AI504: Programming for Artificial Intelligence

Week 5: Variational Autoencoder

Edward Choi
Grad School of AI
edwardchoi@kaist.ac.kr

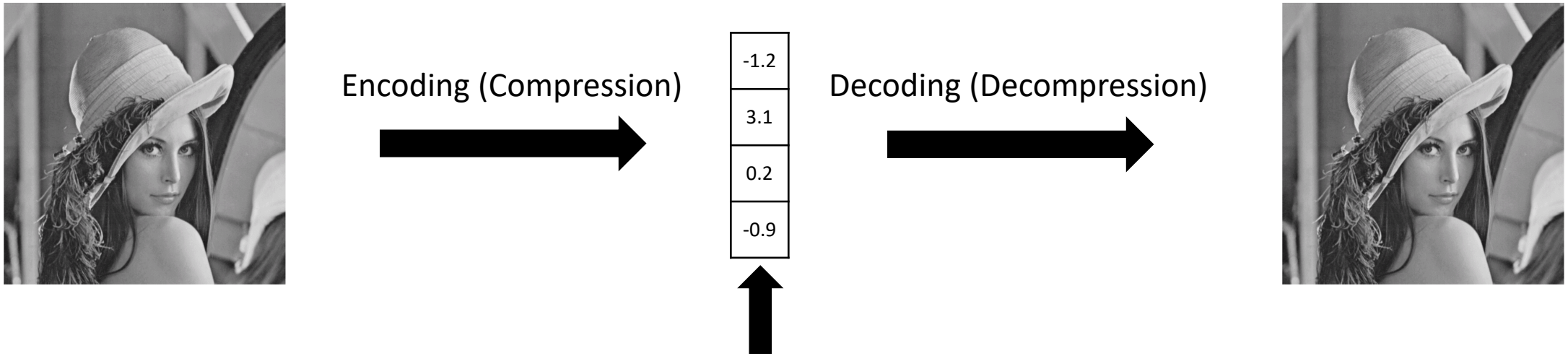
Today’s Topic

- Generative Autoencoder
- Variational Inference
- Variational Autoencoder
- Training VAE
  - Reparametrization Trick

Generative Autoencoder

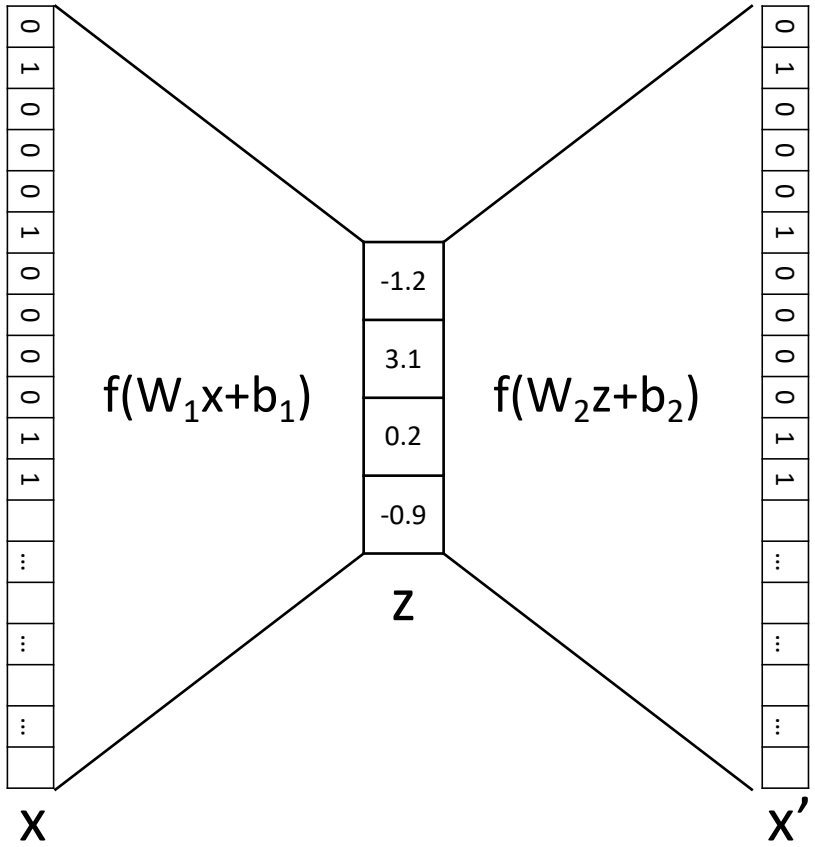
Autoencoder

- Consists of Encoder and Decoder



Autoencoder

- Mean Squared Error (MSE) loss

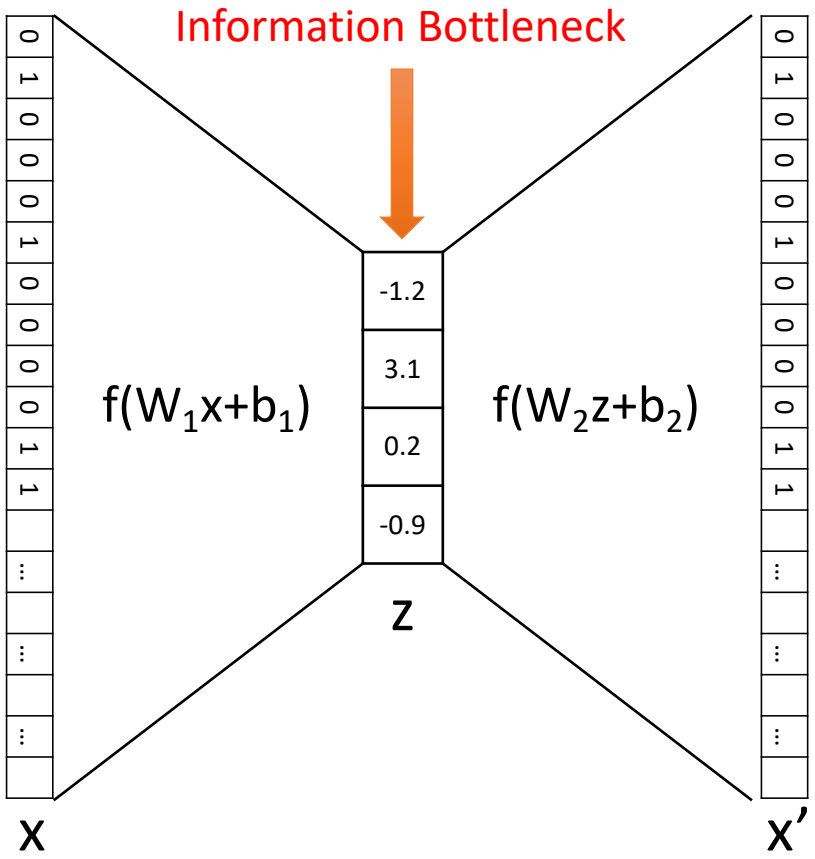


- Encoding
  - $z = f(W_1x+b_1)$
- Decoding
  - $x' = f(W_2z+b_2)$
- Loss
  - $\mathcal{L}(x, x') = \|x - x'\|_2^2$

    (Squared Error)

Autoencoder

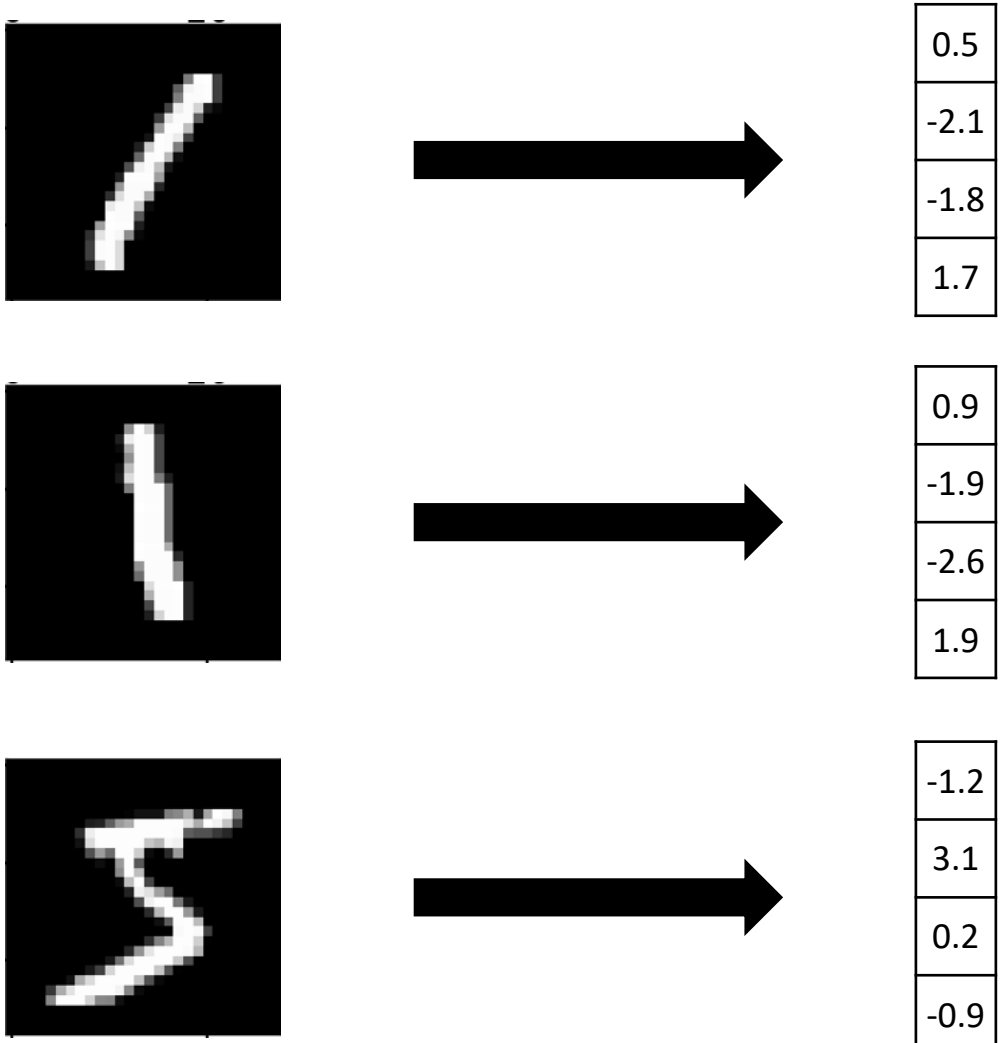
- Compression



Need to pack all information in 4-D

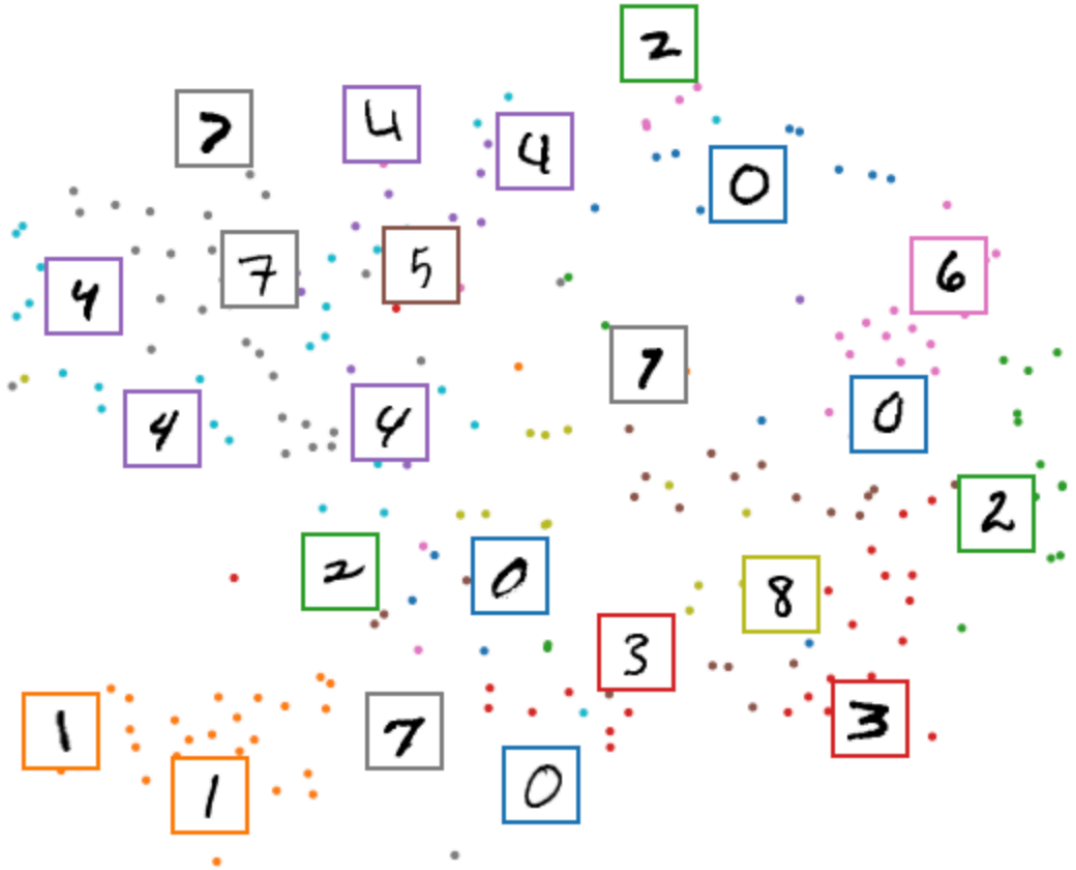
➔ Need to learn some useful hidden features

Latent Representations



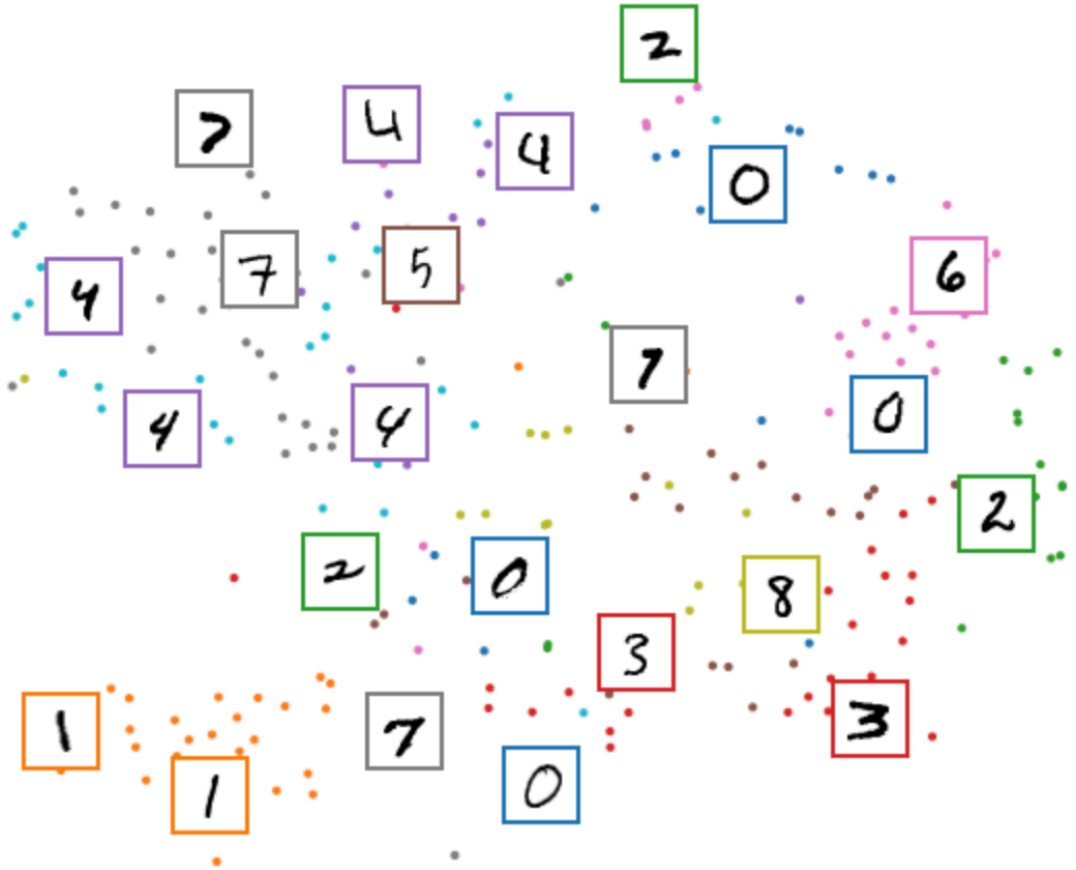
Latent Representation

- Digits are compressed to a 4-D space.



Generating Samples

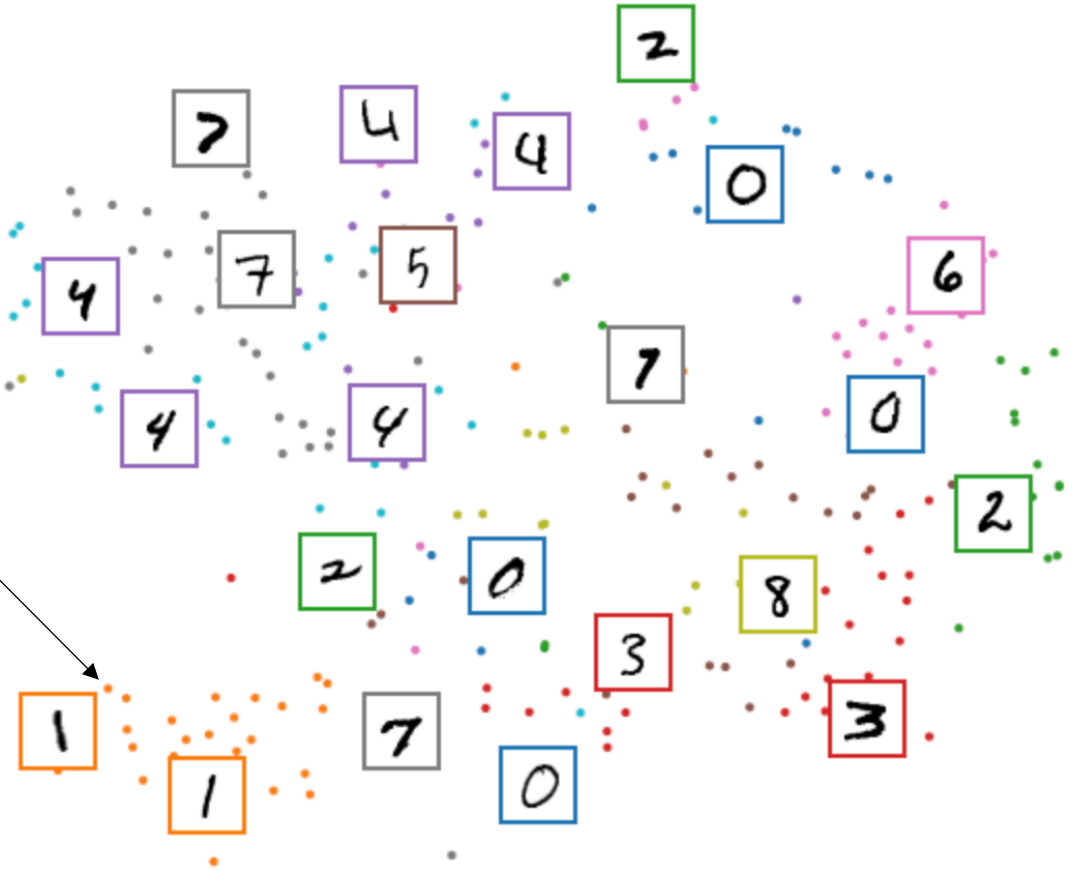
- How can we generate new samples?



Generating New Samples

- Perturb an existing **z**?

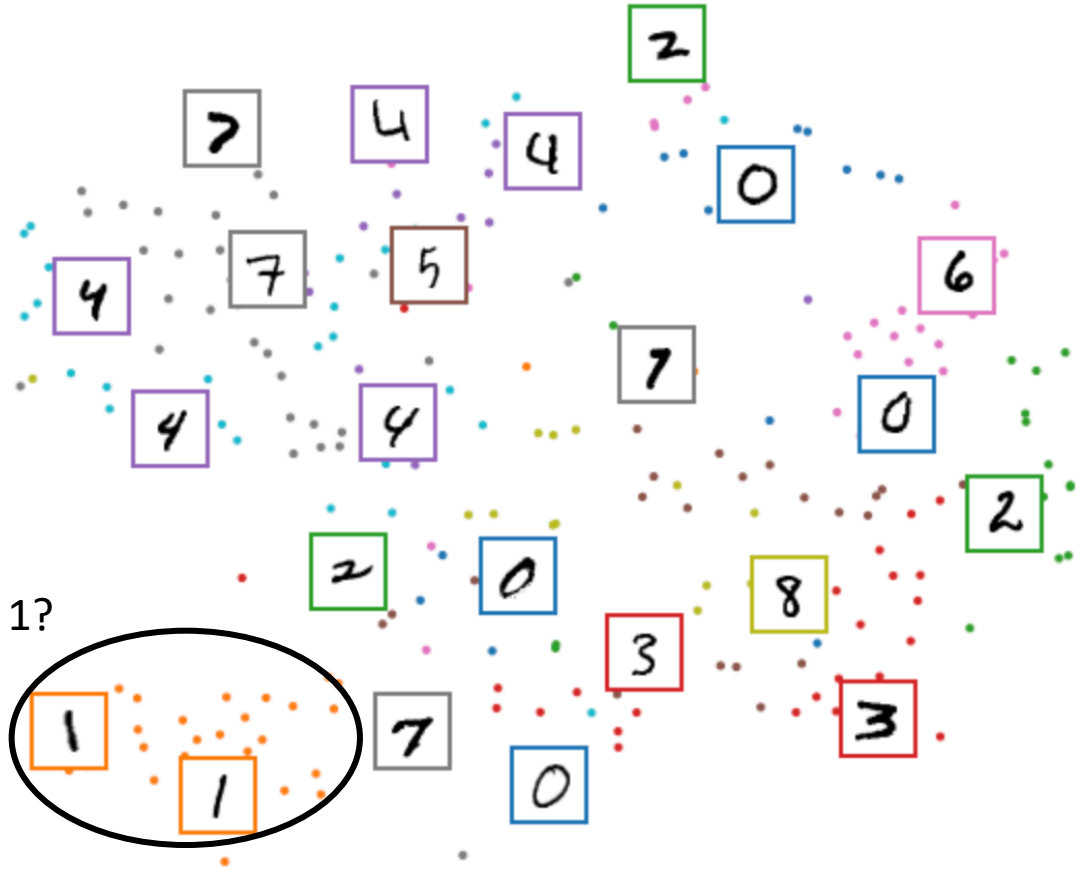
Perturb this sample to generate new images of 1?



Generating New Samples

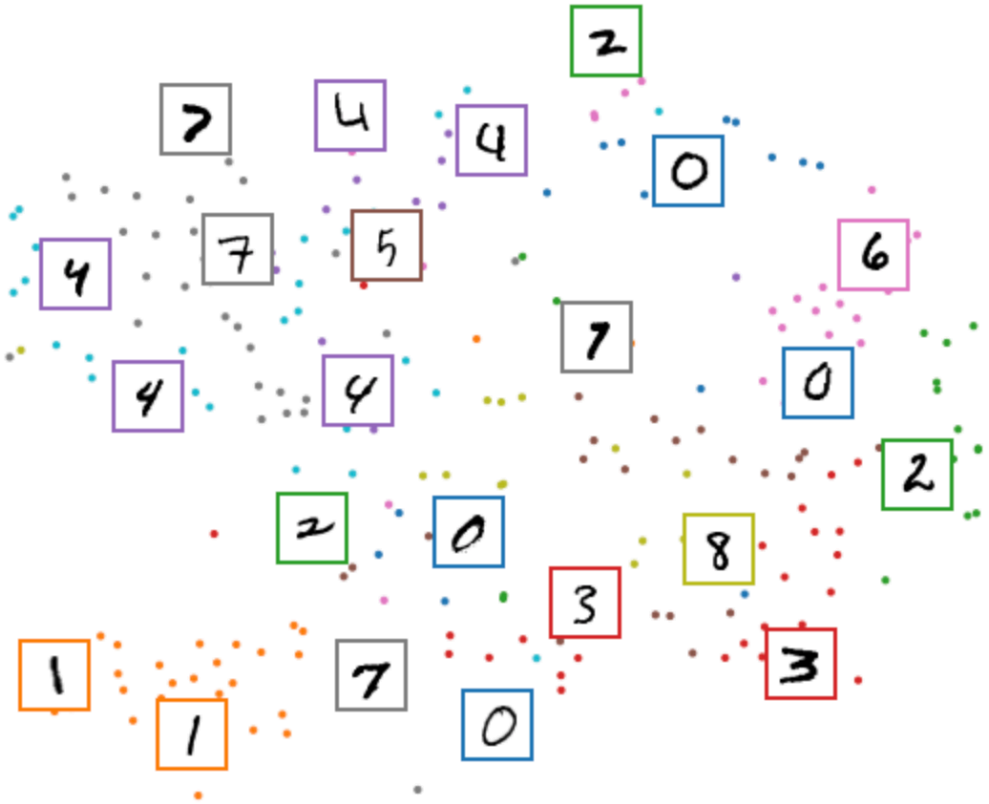
- Sample from a region?

Sample random points from this area to generate images of 1?



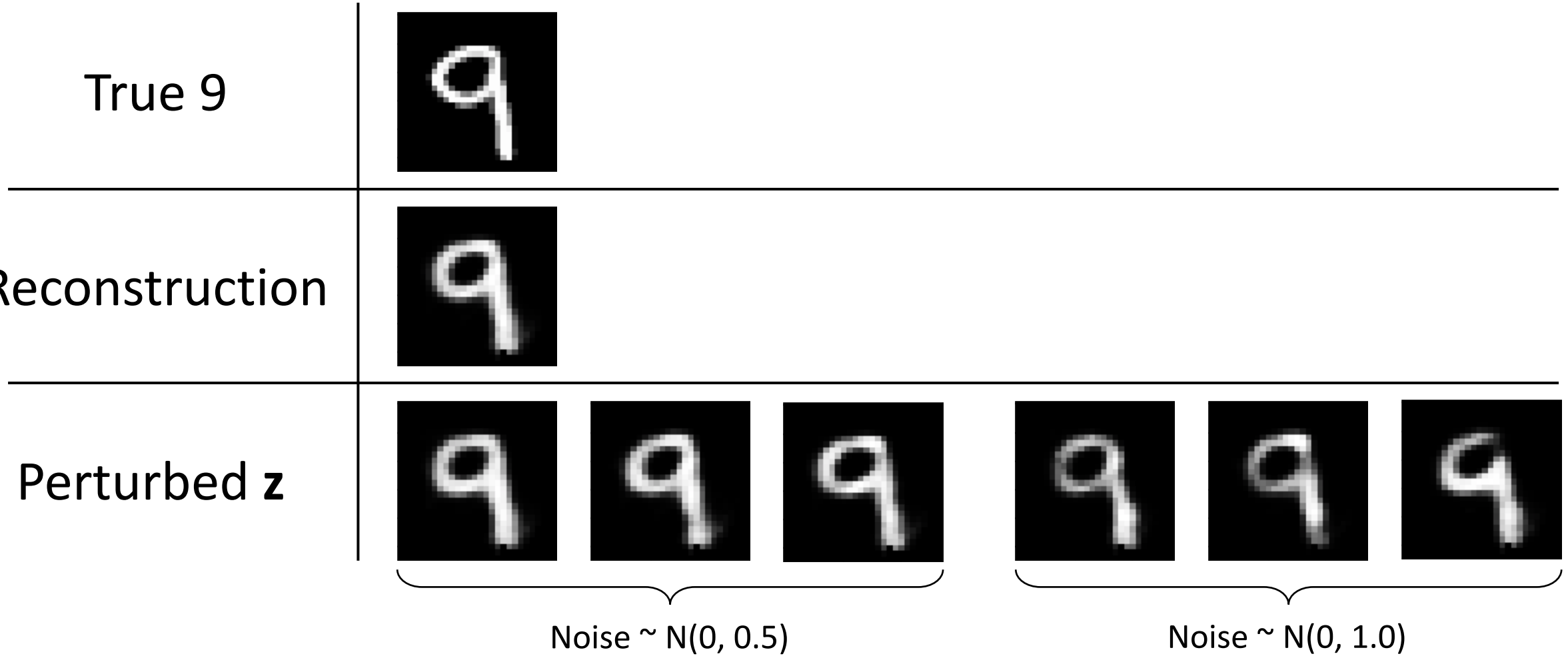
Generating New Samples

- Generating new samples
  - Perturb a known **z**
  - Sample from a region
    - Which region do you sample?
- Both are not guaranteed to work



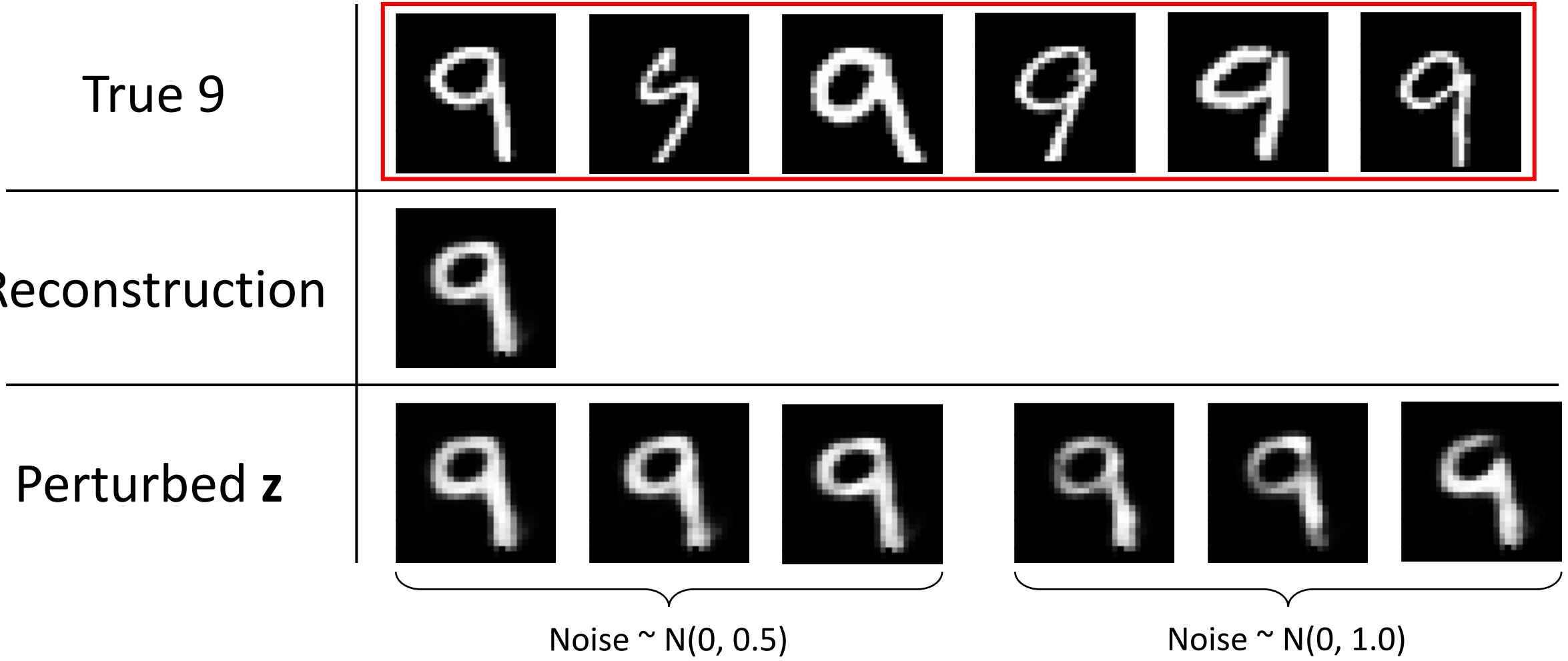
Example: Perturbing **z**

- Using an AE trained on MNIST for 50 epochs.



Example: Perturbing **z**

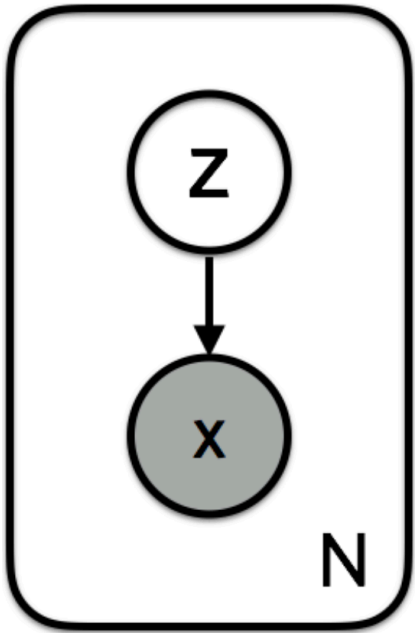
- Cannot generate diverse/novel samples



Variational Inference

Posterior Distribution

- Given data $\mathbf{X}$ and unobserved (latent/hidden) variables $\mathbf{Z}$
  - Assum $\mathbf{Z}$ determines $\mathbf{X}$
  - We are interested in the value $P(\mathbf{Z} \mid \mathbf{X})$



- Draw latent variables $z_i \sim p(z)$
- Draw datapoint $x_i \sim p(x \mid z)$

https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

Posterior Distribution

- Given data $\mathbf{X}$ and unobserved (latent/hidden) variables $\mathbf{Z}$
  - Assum $\mathbf{Z}$ determines $\mathbf{X}$
  - We are interested in the value $P(\mathbf{Z} \mid \mathbf{X})$
- Ex) Topic ($\mathbf{Z}$) of newpaper articles ($\mathbf{X}$)
- Ex) Latent representations ($\mathbf{Z}$) of MNIST images ($\mathbf{X}$)

Posterior Distribution

- Given data $\mathbf{X}$ and unobserved (latent/hidden) variables $\mathbf{Z}$
  - Assum $\mathbf{Z}$ determines $\mathbf{X}$
  - We are interested in the value $P(\mathbf{Z} \mid \mathbf{X})$
- Usually, a true posterior distribution in intractable
  - Using Bayesian principle,

$$P(\mathbf{Z} \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{P(\mathbf{X})} = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{\int_{\mathbf{Z}} P(\mathbf{X}, \mathbf{Z})\, d\mathbf{Z}}$$

Posterior Distribution

- Given data $\mathbf{X}$ and unobserved (latent/hidden) variables $\mathbf{Z}$
  - Assum $\mathbf{Z}$ determines $\mathbf{X}$
  - We are interested in the value $P(\mathbf{Z} \mid \mathbf{X})$
- Usually, a true posterior distribution in intractable
  - Using Bayesian principle,

$$P(\mathbf{Z} \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{P(\mathbf{X})} = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{\int_{\mathbf{Z}} P(\mathbf{X}, \mathbf{Z})\, d\mathbf{Z}}$$

This is often combinatorially large (even infinite!)

Posterior Distribution

- Given data $\mathbf{X}$ and unobserved (latent/hidden) variables $\mathbf{Z}$
  - Assum $\mathbf{Z}$ determines $\mathbf{X}$
  - We are interested in the value $P(\mathbf{Z} \mid \mathbf{X})$
- Usually, a true posterior distribution in intractable
  - Using Bayesian principle,

$$P(\mathbf{Z} \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{P(\mathbf{X})} = \frac{P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Z})}{\int_{\mathbf{Z}} P(\mathbf{X}, \mathbf{Z})\, d\mathbf{Z}}$$

  - Instead, approximate with a simpler function Q (often Gaussian)!

$$P(\mathbf{Z} \mid \mathbf{X}) \approx Q(\mathbf{Z})$$

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize *Kullback-Leibler divergence* (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) \triangleq \sum_{\mathbf{Z}} Q(\mathbf{Z}) \log \frac{Q(\mathbf{Z})}{P(\mathbf{Z} \mid \mathbf{X})}$$

(Assuming $\mathbf{Z}$ is a discrete variable)

  - Note that D_{KL} is not symmetric (i.e. it is not a distance!)
  - This formulation is called “Reverse KL”

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$
\begin{aligned}
D_{\mathrm{KL}}(Q \parallel P) &\triangleq \sum_{\mathbf{Z}} Q(\mathbf{Z}) \log \frac{Q(\mathbf{Z})}{P(\mathbf{Z} \mid \mathbf{X})} \\
&= \sum_{\mathbf{Z}} Q(\mathbf{Z}) \left[\log \frac{Q(\mathbf{Z})}{P(\mathbf{Z}, \mathbf{X})} + \log P(\mathbf{X}) \right] \\
&= \sum_{\mathbf{Z}} Q(\mathbf{Z}) \left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X}) \right] + \sum_{\mathbf{Z}} Q(\mathbf{Z}) \left[\log P(\mathbf{X}) \right] \\
&= \sum_{\mathbf{Z}} Q(\mathbf{Z}) \left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X}) \right] + \log P(\mathbf{X}) \\
&= \mathbb{E}_{\mathbf{Z}} \left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X}) \right] + \log P(\mathbf{X})
\end{aligned}
$$

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

$$\log P(\mathbf{X}) = D_{\mathrm{KL}}(Q \parallel P) - \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right]$$

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

$$\log P(\mathbf{X}) = D_{\mathrm{KL}}(Q \parallel P) \underbrace{- \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right]}_{\mathcal{L}(Q)}$$

$\mathcal{L}(Q)$: Evidence Lower Bound (**ELBO**)

  - Maximizing the ELBO leads to minimizing D_{KL}
    - Because ??

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

$$\log P(\mathbf{X}) = D_{\mathrm{KL}}(Q \parallel P) \underbrace{- \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right]}_{\mathcal{L}(Q)}$$

$\mathcal{L}(Q)$: Evidence Lower Bound (**ELBO**)

  - Maximizing the ELBO leads to minimizing D_{KL}
    - Because $\log P(\mathbf{X})$ is fixed.

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

$$\log P(\mathbf{X}) = D_{\mathrm{KL}}(Q \parallel P) \underbrace{- \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right]}_{\mathcal{L}(Q)}$$

$\mathcal{L}(Q)$: Evidence Lower Bound (**ELBO**)

  - Maximizing the ELBO leads to minimizing D_{KL}
- We converted an inference problem to an optimization problem!
  - For Gaussian Q, we can learn μ and σ.

Learning $Q(\mathbf{Z})$

- We want $Q(\mathbf{Z})$ as similar to $P(\mathbf{Z} \mid \mathbf{X})$ as possible
  - Minimize Kullback-Leibler divergence (KL-Divergence)

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

$$\log P(\mathbf{X}) = D_{\mathrm{KL}}(Q \parallel P) \underbrace{- \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right]}_{\mathcal{L}(Q)}$$

$\mathcal{L}(Q)$: Evidence Lower Bound (**ELBO**)

  - $\mathcal{L}(Q)$ is also called "variational free energy"

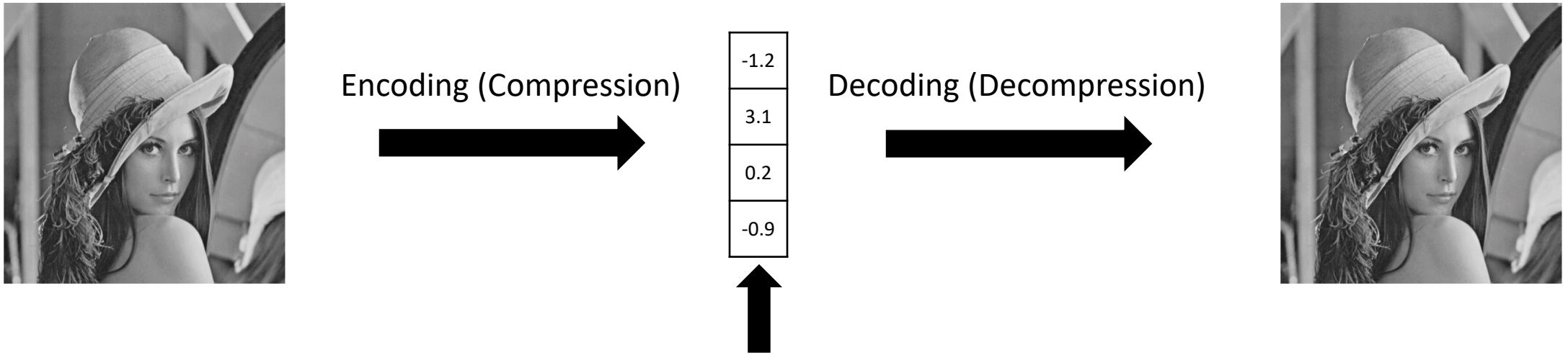
$$\mathcal{L}(Q) = \mathbb{E}_{\mathrm{Z}}[\log P(\mathbf{Z}, \mathbf{X})] + H(Q)$$

(H: Information Entropy)

Variational Autoencoder

VAE

- Objective
  - Compress $\mathbf{x}$ to $\mathbf{z}$ which follows $P(\mathbf{Z} \mid \mathbf{X})$
  - Decompress $\mathbf{z}$ to reconstruct $\mathbf{x}$



VAE

- Objective
  - Compress $\mathbf{x}$ to $\mathbf{z}$ which follows $P(\mathbf{Z} \mid \mathbf{X})$
  - Decompress $\mathbf{z}$ to reconstruct $\mathbf{x}$

Encoding (Compression)

$q_\theta(z \mid x_i)$

Decoding (Decompression)

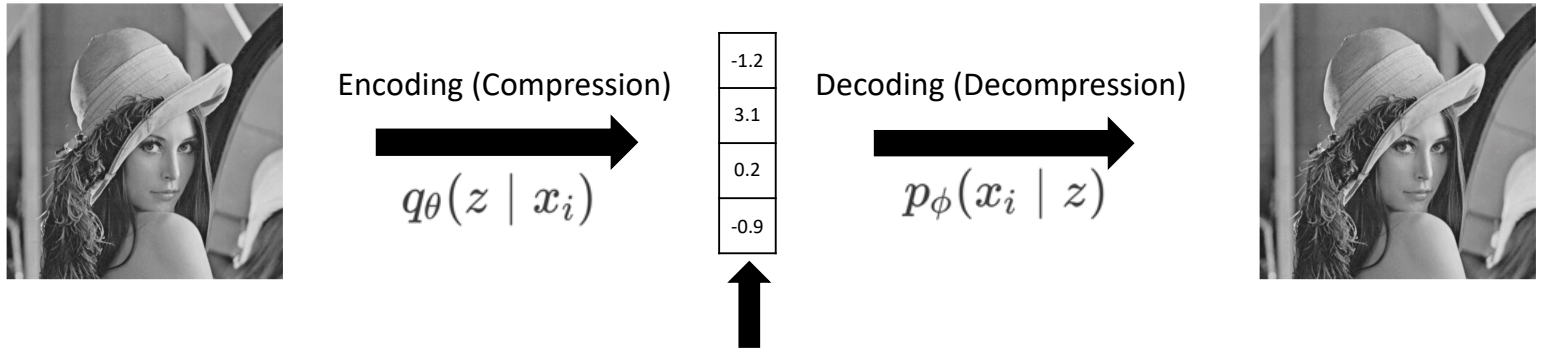
$p_\phi(x_i \mid z)$

This follows the distribution P (e.g. Gaussian $N(0, 1)$)

$$\mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

VAE

- Objective
  - Compress $\mathbf{x}$ to $\mathbf{z}$ which follows $P(\mathbf{Z} \mid \mathbf{X})$
  - Decompress $\mathbf{z}$ to reconstruct $\mathbf{x}$
- This allows us to
  - Map data distribution $P(\mathbf{X})$ to a probability distribution $P(\mathbf{Z})$
  - Sample $\mathbf{z}$ from $P(\mathbf{Z})$, which can be converted to $\mathbf{x}$



VAE Loss

- Loss consists of
  - Reconstruction loss
  - Regularization term
    - Force $\mathbf{z}$ to follow Gaussian distribution
- Loss for a single sample x_i

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

Decoder network parametrized by ϕ

Encoder network parametrized by θ

Usually $\mathcal{N}(0, 1)$

  - In autoencoders, negative log likelihood is reconstruction loss

https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

VAE: Probability Point of View

- We want to infer $P(\mathbf{Z} \mid \mathbf{X})$
  - The posterior distribution given data $\mathbf{X}$

VAE Posterior Distribution

- Since $P(\mathbf{Z} \mid \mathbf{X})$ is intractable, use variational inference

$$\mathbb{KL}(q_\lambda(z \mid x) \,||\, p(z \mid x)) = \mathbf{E}_q[\log q_\lambda(z \mid x)] - \mathbf{E}_q[\log p(x, z)] + \log p(x)$$

$$\Updownarrow$$

$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X}) \qquad \text{(Original VI)}$$

- Use $Q_\lambda(\mathbf{Z} \mid \mathbf{X})$ instead of $Q(\mathbf{Z})$
  - Because $\mathbf{Z}$ is determined by $\mathbf{X}$
  - This will be our encoder
  - λ is (μ, σ) for Gaussian Q ➔ Encoder generates (μ, σ)

VAE Posterior Distribution

- Since $P(\mathbf{Z} \mid \mathbf{X})$ is intractable, use variational inference

$$\mathbb{KL}(q_\lambda(z \mid x) \,||\, p(z \mid x)) = \mathbf{E}_q[\log q_\lambda(z \mid x)] - \mathbf{E}_q[\log p(x, z)] + \log p(x)$$

$$\Updownarrow$$

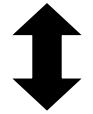
$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X}) \qquad \text{(Original VI)}$$

- Use $Q_\lambda(\mathbf{Z} \mid \mathbf{X})$ instead of $Q(\mathbf{Z})$
  - Because $\mathbf{Z}$ is determined by $\mathbf{X}$
  - This will be our encoder
  - λ is (μ, σ) for Gaussian Q ➔ Encoder generates (μ, σ)
- We want to learn an encoder $q^*_\lambda(z \mid x) = \arg\min_\lambda \mathbb{KL}(q_\lambda(z \mid x) \,||\, p(z \mid x))$

VAE Posterior Distribution

- Since $P(\mathbf{Z} \mid \mathbf{X})$ is intractable, use variational inference

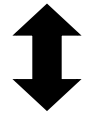
$$\mathbb{KL}(q_\lambda(z \mid x) \,||\, p(z \mid x)) = \mathbf{E}_q[\log q_\lambda(z \mid x)] - \mathbf{E}_q[\log p(x, z)] + \log p(x)$$



$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X}) \qquad \text{(Original VI)}$$

- The new ELBO term

$$ELBO(\lambda) = \mathbf{E}_q[\log p(x, z)] - \mathbf{E}_q[\log q_\lambda(z \mid x)]$$

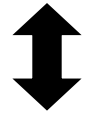


$$\mathcal{L}(Q) = -\,\mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] \qquad \text{(Original ELBO)}$$

VAE Posterior Distribution

- Since $P(\mathbf{Z} \mid \mathbf{X})$ is intractable, use variational inference

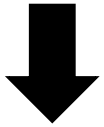
$$\mathbb{KL}(q_\lambda(z \mid x) \,||\, p(z \mid x)) = \mathbf{E}_q[\log q_\lambda(z \mid x)] - \mathbf{E}_q[\log p(x, z)] + \log p(x)$$



$$D_{\mathrm{KL}}(Q \parallel P) = \mathbb{E}_{\mathbf{Z}}\left[\log Q(\mathbf{Z}) - \log P(\mathbf{Z}, \mathbf{X})\right] + \log P(\mathbf{X})$$

- The new ELBO term

$$ELBO(\lambda) = \mathbf{E}_q[\log p(x, z)] - \mathbf{E}_q[\log q_\lambda(z \mid x)]$$



ELBO term for each sample x_i

$$ELBO_i(\lambda) = \mathbb{E}q_\lambda(z \mid x_i)[\log p(x_i \mid z)] - \mathbb{KL}(q_\lambda(z \mid x_i) \,||\, p(z))$$

ELBO Term

- ELBO term for each x_i

$$ELBO_i(\lambda) = \mathbb{E}q_\lambda(z \mid x_i)[\log p(x_i \mid z)] - \mathbb{KL}(q_\lambda(z \mid x_i) \mid\mid p(z))$$

- Parametrize the components
  - Approximate posterior q with θ
  - Likelihood p with ϕ
- The parametrized ELBO term

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

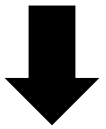
ELBO Term

- ELBO term for each x_i

$$ELBO_i(\lambda) = \mathbb{E}q_\lambda(z \mid x_i)[\log p(x_i \mid z)] - \mathbb{KL}(q_\lambda(z \mid x_i) \mid\mid p(z))$$

- Parametrize the components
  - Approximate posterior q with θ
  - Likelihood p with ϕ
- The parametrized ELBO term

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$



Remind you of something??

ELBO Term

- The parametrized ELBO term

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

VAE ELBO is the negative loss!!

$$ELBO_i(\theta, \phi) = -l_i(\theta, \phi)$$

  - Maximizing the ELBO is equivalent to minimizing the loss!

ELBO Term

- The parametrized ELBO term

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

  - Maximizing the ELBO is equivalent to minimizing the loss!

- ELBO can be maximized by both
  - Updating the encoder network (i.e. learning θ)
  - Updating the decoder network (i.e. learning ϕ)

ELBO Term

- The parametrized ELBO term

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

  - Maximizing the ELBO is equivalent to minimizing the loss!

- ELBO can be maximized by both
  - Updating the encoder network (i.e. learning θ)
  - Updating the decoder network (i.e. learning ϕ)
  - In practice, update both params together.

Training VAE

Training VAE

- Loss function

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

- How do you train your network?

Training VAE

- Loss function

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

- Using a random sample x_i as an example:
  - Push x_i into the encoder $q_\theta(z \mid x_i)$
  - Obtain (μ_i, σ_i)
  - Sample (many) z_i from $\mathcal{N}(\mu_i, \sigma_i)$ ➔ Monte Carlo Estimation of $E_z[f(z)]$
  - Insert z_i into the decoder $p_\phi(x_i \mid z)$
  - Obtain x_i'
  - Calculate squared error $\|x_i' - x_i\|^2$ and $D_{KL}(\mathcal{N}(\mu_i, \sigma_i)||\mathcal{N}(0,1))$
  - Backpropagate
  - Update θ and ϕ

Training VAE

- Loss function

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

- Using a random sample x_i as an example:
  - Push x_i into the encoder $q_\theta(z \mid x_i)$
  - Obtain (μ_i, σ_i)
  - Sample (many) z_i from $\mathcal{N}(\mu_i, \sigma_i)$ ➔ Monte Carlo Estimation of $E_z[f(z)]$
  - Insert z_i into the decoder $p_\phi(x_i \mid z)$
  - Obtain x_i'
  - Calculate squared error $\|x_i' - x_i\|^2$ and $D_{KL}(\mathcal{N}(\mu_i, \sigma_i)||\mathcal{N}(0,1))$
  - Backpropagate
  - Update θ and ϕ

This has an analytical expression,

$$\frac{1}{2}(\mu_i^2 + \sigma_i^2 - \log \sigma_i^2 - 1)$$

which can be used to update θ

Training VAE

- Loss function

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

- Using a random sample x_i as an example:
  - Push x_i into the encoder $q_\theta(z \mid x_i)$
  - Obtain (μ_i, σ_i)
  - Sample (many) z_i from $\mathcal{N}(\mu_i, \sigma_i)$ ➔ Monte Carlo Estimation of $E_z[f(z)]$
  - Insert z_i into the decoder $p_\phi(x_i \mid z)$
  - Obtain x_i'
  - Calculate squared error $\|x_i' - x_i\|^2$ and $D_{KL}(\mathcal{N}(\mu_i, \sigma_i)||\mathcal{N}(0,1))$
  - Backpropagate
  - Update θ and ϕ

How about this?

Stochasticity in the Network

- $\|x_i' - x_i\|^2$ involves a random sample z_i
  - x_i' is decoded based on $z_i \sim \mathcal{N}(\mu_i, \sigma_i)$
- One can backpropagate through the decoder.
  - Consider z_i as an input to the decoder $p_\phi(x_i \mid z)$
  - We can obtain $\frac{\partial \mathcal{L}}{\partial \phi}$ via chain-rule ($\mathcal{L}$ = MSE Loss)

Stochasticity in the Network

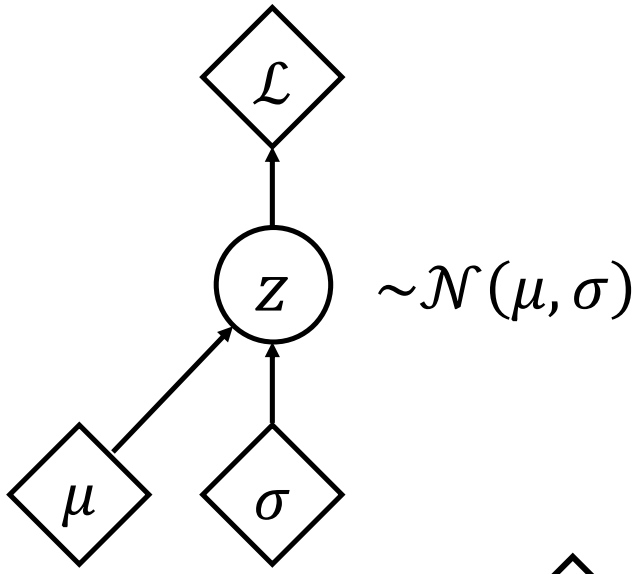
- $\|x_i' - x_i\|^2$ involves a random sample z_i
  - x_i' is decoded based on $z_i \sim \mathcal{N}(\mu_i, \sigma_i)$
- One can backpropagate through the decoder.
  - Consider z_i as an input to the decoder $p_\phi(x_i \mid z)$
  - We can obtain $\frac{\partial \mathcal{L}}{\partial \phi}$ via chain-rule ($\mathcal{L}$ = MSE Loss)
- Can you backpropagate through the encoder to obtain $\frac{\partial \mathcal{L}}{\partial \theta}$?

Stochasticity in the Network

- $\|x_i' - x_i\|^2$ involves a random sample z_i
  - x_i' is decoded based on $z_i \sim \mathcal{N}(\mu_i, \sigma_i)$
- One can backpropagate through the decoder.
  - Consider z_i as an input to the decoder $p_\phi(x_i \mid z)$
  - We can obtain $\frac{\partial \mathcal{L}}{\partial \phi}$ via chain-rule ($\mathcal{L}$ = MSE Loss)
- Can you backpropagate through the encoder to obtain $\frac{\partial \mathcal{L}}{\partial \theta}$?
  - How would you get past the randomly sampled z_i?

Stochasticity in the Network

- Can you backpropagate through the encoder to obtain $\frac{\partial \mathcal{L}}{\partial \theta}$?
  - How would you get past the randomly sampled z_i?

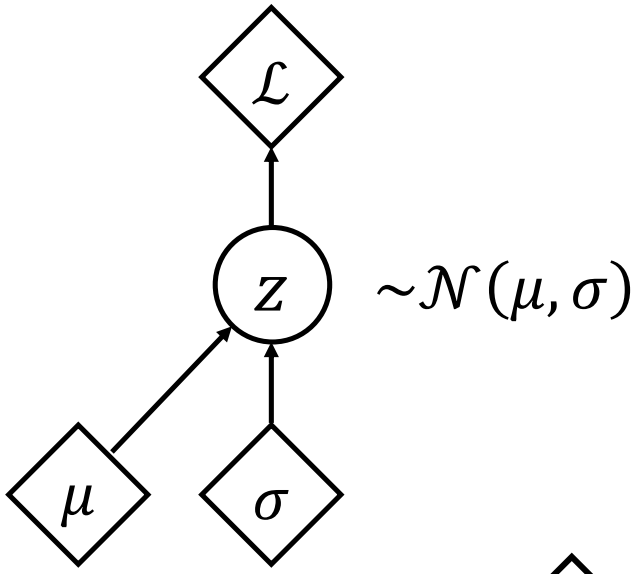


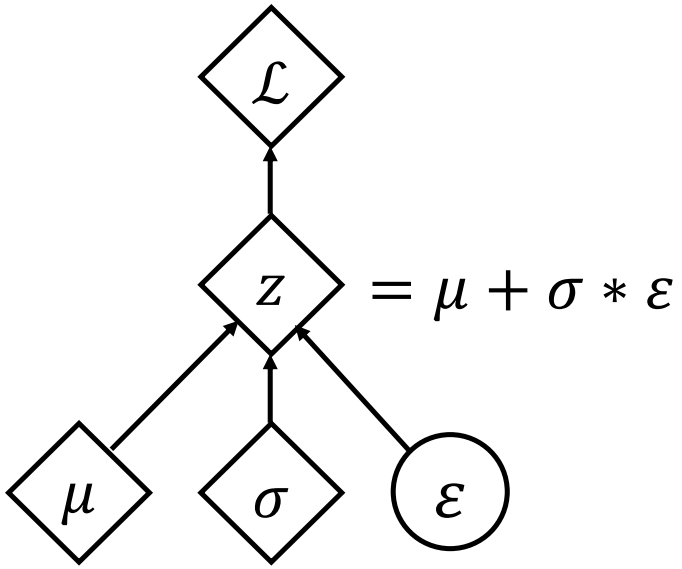
: Deterministic Node

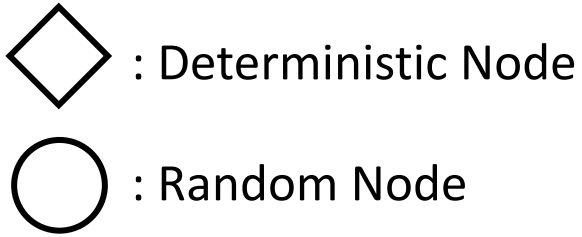
: Random Node

Reparametrization Trick

- Can you backpropagate through the encoder to obtain $\frac{\partial \mathcal{L}}{\partial \theta}$?
  - How would you get past the randomly sampled z_i?

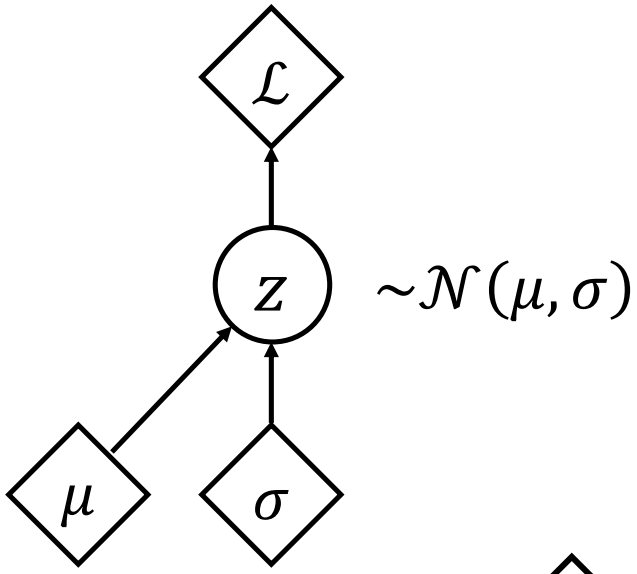






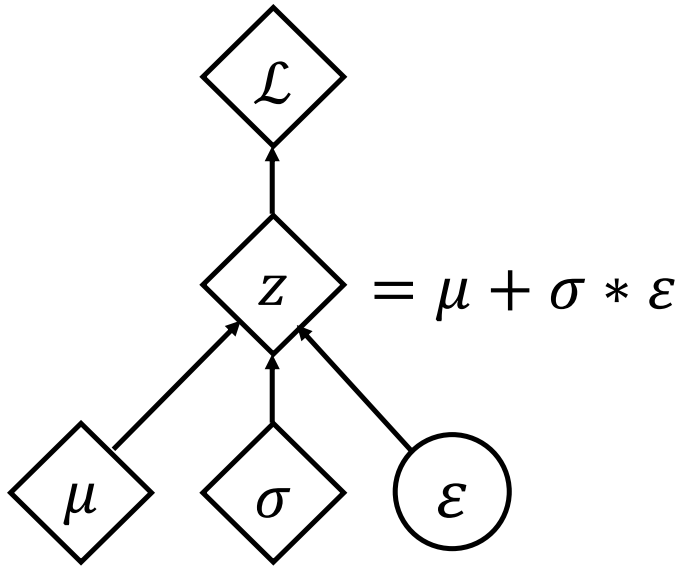
Reparametrization Trick

- Can you backpropagate through the encoder to obtain $\frac{\partial \mathcal{L}}{\partial \theta}$?
  - How would you get past the randomly sampled z_i?



: Deterministic Node

: Random Node

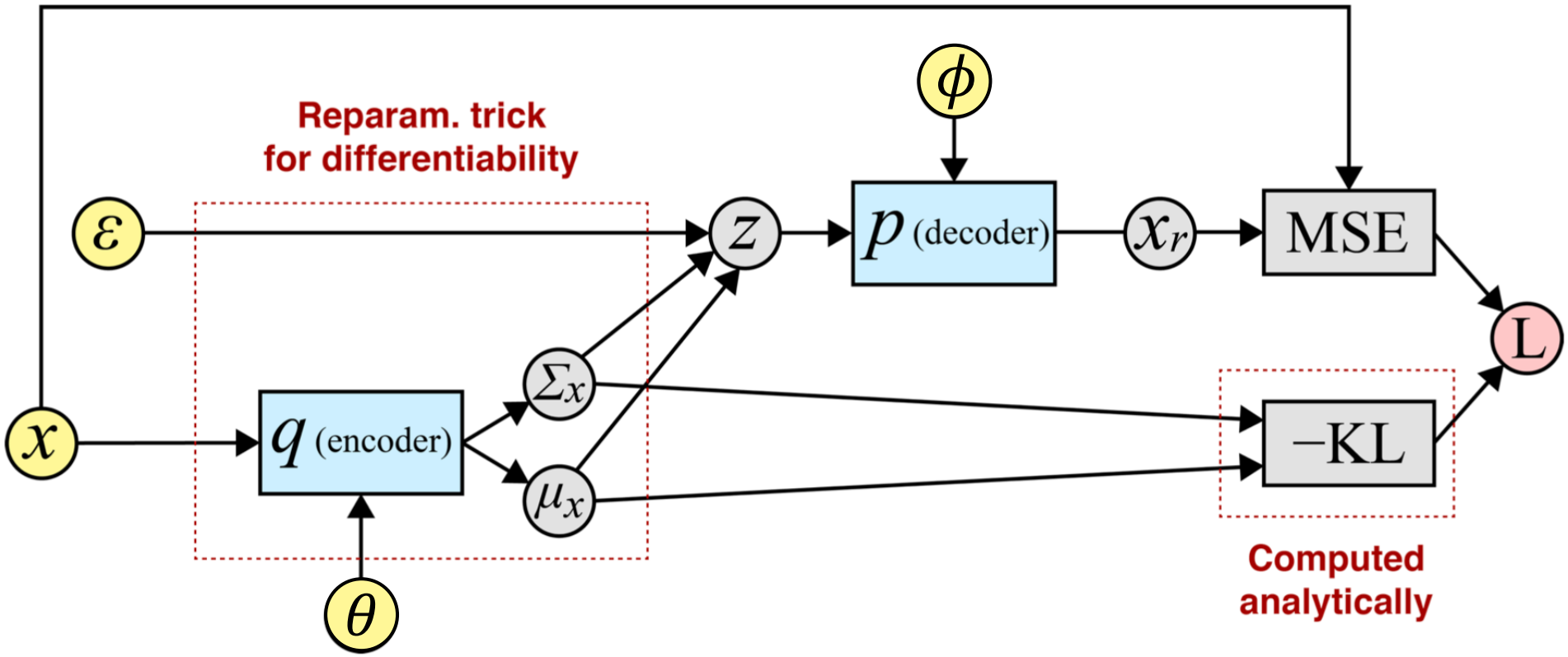


Reparametrization Trick:

Randomness is removed from the backprop path.

The Big Picture

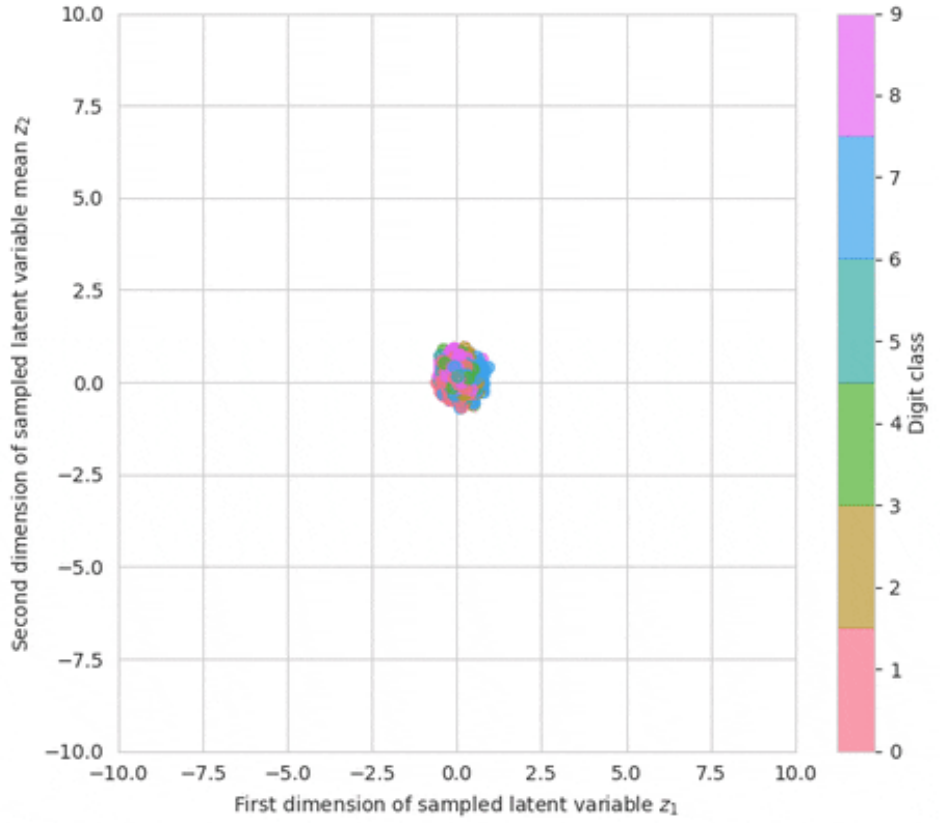
- The entire flow chart



http://gregorygundersen.com/blog/2018/04/29/reparameterization/

Training Process

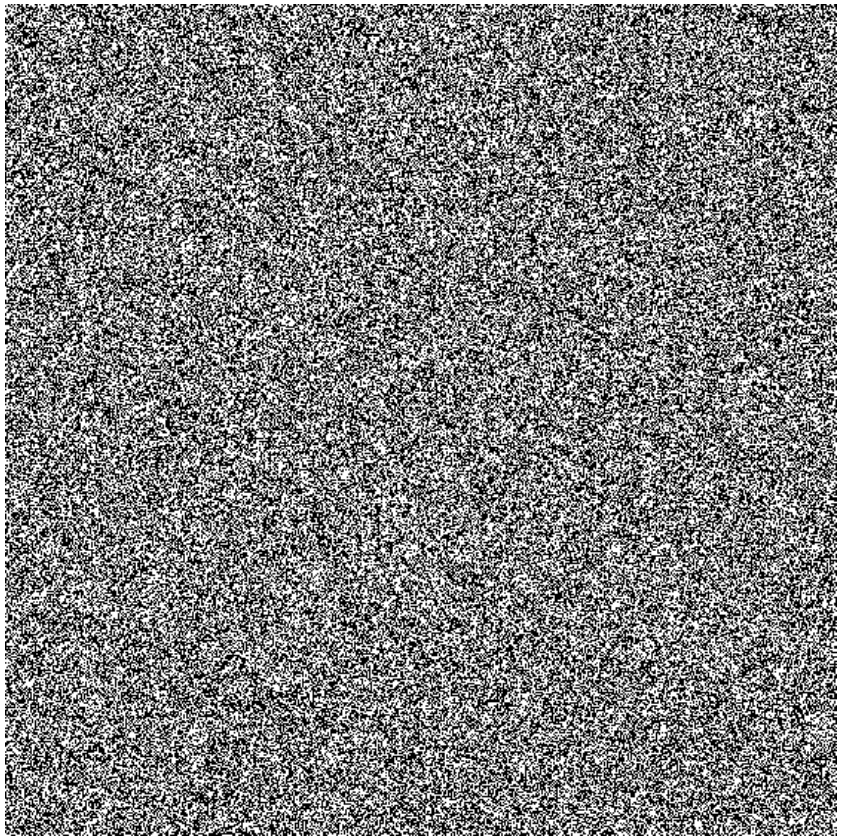
- Training VAE on MNIST images
  - Plotting 2-dimensional z for each encoded image as training proceeds



https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

Image Generation

- Sample z from 2-dimensional space then decode.
  - Range [-3, 3] for both axes.



https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

Conclusion

- VAE is an elegant solution for approximating $P(\mathbf{Z} \mid \mathbf{X})$
  - It is still an approximation though
  - The two mapping functions are disconnected
    - Encoder and decoder use separate parameters
- Normalizing Flow
  - Use invertible functions to map between p(X) and p(Z)
- β-VAE, TC-VAE
  - Disentangling the latent space
  - We want each $\mathbf{z}$ dimension to have an interpretable role.
    - Rotation of the image, skewedness of the image, color of the image, etc.
- You can use VAE to generate sequences or graphs.

AI504: Programming for Artificial Intelligence

Week 5: Variational Autoencoder

Edward Choi
Grad School of AI
edwardchoi@kaist.ac.kr