OENCODER**AUTOENCODER**AUTOE

AUTOENCODER

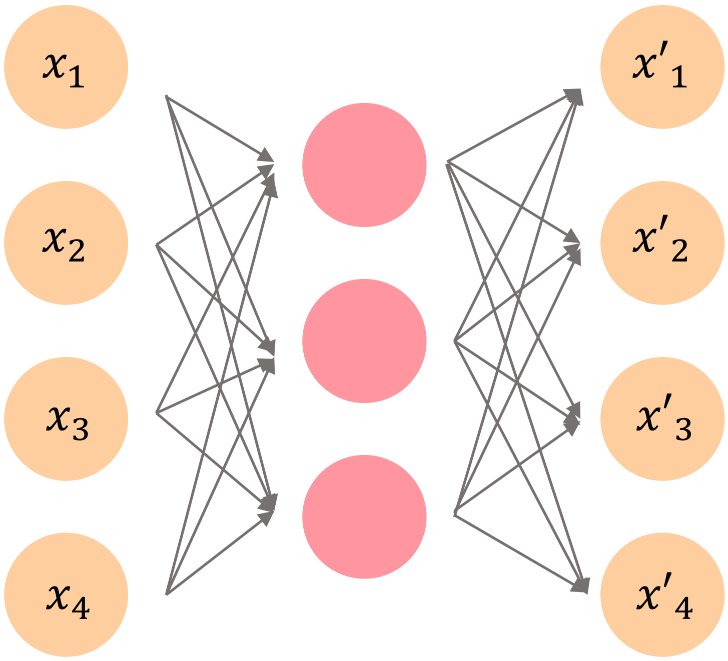
Basics and Extensions of Autoencoder

2020 Jan 8th
Wed 4.pm

KwangWoon University
Machine Learning Study

TEAM A

AUTOENCODER



Basic structure of Autoencoder

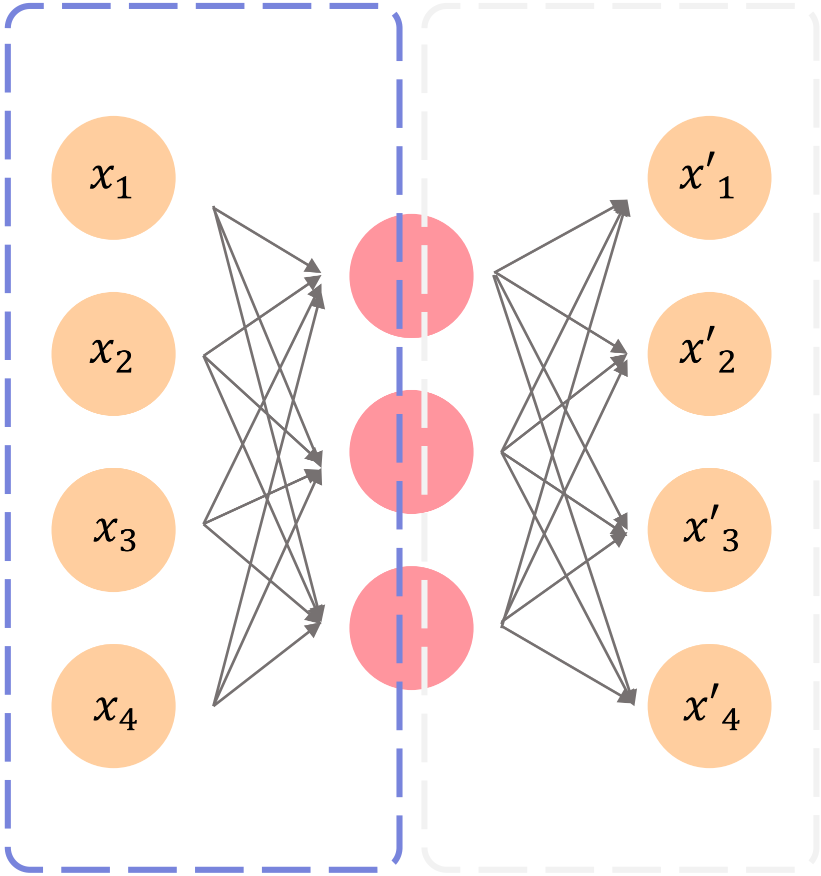
Unsupervised Learning	Reconstructor
Dimension Reduction	Encoder Decoder Base

Key Concepts of Autoencoder

AUTOENCODER

Encoder

Only the key features of the input data entered to Hidden layer by the learning process, and the rest of the information (Non-feature) is being loss



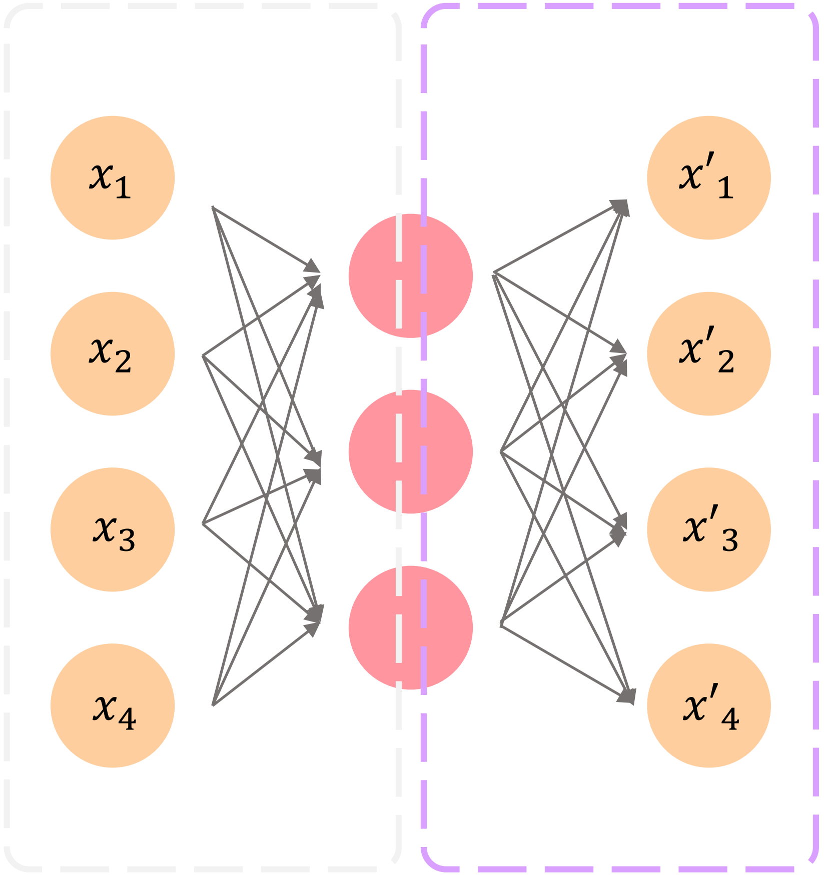
Decoder

Reconstruction input data by approximate value from input, this data based on what has been learned in the Hidden layer

AUTOENCODER

Encoder

Only the key features of the input data entered to Hidden layer by the learning process, and the rest of the information (Non-feature) is being loss



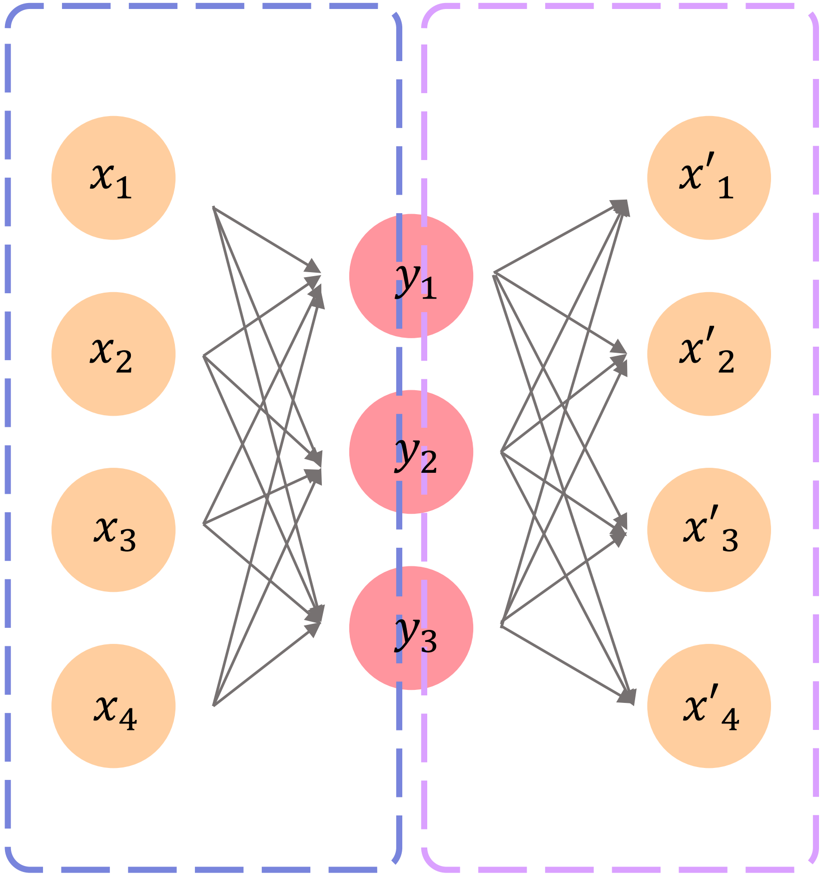
Decoder

Reconstruction input data by approximate value from input, this data based on what has been learned in the Hidden layer

AUTOENCODER

Encoder

Only the key **features** of the input data entered to Hidden layer by the learning process, and the rest of the information (Non-feature) is being loss



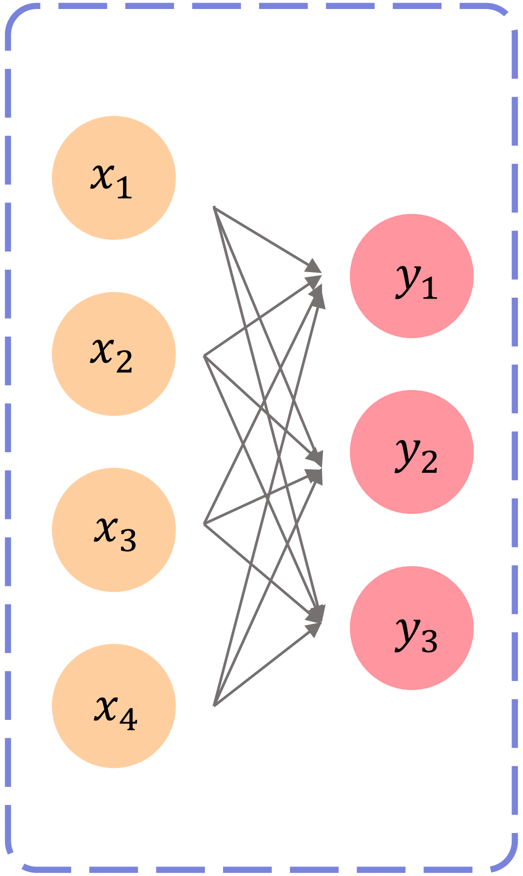
Decoder

Reconstruction input data by approximate value from input, this data based on what has been learned in the Hidden layer

The key of the Autoencoder is let the model **Reconstruction** same output based on input, so that the feature can be extracted at the hidden layer

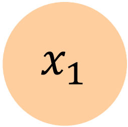
(i.e., find the weight that makes the input value and the output value as similar as possible).

ENCODER

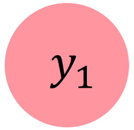


$$\phi : X \to F$$

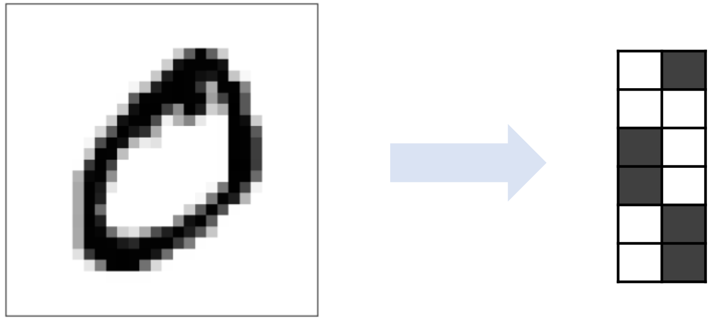
- Feature Extracting by dimension reduction



: Input Data



: Hidden Layer Node



Input Data

Feature
(Weight)

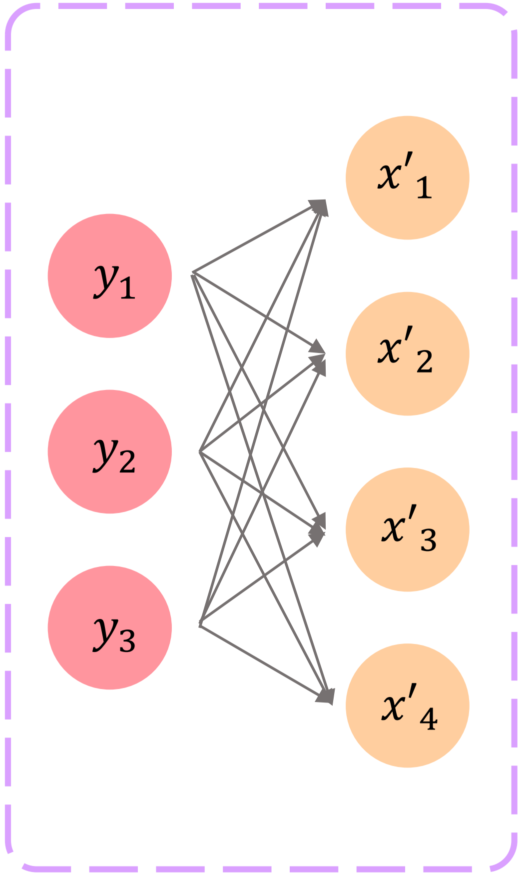
Encoder

Only the key **features** of the input data entered to Hidden layer by the learning process, and the rest of the information (Non-feature) is being loss

Extracts key information from data through dimension reduction

These Features will be uses for reconstructing input

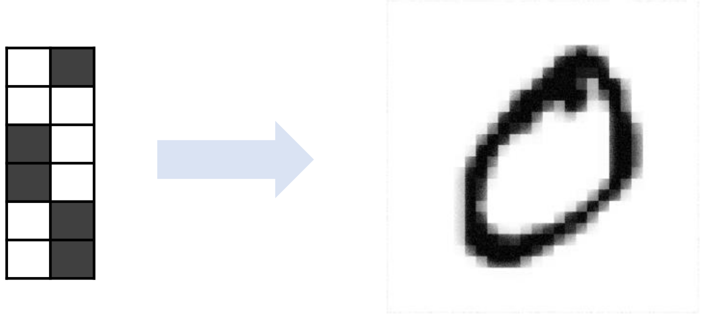
DECODER



$\psi : F \rightarrow X$

- Reconstruction from Feature

y_1 : Hidden Layer Node x_1 : Output Data



Feature
(Weight)

Output Data

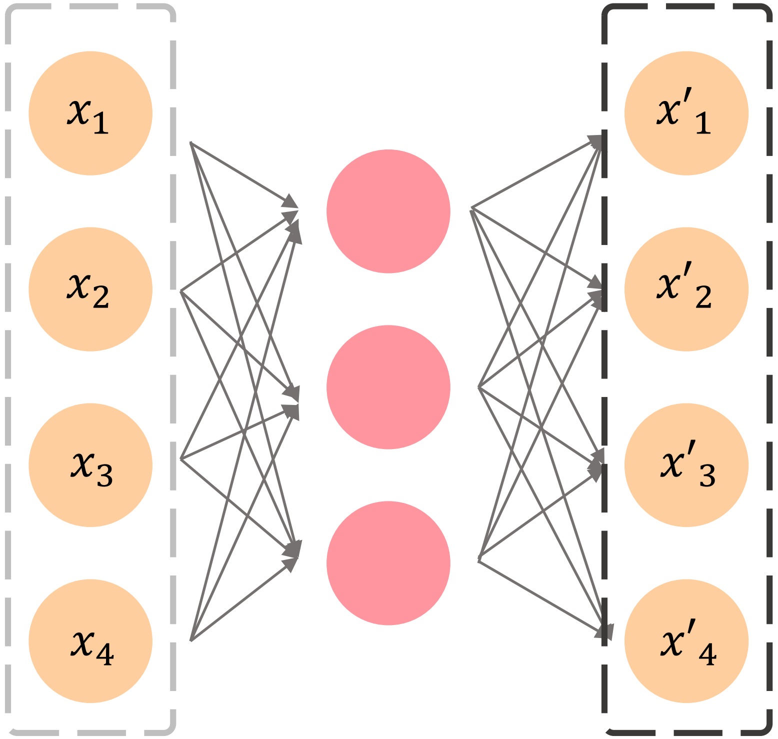
Decoder

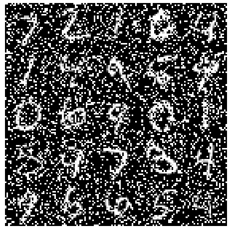
Reconstruction input data by approximate value from input, this data based on what has been learned in the Hidden layer

Reconstruction input data from its Feature

Similarity of input and output data represent models performance

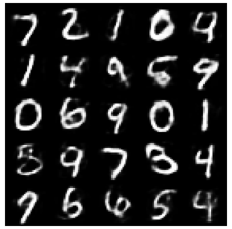
Denoising Autoencoder





$\hat{x} = x + r$

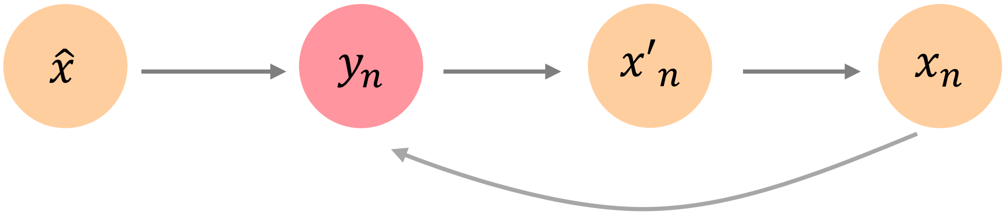
r = Random Noise



x

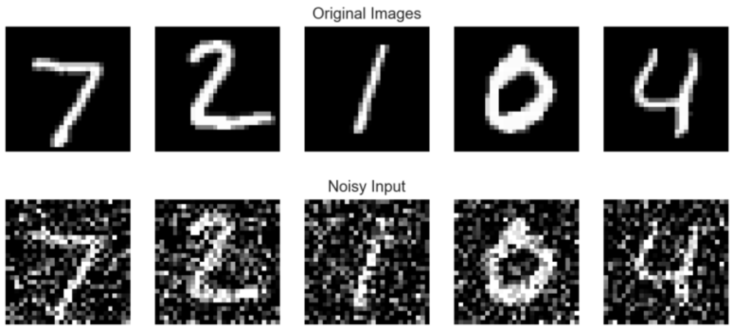
x = Original Image

Training

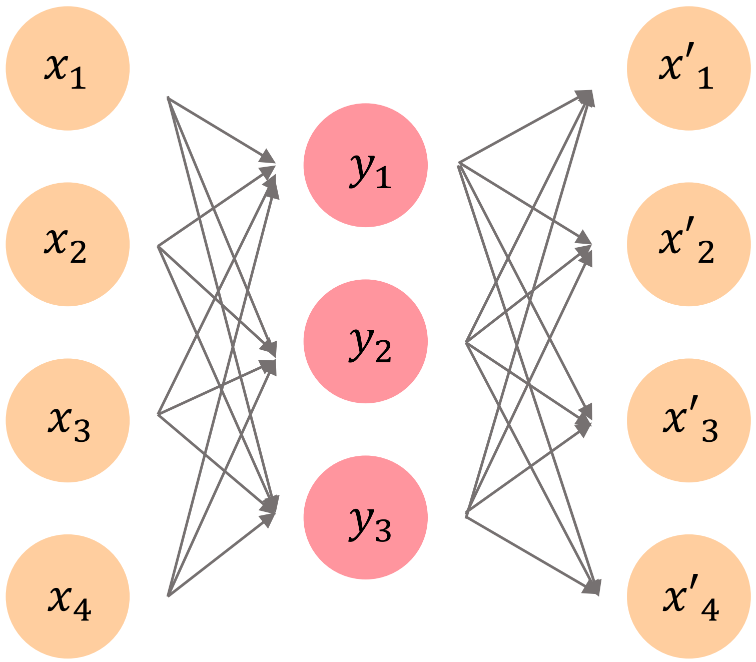


Update y_n let $\hat{x}$ be x_n

Enable extracting of features relate with x_n
(Extraction targeting)



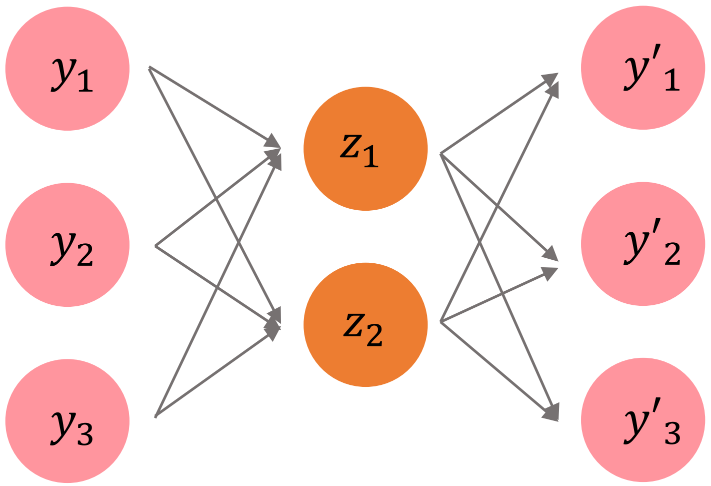
Stacked Autoencoder



Training

Train first autoencoder to reconstruct input x_n

Stacked Autoencoder

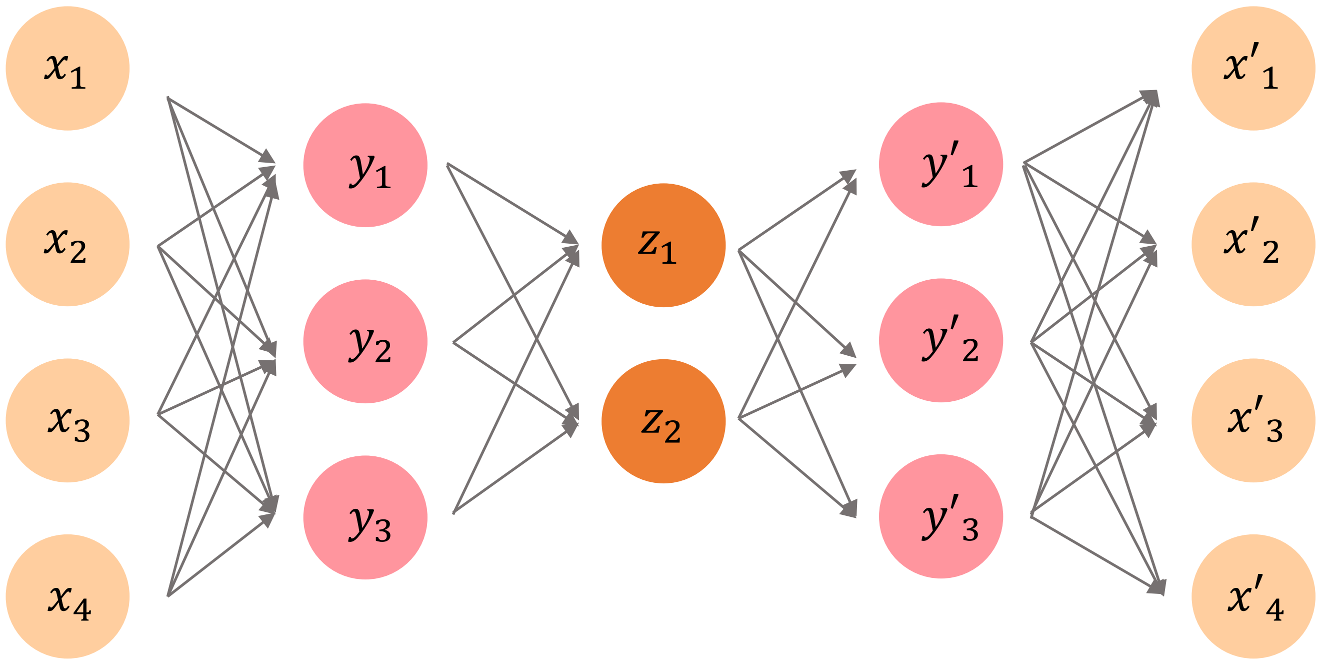


Training

Train first autoencoder to reconstruct input x_n

Train second autoencoder to reconstruct input y_n

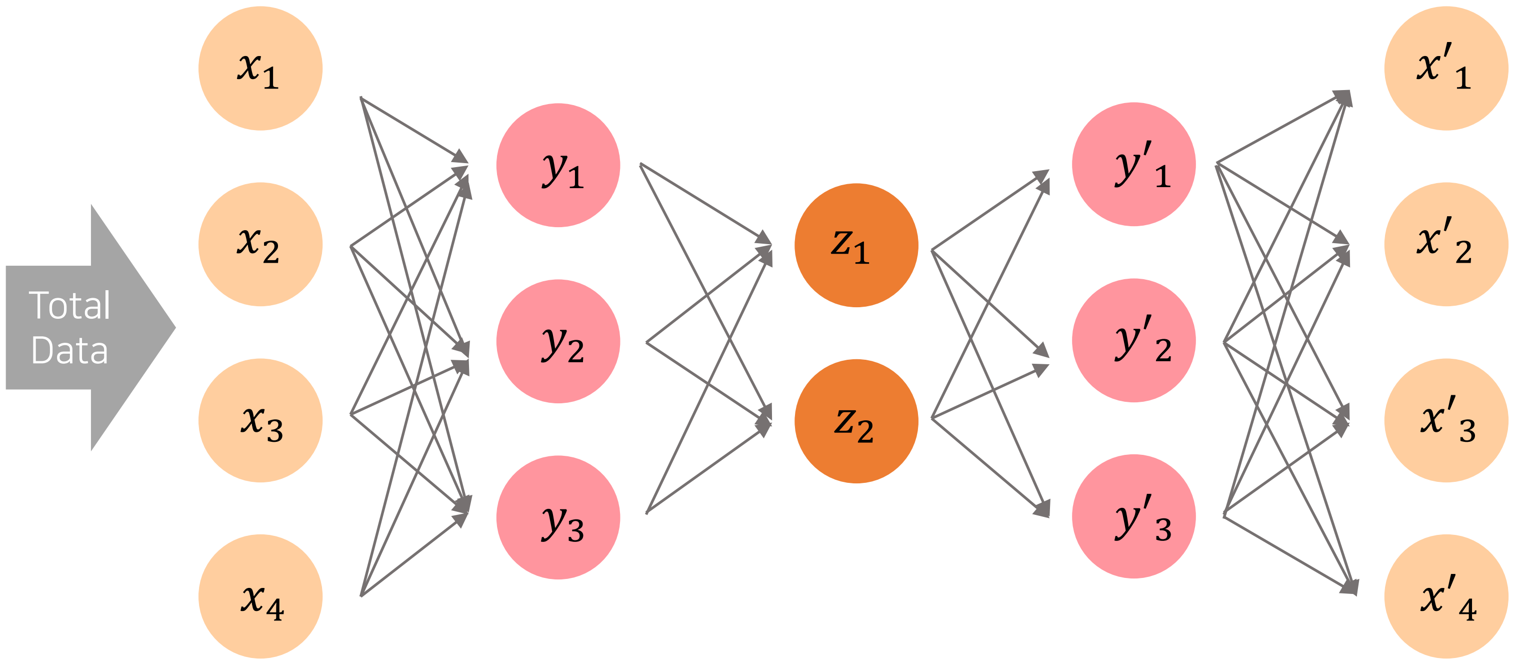
Stacked Autoencoder



Training

Build stacked autoencoder
by stacking autoencoders
trained in the previous step

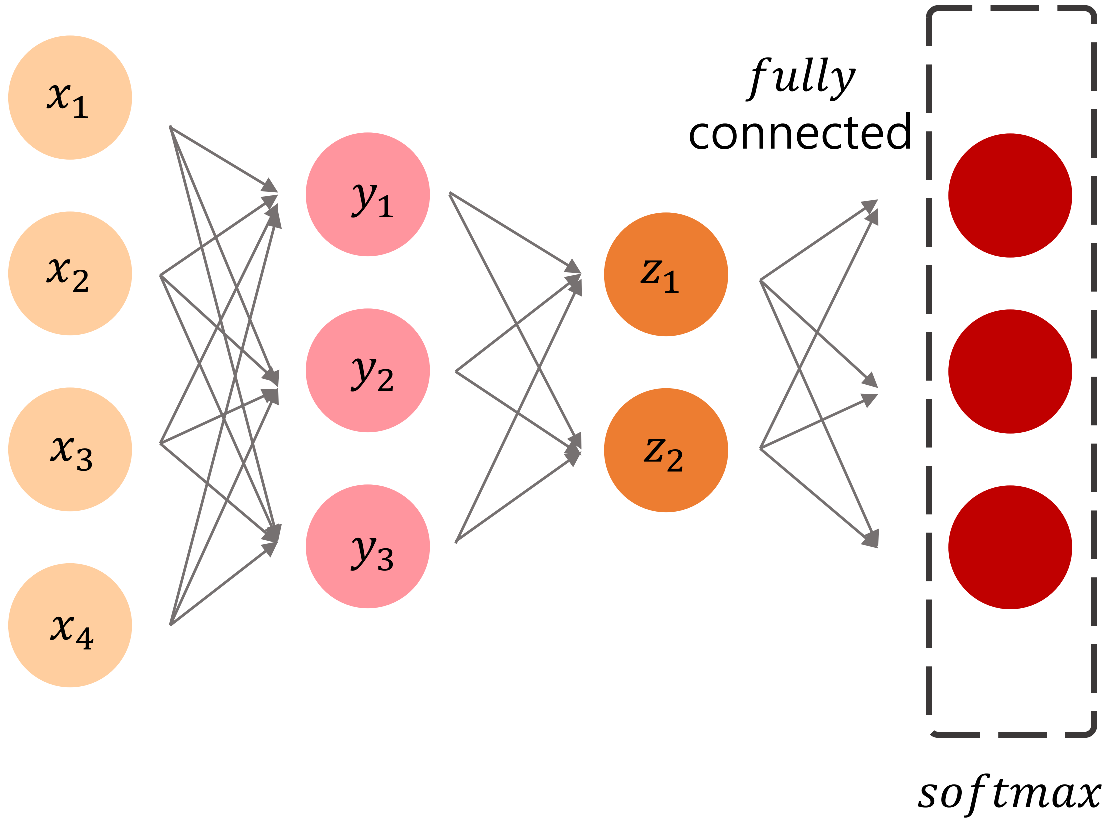
Stacked Autoencoder



Unsupervised pre-training
by using stacked autoencoder

Train stacked autoencoder
by using the total data set

Stacked Autoencoder

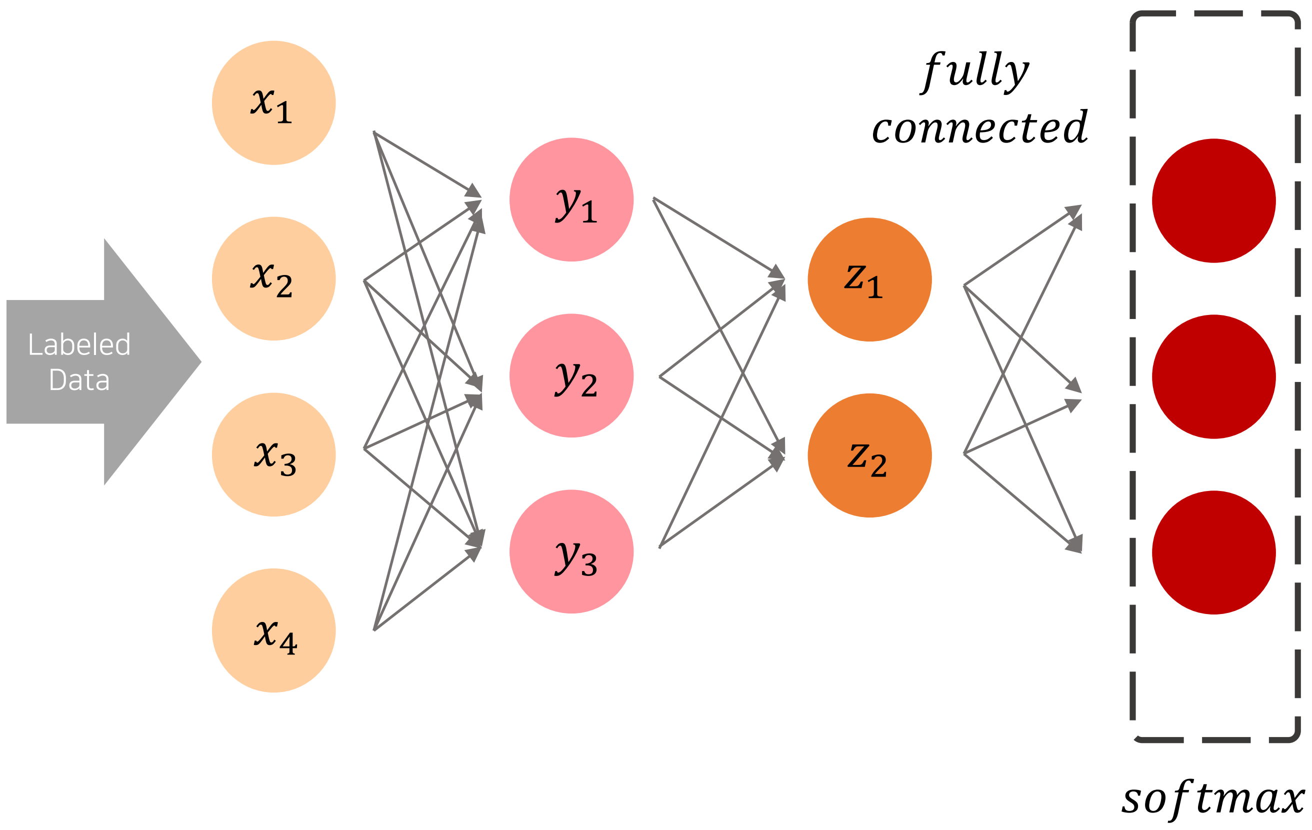


Unsupervised pre-training
by using stacked autoencoder

Copy the parameters of layers from input layer to center layer in the autoencoder.

Then, stack the softmax layer in the top layer, and fully-connect the top two layers.

Stacked Autoencoder



Unsupervised pre-training
by using stacked autoencoder

Train the network by using labeled data.
Finally, the network will operate
as a classifier

Backpropagation algorithm
⇒ Cross Entropy loss

Autoencoder Model also can be working as a **Generator** based on its **Reconstruction** concept

↓

Variational Autoencoder

Autoencoder As Generator

Autoencoder Model also can be working as a

Generator based on its **Reconstruction** concept

Variational Method

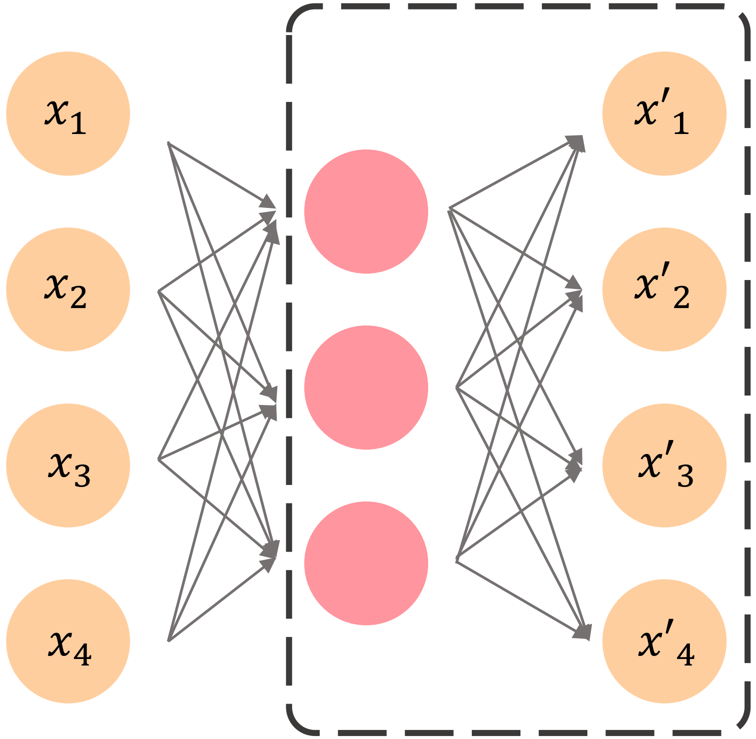
When the problem of finding the extreme point of function p(x),

if it hard to solve by directly access that function(p(x))

Optimize function by replacing it with another easy-to-handle function q(x)

This way we can get an **approximate year for p(x)**

Variational Autoencoder



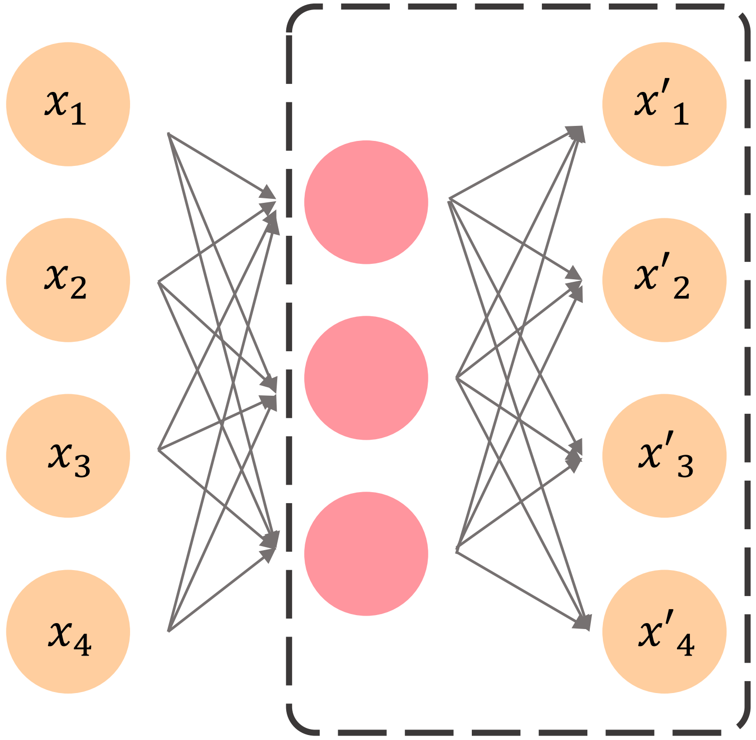
If we can get a low-level representation z from high-level data X

We can Adjust Z to create new images that are not given in the training set

Key is…

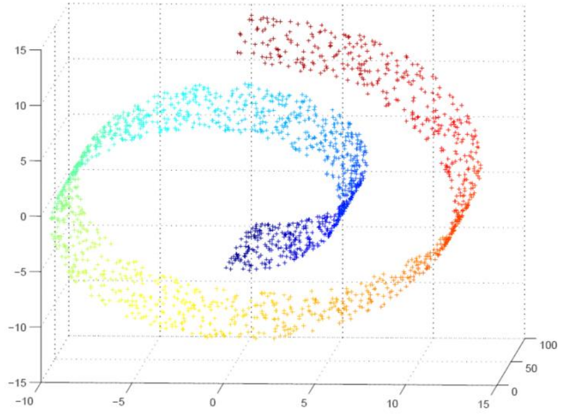
How can we high-level data be expressed as low-level data?

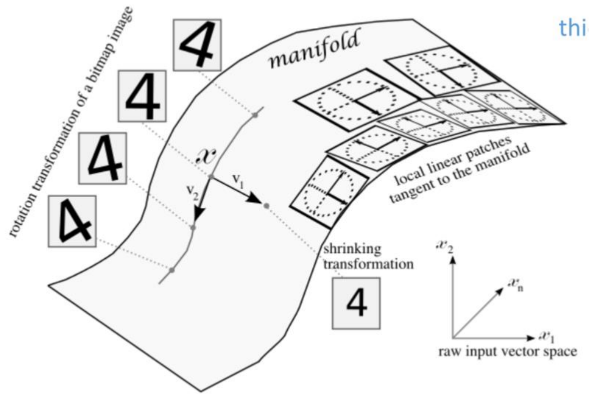
Variational Autoencoder



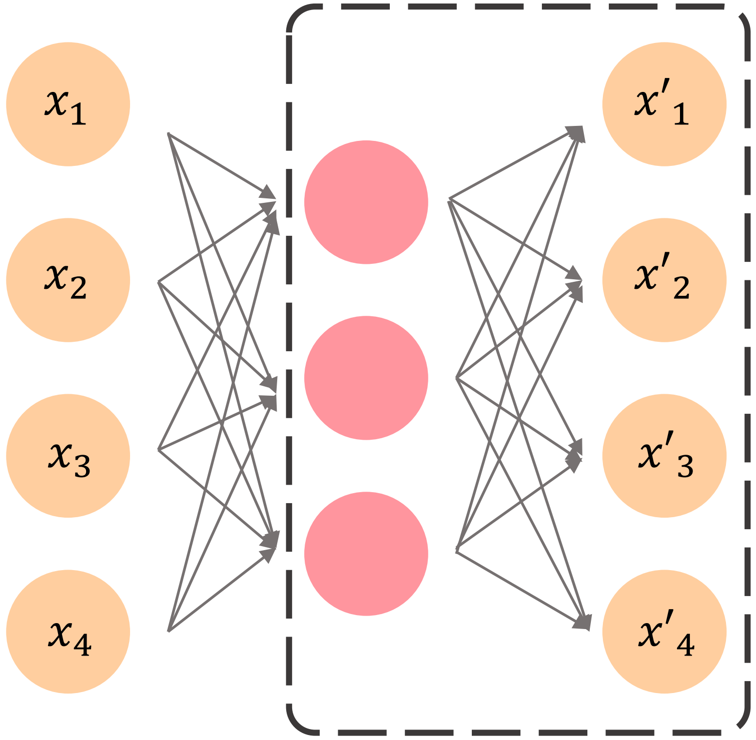
Key is…

How can we high-level data be expressed as low-level data?



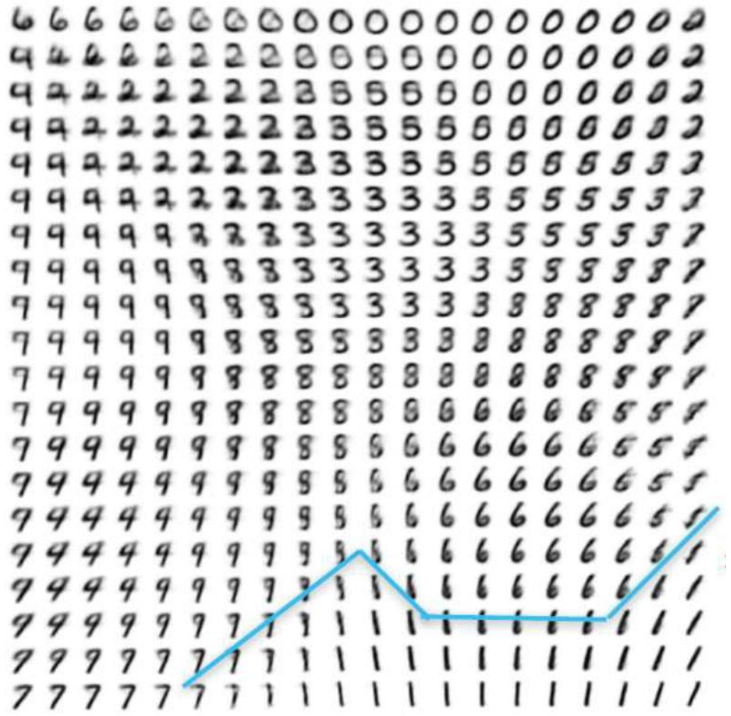


Variational Autoencoder

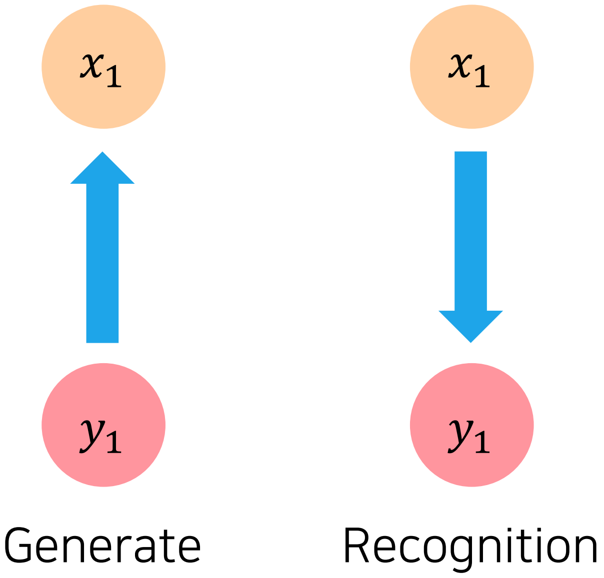


Key is…

How can we high-level data be expressed as low-level data?



Variational Autoencoder



x_n : Input Image

y_n : Latent variable

Example of Latent variable

Type	:	Cam Axis	:	Light
(Smile	:	[14,0,-3]	:	White)

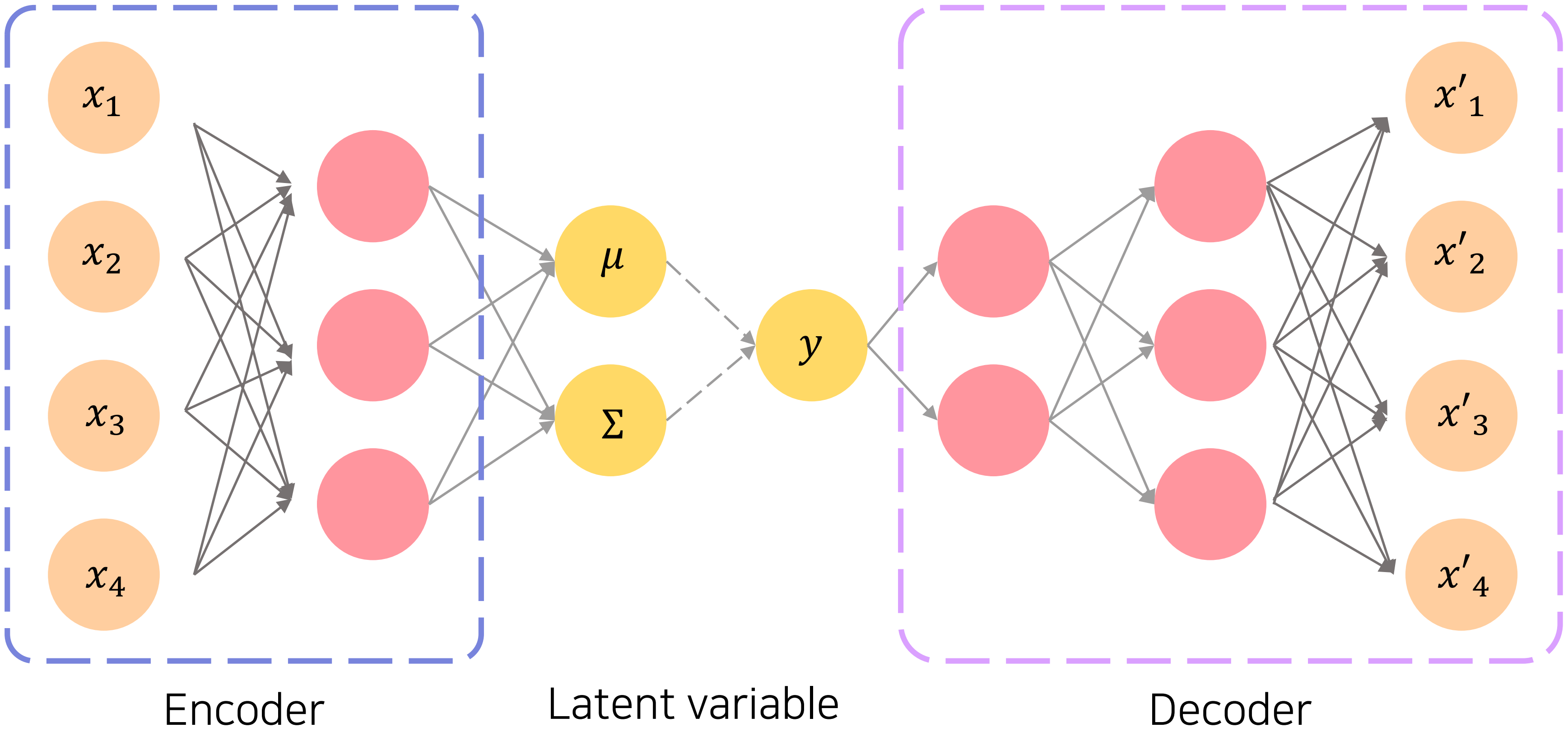
y_n

x_n

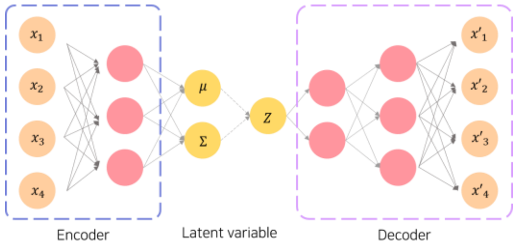
In VAE, Latent variables are
assumed to be **normal distribution**

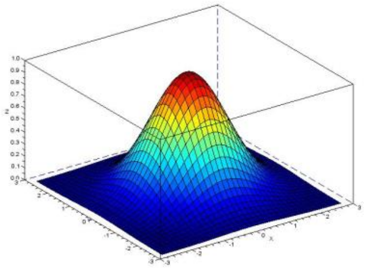
It is easier to update optimized value.

Variational Autoencoder



Variational Autoencoder





Generate Latent Space

$q_\phi(Z|X^{(i)})$

Generate a distribution of Latent variables

Generate Image Based on Latent Space

$P_\theta(X^{(i)}|Z)$

Generate similar output

Generated Image Based on Latent Space

x → y *or* Z → x

Approximate x

Variational Autoencoder

Generate Latent Space

$q_\phi(Z|X^{(i)})$

Generate a distribution of Latent variables

Generate Image Based on Latent Space

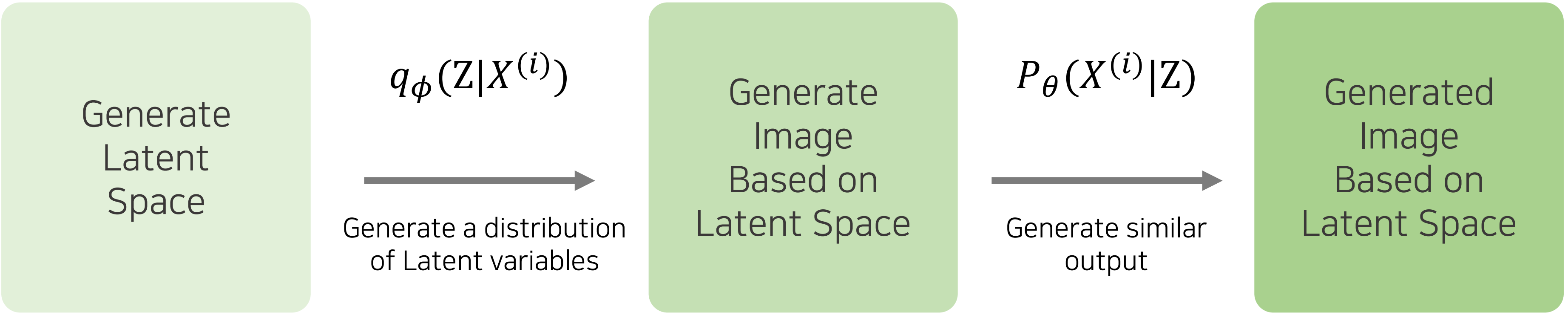
$P_\theta(X^{(i)}|Z)$

Generate similar output

Generated Image Based on Latent Space

HOW TO OPTIMIZE IT?

Variational Autoencoder



Simply…

Maximize $ELBO(\phi)$

Minimize $KL(q_\phi(z|x)||p(z|x))$

Variational Autoencoder

Maximize $ELBO(\phi)$

Minimize $KL(q_\phi(z|x)||p(z|x))$

$$\log(p(x)) = \log\left(\int p(x,z)dz\right) = \log\left(\int p(x|z)p(z)dz\right) = \log\left(\int p(x|z)\frac{p(z)}{q_\phi(z|x)}q_\phi(z|x)dz\right)$$

Jenson's Inequality

$$\geq \int \log\left(p(x|z)\frac{p(z)}{q_\phi(z|x)}\right)q_\phi(z|x)dz = \int \log(p(x|z))q_\phi(z|x)dz - \int \log\left(\frac{q_\phi(z|x)}{p(z)}\right)q_\phi(z|x)dz$$

$$\int \log(p(x|z))q_\phi(z|x)dz = \mathbb{E}_{q_\phi(z|x)}[\log(p(x|z))]$$

$$\int \log\left(\frac{q_\phi(z|x)}{p(z)}\right)q_\phi(z|x)dz = KL(q_\phi(z|x)\|p(z))$$

$$ELBO(\phi) = \mathbb{E}_{q_\phi(z|x)}[\log(p(x|z))] - KL(q_\phi(z|x)\|p(z))$$

Maximize the ELBO and approximate it to q(x) as p(x) <Variational Method>

Find argmax $ELBO(\phi)$

Variational Autoencoder

Maximize $ELBO(\phi)$

Minimize $KL(q_\phi(z|x)||p(z|x))$

<Loss function from ELBO method>

$$\mathcal{L}_{(\theta,\phi;x^i)} = -\mathbb{E}_{q_\phi(z|x^i)}[\log(p_\theta(x^i|z))] + KL(q_\phi(z|x^i)\|p(z))$$

Multivariate Kullback-Leibler Divergence

$$KL(q_\phi(z|x^i)\|p(z) = \frac{1}{2}\left\{tr(\sigma_i^2) + \mu_i^T\mu_i - J + \ln\frac{1}{\prod_{j=1}^{J}\sigma_{j,j}^2}\right\}$$

$$= \frac{1}{2}\left\{\sum_{j=1}^{J}\sigma_{i,j}^2 + \sum_{j=1}^{J}\mu_{i,j}^2 - J - \sum_{j=1}^{J}\ln(\sigma_{i,j}^2)\right\}$$

$$= \frac{1}{2}\sum_{j=1}^{j}(\mu_{i,j}^2 + \sigma_{i,j}^2 - \ln(\sigma_{i,j}^2) - 1)$$

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn}$$

Variational Autoencoder

Expectation

$$\mathbb{E}_{q_\phi(z|x^i)}[\log(p_\theta(x^i|z))]$$
$$= \int log(p_\theta(x_i|z))q_\phi(z|x_i)dz$$

$\downarrow$ Monte-carlo 방식.

$$\mathbb{E}_{q_\phi(z|x^i)}[\log(p_\theta(x^i|z))] \approx \frac{1}{L}\sum_{z^{i,l}} \log(p_\theta(x^i|z^{i,l}))$$

$\downarrow$ Bernoulli 분포

$$\log(p_\theta(x^i|z^i)) = \log \prod_{j=1}^{D} p_\theta(x_{i,j}|z^i)$$
$$= \sum_{j=1}^{D} \log p_\theta(x_{i,j}|z^i)$$
$$= \sum_{j=1}^{D} \log p_{i,j}^{x_{i,j}}(1-p_{i,j})^{x_{i,j}}$$
$$= \sum_{j=1}^{D} x_{i,j}\log p_{i,j} + (1-x_{i,j})\log(1-p_{i,j})$$

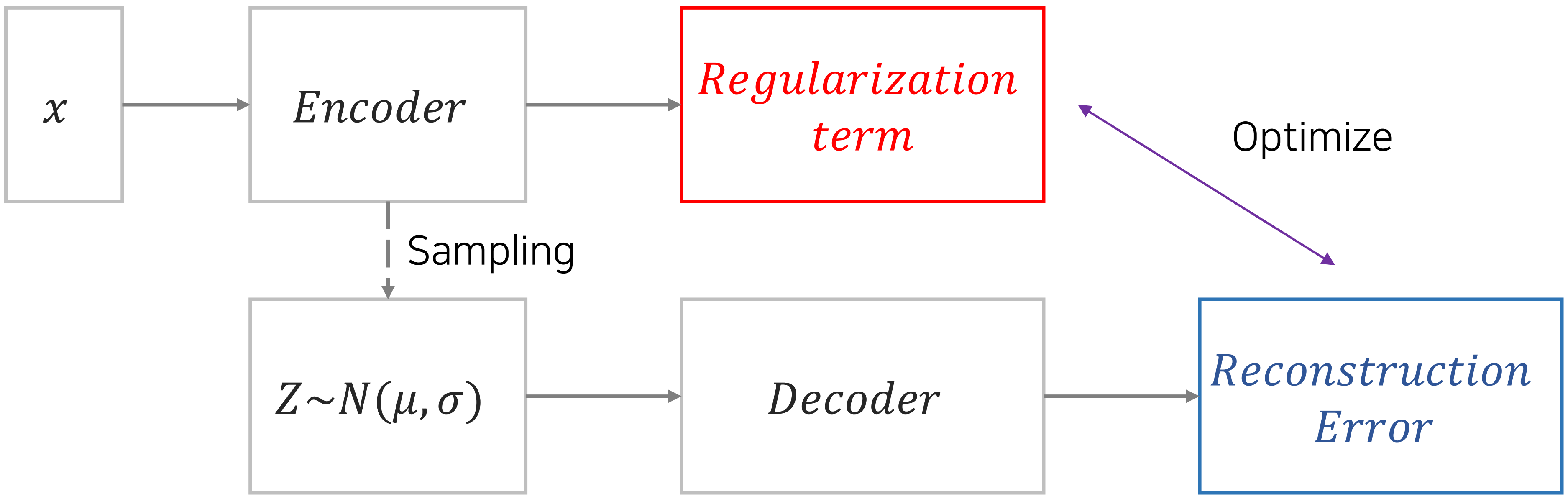
Multivariable Bernoulli Distribution

$$\mathbf{X} = (X_1, \cdots, X_n) \sim (\mathbf{Bern}(\theta_1), \cdots, \mathbf{Bern}(\theta_n)) \stackrel{\text{let}}{=} \mathbf{Bern}_n(\boldsymbol{\theta})$$

$$\mathbf{Bern}_n(\mathbf{x};\boldsymbol{\theta}) = \prod_{i=1}^{n} \theta_i^{x_i}(1-\theta_i)^{1-x_i}$$

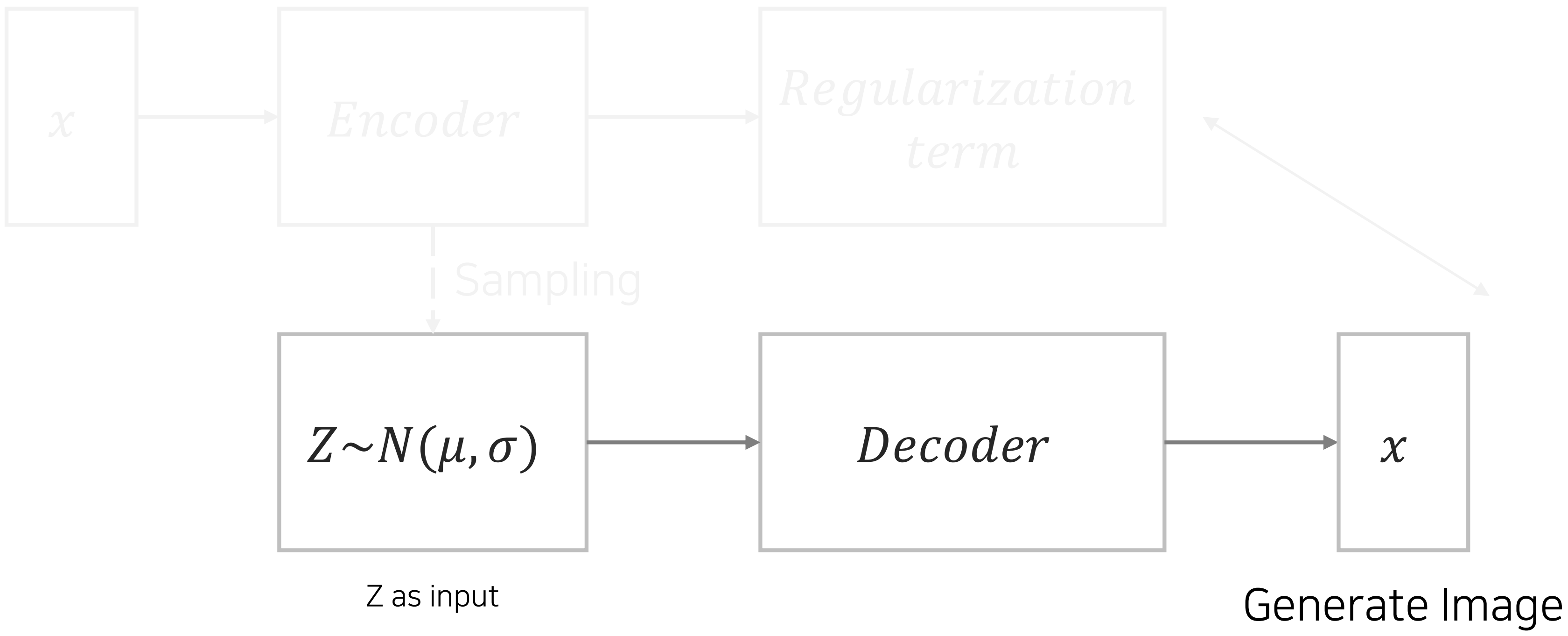
Variational Autoencoder

Training Process



Variational Autoencoder

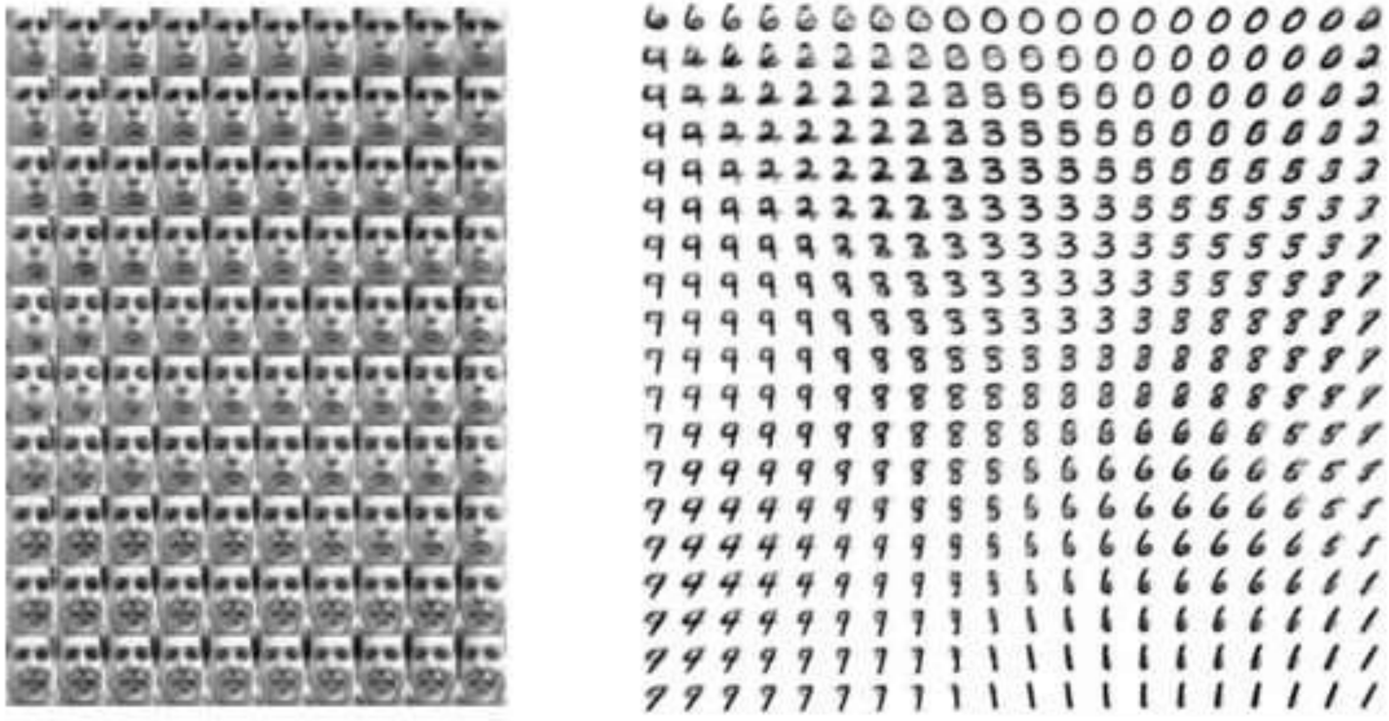
Generate Process



Variational Autoencoder

x

Generate Image



Generate images that correspond to latent space