

# Deep Learning Basics

Lecture 10: Generative Models Part 2

**최성준** (고려대학교 인공지능학과)

**WARNING:** 본 교육 콘텐츠의 지식재산권은 재단법인 네이버커넥트에 귀속됩니다. **본 콘텐츠를 어떠한 경로로든 외부로 유출 및 수정하는 행위를 엄격히 금합니다.** 다만, 비영리적 교육 및 연구활동에 한정되어 사용할 수 있으나 재단의 허락을 받아야 합니다. 이를 위반하는 경우, 관련 법률에 따라 책임을 질 수 있습니다.

# Contents

---

- Maximum Likelihood Learning
- Latent Variable Models
- Generative Adversarial Networks
- Diffusion Models

# Maximum Likelihood Learning

# Maximum Likelihood Learning



45 Best Large Dog Breeds - T...  
goodhousekeeping.com



How dogs contribute to your...  
medicalnewstoday.com



scientists explain puppy dog eyes ...  
theguardian.com



The Best Dogs of BBC Earth | Top 5 ...  
youtube.com



Hot dogs: what soaring pu...  
theguardian.com



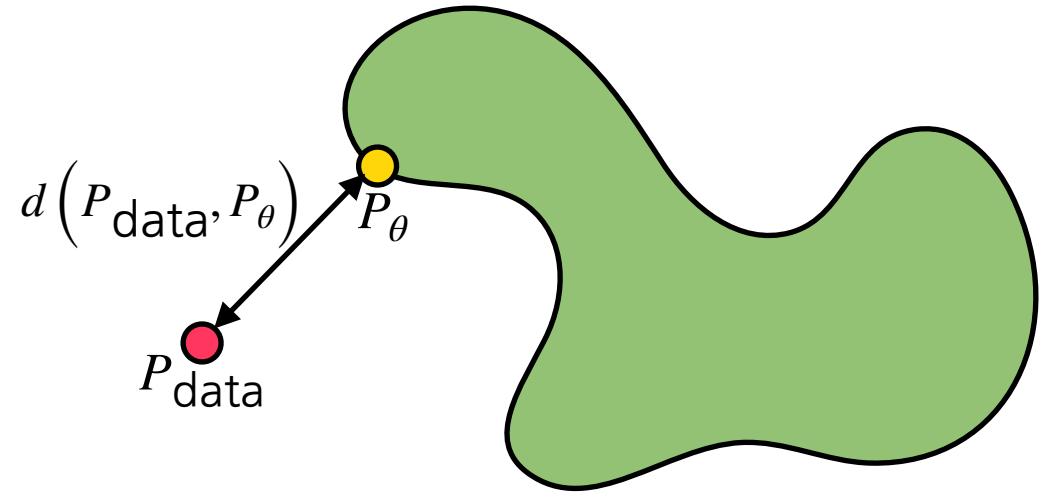
Female Dogs in Heat - zooplus Magazine  
zooplus.co.uk



Dog images - Pexels - Free Stock Phot...  
pexels.com



Dogs caught coronavirus from their ...  
nature.com



Model family  $\theta \in \Theta$

- Given a training set of examples, we can cast the generative model learning process as finding the **best-approximating density** model from the **model family**.
- Then, how can we **evaluate the goodness** of the approximation?

# Maximum Likelihood Learning

---

- KL-divergence

$$\mathbf{D}(P_{\text{data}} || P_{\theta}) = \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} \left[ \log \left( \frac{P_{\text{data}}(\mathbf{x})}{P_{\theta}(\mathbf{x})} \right) \right] = \sum_{\mathbf{x}} P_{\text{data}}(\mathbf{x}) \log \frac{P_{\text{data}}(\mathbf{x})}{P_{\theta}(\mathbf{x})}$$

- We can simplify this with

$$\begin{aligned}\mathbf{D}(P_{\text{data}} || P_{\theta}) &= \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} \left[ \log \left( \frac{P_{\text{data}}(\mathbf{x})}{P_{\theta}(\mathbf{x})} \right) \right] \\ &= \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} [\log P_{\text{data}}(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} [\log P_{\theta}(\mathbf{x})]\end{aligned}$$

- As the first term does not depend on  $P_{\theta}$ , minimizing the **KL-divergence** is equivalent to maximizing the expected log-likelihood.

$$\arg \min_{P_{\theta}} \mathbf{D}(P_{\text{data}} || P_{\theta}) = \arg \min_{P_{\theta}} -\mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} [\log P_{\theta}(\mathbf{x})] = \arg \max_{P_{\theta}} \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} [\log P_{\theta}(\mathbf{x})]$$

# Maximum Likelihood Learning

---

- Approximate the expected log-likelihood

$$\mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} [\log P_{\theta}(\mathbf{x})]$$

with the empirical log-likelihood

$$\mathbf{E}_{\mathcal{D}} [\log P_{\theta}(\mathbf{x})] = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \log P_{\theta}(\mathbf{x})$$

- Maximum likelihood learning is then:

$$\max_{P_{\theta}} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \log P_{\theta}(\mathbf{x})$$

- Variance of Monte Carlo estimate is high:

$$V_P[\hat{g}] = V_P \left[ \frac{1}{T} \sum_{t=1}^T g(x^t) \right] = \frac{V_P[g(x)]}{T}$$

# Empirical Risk Minimization

---

- For maximum likelihood learning, **empirical risk minimization (ERM)** is often used.
- However, **ERM** often suffers from its overfitting.
  - Extreme case: The model remembers all training data

$$p(x) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \delta(x, x_i).$$

- To achieve better generalization, we typically restrict the **hypothesis space** of distributions that we search over.
- However, it could deteriorate the performance of the generative model.

# Maximum Likelihood Learning

---

- Usually, MLL is prone to under-fitting as we often use simple parametric distributions such as spherical Gaussians.
- What about other ways of measuring the similarity?

# Maximum Likelihood Learning

---

- Usually, MLL is prone to under-fitting as we often use simple parametric distributions such as spherical Gaussians.
- What about other ways of measuring the similarity?
  - **KL-divergence** leads to maximum likelihood learning or Variational Autoencoder (VAE).
  - **Jensen-Shannon divergence** leads to Generative Adversarial Network (GAN).
  - **Wasserstein distance** leads to Wasserstein Autoencoder (WAE) or Adversarial Autoencoder (AAE).

# Latent Variable Models

D. Kingma, "Variational Inference and Deep Learning: A New Synthesis," Ph.D. Thesis

# Quick Question

---

Is an **autoencoder** a generative model?

# Variational Autoencoder

---

- The objective is simple.

Maximize  $p_{\theta}(\mathbf{x})$

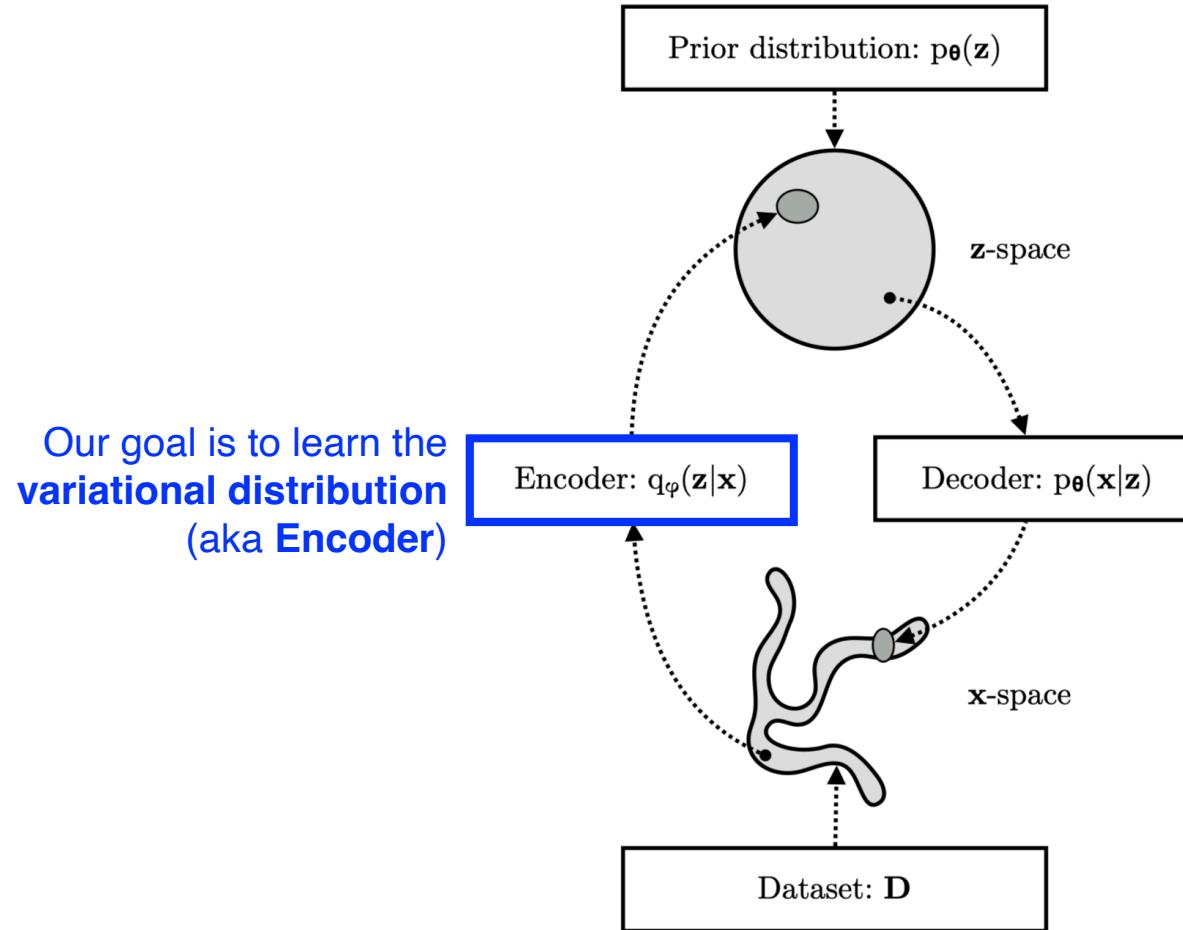
# Variational Autoencoder

---

- Variational inference (VI)
  - The goal of VI is to optimize the **variational distribution** that best matches the **posterior distribution**.
    - **Posterior distribution:**  $p_\theta(z|x)$
    - **Variational distribution:**  $q_\phi(z|x)$
  - In particular, we want to find the **variational distribution** that minimizes the KL divergence between the true posterior.

# Variational Autoencoder

---



# Variational Autoencoder

---

$$\underbrace{\log p_{\theta}(x)}_{\text{Maximum Likelihood Learning } \uparrow} = \int_z \underbrace{q_{\phi}(z|x)}_{\text{For any } q_{\phi}} \log p_{\theta}(x) dz$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x)p_{\theta}(z|x)}{p_{\theta}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{p_{\theta}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)q_{\phi}(z|x)}{p_{\theta}(z|x)q_{\phi}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right] + \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right]$$
$$= \underbrace{\mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right]}_{\text{ELBO } \uparrow} + \underbrace{D_{KL} (q_{\phi}(z|x) \| p_{\theta}(z|x))}_{\text{Variational Gap } \downarrow}$$

# Variational Autoencoder

$$\underbrace{\log p_{\theta}(x)}_{\text{Maximum Likelihood Learning } \uparrow} = \int_z \underbrace{q_{\phi}(z|x)}_{\text{For any } q_{\phi}} \log p_{\theta}(x) dz$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x)p_{\theta}(z|x)}{p_{\theta}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{p_{\theta}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)q_{\phi}(z|x)}{p_{\theta}(z|x)q_{\phi}(z|x)} \right]$$
$$= \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right] + \mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right]$$

Importantly, **ELBO** is a tractable quantity.

$$= \boxed{\mathbb{E}_{z \sim q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right]} + \boxed{\underbrace{D_{KL} (q_{\phi}(z|x) \| p_{\theta}(z|x))}_{\text{Variational Gap } \downarrow}}$$

VI minimizes the (intractable) **objective** via maximizing **ELBO**.

# Evidence Lower Bound

---

$$\underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right]}_{\text{ELBO } \uparrow} = \int \log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} q_\phi(z|x) dz$$
$$= \underbrace{\mathbb{E}_{q_\phi(z|x)} [p_\theta(x|z)]}_{\text{Reconstruction Term}} - \underbrace{D_{KL} (q_\phi(z|x) \| p(z))}_{\text{Prior Fitting Term}}$$

# Evidence Lower Bound

---

$$\underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right]}_{\text{ELBO } \uparrow} = \int \log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} q_\phi(z|x) dz$$
$$= \underbrace{\mathbb{E}_{q_\phi(z|x)} [p_\theta(x|z)]}_{\text{Reconstruction Term}} - \underbrace{D_{KL} (q_\phi(z|x) \| p(z))}_{\text{Prior Fitting Term}}$$

This term minimizes the **reconstruction loss** of an distribution to be similar to the auto-encoder.

This term enforces the latent **prior distribution**.

# Evidence Lower Bound

---

$$\underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right]}_{\text{ELBO } \uparrow} = \int \log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} q_\phi(z|x) dz$$
$$= \underbrace{\mathbb{E}_{q_\phi(z|x)} [p_\theta(x|z)]}_{\text{Reconstruction Term}} - \underbrace{D_{KL} (q_\phi(z|x) \| p(z))}_{\text{Prior Fitting Term}}$$

## Key Limitation

- It is an **intractable** model (hard to evaluate likelihood).
- The prior fitting term should be differentiable, hence it is hard to use diverse latent prior distributions.
- In most cases, we use an isotropic Gaussian where we have a closed-form for the prior fitting term.

$$\underbrace{D_{KL} (q_\phi(z|x) \| \mathcal{N}(0, I))}_{\text{Prior Fitting Term}} = \frac{1}{2} \sum_{i=1}^D (\sigma_{z_i}^2 + \mu_{z_i}^2 - \log(\sigma_{z_i}^2) - 1)$$

# Variational Autoencoder

---



D. Kingma, "Variational Inference and Deep Learning: A New Synthesis," Ph.D. Thesis

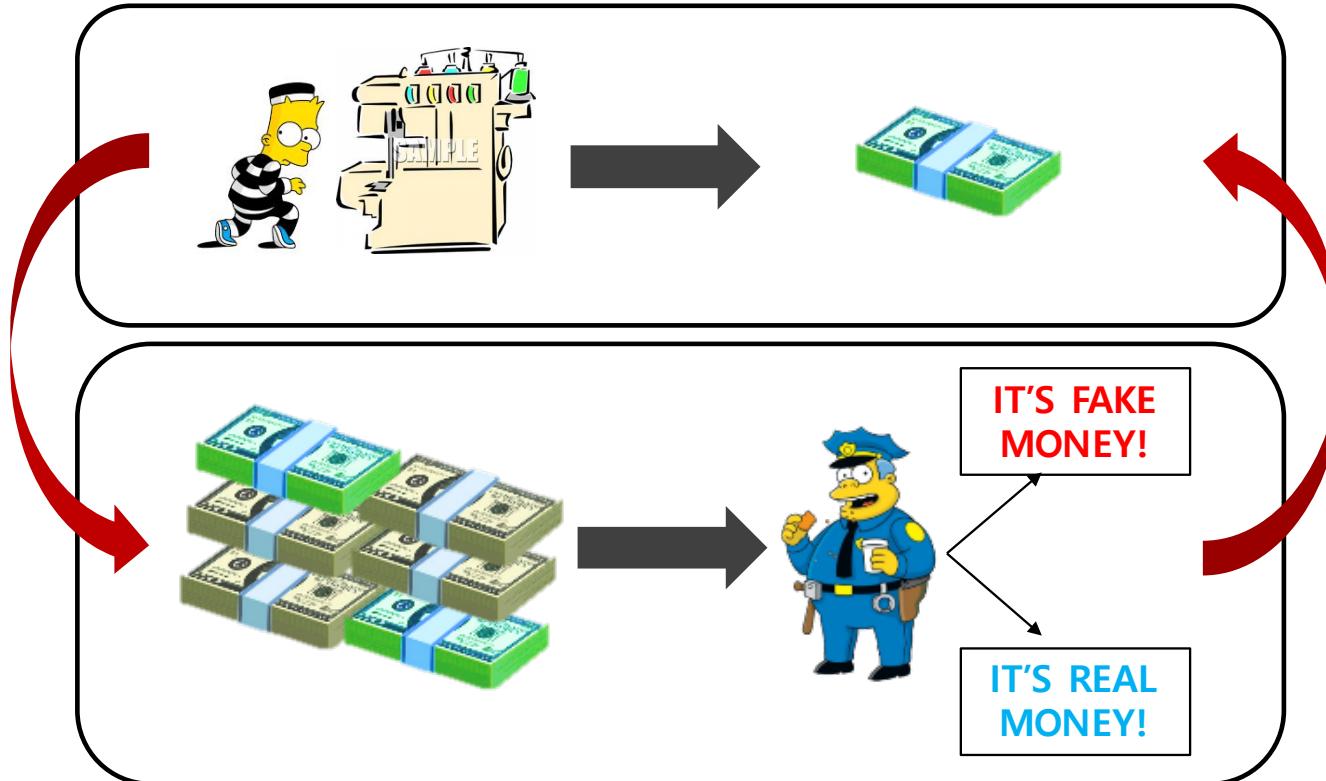
© NAVER Connect Foundation

# Generative Adversarial Networks

Goodfellow et al., "Generative Adversarial Networks", NIPS, 2014

# Generative Adversarial Networks

---



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

# GAN Objective

---

- GAN is a two player minimax game between **generator** and **discriminator**.
  - **Discriminator** objective

$$\max_D V(G, D) = E_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x})] + E_{\mathbf{x} \sim p_G} [\log(1 - D(\mathbf{x}))]$$

- The **optimal discriminator** is

$$D_G^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}$$

# GAN Objective

---

- GAN is a two player minimax game between **generator** and **discriminator**.

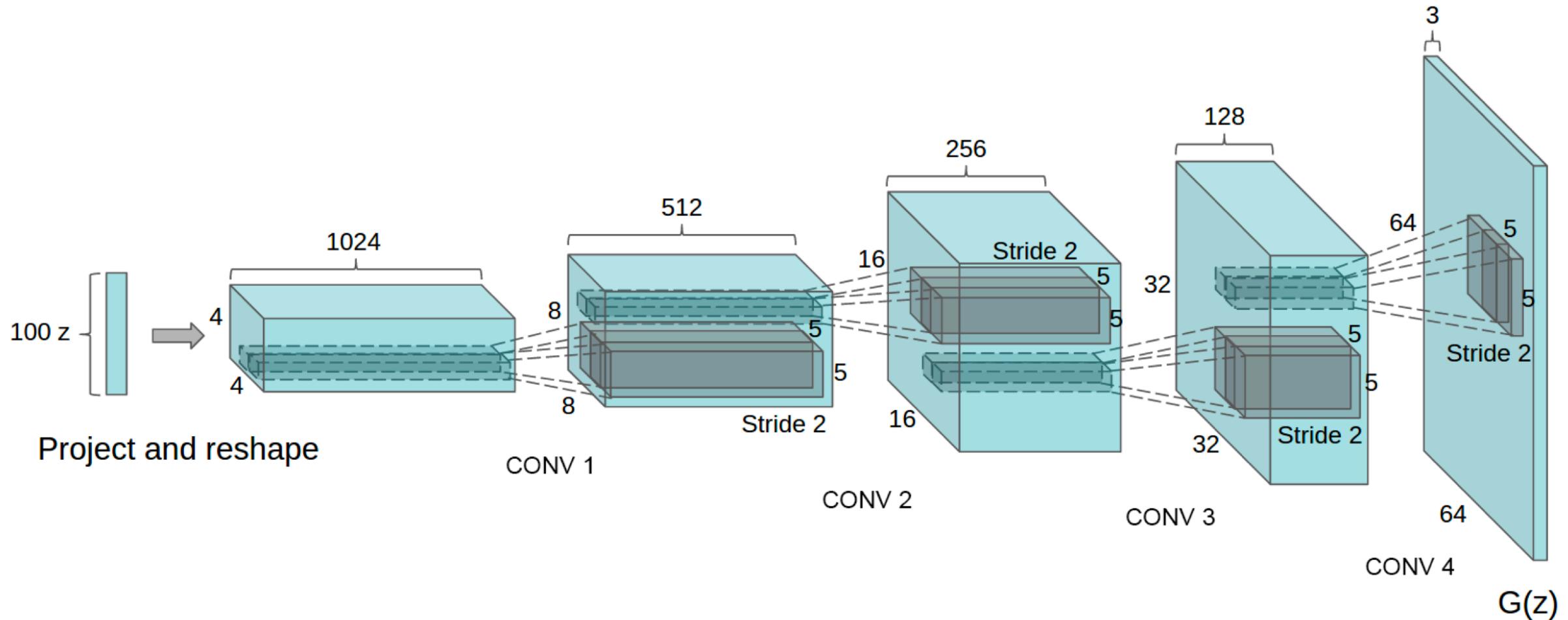
- **Generator** objective

$$\min_G V(G, D) = E_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x})] + E_{\mathbf{x} \sim p_G} [\log(1 - D(\mathbf{x}))]$$

- Plugging in the **optimal discriminator**, we get

$$\begin{aligned} V(G, D_G^*(\mathbf{x})) &= E_{\mathbf{x} \sim p_{\text{data}}} \left[ \log \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})} \right] + E_{\mathbf{x} \sim p_G} \left[ \log \frac{p_G(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})} \right] \\ &= E_{\mathbf{x} \sim p_{\text{data}}} \left[ \log \frac{p_{\text{data}}(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}{2}} \right] + E_{\mathbf{x} \sim p_G} \left[ \log \frac{p_G(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}{2}} \right] - \log 4 \\ &= \underbrace{D_{KL} \left[ p_{\text{data}}, \frac{p_{\text{data}} + p_G}{2} \right] + D_{KL} \left[ p_G, \frac{p_{\text{data}} + p_G}{2} \right]}_{2 \times \text{Jenson-Shannon Divergence (JSD)}} - \log 4 \\ &= 2D_{JSD}[p_{\text{data}}, p_G] - \log 4 \end{aligned}$$

# Deep Convolutional GAN



# Diffusion Models

Ho et al., "Denoising Diffusion Probabilistic Models," 2020

# Diffusion Models

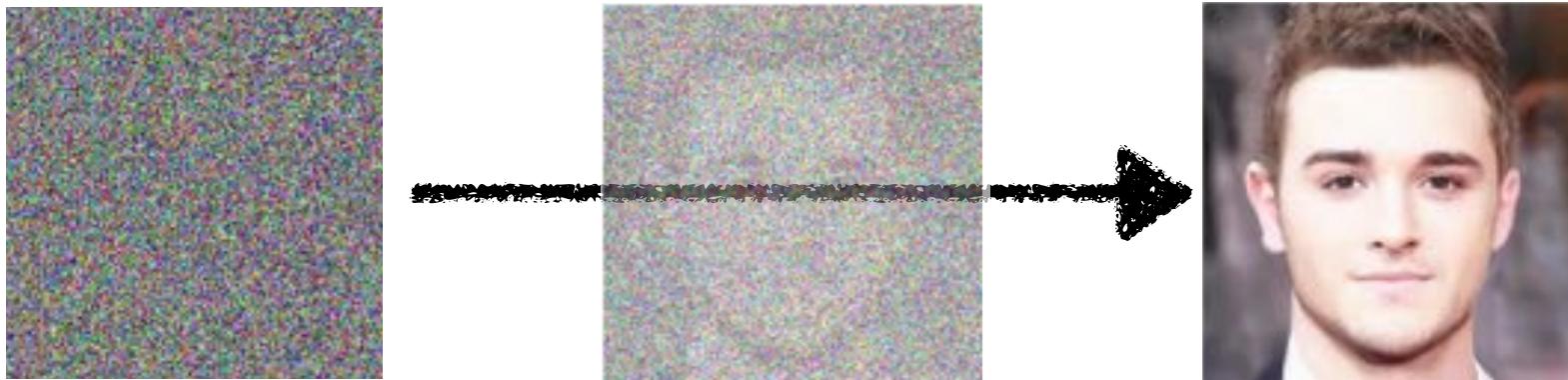
---



Diffusion models generate images from noise.

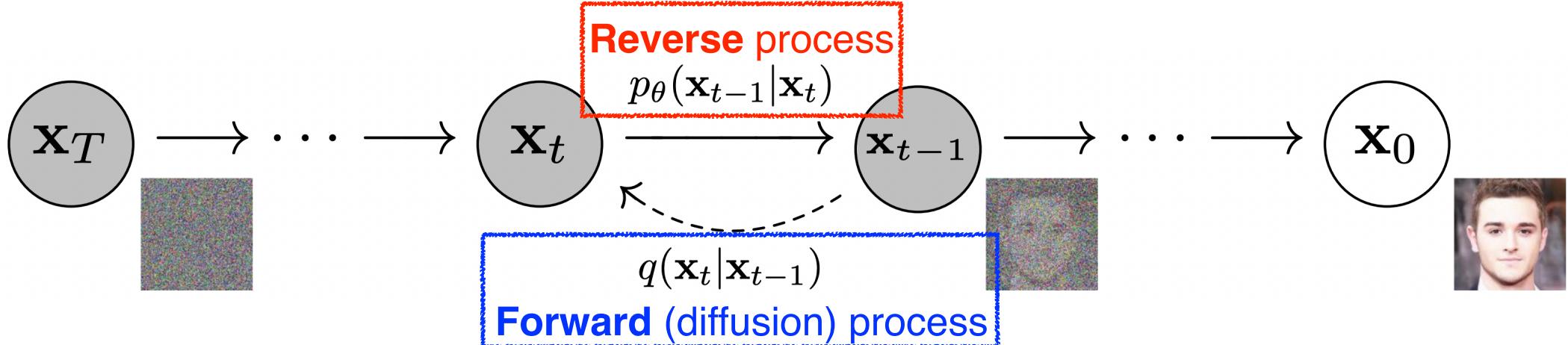
# Diffusion Models

---



Diffusion models **progressively** generate images from noise.

# Diffusion Models



- **Forward (diffusion) process** progressively injects noise to an image.

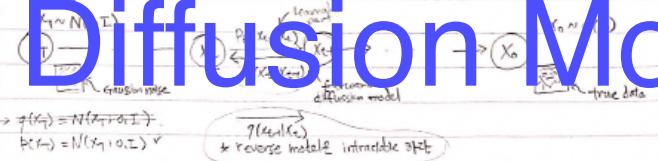
$$q(\mathbf{x}_{1:T} | \mathbf{x}_0) := \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}) := \prod_{t=1}^T \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

- The **reverse process** is learned in such a way to **denoise** the perturbed image back to a clean image.

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

# Diffusion Models

Denoising Diffusion Probabilistic Model (DDPM) for Idiots



MC objective DDPM이 주는  $x_T \sim N(0, I)$  를 사용해서 T번의 forward process  $p_\theta(x_{t-1}|x_t)$  을 거친 결과가 true data distribution  $q(x_t)$  과 비슷하게 만드는 것이다.

기억적으로  $p_\theta(x_{t-1}|x_t)$  를 학습을 하면,  $p_\theta(x_t) = \int p_\theta(x_0, x_1, \dots, x_T) dx_0 dx_1 \dots dx_T$  를 model을 만들수 있다. 물론,  $p_\theta(x_0, x_1, \dots, x_T) = p_\theta(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t)$  와 같이 계산된다.

기억적으로  $p_\theta(x_{t-1}|x_t) \sim N(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$ ,  
 $q(x_0, x_1, \dots, x_T) \equiv \prod_{t=1}^T q(x_t|x_{t-1})$ ,  $q(x_t|x_{t-1}) \equiv N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$   
 $\downarrow$   $x_0$  conditioning of  $q(x_t|x_{t-1})$ ,  $\beta_t$  of forward diffusion process를 계산할 때.

Forward diffusion process  $q(x_t|x_{t-1})$  는 짐작가 텁텁지만, 이것으로 induce the backward process를

$q(x_{t-1}|x_t)$ 은 구할 수가 있다. 하지만  $x_0$ 에 condition 했을 때  $q(x_{t-1}|x_t, x_0)$ 은 구할 수가 없다.

backward model  
and diffusion process

rational Bound

$$\begin{aligned} E_{x_0 \sim q(x_0)} [-\log p_\theta(x_0)] &= E_{x_0 \sim q(x_0)} [-\log \int p_\theta(x_0, x_1, \dots, x_T) dx_1 dx_2 \dots dx_T] \\ &= E_{x_0 \sim q(x_0)} [-\log \int p_\theta(x_0, x_1, \dots, x_T) \frac{q(x_1, x_2, \dots, x_T)}{q(x_1, x_2, \dots, x_T)} dx_1 dx_2 \dots dx_T] \\ &= E_{x_0 \sim q(x_0)} [-\log E_{x_1, x_2, \dots, x_T \sim q(x_1, x_2, \dots, x_T)} [\frac{p_\theta(x_0, x_1, \dots, x_T)}{q(x_1, x_2, \dots, x_T)}]] \\ &\leq E_{x_0 \sim q(x_0)} E_{x_1, x_2, \dots, x_T \sim q(x_1, x_2, \dots, x_T)} [-\log \frac{p_\theta(x_0, x_1, \dots, x_T)}{q(x_1, x_2, \dots, x_T)}] \\ &= E_{x_0 \sim q(x_0)} [-\log p_\theta(x_0)] \\ &= E_{x_0 \sim q(x_0)} [\log p(x_0) p_\theta(x_{t-1}|x_0) \dots p_\theta(x_1|x_0) p_\theta(x_0|x_1)] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log p_\theta(x_t|x_{t-1})] \quad (19) \text{ or } (20) \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \frac{1}{2} \log p_\theta(x_{t-1}|x_t) - \log p_\theta(x_t|x_{t-1})] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \frac{1}{2} \log p_\theta(x_{t-1}|x_t) - \log p_\theta(x_t|x_{t-1})] \\ &\quad \cdot q(x_t|x_{t-1}, x_0) \because (x_t \perp x_0 \text{ given } x_{t-1}) \\ &\quad \cdot \frac{1}{2} \log p_\theta(x_t|x_{t-1}) \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log p_\theta(x_t|x_{t-1}) - \log p_\theta(x_t|x_{t-1})] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} - \log p_\theta(x_t|x_{t-1})] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} - \log p_\theta(x_t|x_{t-1})] \\ &\quad \cdot \frac{1}{2} \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} = \frac{1}{2} \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} - \log p_\theta(x_t|x_{t-1})] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} - \log p_\theta(x_t|x_{t-1})] \\ &= E_{x_0 \sim q(x_0)} [-\log p(x_0) - \sum_{t=1}^T \log \frac{p_\theta(x_t|x_{t-1})}{q(x_t|x_{t-1})} - \log p_\theta(x_t|x_{t-1})] \\ &\quad \cdot L_T \quad L_{t-1} \quad L_t \quad -a \end{aligned}$$

forward diffusion process  $q(x_t|x_0) = N(x_t; \sqrt{1-\beta_t}x_0, \beta_t I)$   
forward process variance

jump diffusion process  $q(x_t|x_0) = N(x_t; \sqrt{1-\beta_t}x_0, (1-\beta_t)I)$

backward process posterior  $q(x_{t-1}|x_t, x_0) = N(x_{t-1}; \frac{\beta_t}{1-\beta_t}x_0 + \frac{(1-\beta_t)}{1-\beta_t}\sqrt{\beta_t}x_t, \frac{\beta_t(1-\beta_t)}{1-\beta_t}I)$

Others: backward process posterior  $q(x_{t-1}|x_t)$



<https://github.com/sjchoi86/2022-1-deep-learning-applications/>

따라서  $x_t$  라고 reparameterized  $\tilde{x}_t(\epsilon, x_0)$  는  
 $\tilde{x}_t = \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta_t}{\sqrt{1-\beta_t}}\epsilon)$  이 된다.

$\rightarrow L_{t-1} + C = E_{x_0 \sim q(x_0), \epsilon \sim N(0, I)} \left[ \frac{1}{2\sigma^2} \left\| \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta_t}{\sqrt{1-\beta_t}}\epsilon) - \mu_q(x_t|x_0) \right\|^2 \right]$  (1)

2. this is Gaussian due to the conjugacy of Gaussians

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= q(x_t|x_{t-1}, x_0) q(x_{t-1}|x_0) \\ &\quad + N(x_0|a, b) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x_{t-1}-\mu_q(x_{t-1}|x_0))^2} \\ &= \frac{1}{\sqrt{1-\beta_t}} e^{-\frac{1}{2\beta_t}(x_{t-1}-\mu_q(x_{t-1}|x_0))^2} \\ &= N(x_{t-1}; \sqrt{1-\beta_t}x_t, (1-\beta_t)I) \\ &\propto \exp \left( -\frac{1}{2} \left( \frac{(x_{t-1}-\sqrt{1-\beta_t}x_t)^2}{\beta_t} + \frac{(x_t-\sqrt{1-\beta_t}x_{t-1})^2}{1-\beta_t} \right) \right) \\ &= \exp \left( -\frac{1}{2} \left( \frac{1}{\beta_t} x_{t-1}^2 - \frac{2}{\beta_t} \sqrt{1-\beta_t} x_t x_{t-1} + \frac{1-\beta_t}{\beta_t} x_t^2 \right. \right. \\ &\quad \left. \left. + \frac{x_{t-1}^2}{(1-\beta_t)} - \frac{2\sqrt{1-\beta_t}}{(1-\beta_t)} x_t x_{t-1} + \frac{1}{(1-\beta_t)} x_t^2 \right) \right. \\ &\quad \left. - \frac{x_{t-1}^2}{(1-\beta_t)} + \frac{2\sqrt{1-\beta_t} x_t x_{t-1}}{(1-\beta_t)} - \frac{x_t^2}{(1-\beta_t)} \right) \\ &= \exp \left( -\frac{1}{2} \left( \frac{1-\beta_t}{\beta_t} x_{t-1}^2 - \left( \frac{2}{\beta_t} \sqrt{1-\beta_t} x_t + \frac{2\sqrt{1-\beta_t}}{(1-\beta_t)} x_0 \right) x_{t-1} + a \right) \right) \\ &= \frac{1}{\sqrt{b}} \cdot \frac{2\pi}{b} \\ i) \frac{1}{b} = \frac{\partial z}{\partial x} + \frac{1}{(1-\beta_t)x_t} = \frac{\frac{\partial z}{\partial x} - \frac{1-\beta_t}{\beta_t}x_t}{\beta_t(1-\beta_t)x_t} = \frac{1-\beta_t}{\beta_t(1-\beta_t)x_t} \therefore \frac{\partial z}{\partial x} = \frac{\beta_t(1-\beta_t)}{1-\beta_t}x_t \end{aligned}$$

ii)  $\frac{\partial a}{\partial x} = \frac{2}{\beta_t} \sqrt{1-\beta_t} x_t + \frac{x_{t-1}}{(1-\beta_t)} \xrightarrow{\text{부분분석}} \frac{(1-\beta_t)}{\sqrt{1-\beta_t}} \sqrt{1-\beta_t} x_t + \frac{\beta_t \sqrt{1-\beta_t}}{(1-\beta_t)} x_0$

$a_t = \frac{\beta_t(-\sqrt{1-\beta_t})}{1-\beta_t} \left( \frac{\sqrt{1-\beta_t} x_t}{\beta_t} + \frac{\sqrt{1-\beta_t}}{(1-\beta_t)} x_0 \right) \xrightarrow{\text{부분분석}} x_t = \frac{\beta_t(-\sqrt{1-\beta_t})}{1-\beta_t} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0$

$\downarrow x_0 = \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0$

$a_t = \frac{\beta_t(-\sqrt{1-\beta_t})}{1-\beta_t} \left( \frac{\sqrt{1-\beta_t} x_t}{\beta_t} + \frac{\sqrt{1-\beta_t}}{(1-\beta_t)} \left( \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0 \right) \right)$

$= \frac{(1-\beta_t)\sqrt{1-\beta_t}}{1-\beta_t} x_t + \frac{\beta_t \sqrt{1-\beta_t}}{1-\beta_t} \left( \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0 \right)$

$= \frac{(1-\beta_t)\sqrt{1-\beta_t}}{1-\beta_t} x_t + \frac{\beta_t}{1-\beta_t} \left( \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0 \right)$

$= \frac{\beta_t}{1-\beta_t} \left( \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0 \right)$

$= \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\beta_t}{1-\beta_t} \left( \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{\sqrt{1-\beta_t}}{\beta_t} x_0 \right)$

$= \frac{1}{\sqrt{1-\beta_t}} x_t - \frac{1}{\sqrt{1-\beta_t}} \beta_t x_t + \frac{1}{\sqrt{1-\beta_t}} \beta_t x_0 = \frac{1}{\sqrt{1-\beta_t}} (x_t - \beta_t x_t + \beta_t x_0) = \frac{1}{\sqrt{1-\beta_t}} x_0$

따라서  $q(x_{t-1}|x_t, x_0) = N(x_{t-1}; \frac{1}{\sqrt{1-\beta_t}} x_0, \frac{\beta_t}{1-\beta_t} I)$

3. 6

4. 6

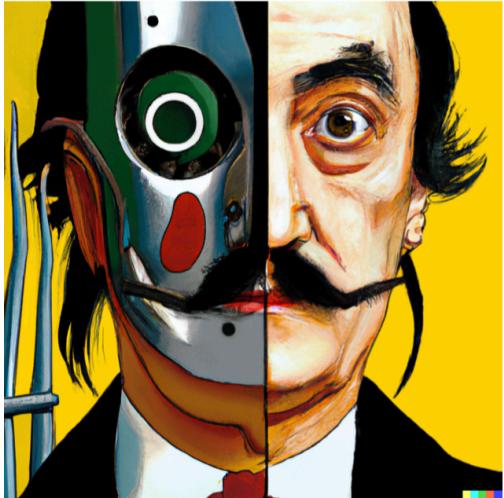
5. 6

6. 6

Lo 미리보아 남은 term은  $Lo = E_{x_0 \sim q(x_0)} [-\log p_\theta(x_0|x_1)]$  ↓  
즉  $x_0$ 에서부터  $x_1$ 까지 sampling 될 reverse process의 결과로 나온 Gaussian을 따르는  $x_0$ 을 실제 데이터의 분포인  $q(x_0)$ 의 Domain으로 보내줘야 한다.  
이 때는 independent discrete denoiser로 연결된다.

# DALL-E2

---



vibrant portrait painting of Salvador Dalí with a robotic half face



a shiba inu wearing a beret and black turtleneck



a close up of a handpalm with leaves growing from it



an espresso machine that makes coffee from human souls, artstation



panda mad scientist mixing sparkling chemicals, artstation



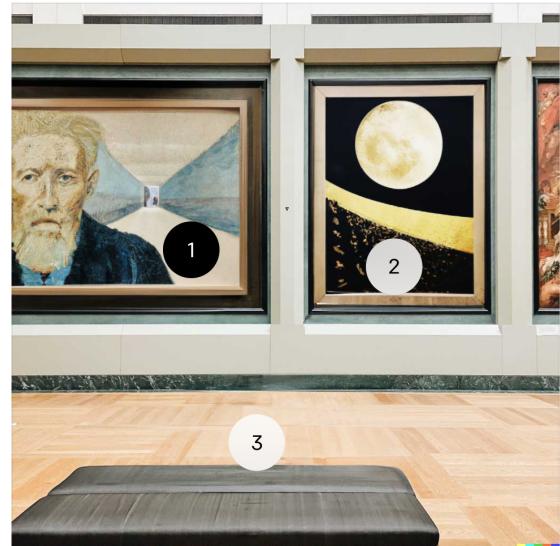
a corgi's head depicted as an explosion of a nebula

<https://openai.com/dall-e-2>

# DALL-E2

---

ORIGINAL IMAGE



DALL-E 2 EDITS



<https://openai.com/dall-e-2>

# Thank you for listening

---