Auto-Encoding Variational Bayes

Kingma, D. P. & Welling, M. Auto-encoding variational Bayes.
In *International Conference on Learning Representations* (2014)

2025년 7월 24일

이규원

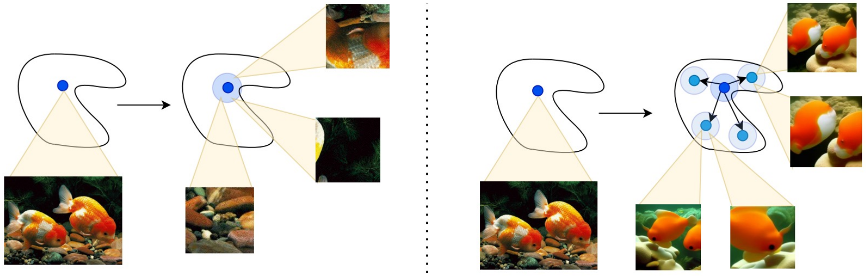
Department of Computer Science and Engineering
Chung-Ang University

Index

- Backgrounds
- Motivation
- Methods
- Experimental Results
- Conclusions

Backgrounds

- **Generative model**
  - Models that learn the data generation process and generate new samples
- **Applications of generative models**
  - Creative work (e.g., art, music)
  - Data augmentation – when the dataset is small
  - Representation learning – when labeled data is scarce
  - Anomaly detection – when decision boundaries are unclear



a) Standard Augmentations

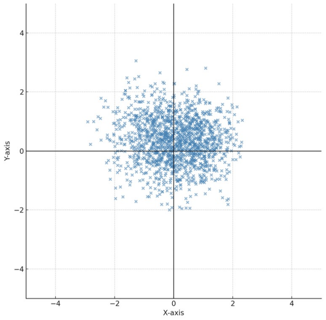
b) Generative Augmentations

Ayromlou, Sana, et al. "Can Generative Models Improve Self-Supervised Representation Learning?." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. No. 2. 2025.

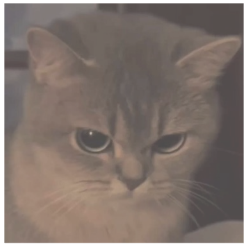
Backgrounds

- **Example: Auto-Encoding Variational Bayes**
  - Learn the data generation process based on probability distributions
  - Estimate the latent probability distribution of training data

Train Dataset



Latent Space

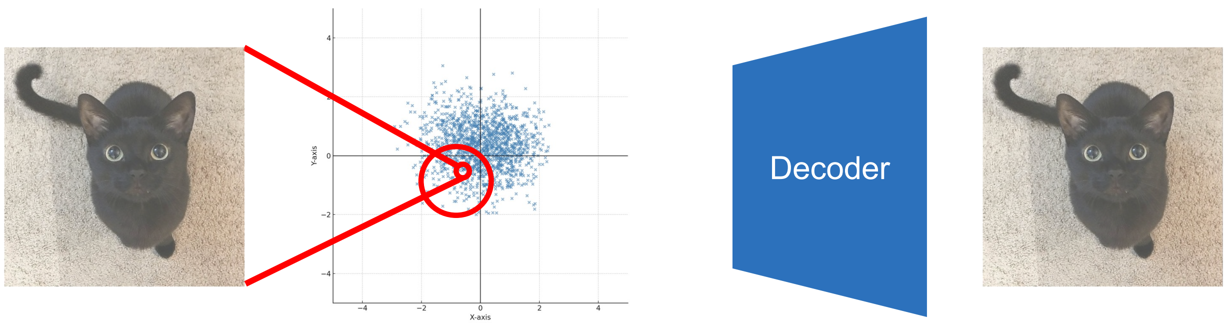


Generated Sample

Motivation

- **Challenges in learning**
  - Train the decoder to make training data highly probable (maximize p(x))
  - Since the posterior p(z|x) is intractable, the input to the decoder cannot be obtained

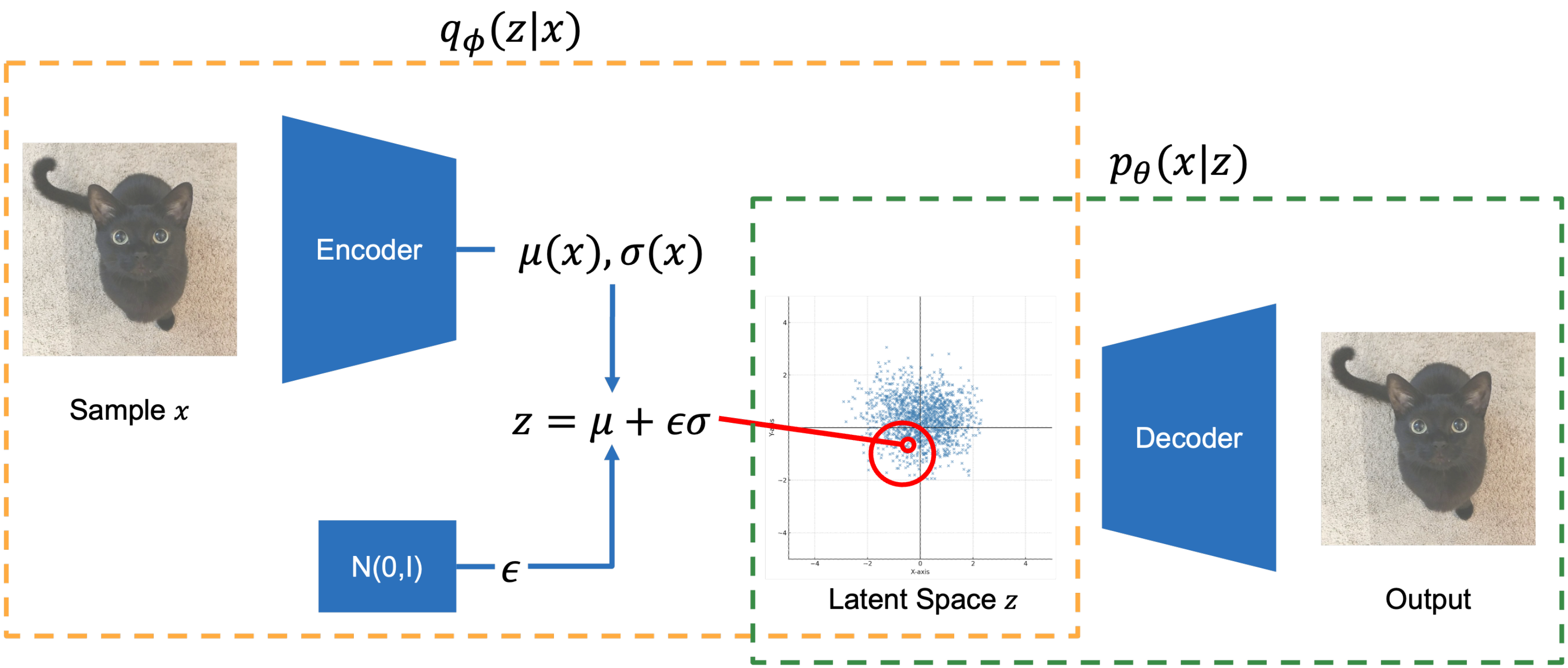
- **Limitation of previous methods**
  - Encoder and decoder are trained separately, leading to unstable training
  - Approximate p(z|x) by sampling from p(x|z)p(z), which is computationally expensive
  - Assume a tractable approximate posterior q(z|x), but the approximation quality is limited



Train Sample   Latent Space   Train Sample

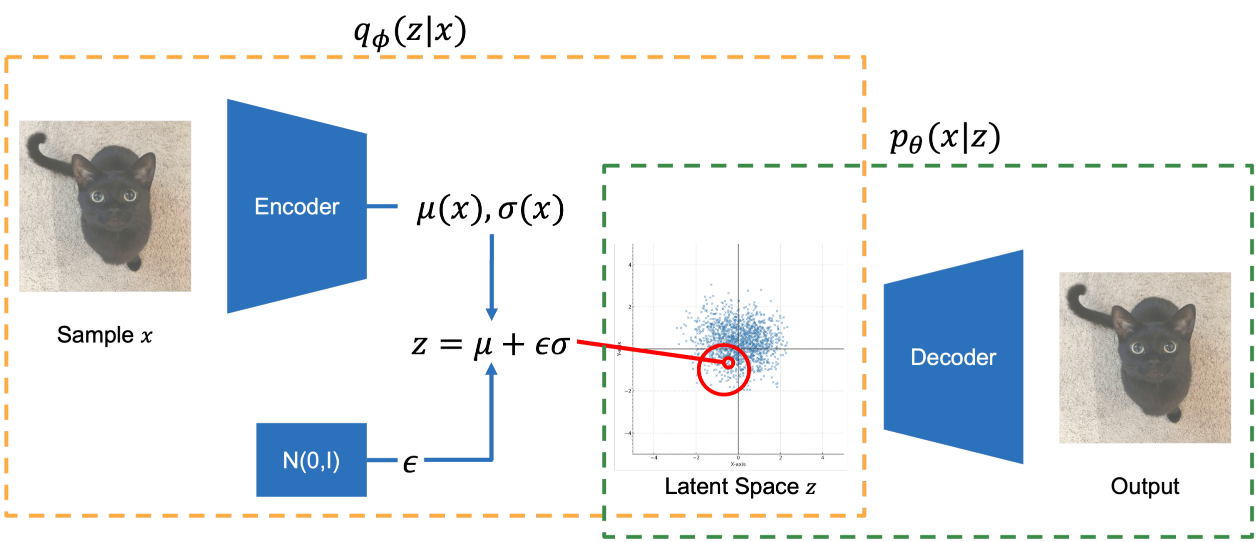
Methods - Overview

- **Idea**
  - Use an encoder to sample from p(z|x) without explicitly estimating it.
  - Use the reparameterization trick to enable end-to-end training of the encoder and decoder



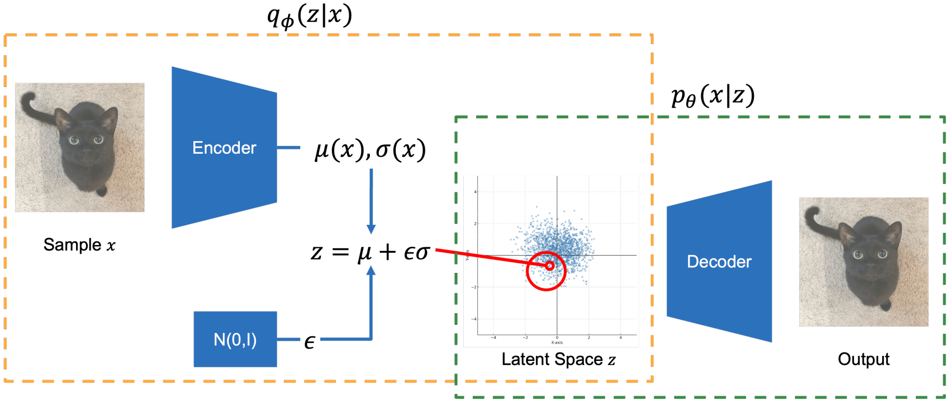
Methods – Encoder

- **Learning objective**
  - Learns to predict the parameters (mean and variance) of the $q_\phi(z|x^{(i)})$ for each input $x^{(i)}$
- **Reparameterization trick**
  - Sampling from $z \sim q_\phi(z|x^{(i)})$ is non-differentiable w.r.t. ϕ
  - Decompose the stochastic component ϵ, and define $z = g_\phi(\epsilon, x)$ as a deterministic function

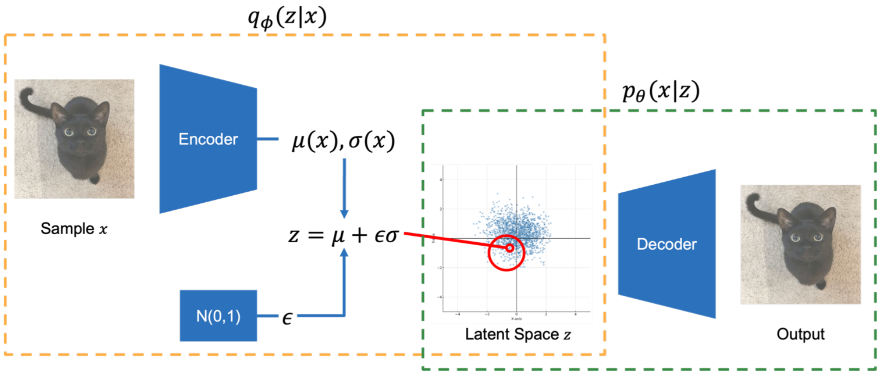


Methods - Decoder

- **Training phase**
  - Use z from the reparameterization trick as input
  - Decoder estimates the parameters of the probability distribution for each pixel intensity
    - e.g., Bernoulli or Gaussian distribution
  - Likelihood is calculated from the distribution over pixel intensities
- **Generation phase**
  - Sample z from the prior distribution
  - Estimate parameters of the pixel intensity distribution
  - Sample pixel values from the predicted distributions



Methods – ELBO



- **ELBO derivation**

$$\mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right]$$

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

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

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

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

Reconstruction Regularization ≥ 0

Methods

- **ELBO computation**
  - $\mathbb{E}_z\left[\log p_\theta(x^{(i)}|z)\right] - D_{KL}\left(q_\phi(z|x^{(i)}) \parallel p_\theta(z)\right) + D_{KL}\left(q_\phi(z|x^{(i)}) \parallel p_\theta(z|x^{(i)})\right)$

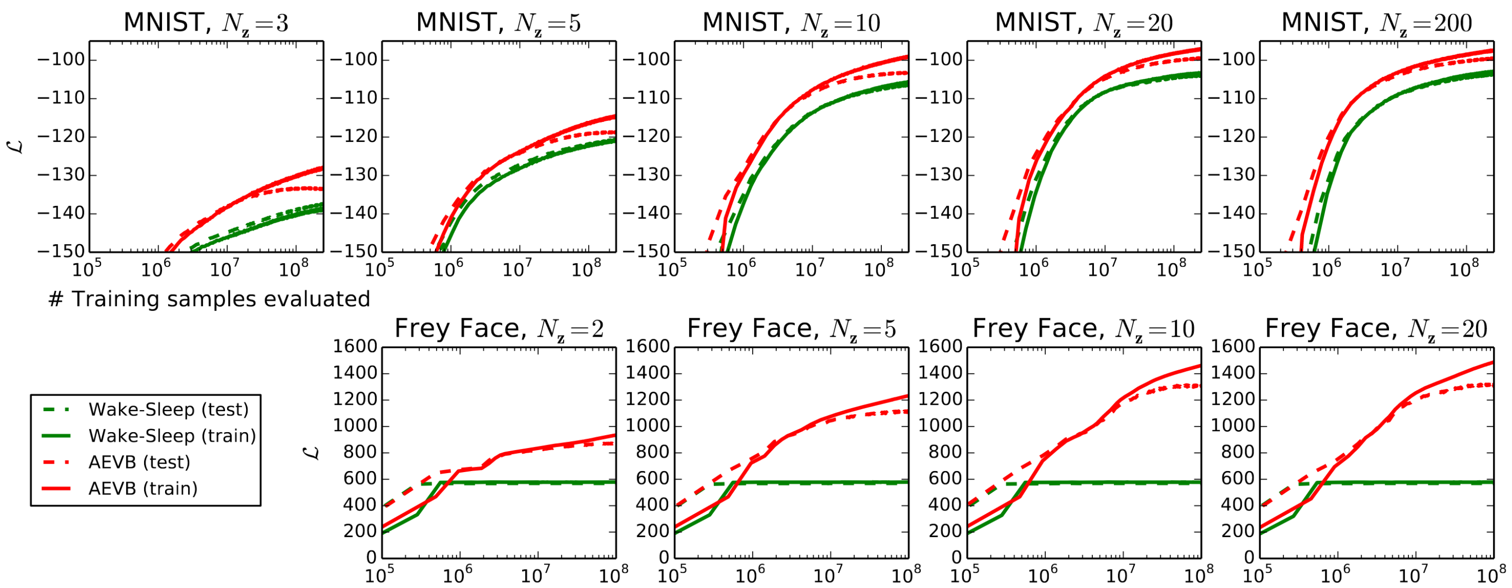
    Reconstruction　　Regularization　　≥ 0
  - Reconstruction term

    $$\frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x^{(i)} \mid z^{(i,l)}), \quad \text{where } z^{(i,l)} = g_\phi(\epsilon^{(l)}, x^{(i)}), \ \epsilon^{(l)} \sim \mathcal{N}(0, I)$$

    - For each input $x^{(i)}$, draw L samples $z^{(i,l)} \sim q_\phi(z|x^{(i)})$
    - In practice, set L = 1 and use minibatch size M = 100
  - Regularization term

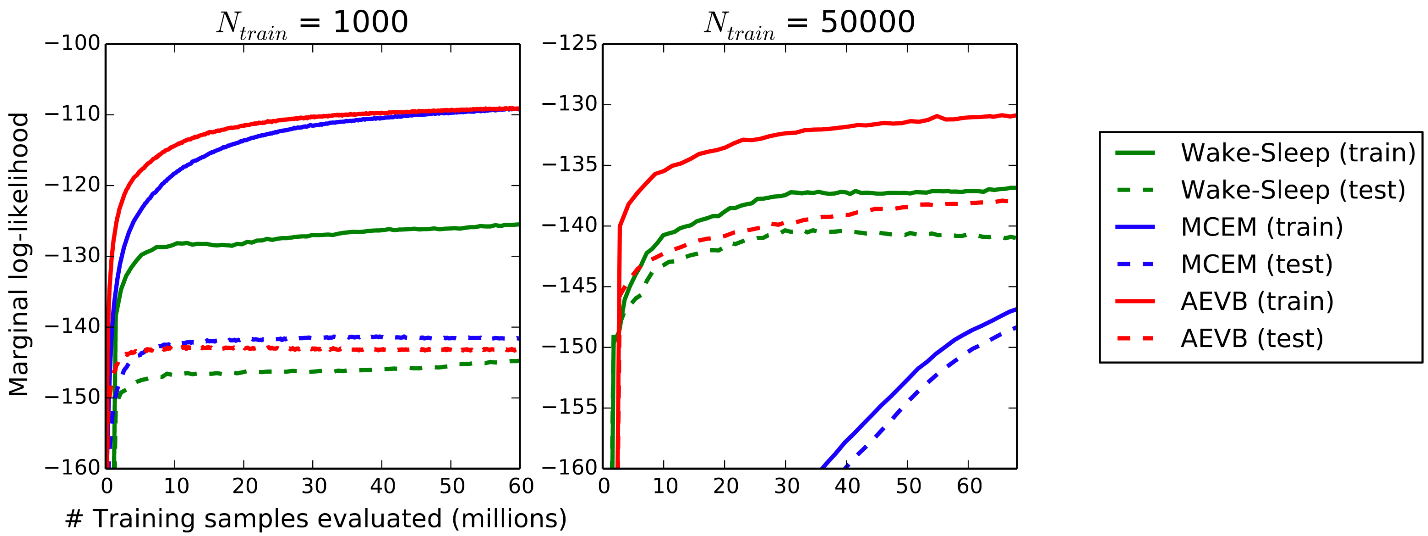
    $$D_{\mathrm{KL}}\left(\mathcal{N}(\mu, \sigma^2) \parallel \mathcal{N}(0,1)\right) = \frac{1}{2}\sum_{j=1}^{d}\left(\sigma_j^2 + \mu_j^2 - 1 - \log \sigma_j^2\right)$$

Experimental Results



- **Likelihood lower bound**
  - x: # of training samples evaluated / y: variational lower bound
  - Wake-Sleep trains the decoder and encoder separately
  - Across all datasets, AEVB consistently achieved a higher variational lower bound
  - AEVB remained stable with higher latent dimensions on Frey Face.

Experimental Results



- **Marginal likelihood (AEVB vs Wake-Sleep vs MCEM)**
  - AEVB dominates in convergence speed and final performance across dataset sizes
  - MCEM may exceed AEVB in small-scale settings, but becomes impractical in large-scale training
  - Wake-Sleep is fast but consistently underperforms in marginal likelihood

Conclusion

- **Reparameterization trick enables end-to-end training with backpropagation**
- **Encoder approximates the posterior p(z|x) for efficient, scalable inference**
- **AEVB(VAE) unifies probabilistic generative modeling and deep learning**