drawback : sequential하게 generate ⇒ 매우 느림.

학습도 느리고, generation도 느림

③ Pixel CNN : 현재 위치에서의 픽셀 값을 generate하기 위해 주변 region의 정보를 이용

→ train time : spatial region data를 이용해 dependency ($p(x_i | \cdots)$)를 학습

⇒ pixel RNN보다는 더 빠름. 그러나 여전히 generation이 sequential하게 일어나기 때문에 느림



④ Pixel RNN / CNN Pros & Cons

(1) Pros : - likelihood $p(x)$를 explicitly compute 가능

- train data를 가지고 모델을 평가하기 좋음
- 그럴듯한 이미지 생성 ; semantic한 부분에서 차이 있을 수 있지만.

(2) Cons : Slow

4. Variational AutoEncoder : explicit but intractable density function with latent z ⇒ 더 정교한 모델!

$$p_\theta(x) = \int p_\theta(z)\, p_\theta(x|z)\, dz$$

↳ 바로 optimize 불가. ⇒ lower bound를 만들어놓고 이를 optimize

① Autoencoder : 낮은 차원의 feature representation을 도출하는 unsupervised approach

L2 loss function: $\|x - \hat{x}\|^2$

$\hat{x}$

↑ Decoder : reconstruct original data

z → Why dimension reduction? 즉 x의 중요한 feature를 represent해야하므로.

↑ Encoder : 더 낮은 차원으로 e.g. CNN, Deep FC layers, ...

x

⇒

Loss function

$\hat{y}$ y

↑ Classifier

z

↑ Encoder

x

Fine-tune encoder jointly with classifier

→ encoder, decoder should be symmetric e.g. encoder: 4-layer conv. decoder: 4-layer upconv.

⇒ 충분한 양의 data가 없는 경우, supervised learning initialize하기위해 사용.

⇝ ※ z (features)는 training data의 variation 요소를 잡아냄 ⇒ 그렇다면 이걸 가지고 new img generate할 수 있지 않을까?

② Variational Autoencoders :

Assume $\{x^{(i)}\}_{i=1}^N$ is generated from unobserved representation (z)

우리의 목표 ↓

We want to estimate the true parameters θ^* of this generative model.

x ← Sample from true conditional : $p_{\theta^*}(x|z^{(i)})$

↑ Decoder Network

z ← Sample from true prior : $p_{\theta^*}(z)$

(1) p(z)를 가우시안 분포로 정함.

(2) p(x|z)는 neural network로 구함.

How to train the model? 앞서 본 pixel RNN/CNN 같은 FVBN model의 경우 training data x의 likelihood를 maximize

$p_\theta(x) = \int p_\theta(x|z)\, p_\theta(z)\, dz$ → 이걸 maximize? 무슨 문제가 있는가? Intractable하다!

→ Data likelihood : $p_\theta(x) = \int p_\theta(x|z)\, p_\theta(z)\, dz$ ← Intractable

($p_\theta(x|z)$: Decoder neural network, $p_\theta(z)$: simple Gaussian prior)

Posterior density : $p_\theta(z|x) = p_\theta(x|z)\, p_\theta(z) / p_\theta(x)$ ← Intractable

⇒ cannot optimize directly ⇒ $p_\theta(z|x)$를 approximate하는 인코더 네트워크 $q_\phi(z|x)$를 사용

Sample z from $z|x \sim N(\mu_{z|x}, \Sigma_{z|x})$

$\mu_{z|x}$ $\Sigma_{z|x}$

Encoder $q_\phi(z|x)$

x

Sample x|z from $x|z \sim N(\mu_{x|z}, \Sigma_{x|z})$

$\mu_{x|z}$ $\Sigma_{x|z}$

Decoder $p_\theta(x|z)$

z

$$\log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (q_\phi: \text{encoder})$$

$$= E_z\left[\log \frac{p_\theta(x^{(i)}|z)\, p_\theta(z)}{p_\theta(z|x^{(i)})}\right]$$

$$= E_z\left[\log \frac{p_\theta(x^{(i)}|z)\, p_\theta(z)}{p_\theta(z|x^{(i)})} \frac{q_\phi(z|x^{(i)})}{q_\phi(z|x^{(i)})}\right]$$

$$= E_z\left[\log p_\theta(x^{(i)}|z)\right] - E_z\left[\log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)}\right] + E_z\left[\log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})}\right]$$

$$= E_z\left[\log p_\theta(x^{(i)}|z)\right] - D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z|x^{(i)}))$$

($E_z[\log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z))$: Variational lower bound; $D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z|x^{(i)}))$: Intractable but ≥0)

find θ^*, ϕ^* that maximize this term

$L(x^{(i)}, \theta, \phi)$

cf) KL Divergence ↑ → 두 분포 서로 다름

→ $E_z[\log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z))$ 최적화 ; $q_\phi(z|x^{(i)})$가 $p_\theta(z)$ 분포 비슷하도록

Maximize likelihood of original image being reconstructed

Make approximate posterior distribution close to prior

→ 이를 maximize하는 param θ, ϕ 찾기 위해 각 batch 마다 forward pass, backprop 함.

③ VAE 장단점

(1) Pros : - principled approach for generative model

- $q(z|x)$ inference → feature representation 다른 tasks에도 유용

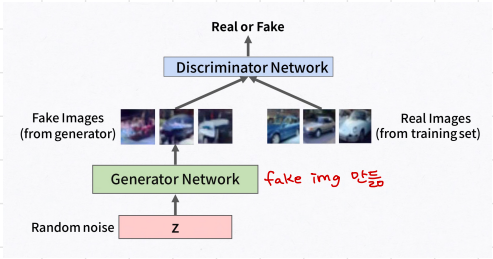
(2) Cons : - use lowerbound → 좋은 evaluation은 아님.

- GAN에 비해 blurrier & lower quality

5. GAN : Implicit density function. → take game - theoretic approach

① Sample from noise. Then learn transformation to training distribution.

②

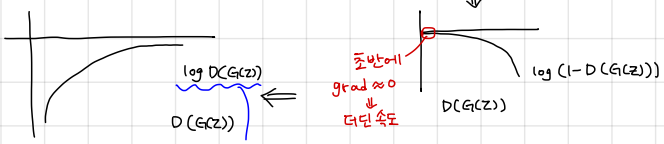


$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

G는 이를 minimize하고 싶음.

<objective function>

D는 Gradient Ascent, G는 Gradient Descent



이걸 maximize하는 방향

objective : $\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$

즉 D는 전체식에 GA, G는 log(D(G(z))에서 GA

③ GAN Pros & Cons

(1) Pros : - 좋은 성능

(2) Cons : - train하기 어려움 (어느 하나가 성능이 좋아지면 x)

- P(x), P(z|x) 같은 query에 대한 inference가 없다.