Variational Autoencoders

Reference

- Auto-encoding variational Bayes. Diederik P Kingma and Max Welling, ICLR, 2014.
- Tutorial on Variational Autoencoders, Carl Doersch, arxiv:1606.05908v2.
- Variational Inference: A Review for Statisticians, David M Blei, Alp Kucukelbir, Jon D McAuliffe, arXiv:1601.00670v9.
- https://www.youtube.com/watch?v=uaaqyVS9-rM&feature=youtu.be&t=19m42s (mathematical derivation)
- https://www.jeremyjordan.me/variational-autoencoders/ (good read)

Motivation

- Autoencoder has “encoding” part and “decoding” part.
  - Encoding part can be “learning” latent space
  - From latent space, observation is decoding the latent space
  - Latent space can only be observed “probabilistically”
  - Assign probability distribution to the interface of encoder and decoder
  - Simple autoencoder is learning definite value of encoder and decoder
  - Variational autoencoder is learning probability distribution of latent space (interface between encoder and decoder)

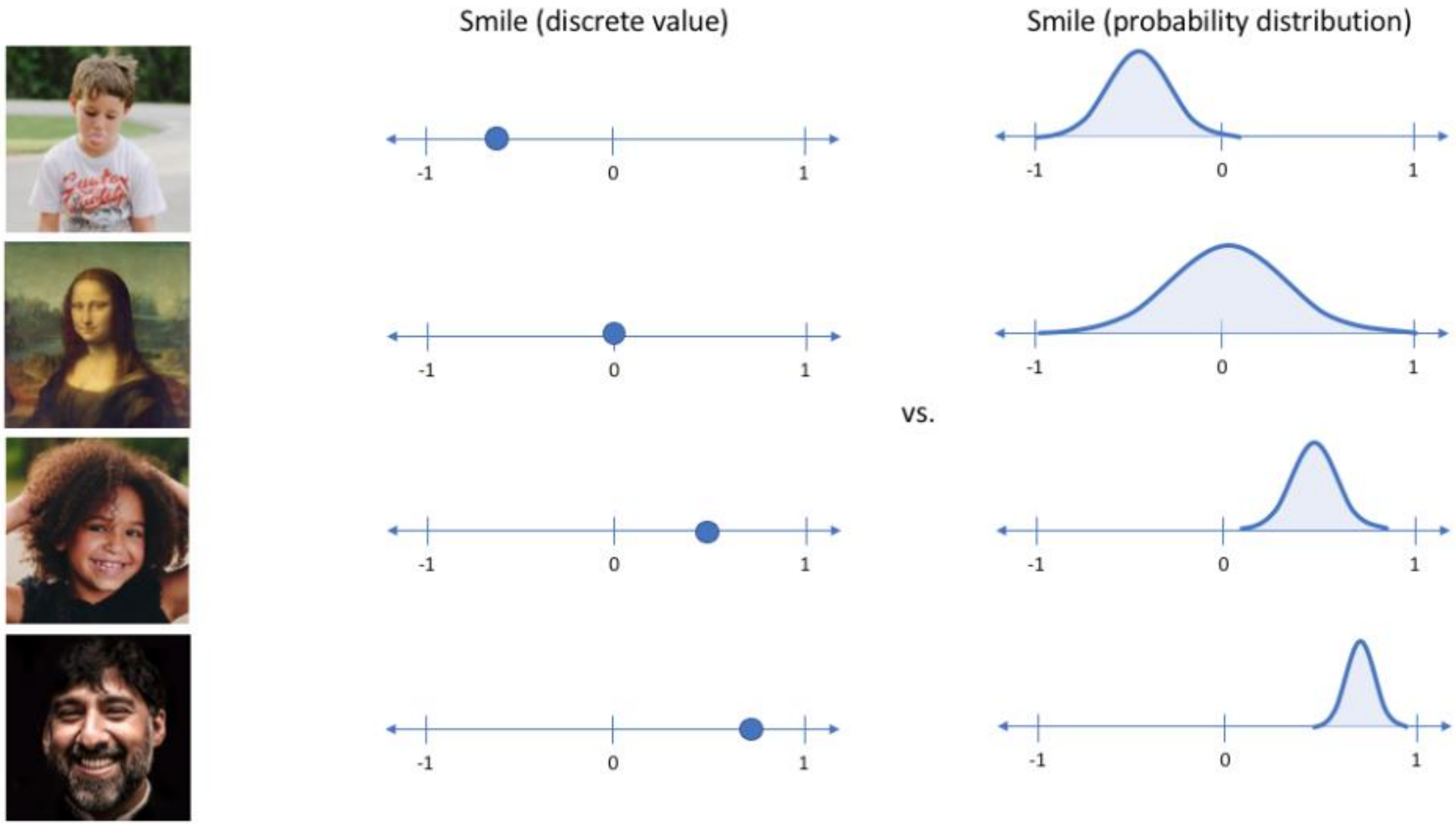
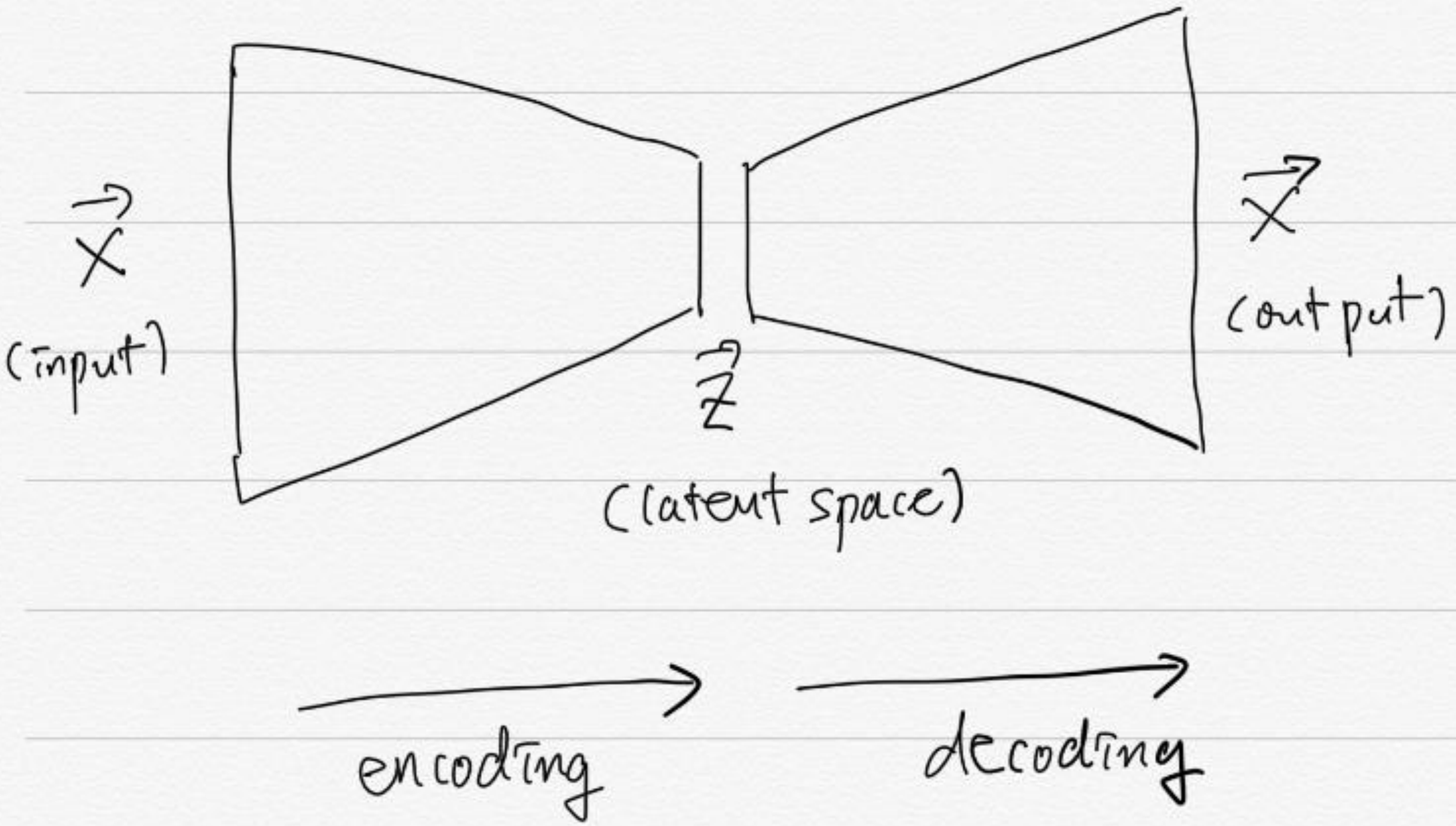
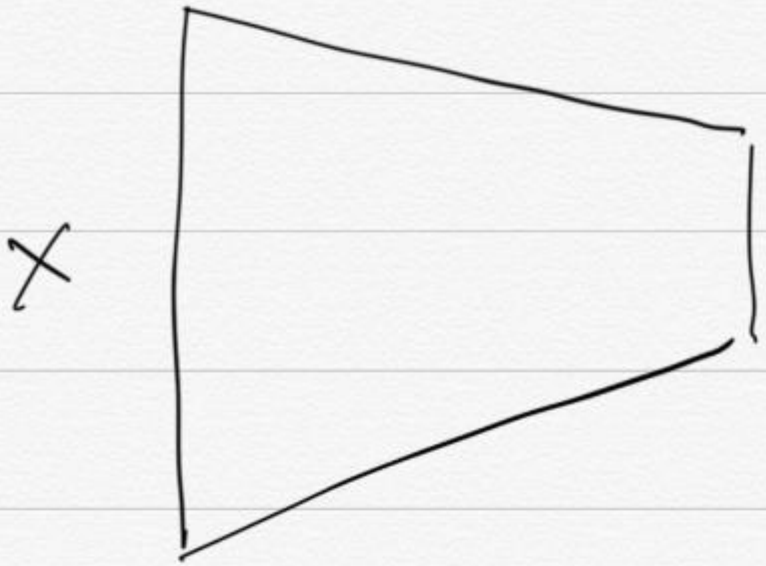


Image from https://www.jeremyjordan.me/variational-autoencoders/

Autoencoder



"Encoding"



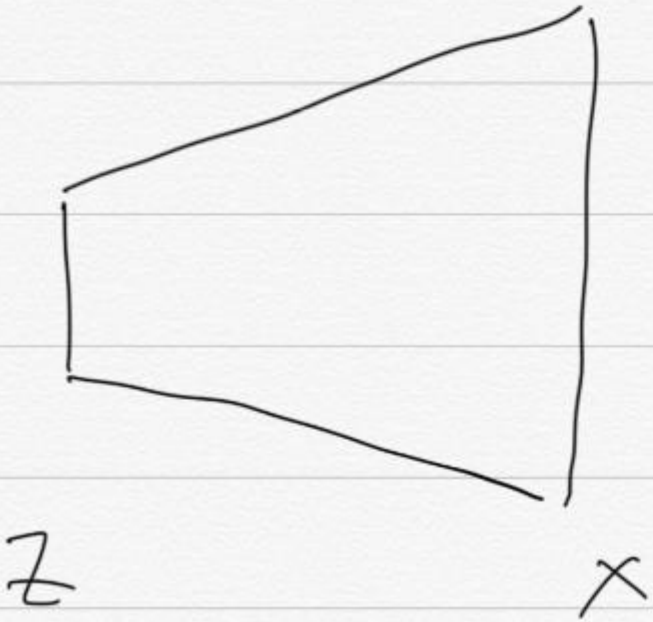
Prior

$$P(z|x) = \frac{P(x|z)\,P(z)}{P(x)}$$

posterior

Goal: we want to learn $P(z|x)$ posterior distribution

"Decoding" : reconstruct data



$p(x|z)$

sample $\vec{z}$ from distribution $p(z|x)$ and then insert to "deterministic" decoder $p(x|z)$

Conventional autoencoder:

doesn't learn the distribution of

$p(z|x)$ but simply train $p(z|x) \to p(x|z)$

$\Rightarrow p(x|x)$

deterministically.

Challenge:

prior: given

$$p(z|x) = \frac{p(x|z)\,p(z)}{p(x)}$$

learnable

$$p(x) = \int p(x|z)\,p(z)\,\underline{dz}$$

intractable integration

Solution:

(1) Monte Carlo (evaluate $\int p(x|z)p(z)dz$ directly $\rightarrow$ computationally expensive)

(2) Variational Inference:

approximate $p(z|x)$ with $q(z|x)$.

choose $q(z|x)$ in tractable form

Measure of "closeness"

KL divergence

$$KL(q(x) \,||\, p(x)) = -\sum_x q(x) \log\left[\frac{p(x)}{q(x)}\right]$$

$$\geq 0$$

$$\neq KL(p(x) \,||\, q(x))$$

Goal : minimize KL divergence

$$KL\left(q(z|x) \,\|\, p(z|x)\right) = -\sum_{z} q(z|x) \log\left[\frac{p(z|x)}{q(z|x)}\right]$$

x is given

$$= -\sum_{z} q(z|x) \log\left[\frac{p(x,z)}{q(z|x)} \frac{1}{p(x)}\right]$$

$$= -\sum_{z} q(z|x) \log\left[\frac{p(x,z)}{q(z|x)}\right] + \underbrace{\sum_{z} q(z|x)}_{=1} \log p(x)$$

$$= -\sum_{z} q(z|x) \log\left[\frac{p(x,z)}{q(z|x)}\right] + \log p(x)$$

$$= KL\left(q(z|x) \,||\, p(z|x)\right)$$

$$\Rightarrow \quad KL(q(z|x) \,\|\, p(z|x)) + \sum_{z} q(z|x) \log\left[\frac{p(x,z)}{q(z|x)}\right]$$

$$= \log p(x) \quad : \text{constant. (independent of choice of } q(z|x))$$

$\Rightarrow$ minimize KL divergence is equivalent to

maximize $\mathcal{L} = \sum_{z} q(z|x) \log\left[\frac{p(x,z)}{q(z|x)}\right]$

Since KL divergence ≥ 0,

$\mathcal{L} \leq \log p(x)$: $\mathcal{L}$ is a lower bound of $\log p(x)$

$\Rightarrow \mathcal{L}$ is a "variational" lower bound of $\log p(x)$.

Now, maximize $\mathcal{L}$:

$$\sum_z q(z|x) \log \frac{p(x,z)}{q(z|x)}$$

$p(x,z) = p(x|z)\,p(z)$

prior of latent space z

$$= \sum_z q(z|x) \log \frac{p(x|z)\,p(z)}{q(z|x)}$$

$$= \sum_z q(z|x) \log p(x|z) - \Big(\underbrace{-\sum_z q(z|x) \log \frac{q(z|x)}{p(z)}}_{= KL\,(q(z|x)\,\|\,p(z))} \Big)$$

$$= \mathop{E}_{z \sim q} \left[\log p(x|z) \right] - KL\,(q(z|x)\,\|\,p(z))$$

Maximize :

$$\underbrace{\mathop{E}_{z \sim q}\left[\log p(x|z)\right]}_{\Downarrow} - \underbrace{KL\left(q(z|x) \,\|\, p(z)\right)}_{\Downarrow}$$

maximize log likelihood of reconstruction

minimize KL

→ make $q(z|x)$ as similar as to prior $p(z)$

Graphically :

$$
\begin{tikzcd}
z \arrow[d, bend right=60, "p(x|z)"'] & \text{latent space} \arrow[l] \\
x \arrow[u, bend right=60, "p(z|x) \simeq q(z|x)"'] & \text{observation space} \arrow[l]
\end{tikzcd}
$$

$p(x|z)$

$\rightarrow$ learning is doable

by maximizing

log expectation

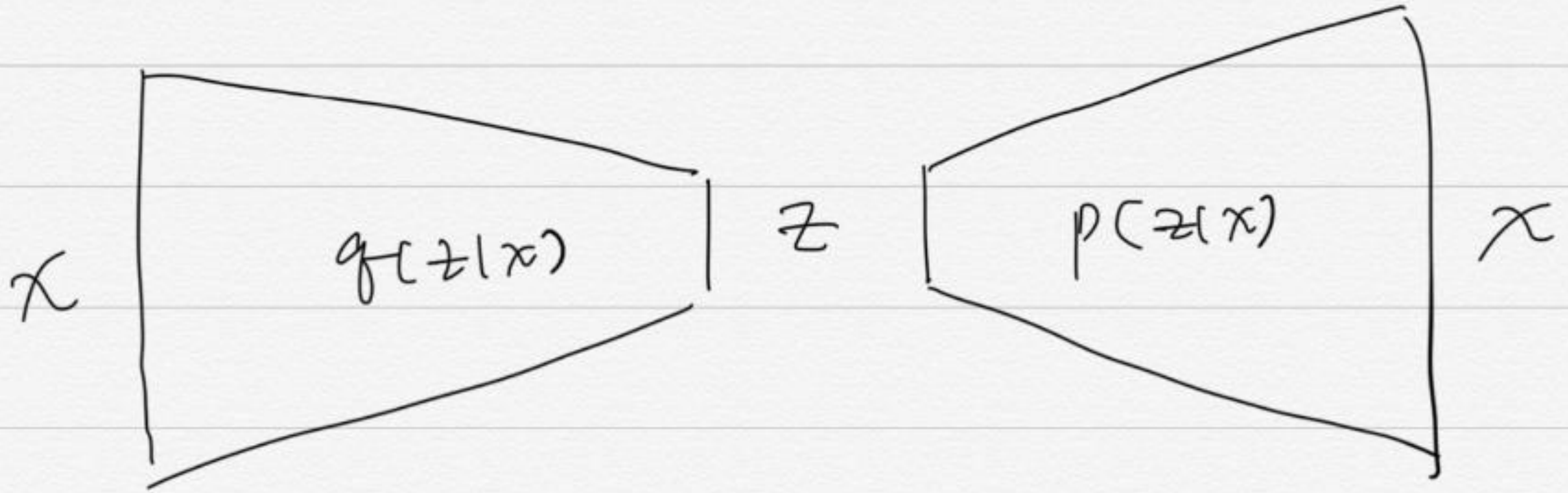
$p(z|x)$

$\rightarrow$ learning is hard

so approximate by

$q(z|x)$

Variational Autoencoder



encoder : $-KL(q(z|x)\,||\,p(z))$

decoder : $\mathbb{E}_{z}\left[\log p(x|z)\right]$

make $q(z|x)$ similar to "my choice" $p(z)$

minimize reconstruction error
deterministic algorithm.

(1) maximize $\underset{z}{E}[\log p(x|z)]$

if $p(x|z) \sim N(f(z), I)$: $f(z)$ = decoder learning

$$\sim e^{-|x-f(z)|^2}$$

$$\Rightarrow \min\left[\sum_z |x-f(z)|^2\right] \rightarrow \text{standard square loss minimization}$$

if $p(x|z) \sim$ Bernouli (learning $f(z) \in [0,1]$)

$$\Rightarrow \min\left[\sum\left[x \log f(z) + (1-x)\log(1-f(z))\right]\right]$$

$\rightarrow$ standard cross entropy minimization

(2)
minimize $KL(q(z|x) \,||\, p(z))$

: Choose $p(z) \sim \mathcal{N}(0, I)$ → prior choice

$q(z|x) \sim \mathcal{N}(\mu, \Sigma)$

↳ choose $D \times D$ diagonal

⇒ Tractable choice (Gaussian)

$KL(q(z|x) \| p(z))$

$$= \frac{1}{2}\left(tr(\Sigma) + \mu^T \cdot \mu - D - \log\det(\Sigma) \right)$$

(general form of $q(z|x) \sim N(\vec{\mu}, \Sigma)$

and $p(z) \sim N(0, I)$)

learn μ & Σ from gradient descent algorithm

and generate $\vec{z} \sim N(\mu, \Sigma)$

feed generated $\vec{z}$ in to decoder $p(x|z)$

Or, generate z from $N(0, I)$ and feed into

$p(x|z)$ to generate artificial output.

"Generative Model"

Challenge: how to train the network?

Each time, input $X \longrightarrow$ compute μ & Σ

$\rightarrow$ "sample" z from $\mathcal{N}(\mu, \Sigma)$

$\rightarrow$ feed z to decoder $p(x|z)$

$\rightarrow$ compute log likelihood

$\rightarrow$ Backpropagate: impossible for "sampling" stage

Solution : Reparametrization Trick :

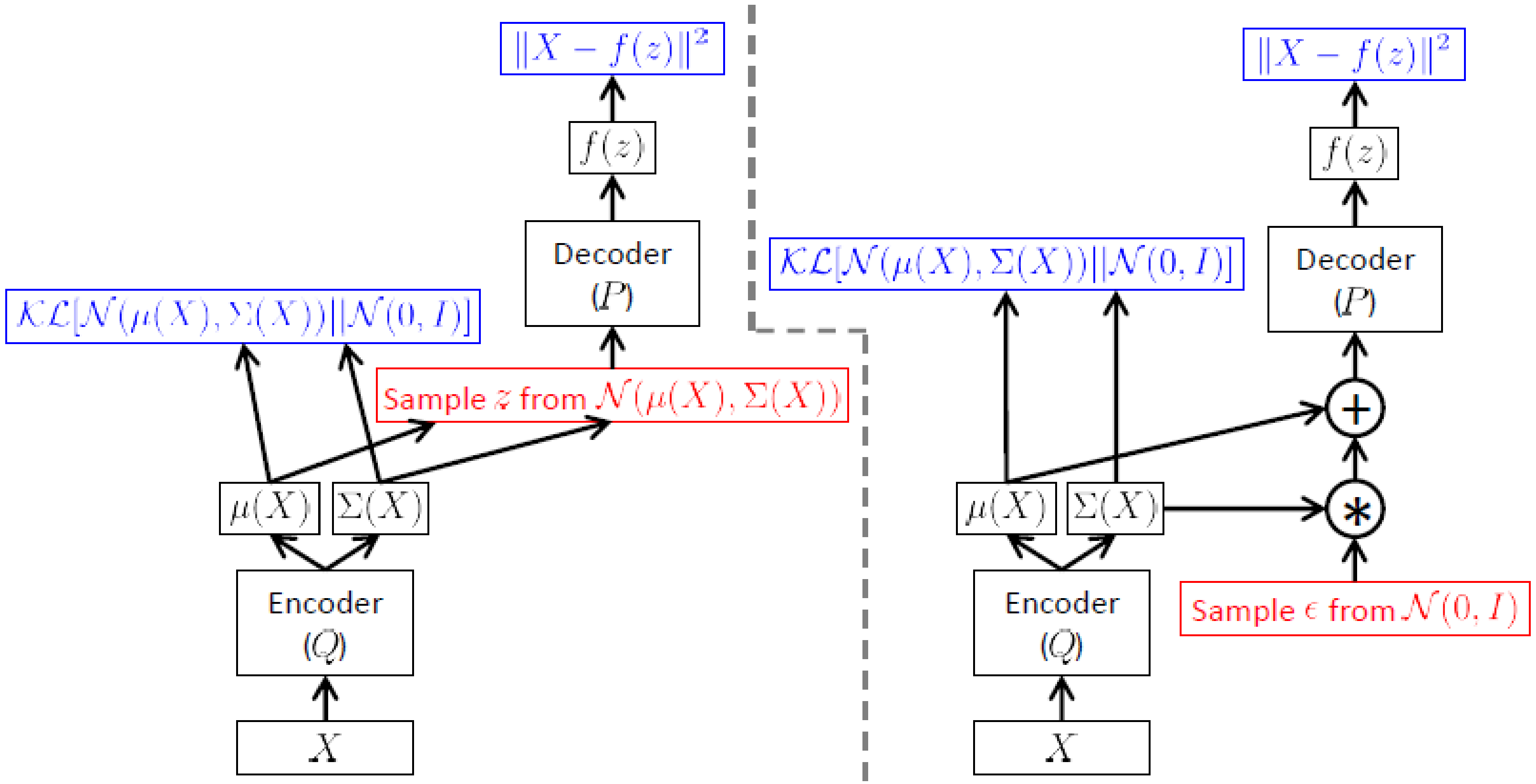


Image: Figure 4 from arXiv:1606.05908v2

$$z = \mu + \Sigma^{1/2} \cdot \varepsilon \rightarrow z \sim N(\mu, \Sigma)$$

Since Σ is positive semi definite,
train $\log(\Sigma)$ instead and exponentiate.

Learn mu and sigma from input X and treat epsilon as another input variables to generate random z

Q: Why $p(z) \sim N(0, I)$ prior would be enough to generate complex output?

A: With enough non-linearity in $p(x|z)$ (decoder network), we can learn complex structure from simple $N(0, I)$

Example:

$$z \sim N\left(0, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$$

$$u = \frac{z}{10} + \frac{z}{\|z\|}$$ → distributed on a ring.

Meaning of $\underbrace{E_z[\log p(x|z)]}_{①} - \underbrace{KL(q(z|x) \| p(z))}_{②}$

① : Try to reconstruct input as much as possible (usual learning algorithm)

② : Enforce $q(z|x)$ to have simple distribution

→ Regularization.