고려대학교
빅데이터 연구회
KU-BIG

Credit Card Fraud Detection
- VAE algorithm

EBB

목 차

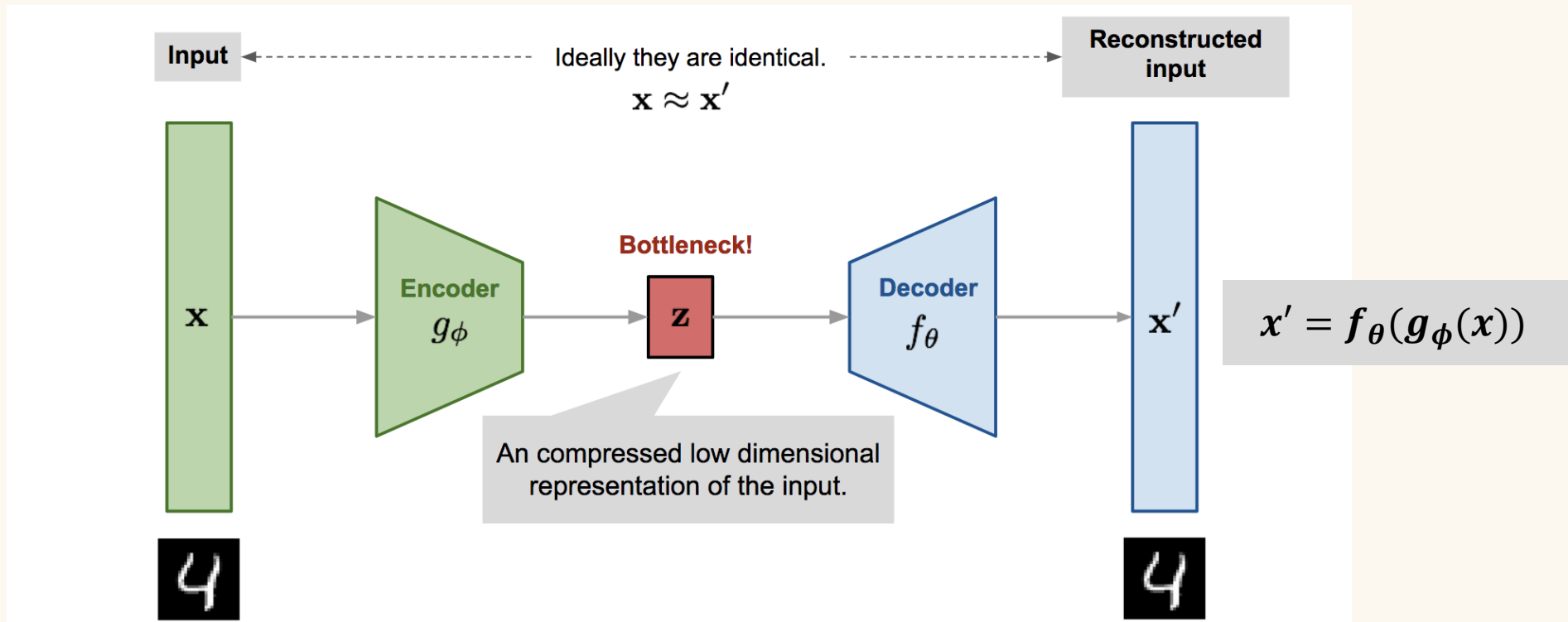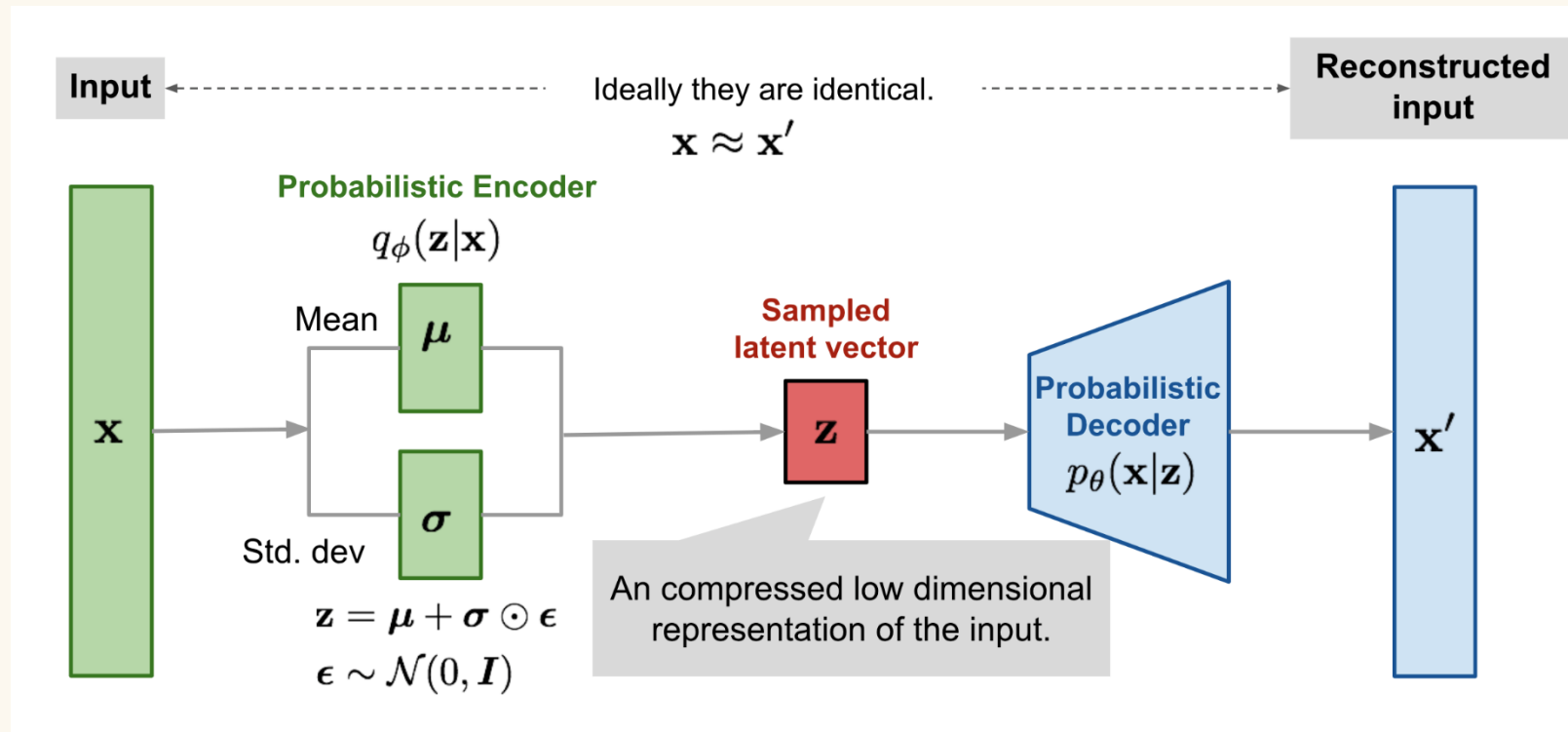# Autoencoder?

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html
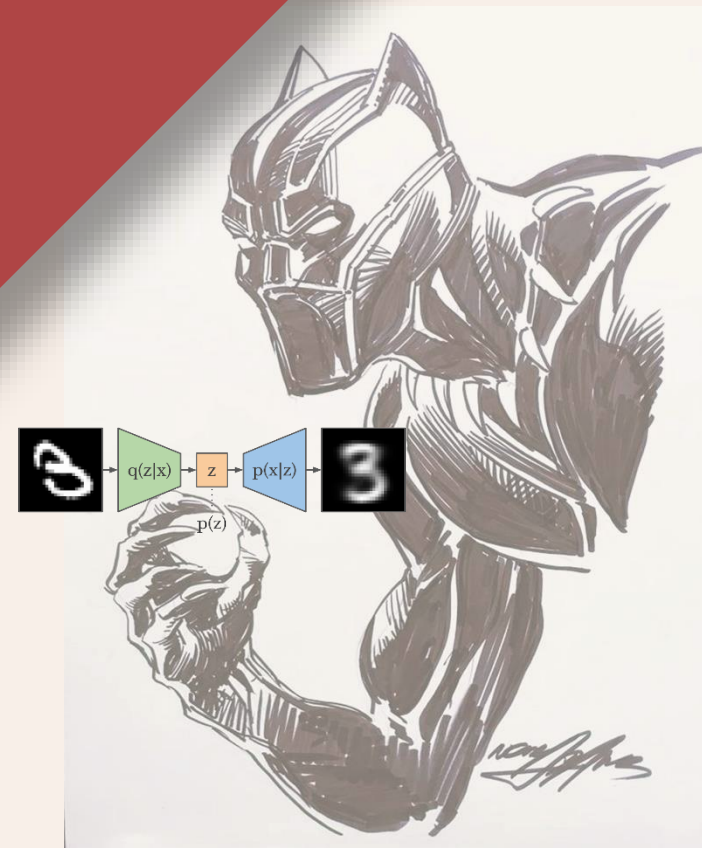
MSE loss function : $L_{AE}(\phi, \theta) = \frac{1}{n}\sum_{i=1}^{n} (x^{(i)} - f_\theta\left(g_\phi(x^{(i)})\right))^2$

Variational Autoencoder (VAE)?

Input ← Ideally they are identical. → **Reconstructed input**

$$\mathbf{x} \approx \mathbf{x}'$$

Probabilistic Encoder

$$q_\phi(\mathbf{z}|\mathbf{x})$$

Mean $\boldsymbol{\mu}$

$\mathbf{x}$

Std. dev $\boldsymbol{\sigma}$

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

Sampled latent vector

$\mathbf{z}$

An compressed low dimensional representation of the input.

Probabilistic Decoder

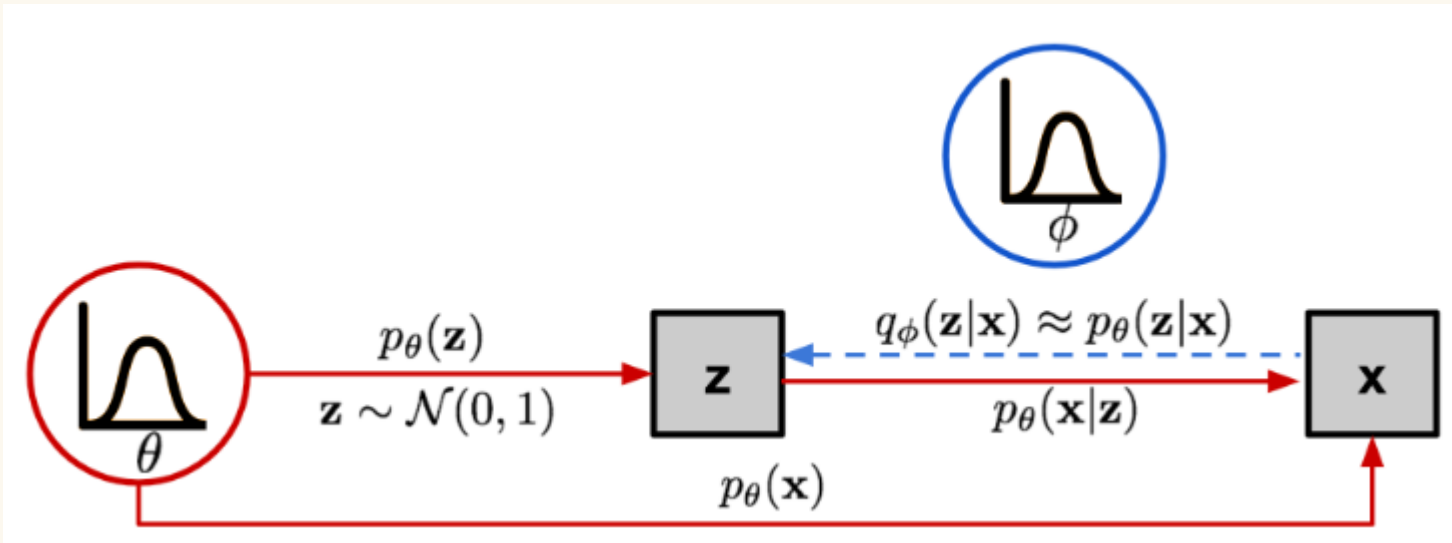$$p_\theta(\mathbf{x}|\mathbf{z})$$

$\mathbf{x}'$

〈Decoder〉

θ^* : true parameter

1. $p_{\theta^*}(\mathbf{z})$에서 $\mathbf{z}^{(i)}$ 추출.
2. $p_{\theta^*}(\mathbf{x}|\mathbf{z}^{(i)})$에서 $\mathbf{x}^{(i)}$ 생성

Optimal parameter : $\theta^* = \arg\max_\theta \sum_{i=1}^{n} \log \boxed{p_\theta(\mathbf{x}^{(i)})} \longrightarrow$ **Intractable !!**

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Variational Autoencoder (VAE)?

$\mathbf{z}$: latent variable

$p_\theta(\mathbf{z})$: prior

$p_\theta(\mathbf{x}|\mathbf{z})$: Likelihood

$p_\theta(\mathbf{z}|\mathbf{x})$: posterior

알려져 있는 분포 활용

$p_\theta(\mathbf{z})$를 추론하고자 $p_\theta(\mathbf{z}|\mathbf{x})$를 이용하고, $p_\theta(\mathbf{z}|\mathbf{x})$를 추론하고자 $q_\phi(\mathbf{z}|\mathbf{x})$를 활용한다.

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html
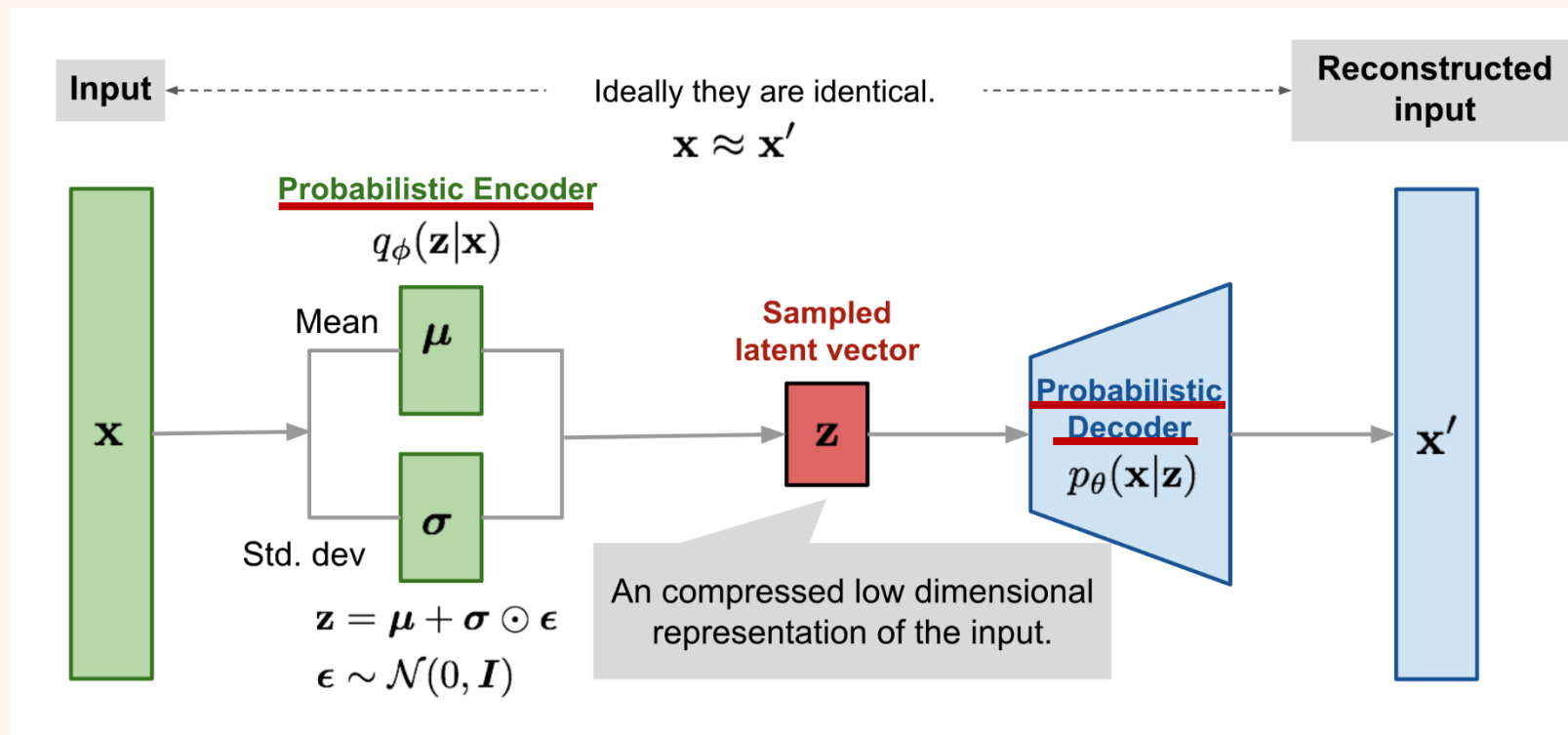
Variational Autoencoder (VAE)?

$p_\theta(\mathbf{z})$를 추론하고자 $p_\theta(\mathbf{z}|\mathbf{x})$를 이용하고, $p_\theta(\mathbf{z}|\mathbf{x})$를 추론하고자 $q_\phi(\mathbf{z}|\mathbf{x})$를 활용한다.

Variational Inference
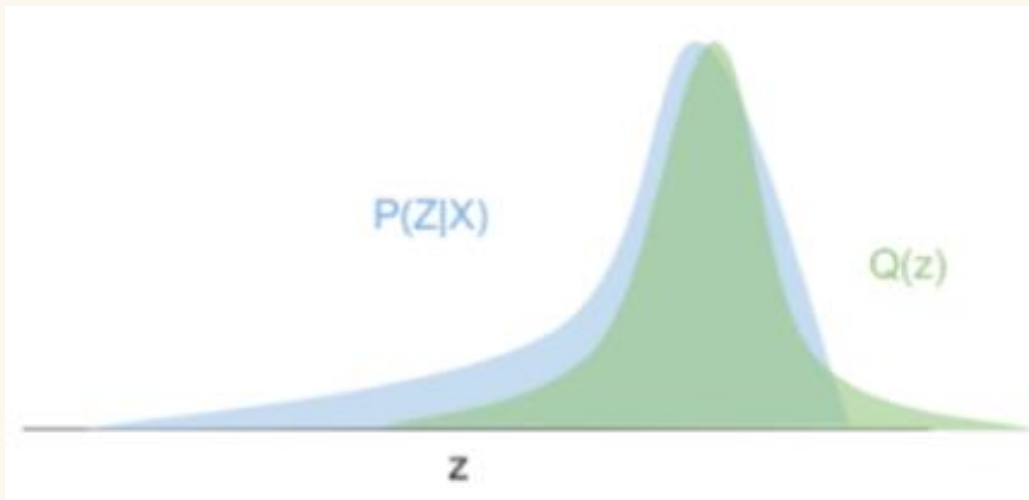
1. $p_\theta(\mathbf{x})$를 계산하기 힘든 경우

2. $p_\theta(\mathbf{z}), p_\theta(\mathbf{x}|\mathbf{z})$ 를 더 복잡하게 모델링하고 싶은 경우

https://ratsgo.github.io/generative%20model/2017/12/19/vi/

Variational Autoencoder (VAE)?

Input ← - - - - - Ideally they are identical. - - - - - → **Reconstructed input**

$$\mathbf{x} \approx \mathbf{x}'$$

Probabilistic Encoder

$$q_\phi(\mathbf{z}|\mathbf{x})$$

Mean $\boldsymbol{\mu}$

Sampled latent vector

Probabilistic Decoder

$$p_\theta(\mathbf{x}|\mathbf{z})$$

$\mathbf{x}$ $\mathbf{z}$ $\mathbf{x}'$

Std. dev $\boldsymbol{\sigma}$

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

An compressed low dimensional representation of the input.

Loss function :

$$L_{\text{VAE}}(\theta, \phi) = -\log p_\theta(\mathbf{x}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$$
$$= -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} p_\theta(\mathbf{x}|\mathbf{z}) + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}))$$
$$\theta^*, \phi^* = \arg\min_{\theta, \phi} L_{\text{VAE}}$$

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

$L_{VAE}(\theta, \phi)$ 뜯어보기.

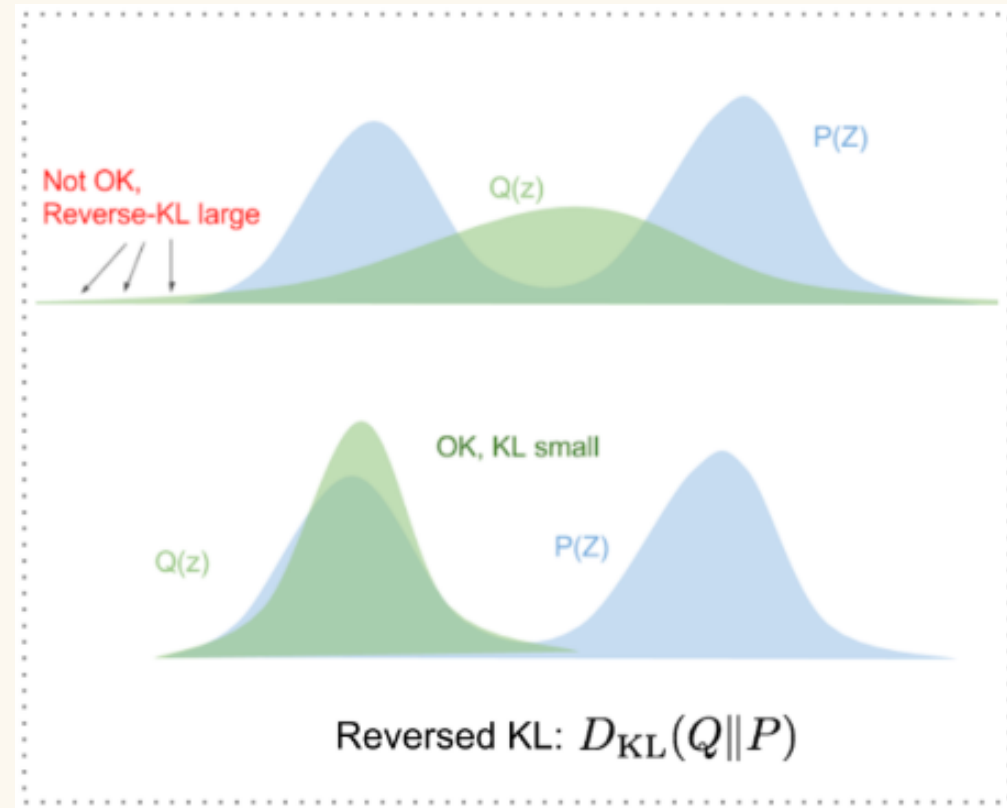$$L_{\mathrm{VAE}}(\theta, \phi) = -\log p_\theta(\mathbf{x}) + \underline{D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))}$$
$$= -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} p_\theta(\mathbf{x}|\mathbf{z}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}))$$
$$\theta^*, \phi^* = \arg\min_{\theta,\phi} L_{\mathrm{VAE}}$$

$q_\phi(\mathbf{z}|\mathbf{x})$는 $p_\theta(\mathbf{z}|\mathbf{x})$에 최대한 비슷해야 한다.

→ | KL–Divergence |

$$D_{\mathrm{KL}}(Q\|P) = \mathbb{E}_{z\sim Q(z)} \log \frac{Q(z)}{P(z)} \geq 0$$

→ **Minimize** $D_{\mathrm{KL}}(Q\|P)$

Not OK, Reverse-KL large

P(Z) Q(z)

OK, KL small

Q(z) P(Z)

Reversed KL: $D_{\mathrm{KL}}(Q\|P)$

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$$

$$= \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})} d\mathbf{z}$$

$$= \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})p_\theta(\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} d\mathbf{z}$$

$$= \int q_\phi(\mathbf{z}|\mathbf{x}) \left(\log p_\theta(\mathbf{x}) + \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} \right) d\mathbf{z}$$

$$= \log p_\theta(\mathbf{x}) + \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z},\mathbf{x})} d\mathbf{z}$$

$$= \log p_\theta(\mathbf{x}) + \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})} d\mathbf{z}$$

$$= \log p_\theta(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z})} - \log p_\theta(\mathbf{x}|\mathbf{z}) \right]$$

$$= \log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})$$

$$D_{\mathrm{KL}}(Q\|P) = \mathbb{E}_{z\sim Q(z)} \log \frac{Q(z)}{P(z)}$$

; Because $p(z|x) = p(z,x)/p(x)$

; Because $\int q(z|x) dz = 1$

$g(\mathbf{z})$

; Because $p(z,x) = p(x|z)p(z)$

$\mathbb{E}_{z\sim q_\phi(\mathbf{z}|\mathbf{x})}[g(\mathbf{z})]$

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

$$D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})\right) = \log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\log p_\theta(\mathbf{x}|\mathbf{z})$$

$$\log p_\theta(\mathbf{x}) - D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})\right) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\log p_\theta(\mathbf{x}|\mathbf{z}) - D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right)$$

$$\geq 0$$

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\log p_\theta(\mathbf{x}|\mathbf{z}) - D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right) = -L_{\mathrm{VAE}}$$

Lower bound of $\log p_\theta(\mathbf{x})$ ⟶ **Maximize**

$$L_{\mathrm{VAE}}(\theta,\phi) = -\log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})\right)$$
$$= -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}p_\theta(\mathbf{x}|\mathbf{z}) + D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right)$$
$$\theta^*,\phi^* = \arg\min_{\theta,\phi} L_{\mathrm{VAE}}$$

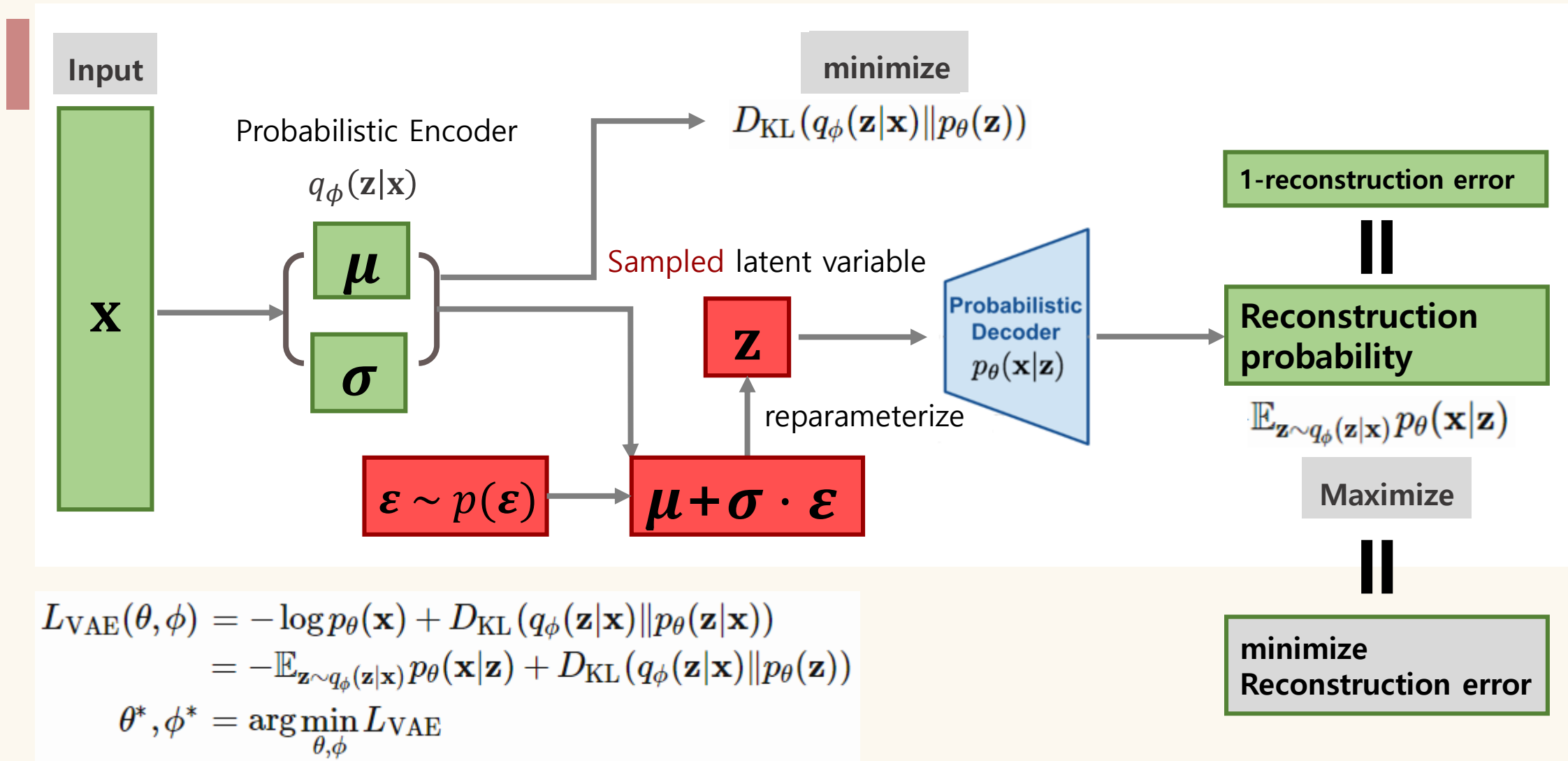**Reconstruction probability**

Minimize Loss -> Maximize real data sample을 생성하는 확률의 lower bound

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Reparameterization trick

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\boldsymbol{I})$$

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

$\odot$ refers to element-wise product.

Stochastic process(확률과정)라
Backprop이 불가능한 z를
미분 가능하게 만들어서 (수식화하여)
Backprop을 가능하게 만든다!!!!!!!!!!!!!!

Original form

Reparameterised form

Backprop

$\partial f/\partial z_j$

$\partial f/\partial \varphi_i$

$\simeq \partial L/\partial \varphi_i$

$\sim q(z|\phi,x)$

$= g(\phi,x,\epsilon)$

$\sim p(\epsilon)$

◇ : Deterministic node

● : Random node

[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Wakanda Forever

Input

Probabilistic Encoder

$q_\phi(\mathbf{z}|\mathbf{x})$

minimize

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}))$$

1-reconstruction error

$\boldsymbol{\mu}$

$\boldsymbol{\sigma}$

$\mathbf{x}$

Sampled latent variable

z

Probabilistic Decoder

$p_\theta(\mathbf{x}|\mathbf{z})$

reparameterize

$\boldsymbol{\varepsilon} \sim p(\boldsymbol{\varepsilon})$

$\boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\varepsilon}$

Reconstruction probability

$$\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\, p_\theta(\mathbf{x}|\mathbf{z})$$

Maximize

$$L_{\mathrm{VAE}}(\theta,\phi) = -\log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$$
$$= -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\, p_\theta(\mathbf{x}|\mathbf{z}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}))$$
$$\theta^*, \phi^* = \arg\min_{\theta,\phi} L_{\mathrm{VAE}}$$

minimize Reconstruction error

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Wakanda Forever

Algorithm 3 Variational autoencoder training algorithm

INPUT: Dataset $x^{(1)}, \cdots, x^{(N)}$

OUTPUT: probabilistic encoder f_ϕ, probabilistic decoder g_θ

$\quad \phi, \theta \leftarrow$ Initialize parameters

$\quad$ **repeat**

$\qquad$ **for** i=1 **to** N **do**

$\qquad\quad$ Draw L samples from $\epsilon \sim \mathcal{N}(0, 1)$

$\qquad\quad z^{(i,l)} = h_\phi(\epsilon^{(i)}, x^{(i)}) \quad i = 1, \cdots, N$ $\leftarrow$ Reparameterization trick

$\qquad$ **end for**

$\qquad E = \sum_{i=1}^{N} -D_{KL}(q_\phi(z|x^{(i)})\|p_\theta(z)) + \frac{1}{L}\sum_{l=1}^{L}(\log p_\theta(x^{(i)}|z^{(i,l)}))$ $\leftarrow$ loss

$\qquad \phi, \theta \leftarrow$ Update parameters using gradients of E (e.g. Stochastic Gradient Descent)

$\quad$ **until** convergence of parameters ϕ, θ

Jinwon An, Sungzoon Cho, (2015). Variational Autoencoder based Anomaly Detection using Reconstruction Probability

Algorithm 4 Variational autoencoder based anomaly detection algorithm

INPUT: Normal dataset X,   Anomalous dataset $x^{(i)}$ $i = 1, \cdots, N$,   threshold α

OUTPUT: reconstruction probability $p_\theta(x|\hat{x})$

$\phi, \theta \leftarrow$ train a variational autoencoder using the normal dataset X

for i=1 to N **do**

    $\boldsymbol{\mu}_{z^{(i)}}, \boldsymbol{\sigma}_{z^{(i)}} = f_\theta(z|x^{(i)})$

    draw L samples from $z \sim \mathcal{N}(\boldsymbol{\mu}_{z^{(i)}}, \boldsymbol{\sigma}_{z^{(i)}})$

    for l=1 to L **do**

      $\boldsymbol{\mu}_{\hat{x}^{(i,l)}}, \boldsymbol{\sigma}_{\hat{x}^{(i,l)}} = g_\phi(x|z^{(i,l)})$

    end for

    $reconstruction\ probability(i) = \frac{1}{L}\sum_{l=1}^{L} p_\theta(x^{(i)}|\boldsymbol{\mu}_{\hat{x}^{(i,l)}}, \boldsymbol{\sigma}_{\hat{x}^{(i,l)}})$

    if $reconstruction\ probability(i) < \alpha$ **then**

      $x^{(i)}$ is an anomaly

    else

      $x^{(i)}$ is not an anomaly

    end if

end for

Jinwon An, Sungzoon Cho, (2015). Variational Autoencoder based Anomaly Detection using Reconstruction Probability

---

Encoder로부터 z분포에 대한 parameter 추출

Decoder로부터 x̂분포에 대한 (reconstruction) parameter 추출

reconstruction parameter가 주어졌을 때, original input x가 생성될 확률의 평균
Reconstruction error = 1- reconstruction probability

Wakanda Forever

```python
class VAE(nn.Module):
    def __init__(self, image_size = 784, h_dim = 400, z_dim = 20):
        super(VAE, self).__init__()
        self.fc1 = nn.Linear(image_size, h_dim)
        self.fc2 = nn.Linear(h_dim, z_dim)
        self.fc3 = nn.Linear(h_dim, z_dim)
        self.fc4 = nn.Linear(z_dim, h_dim)
        self.fc5 = nn.Linear(h_dim, image_size)

    def encode(self, x):
        h = F.relu(self.fc1(x))
        return self.fc2(h), self.fc3(h)
    def reparameterize(self, mu, log_var):
        std = torch.exp(log_var/2)
        eps = torch.randn_like(std)
        return mu+eps*std
    def decode(self, z):
        h = F.relu(self.fc4(z))
        return self.fc5(h)

    def forward(self, x):
        mu, log_var = self.encode(x)
        z = self.reparameterize(mu, log_var)
        x_reconst = self.decode(z)
        return x_reconst, mu, log_var

model = VAE().to(device)
optimizer = torch.optim.Adam(model.parameters(), lr = learning_rate)
```
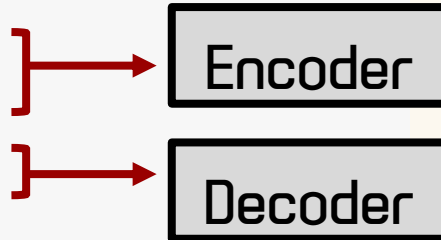
Encoder

Decoder

z_dim : parameter쌍의 개수

fc2, fc3는 mu와 sigma를 도출함.

fc2, fc3 중에 어느 것이 mu인지,

sigma인지는 알 수 없음.

빛희정 님의 은혜로운 코드. https://github.com/KU-BIG/novelty-detection/blob/master/novelty_detection_pytorch.ipynb