Variational Inference

Spring 2024

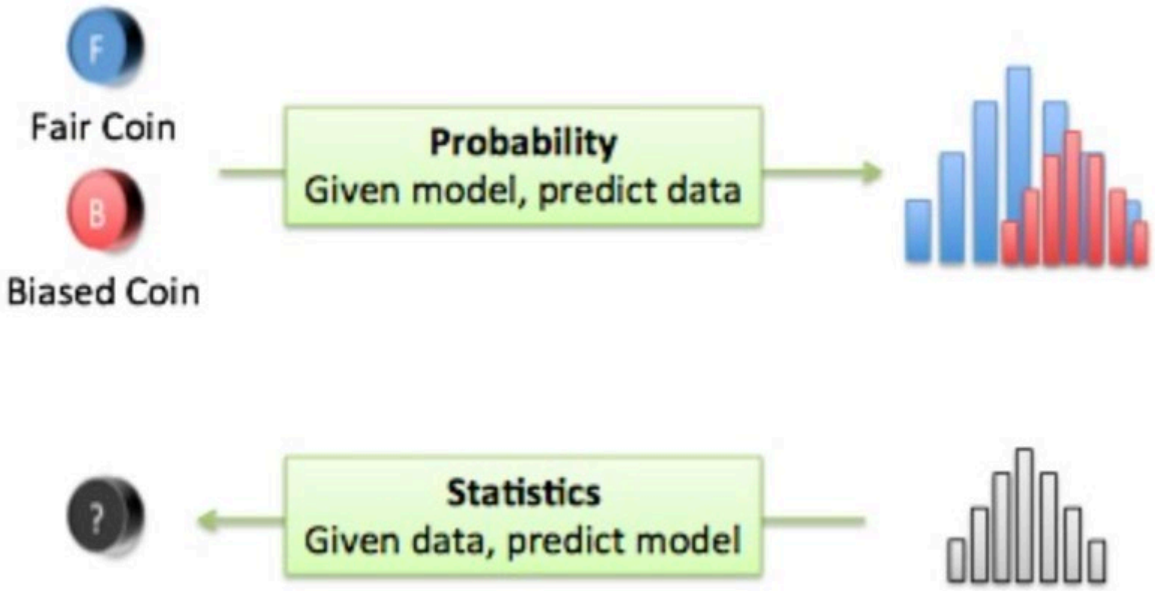
Introduction

- **Probability and Statistical Inference**
- **Expectation Maximization**
- **Variational Inference**
- **Variational Autoencoder & Bayesian NN**

Probability and Statistical Inference

- **The basic problem that we study in Probability is:**
  Given a data generating process, what are the properties of the outcomes?
- **The basic problem of Statistical Inference is the inverse of Probability:**
  Given the outcomes, what can we say about the process that generated the data?

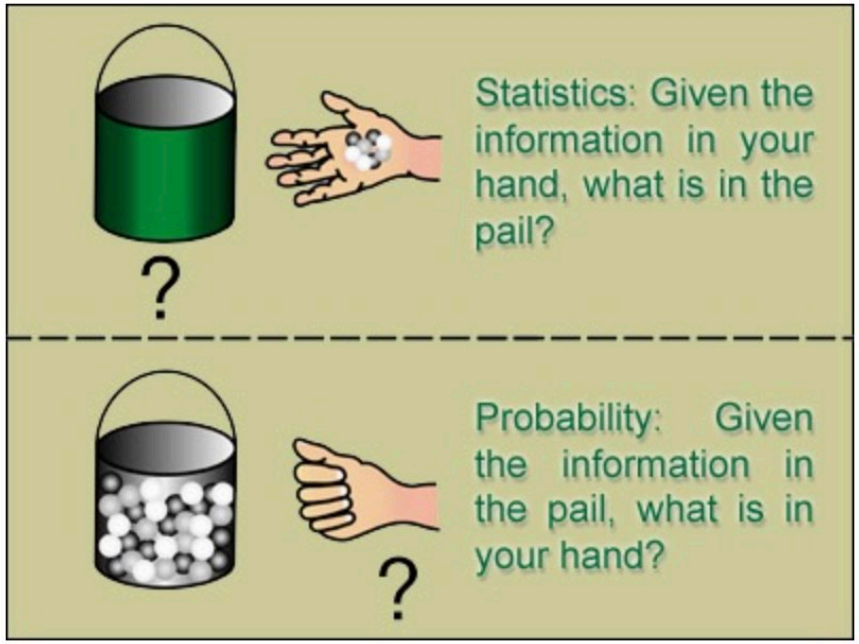
—— Larry Wasserman《All of Statistics》



Probability and Statistical Inference

- **The basic problem that we study in Probability is:**
  Given a data generating process, what are the properties of the outcomes?
- **The basic problem of Statistical Inference is the inverse of Probability:**
  Given the outcomes, what can we say about the process that generated the data?

—— Larry Wasserman《All of Statistics》



Probability and Statistical Inference

Likelihood & Probability（似然与概率）

In English, they are synonyms. In Colins,

The **likelihood/probability** of something happening is how likely it is to happen.

In statistics, they have different definitions.

- **Probability：** Given the model, to predict the outcomes
- **Likelihood：** Given the data, to predict the values of some parameters

Likelihood function & Probability function: $P(X/\Theta)$

- **Probability function:** the probability of different sample X, in which Θ is given and fixed, X is variable.
- **Likelihood function:** the probability of X under different Θ, in which X is given and fixed, Θ is variable. Also noted as $\mathcal{L}(\Theta/X)$

MLE：Maximum Likelihood Estimation

最大似然估计（MLE）

- 估计概率模型参数的一种方法（model parameter estimation）
- **Basic idea：pick Θ that "makes data most likely"**

$$\hat{\theta}_{\mathrm{ML}} = \arg\max_{\theta} p_X(x;\theta)$$

- 对 i.i.d. 的样本集来说，总体的似然就是每个样本似然的乘积。

$$\mathcal{L}(\Theta/X) = P(X/\Theta) = P(x_1, x_2, ..., x_n/\Theta) = \prod_{i=1}^{n} P(x_i/\Theta)$$

Procedures for MLE

- **Find likelihood function**
- **Transform to log-likelihood function** $ln\mathcal{L}(\Theta/X) = lnP(X/\Theta) = \sum_{i=1}^{n} lnP(x_i/\Theta)$
- **Maximize log-likelihood function**

MLE：Maximum Likelihood Estimation

正态分布的最大似然估计

假设样本服从正态分布 $N \sim (\mu, \sigma^2)$，则其似然函数为

$$L(\mu, \sigma^2) = \prod_{i=0}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

对其取对数得：

$$lnL(\mu, \sigma^2) = -\frac{n}{2} ln(2\pi) - \frac{n}{2} ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=0}^{n} (x_i - \mu)^2$$

MLE：Maximum Likelihood Estimation

分别对 μ，σ^2 求偏导，并令偏导数为0，得：

$$\begin{cases} \frac{\partial lnL(\mu,\sigma^2)}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=0}^{n}(x_i-\mu) = 0 \\ \\ \frac{\partial lnL(\mu,\sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}\sum_{i=0}^{n}(x_i-\mu)^2 = 0 \end{cases}$$

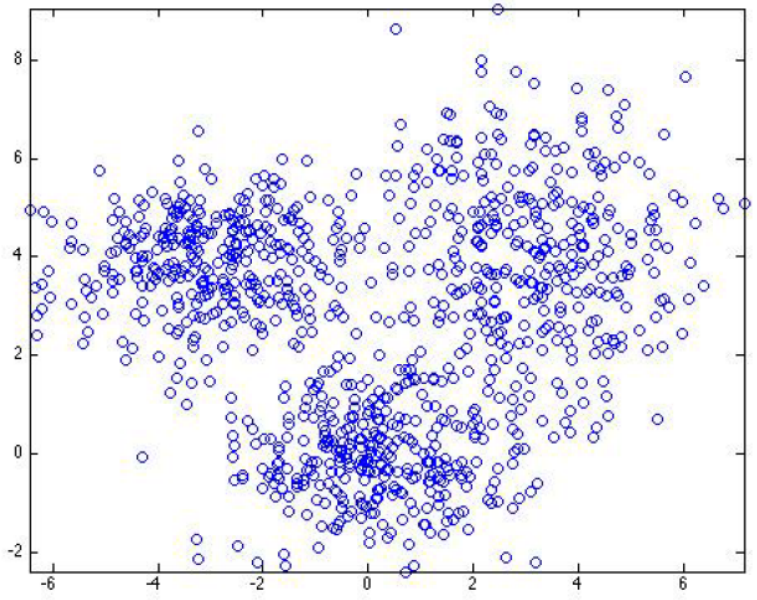
解得：

$$\begin{cases} \widehat{\mu} = \frac{1}{n}\sum_{i=0}^{n} x_i = x^2 \\ \\ \widehat{\sigma^2} = \frac{1}{n}\sum_{i=0}^{n}(x_i-x^2)^2 \end{cases}$$

$\widehat{\mu}$，$\widehat{\sigma^2}$ 就是正态分布中 μ，σ^2 的最大似然估计。

Expectation Maximization

When you have data that looks like:



Can you fit them using a single-mode Gaussian distribution, i.e.,:

$$p(X) = \mathcal{N}(X|\mu, \Sigma)$$
$$= (2\pi)^{-k/2}|\Sigma|^{-\frac{1}{2}} \exp^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

Clearly NOT! This is typically modelling using Mixture Densities, in the case of Gaussian Mixture Model (k-mixture) (GMM):

$$p(X) = \sum_{l=1}^{k} \alpha_l \mathcal{N}(X|\mu_l, \Sigma_l) \qquad \sum_{l=1}^{k} \alpha_l = 1$$

Gaussian Mixture Model

$$P(x) = \sum_z P(z)P(x|z)$$

How to sample?

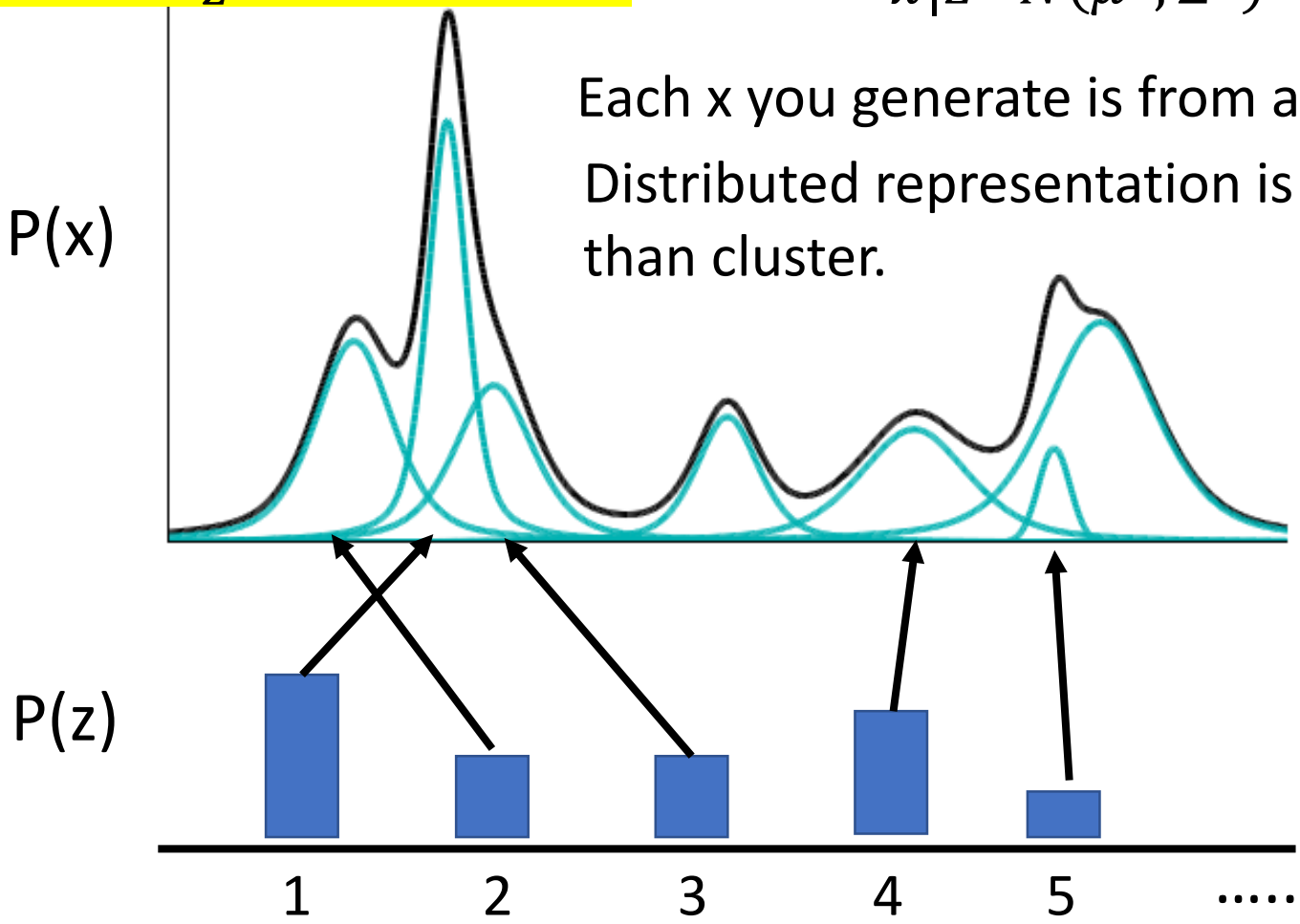
$z \sim P(z)$ (multinomial)

z is an integer

$x|z \sim N(\mu^z, \Sigma^z)$

Each x you generate is from a mixture

Distributed representation is better than cluster.



Expectation Maximization

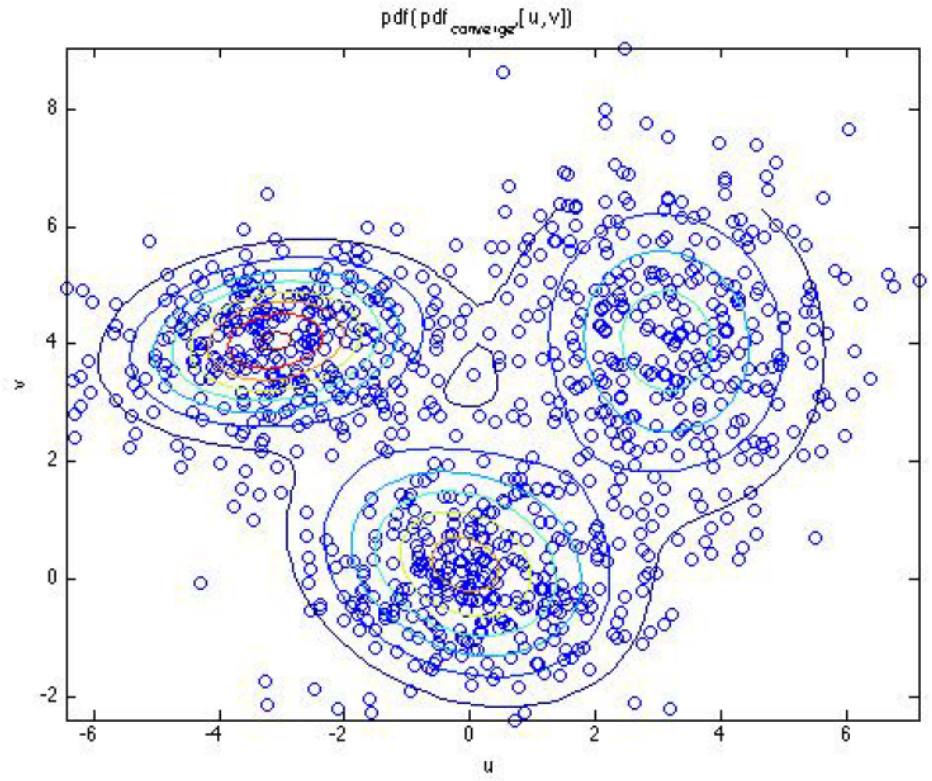


Figure: gmm fitting result

Let $\Theta = \{\alpha_1, \ldots \alpha_k, \mu_1, \ldots \mu_k, \Sigma_1, \ldots \Sigma_k\}$

$$\Theta_{\text{MLE}} = \arg\max_{\Theta} \mathcal{L}(\Theta|X)$$

$$= \arg\max_{\Theta} \left(\sum_{i=1}^{n} \log \sum_{l=1}^{k} \alpha_l \mathcal{N}(X|\mu_l, \Sigma_l) \right)$$

- Unlike single mode Gaussian, we can't just take derivatives and let it equal zero easily.
- We need to use Expectation-Maximization to help us solving this

Expectation Maximization

Instead of perform:

$$\theta^{\text{MLE}} = \arg\max_{\theta}(\mathcal{L}(\theta)) = \arg\max_{\theta}(\log[p(X|\theta)])$$

- **The trick** is to assume some "latent" variable Z to the model.
- such that we generate a series of $\Theta = \{\theta^{(1)}, \theta^{(2)}, \ldots \theta^{(t)}\}$

For each iteration of the E-M algorithm, we perform:

$$\Theta^{(g+1)} = \arg\max_{\theta}\left(\int_{z} \log\left(p(X, Z|\theta)\right) p(Z|X, \Theta^{(g)})\right) dz$$

However, we must ensure convergence:

$$\log[p(X|\Theta^{(g+1)})] = \mathcal{L}(\Theta^{(g+1)}) \geq \mathcal{L}(\Theta^{(g)}) \qquad \forall i$$

Expectation Maximization

$$\mathcal{L}(\theta|X) = \ln\left(p(X|\theta)\right) \qquad \frac{p(X,Z/\theta)}{p(Z/X,\theta)} = \frac{p(X/\theta)p(Z/X,\theta)}{p(Z/X,\theta)} = p(X/\theta)$$

$$= \ln\left(\frac{p(X,Z|\theta)}{p(Z|X,\theta)}\right) = \ln\left(\frac{\frac{p(X,Z|\theta)}{Q(Z)}}{\frac{p(Z|X,\theta)}{Q(Z)}}\right)$$

$$= \ln\left(\frac{p(X,Z|\theta)}{Q(Z)} \times \frac{Q(Z)}{p(Z|X,\theta)}\right)$$

$$= \ln\left(\frac{p(X,Z|\theta)}{Q(Z)}\right) + \ln\left(\frac{Q(Z)}{p(Z|X,\theta)}\right)$$

$$\implies \ln\left(p(X|\theta)\right) = \int_Z \ln\left(\frac{p(X,Z|\theta)}{Q(Z)}\right) Q(Z) + \int_Z \ln\left(\frac{Q(Z)}{p(Z|X,\theta)}\right) Q(Z)$$

$$= \int_Z \ln\left(\frac{p(X,Z|\theta)}{Q(Z)}\right) Q(Z) + \underbrace{\mathrm{KL}(Q(Z)\|p(Z|X,\theta))}_{\geq 0}$$

$$= F(\theta, Q) + \int_Z \ln\left(\frac{Q(Z)}{p(Z|X,\theta)}\right) Q(Z)$$

Expectation Maximization

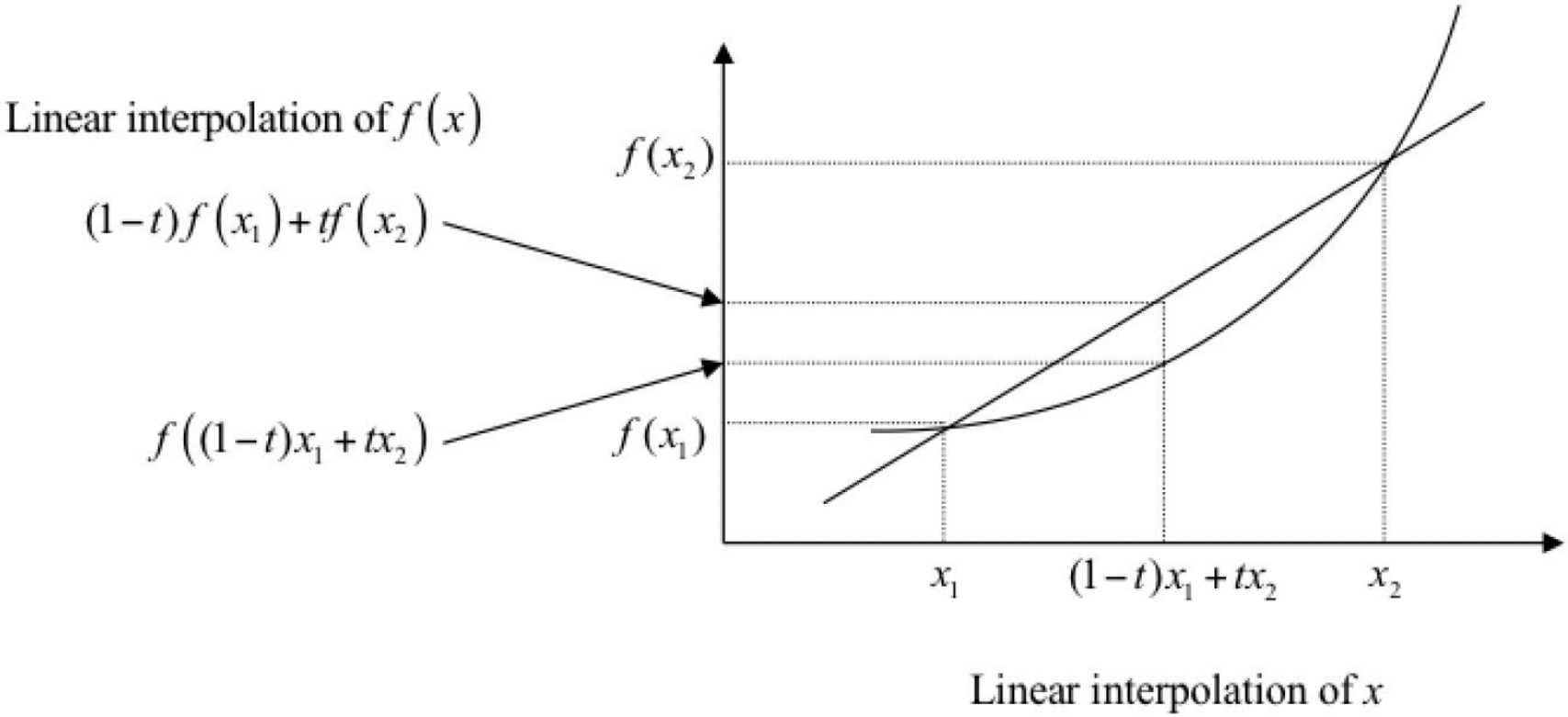
Another way of knowing:

$$\mathcal{L}(\theta|X) = \ln\left(p(X|\theta)\right) \geq \int_Z \ln\left(\frac{p(X,Z|\theta)}{Q(Z)}\right) Q(Z)$$

is to use Jensen's inequality:

$$\begin{aligned}
\mathcal{L}(\theta|X) &= \ln p(X|\theta) = \ln \int_Z p(X,Z|\theta) \\
&= \underbrace{\ln\left(\int_Z \frac{p(X,Z|\theta)}{Q(Z)} Q(Z)\right)}_{\ln \mathbb{E}_{Q(Z)}[f(Z)]} \\
&\geq \underbrace{\int_Z \ln\left(\frac{p(X,Z|\theta)}{Q(Z)}\right) Q(Z)}_{\mathbb{E}_{Q(Z)} \ln[f(Z)]}
\end{aligned}$$

Jensen's inequality



$$f((1-t)x_1 + tx_2) \leq (1-t)f(x_1) + tf(x_2) \qquad t \in (0 \ldots 1)$$

Jensen's inequality

Using notation Φ instead of f:

$$\Phi\left((1-t)x_1 + tx_2\right) \leq (1-t)\Phi(x_1) + t\Phi(x_2) \qquad t \in (0 \ldots 1)$$

Can be generalised further, let $\sum_{i=1}^{n} p_i = 1$:

$$\Phi\left(p_1 x_1 + p_2 x_2 + \ldots p_n x_n\right) \leq p_1\Phi(x_1) + p_2\Phi(x_2) \ldots p_n\Phi(x_n) \qquad \sum_{i=1}^{n} p_i = 1$$

$$\implies \Phi\left(\sum_{i=1}^{n} p_i x_i\right) \leq \sum_{i=1}^{n} p_i \Phi(x_i)$$

$$\implies \Phi\left(\sum_{i=1}^{n} p_i f(x_i)\right) \leq \sum_{i=1}^{n} p_i \Phi(f(x_i)) \qquad \text{by replacing } x_i \text{ with } f(x_i)$$

Can also generalised to the continous case, by letting $\int_{x\in\mathbb{S}} p(x) = 1$:

$$\Phi\left(\int_{x\in\mathbb{S}} f(x)p(x)\right) \leq \int_{x\in\mathbb{S}} \Phi(f(x_i))p(x) \implies \Phi\mathbb{E}[f(x)] \leq \mathbb{E}[\Phi(f(x_i))]$$

Jensen's inequality

$\Phi(x) = -\log(x)$ is a convex function:

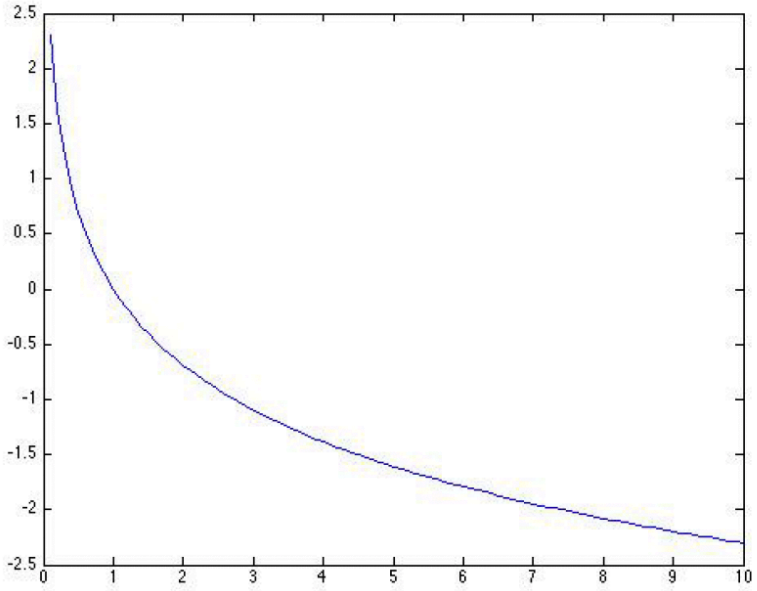


Figure: plot of $\Phi(x) = -\log(x)$

when $\Phi(.)$ is convex

$$\Phi\mathbb{E}[f(x)] \leq \mathbb{E}[\Phi(f(x_i))]$$
$$\implies -\log\mathbb{E}[f(x)] \leq \mathbb{E}[-\log(f(x_i))]$$

when $\Phi(.)$ is concave

$$\Phi\mathbb{E}[f(x)] \geq \mathbb{E}[\Phi(f(x_i))]$$
$$\implies -\log\mathbb{E}[f(x)] \geq \mathbb{E}[-\log(f(x_i))]$$

Expectation Maximization

E-M becomes a M-M algorithm

$$\begin{aligned}\mathcal{L}(\Theta|X) &= \int_Z \ln\left(\frac{p(X,Z|\Theta)}{Q(Z)}\right) Q(Z) + \int_Z \ln\left(\frac{Q(Z)}{p(Z|X,\Theta)}\right) Q(Z) \\ &= F(\Theta, Q) + \mathrm{KL}(Q(Z)||p(Z|X,\Theta))\end{aligned}$$

STEP 1 Fix $\Theta = \Theta^{(g)}$, maximize $Q(Z)$

- $\mathcal{L}(\Theta|X)$ is fixed, i.e., indepedant of $Q(Z)$. Therefore, $\mathcal{L}(\Theta|X)$ is the upper bound of $F(\Theta, Q)$.
- To make $\mathcal{L}(\Theta|X) = F(\Theta, Q)$, i.e, $\mathrm{KL}(.) = 0$, we choose $Q(Z) = p(Z|X, \Theta^{(g)})$. Therefore:

$$\mathcal{L}(\Theta|X) = \int_Z \ln\left(\frac{p(X,Z|\Theta)}{p(Z|X,\Theta^{(g)})}\right) p(Z|X,\Theta^{(g)})$$

STEP 2 Fix $Q(Z)$, maximize Θ

$$\Theta^{(g+1)} = \arg\max_{\Theta}\left(\int_z \log\left(p(X,Z|\Theta)\right) p(Z|X,\Theta^{(g)})\right) dz$$

MAP：Maximum A Posteriori estimation

最大后验概率估计（MAP）

- Assume θ has a prior distribution
- Basic idea：pick θ that maximizes $P(X|\theta)P(\theta)$

$$\underset{\theta}{argmax}\, P(\theta|X) = \underset{\theta}{argmax}\, \frac{P(X|\theta)P(\theta)}{P(X)} \propto \underset{\theta}{argmax}\, P(X|\theta)P(\theta)$$

Posterior $\propto$ Likelihood · Prior

Procedures for MAP

- Find prior distribution and likelihood function
- Find posteriori distribution and transform it to logarithmic function
- Maximize logarithmic function

Bayesian Estimation

贝叶斯估计（Bayesian Estimation）

- **Extension of MAP**
- **MLE 和 MAP 都是假设 θ 未知，但是确定的值，将使函数取得最大值的 θ 作为估计值**
- **贝叶斯估计，假设参数 θ 是未知的随机变量，不是确定值，求解的是参数 θ 在样本 X 上的后验分布。**

Bayesian Theorem：

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)} = \frac{P(X|\theta)P(\theta)}{\int_{\Theta} P(X|\theta)P(\theta)d\theta}$$

$P(X)$：also called as evidence, which is generally intractable

Variational Inference：approximate to posteriori distribution with simple distribution

Variational Inference

- It is universally true that:

$$p(X, Z) = p(X)p(Z/X)$$

$$\ln\left(p(X)\right) = \ln\left(p(X, Z)\right) - \ln\left(p(Z|X)\right)$$

- It's also true (a bit silly) that:

$$\ln\left(p(X)\right) = [\ln(p(X, Z)) - \ln(q(Z))] - [\ln(p(Z|X)) - \ln(q(Z))]$$

- The above is so that we can insert an arbitrary pdf $q(Z)$ into, now we get:

$$\ln\left(p(X)\right) = \ln\left(\frac{p(X, Z)}{q(Z)}\right) - \ln\left(\frac{p(Z|X)}{q(Z)}\right)$$

- Taking the expectation on both sides, given $q(Z)$:

$$\begin{aligned}
\ln\left(p(X)\right) &= \int q(Z) \ln\left(\frac{p(X, Z)}{q(Z)}\right) \mathrm{d}Z - \int q(Z) \ln\left(\frac{p(Z|X)}{q(Z)}\right) \mathrm{d}Z \\
&= \underbrace{\int q(Z) \ln(p(X, Z)) \mathrm{d}Z - \int q(Z) \ln(q(Z)) \mathrm{d}Z}_{\mathcal{L}(q)} + \underbrace{\left(- \int q(Z) \ln\left(\frac{p(Z|X)}{q(Z)}\right) \mathrm{d}Z\right)}_{\mathbb{KL}(q\|p)} \\
&= \mathcal{L}(q) + \mathbb{KL}(q\|p)
\end{aligned}$$

Variational Inference

$$\ln\left(p(X)\right) = \mathcal{L}(q) + \mathbb{KL}(q\|p)$$

- We can give a name to both terms:

Evidence Lower Bound (ELOB): $$\mathcal{L}(q) = \int q(Z)\ln(p(X,Z))\mathrm{d}Z - \int q(Z)\ln(q(Z))\mathrm{d}Z$$

KL divergence: $$\mathbb{KL}(q\|p) = \int q(Z)\ln\left(\frac{p(Z|X)}{q(Z)}\right)dZ$$

- Notice $p(X)$ is fixed with respect to the choice of $q(Z)$. We wanted to choose a $q(Z)$ function that minimize KL divergence, so that $q(Z)$ becomes closer and closer to $p(Z|X)$. Of course, let's see what happens when $q(Z) = p(Z|X)$:

$$\mathbb{KL}(q\|p) = -\int p(Z|X)\ln\left(\frac{p(Z|X)}{p(Z|X)}\right)\mathrm{d}Z = 0$$

- We know that $p(X) = \mathcal{L}(q) + \mathbb{KL}(q\|p)$. Minimizing $\mathbb{KL}(q\|p)$ is the same as maximizing the Evidence Lower Bound $\mathcal{L}(q)$.

Variational Inference

- Suppose let's choose $q(Z)$, such that:

$$q(Z) = \prod_{i=1}^{M} q_i(Z_i)$$

- Substitute this choice into Evidence Lower Bound (ELOB):

$$\begin{aligned}\mathcal{L}(q) &= \int q(Z) \ln(p(X,Z)) \mathrm{d}Z - \int q(Z) \ln(q(Z)) \mathrm{d}Z \\ &= \underbrace{\int \prod_{i=1}^{M} q_i(Z_i) \ln\left(p(X,Z)\right) \mathrm{d}Z}_{\text{part (1)}} - \underbrace{\int \prod_{i=1}^{M} q_i(Z_i) \sum_{i=1}^{M} \ln\left(q_i(Z_i)\right) \mathrm{d}Z}_{\text{part (2)}}\end{aligned}$$

Coordinate Ascent Variational Inference (CAVI)

Variational Inference

$$\text{(Part 1)} = \int \prod_{i=1}^{M} q_i(Z_i) \ln\left(p(X,Z)\right) \mathrm{d}Z$$

$$\int_{Z_1} \int_{Z_2} \cdots \int_{Z_M} \prod_{i=1}^{M} q_i(Z_i) \ln\left(p(X,Z)\right) \mathrm{d}Z_1, \mathrm{d}Z_2, \ldots, \mathrm{d}Z_M$$

- Rearrange the expression by taking a particular $q_j(Z_j)$ out of the integral:

$$\text{(Part 1)} = \int_{Z_j} q_j(Z_j) \left(\int_{Z_{i \neq j}} \cdots \int \prod_{i \neq j}^{M} q_i(Z_i) \ln\left(p(X,Z)\right) \prod_{i \neq j}^{M} dZ_i \right) dZ_j$$

- or, more compactly:

$$\text{(Part 1)} = \int_{Z_j} q_j(Z_j) \left(\int_{Z_{i \neq j}} \cdots \int \ln\left(p(X,Z)\right) \prod_{i \neq j}^{M} q_i(Z_i) dZ_i \right) dZ_j$$

- or, even more meaningfully, it can be put into an expectation function, and since $\prod_{i \neq j}^{M} q_i(Z_i)$ is a joint probability density

$$\text{(Part 1)} = \int_{Z_j} q_j(Z_j) \left[\mathbb{E}_{i \neq j} \left[\ln\left(p(X,Z)\right) \right] \right] dZ_j$$

Variational Inference

$$(\text{Part 2}) = \int \prod_{i=1}^{M} q_i(Z_i) \sum_{i=1}^{M} \ln\left(q_i(Z_i)\right) dZ$$

- Note that the above needs to integrate out all $Z = \{z_1, \ldots, z_M\}$, which is quite daunting. However, notice that each term in the sum, $\sum_{i=1}^{M} \ln\left(q_i(Z_i)\right)$ involves only a single i, therefore, we are able to simplify the above into the following:

$$(\text{Part 2}) = \sum_{i=1}^{M} \left(\int_{Z_i} q_i(Z_i) \ln\left(q_i(Z_i)\right) dZ_i \right)$$

- For a particular $p_j(Z_j)$, the rest of the sum can be treated like a constant, part 2 can be written as:

$$(\text{Part 2}) = \int_{Z_j} q_i(Z_i) \ln\left(q_i(Z_i)\right) dZ_j + \text{const.}$$

where const. are the term does not involve Z_j.

Variational Inference

$$\mathcal{L}(q) = \text{Part (1)} - \text{Part (2)} = \int_{Z_j} q_j(Z_j)\mathbb{E}_{i\neq j}\Big[\ln\left(p(X,Z)\right)\Big]\mathrm{d}Z_j - \int_{Z_j} q_j(Z_j)\ln\left(q_j(Z_j)\right)\mathrm{d}Z_j + \text{const.}$$

- Note that $\mathbb{E}_{i\neq j}\left[\ln\left(p(X,Z)\right)\right]$ would be some $\ln[p(Z_i)]$, we name it $\ln(\tilde{p}_j(X,Z_j))$, i.e.,:

$$\ln(\tilde{p}_j(X,Z_j)) = \mathbb{E}_{i\neq j}\left[\ln\left(p(X,Z)\right)\right]$$

- Or equivalently we can express Evidence Lower Bound (ELOB) in terms of:

$$\mathcal{L}(q_j) = \int_{Z_j} q_j(Z_j)\ln\left[\frac{\tilde{p}_j(X,Z_j)}{q_j(Z_j)}\right] + \text{const.}.$$

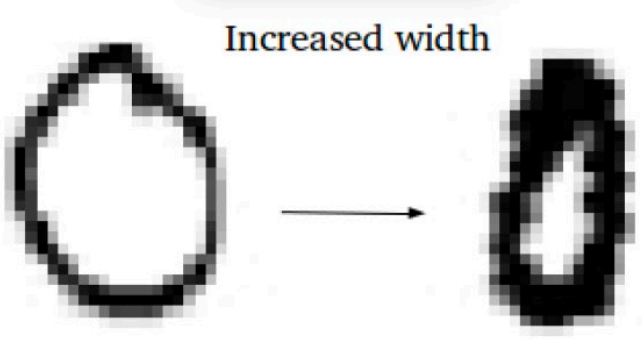
  This is the same as $-\mathbb{KL}\Big(\mathbb{E}_{i\neq j}\left[\ln\left(p(X,Z)\right)\right] \| q_i(Z_i)\Big)$

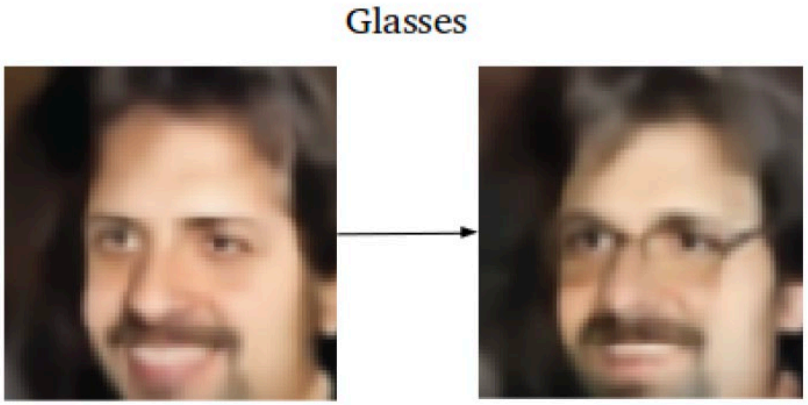
- **This is the key**: We can maximize ELOB, or $\mathcal{L}(q)$, by minimizing this special KL divergence, where we can find approximate and optimal $q_i^*(Z_i)$, such that:

$$\ln\left(q_i^*(Z_i)\right) = \mathbb{E}_{i\neq j}\left[\ln\left(p(X,Z)\right)\right]$$

Variational Autoencoder (VAE)

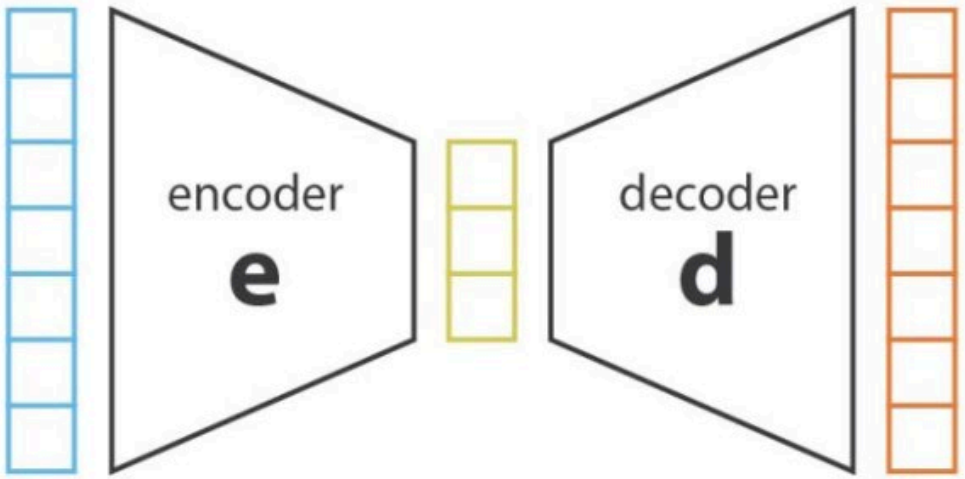
- VAEs are powerful generative models, for generating a random, new output, that looks similar to the training data.
- More often, to alter or explore variations on data you already have





Variational Autoencoder (VAE)

- **Standard autoencoder**



x — initial data in space R^n

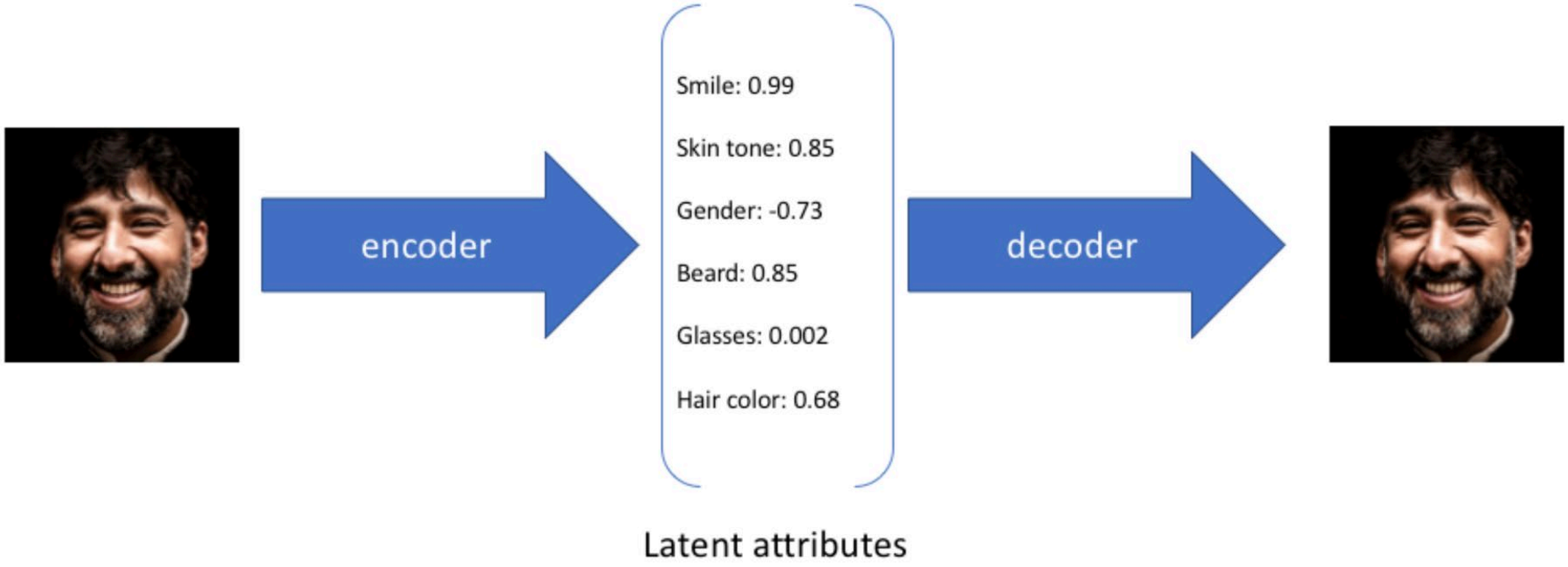
$z = e(x)$ — encoded data in latent space R^m (with $m<n$)

$\hat{x} = d(z)$ — encoded-decoded data back in the initial space R^n

$$\text{loss} = \| x - \hat{x} \|^2 = \| x - d(z) \|^2$$

Variational Autoencoder (VAE)

- Learning descriptive attributes of faces to describe an observation in some compressed representation.
- Each latent attribute is described using a single value.



Variational Autoencoder (VAE)

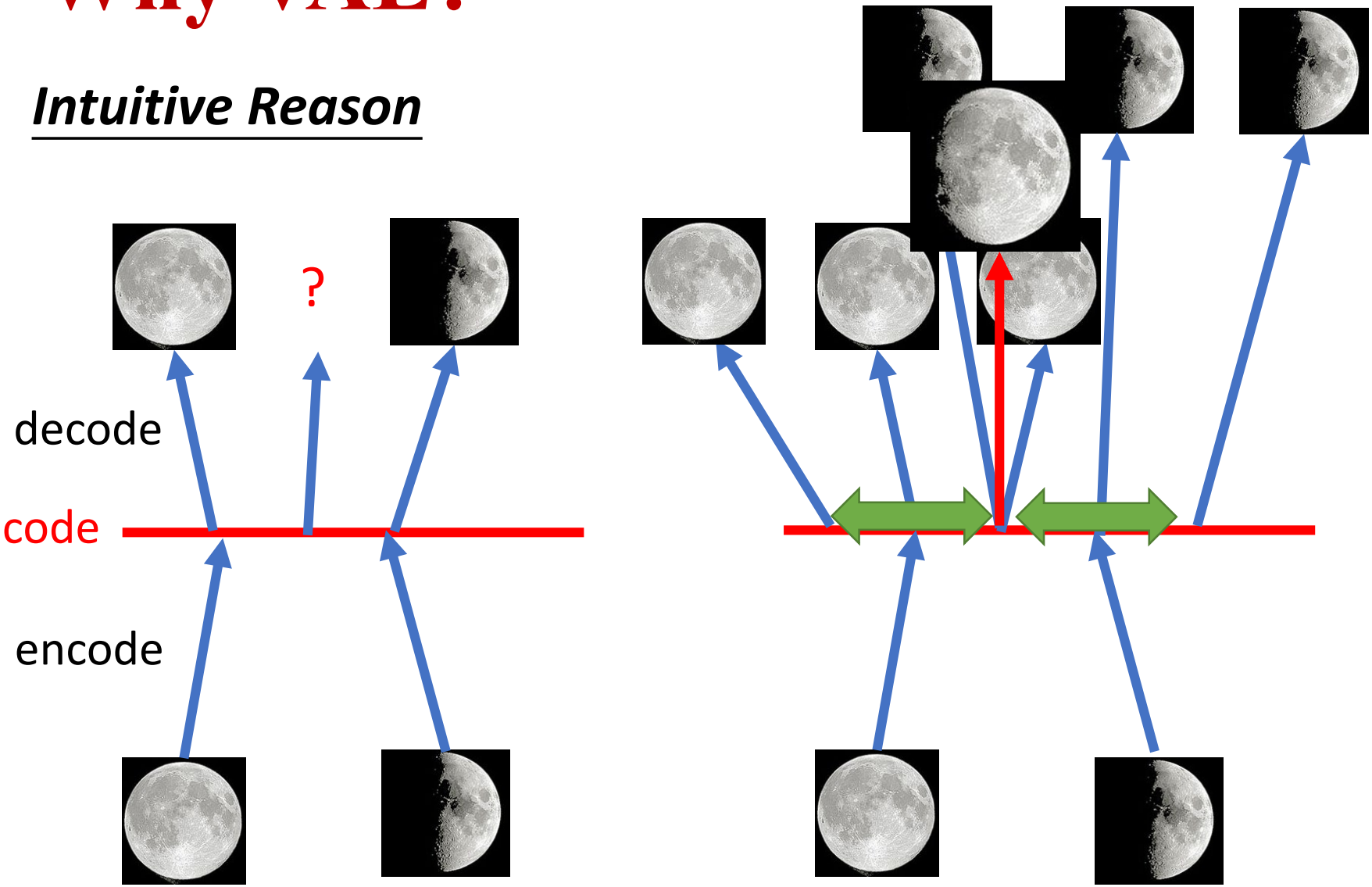
Why not use standard autoencoder for generation ?

- For standard autoencoder, the latent space is discontinuous
- Only replicate the same image that was put in
- When sample a variation, the decoder generates unrealistic output
- The decoder has no idea how to deal with never saw region

A continuous latent space is required for generation

Why VAE?

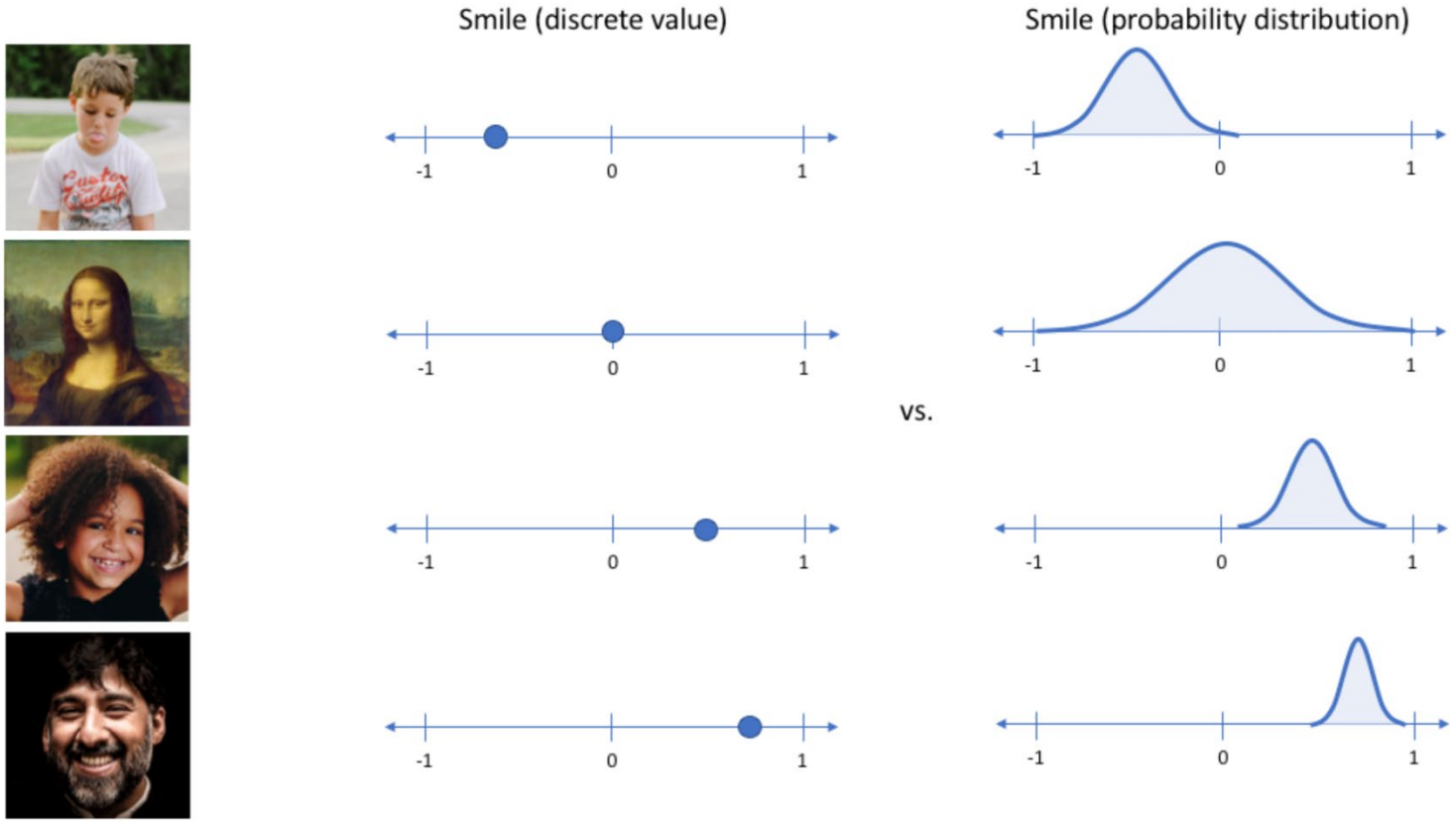
Intuitive Reason



Variational Autoencoder (VAE)

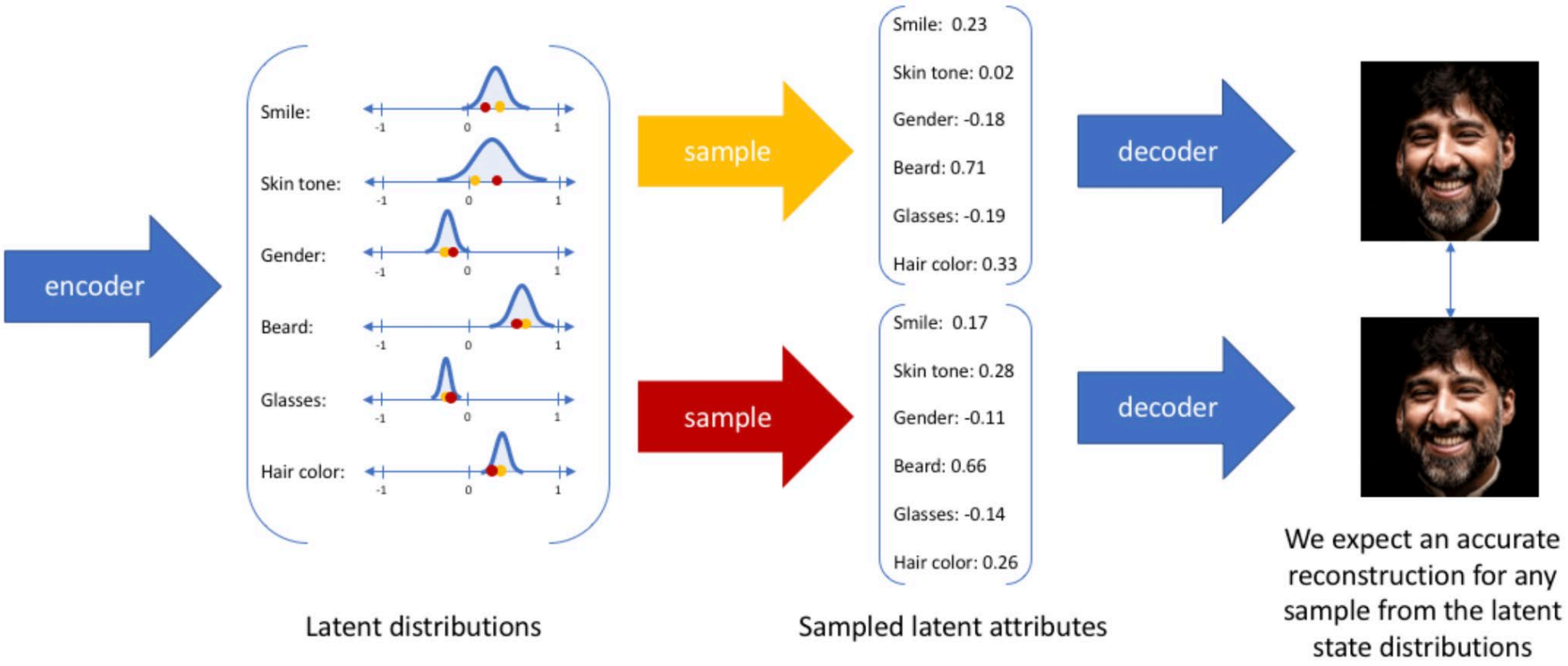
The latent space of VAE is continuous, allowing random sampling

- represent each latent attribute as a probability distribution



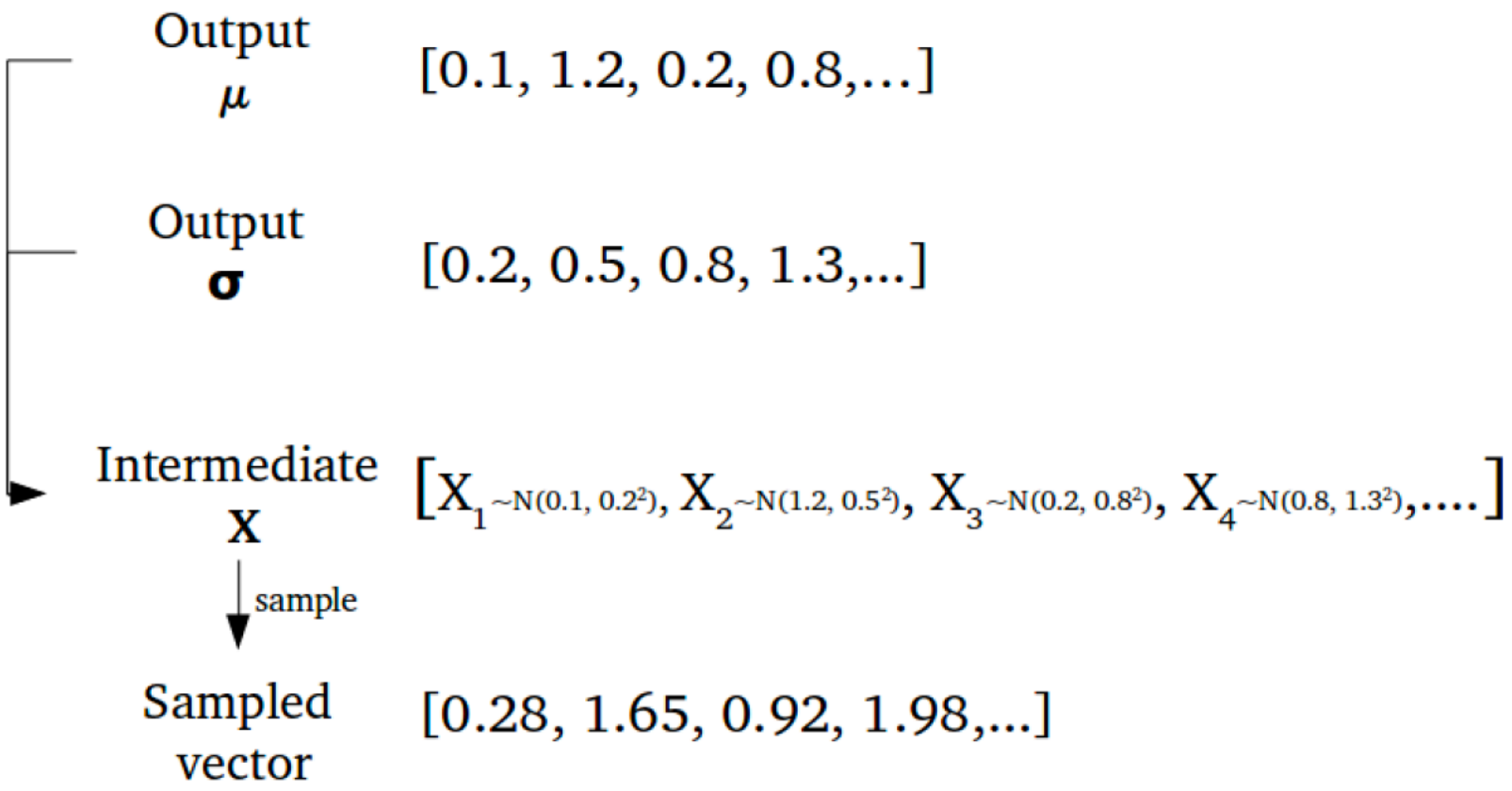
Variational Autoencoder (VAE)

- Values nearby to one another in latent space should correspond with very similar reconstructions.

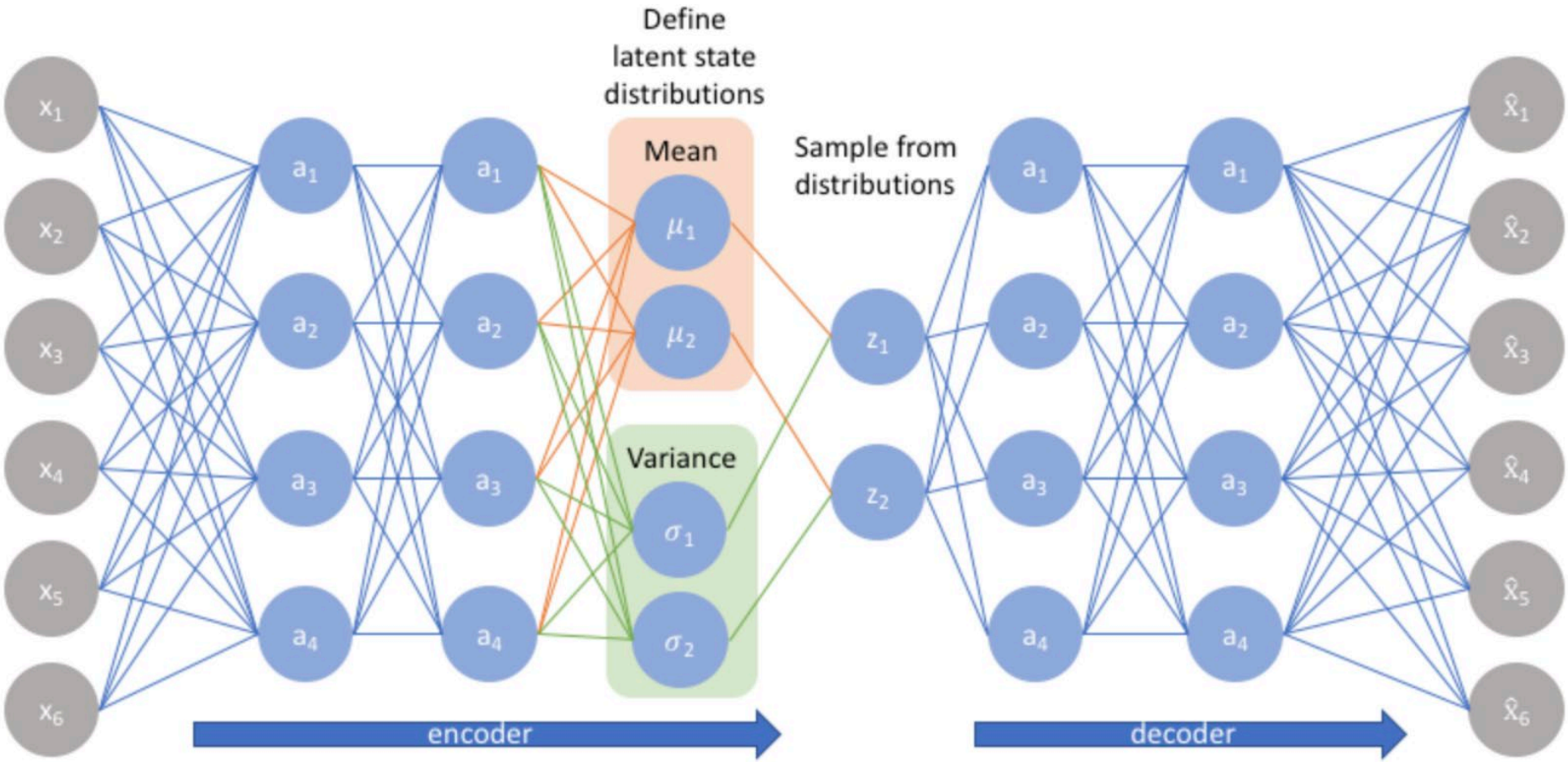


Variational Autoencoder (VAE)

- The output of the encoder is the means and standard deviations



Variational Autoencoder (VAE)



Structure of Variational Autoencoder

Variational Autoencoder (VAE)

- Suppose some hidden variable z generates an observation x

- We'd like to compute $p(z/x)$ $\quad p(x,z) = p(z)p(z/x)$

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

- Computing $p(z/x)$ is intractable, varitational inference is used. Namely

$$\min KL\left(q(z|x)\,||p(z|x)\right) \qquad q(Z) = \prod_{j=1}^{m} q_j(z_j)$$

- Minimizing $\min KL\left(q(z|x)\,||p(z|x)\right)$ equals to maximize the ELBO

$$\int q(z|x)log[p(x|z)p(z)]dz - \int q(z|x)logq(z|x)dz$$

Variational Autoencoder (VAE)

Proof

$$KL(q(z|x)||p(z|x)) = \int q(z|x) log \frac{q(z|x)}{p(z|x)} dz$$

$$= \int q(z|x) log \frac{q(z|x)}{\frac{p(x|z)p(z)}{p(x)}} dz$$

$$= \int q(z|x) log q(z|x) dz + \int q(z|x) log p(x) dz - \int q(z|x) log[p(x|z)p(z)] dz$$

$$= \int q(z|x) log q(z|x) dz + log p(x) \int q(z|x) dz - \int q(z|x) log[p(x|z)p(z)] dz \qquad \int q(z|x) dz = 1$$

$$= log p(x) + \int q(z|x) log q(z|x) dz - \int q(z|x) log[p(x|z)p(z)] dz$$

Given x , $p(x)$ is a constant. $\min KL(q(z|x)||p(z|x))$ equals to maximize the ELBO

$$\int q(z|x) log[p(x|z)p(z)] dz - \int q(z|x) log q(z|x) dz$$

Variational Autoencoder (VAE)

$$\int q(z|x)log[p(x|z)p(z)]dz - \int q(z|x)logq(z|x)dz$$

$$= \int q(z|x)logp(x|z)dz + \int q(z|x)logp(z)dz - \int q(z|x)logq(z|x)dz$$

$$= \int q(z|x)logp(x|z)dz - \int q(z|x)log\frac{q(z|x)}{p(z)}dz$$

$$= E_{q(z|x)} \log p(x|z) - KL(q(z|x)||p(z))$$

Thus

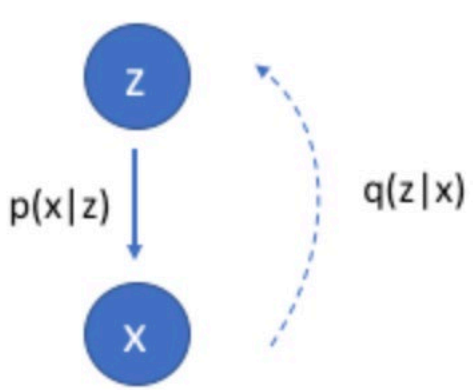
$$\min KL(q(z|x)||p(z|x))$$

$$\text{Max } E_{q(z|x)} \log p(x|z) - KL(q(z|x)||p(z)) \quad p(z) \sim N(0,1)$$

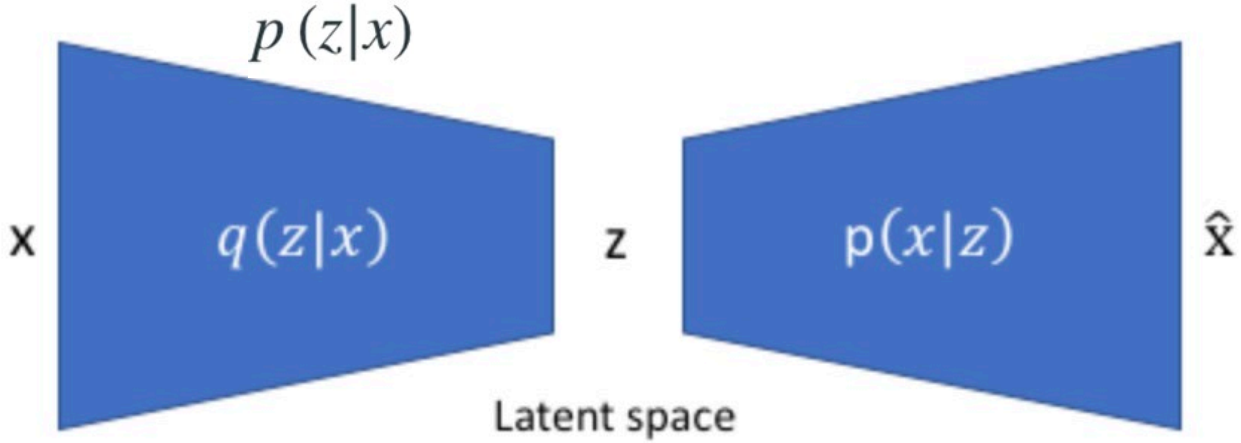
Variational Autoencoder (VAE)

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

Max $E_{q(z|x)} \log p\left(x|z\right) - KL\left(q\left(z|x\right)||p\left(z\right)\right)$ $p(z) \sim N(0, I)$



We'd like to use our observations to understand the hidden variable.



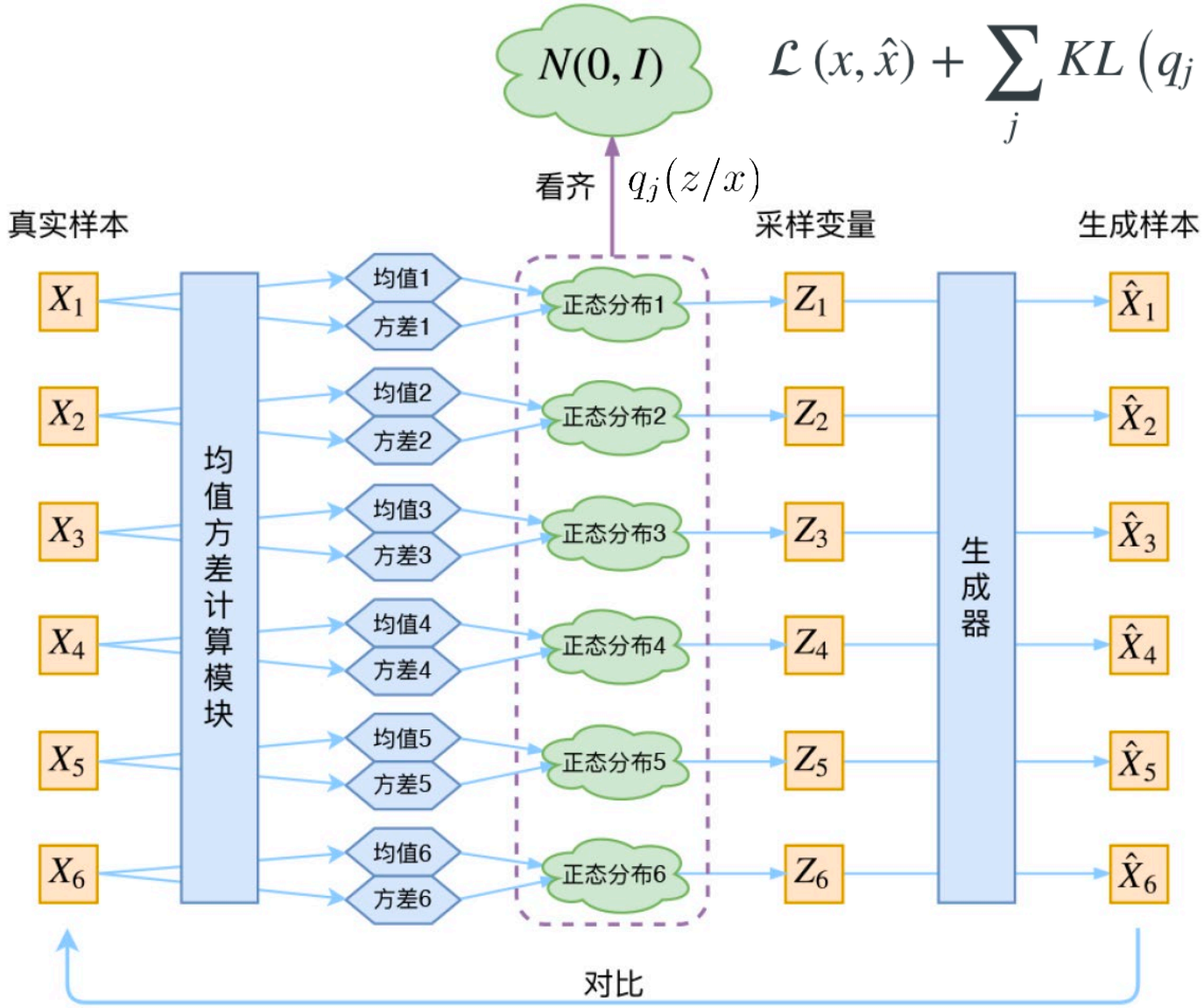
Latent space representation.

Neural network mapping x to z.

Neural network mapping z to x.

Loss function $\mathcal{L}\left(x, \hat{x}\right) + \sum_{j} KL\left(q_j\left(z|x\right)||p\left(z\right)\right)$

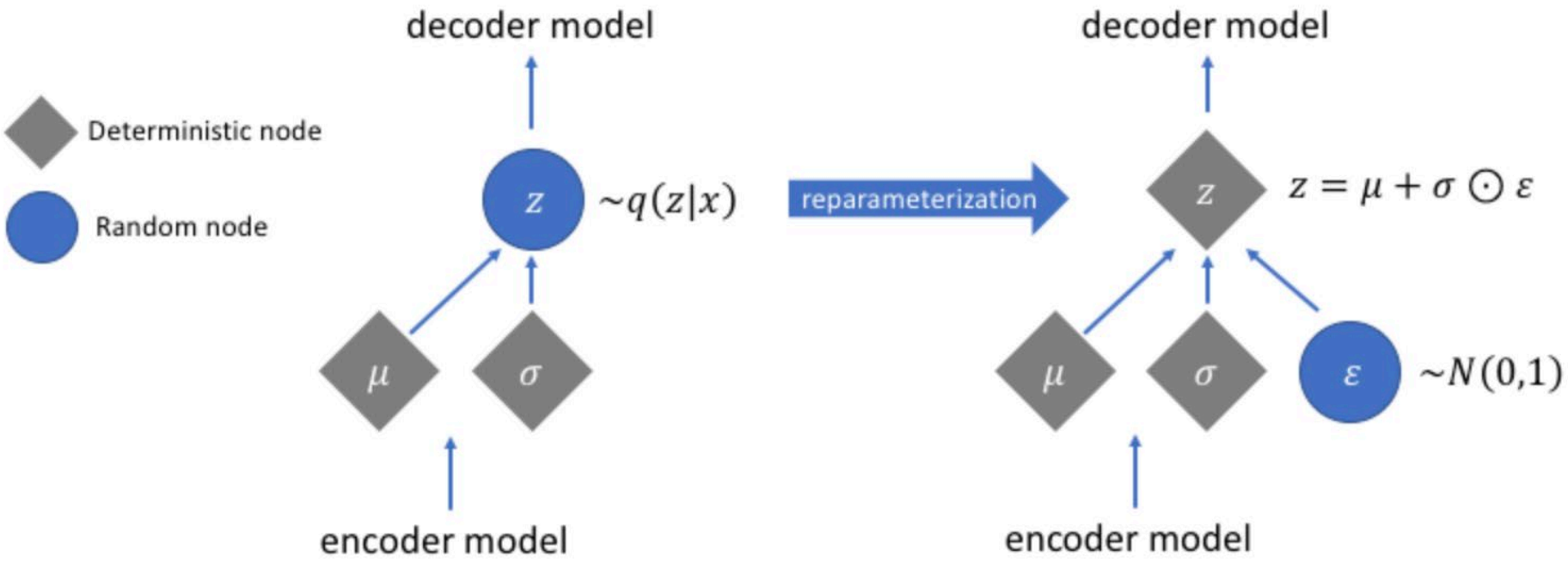
Variational Autoencoder (VAE)



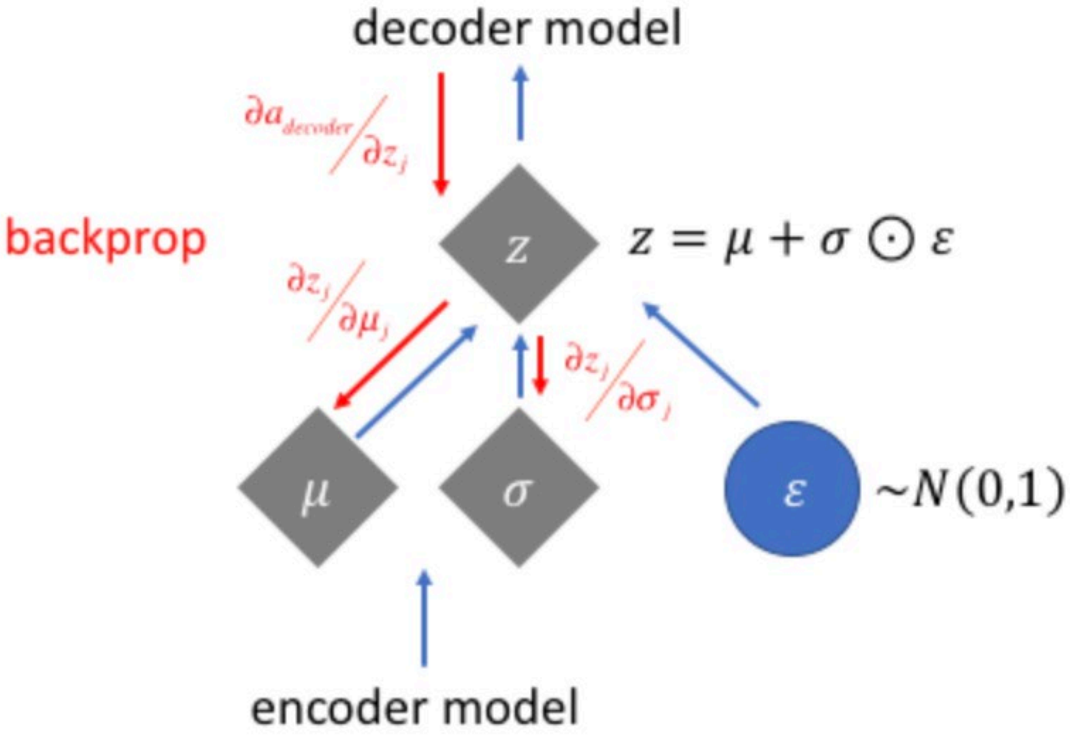
Variational Autoencoder (VAE)

Reparameterization

$$z \sim N(\mu, \sigma^2) \qquad \frac{z-\mu}{\sigma} = \varepsilon \sim N(0,1)$$



Variational Autoencoder (VAE)

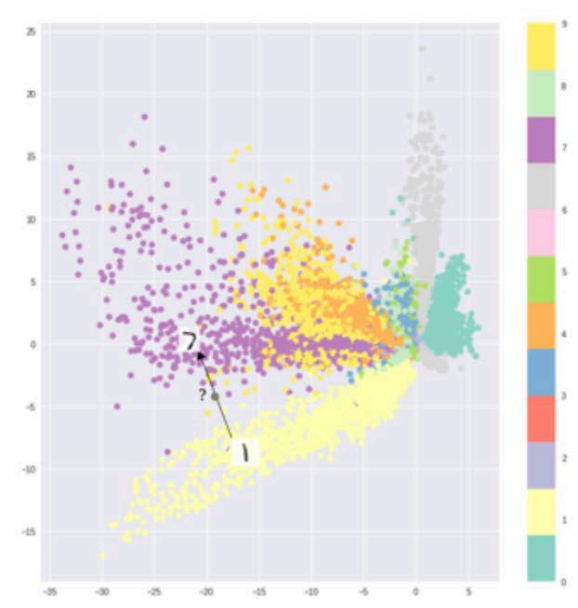


To avoid negative σ , We often learn $\sigma = log(1 = exp(\rho))$

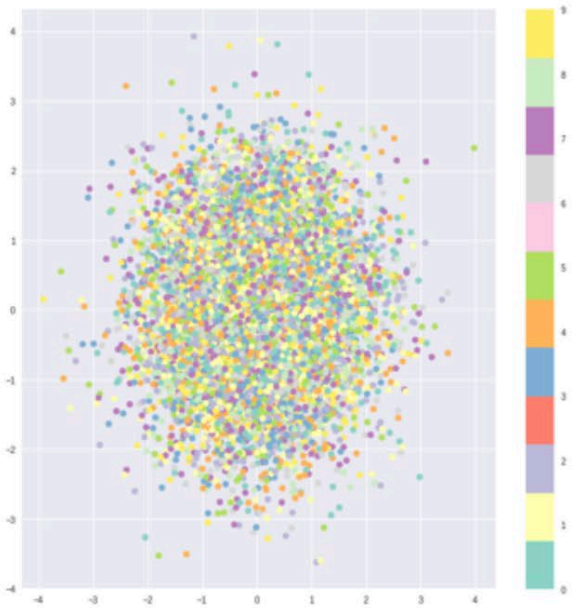
Variational Autoencoder (VAE)

$$\mathcal{L}(x, \hat{x}) + \sum_j KL\left(q_j(z|x) \,||p(z)\right)$$

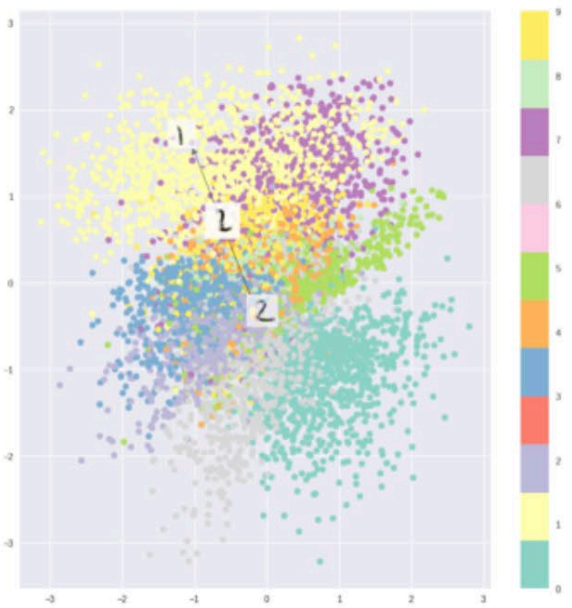
Only reconstruction loss



Only KL divergence



Combination



Point Estimates of Neural Networks

A neural network is viewed as a probabilistic model: $P(\mathbf{y}|\mathbf{x}, \mathbf{w})$

- Given input x, a neural network assigns a probability to each possible y, using the set of weights w.
- For classification, $P(\mathbf{y}|\mathbf{x}, \mathbf{w})$ is categorical distribution, corresponds to the cross-entropy or softmax loss.
- For regression, $P(\mathbf{y}|\mathbf{x}, \mathbf{w})$ is Gaussian distribution, corresponds to MSE loss.
- The weights can be learnt by Maximum Likelihood Estimation (MLE).

$$\begin{aligned}\mathbf{w}^{\text{MLE}} &= \arg\max_{\mathbf{w}} \log P(\mathcal{D}|\mathbf{w}) \\ &= \arg\max_{\mathbf{w}} \sum_i \log P(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w})\end{aligned}$$

Point Estimates of Neural Networks

- If a prior upon the weights w is placed, then we have the Maximum Posteriori (MAP).

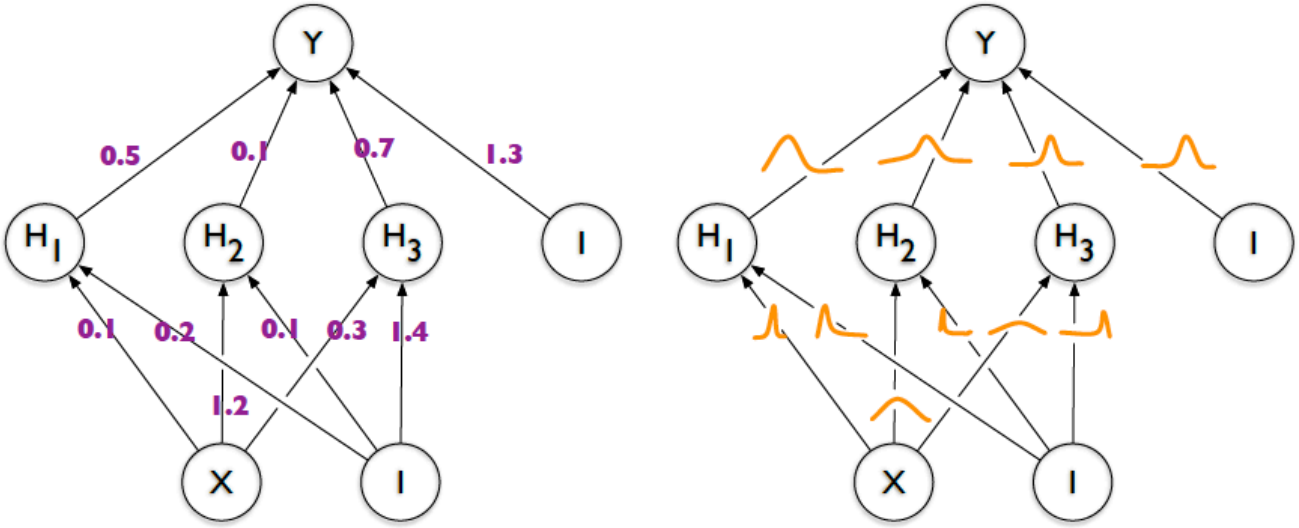
$$\begin{aligned}\mathbf{w}^{\mathrm{MAP}} &= \arg\max_{\mathbf{w}} \log P(\mathbf{w}|\mathcal{D}) \\ &= \arg\max_{\mathbf{w}} \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w})\end{aligned}$$

If w are given a Gaussian prior, this yields L2 regularisation.

If w are given a Laplace prior, then L1 regularisation is recovered.

Bayesian Neural Networks (BNN)

- 传统的神经网络，其权重参数是确定的值；贝叶斯神经网络，权重参数是随机变量。
- 传统的神经网络，用交叉熵，MSE 等损失函数去拟合标签值；贝叶斯神经网络，拟合后验分布。
- 贝叶斯神经网络，将概率建模和神经网络结合起来，不仅可以给出预测值，而且可以给出预测值的置信度。



Bayesian Neural Networks (BNN)

- Given the training data $\mathcal{D} = (\mathbf{x}_i, \mathbf{y}_i)_i$, a Bayesian neural network predicts the distribution of an unknown label $\hat{\mathbf{y}}$ of a test data item $\hat{\mathbf{x}}$

$$P(\hat{\mathbf{y}}|\hat{\mathbf{x}}) = \mathbb{E}_{P(\mathbf{w}|\mathcal{D})}[P(\hat{\mathbf{y}}|\hat{\mathbf{x}}, \mathbf{w})]$$

- The weights are set according to the posterior distribution $P(\mathbf{w}|\mathcal{D})$
- Taking an expectation under the posterior distribution $P(\mathbf{w}|\mathcal{D})$, equals to using an ensemble of infinite number of neural networks, which is intractable for any practical neural networks.
- Also, the posterior distribution $P(\mathbf{w}|\mathcal{D})$ is intractable

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathbf{w}, \mathcal{D})}{P(\mathcal{D})} = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$$

Bayesian Neural Networks (BNN)

- **Variational inference used to approximate posterior distribution** $P(\mathbf{w}|\mathcal{D})$

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ \mathrm{KL}\left[q(w|\theta)||P(w|\mathcal{D})\right]$$

$$= \underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{q(w|\theta)}\left[\log\left[\frac{q(w|\theta)}{P(w|\mathcal{D})}\right]\right] \qquad \text{(definition of KL divegence)}$$

$$= \underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{q(w|\theta)}\left[\log\left[\frac{q(w|\theta)P(\mathcal{D})}{P(\mathcal{D}|w)P(w)}\right]\right] \qquad \text{(Bayes Theorem)}$$

$$= \underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{q(w|\theta)}\left[\log\left[\frac{q(w|\theta)}{P(\mathcal{D}|w)P(w)}\right]\right] \qquad \text{(Drop } P(\mathcal{D}) \text{ because it doesn't depend on } \theta\text{)}$$

- **The loss function is the expected lower bound (ELBO)**

$$\mathcal{L} = \mathbb{E}_{q(w|\theta)}\left[\log\left[\frac{q(w|\theta)}{P(\mathcal{D}|w)P(w)}\right]\right]$$

Bayesian Neural Networks (BNN)

- We approximate the exact cost as:

$$\mathcal{L} \approx \sum_{i=1}^{n} \log q(\mathbf{w}^{(i)}|\theta) - \log P(\mathbf{w}^{(i)}) - \log P(\mathcal{D}|\mathbf{w}^{(i)})$$

where $\mathbf{w}^{(i)}$ denotes the ith Monte Carlo sample drawn from $q(\mathbf{w}^{(i)}|\theta)$.

- **Reparameterization:**

Suppose that the variational posterior is a diagonal Gaussian distribution, then a sample of the weights $\mathbf{w}$ can be obtained by sampling a unit Gaussian, shifting it by a mean μ and scaling by a standard deviation σ. We parameterise the standard deviation pointwise as $\sigma = \log(1 + \exp(\rho))$ and so σ is always non-negative. The variational posterior parameters are $\theta = (\mu, \rho)$. Thus the transform from a sample of parameter-free noise and the variational posterior parameters that yields a posterior sample of the weights $\mathbf{w}$ is: $\mathbf{w} = t(\theta, \epsilon) = \mu + \log(1 + \exp(\rho)) \circ \epsilon$ where $\circ$ is pointwise multiplication. Each step of optimisation proceeds as follows:

Bayesian Neural Networks (BNN)

1. Sample $\epsilon \sim \mathcal{N}(0, I)$.
2. Let $\mathbf{w} = \mu + \log(1 + \exp(\rho)) \circ \epsilon$.
3. Let $\theta = (\mu, \rho)$.
4. Let $f(\mathbf{w}, \theta) = \log q(\mathbf{w}|\theta) - \log P(\mathbf{w})P(\mathcal{D}|\mathbf{w})$.
5. Calculate the gradient with respect to the mean

$$\Delta_\mu = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \mu}. \tag{3}$$

6. Calculate the gradient with respect to the standard deviation parameter ρ

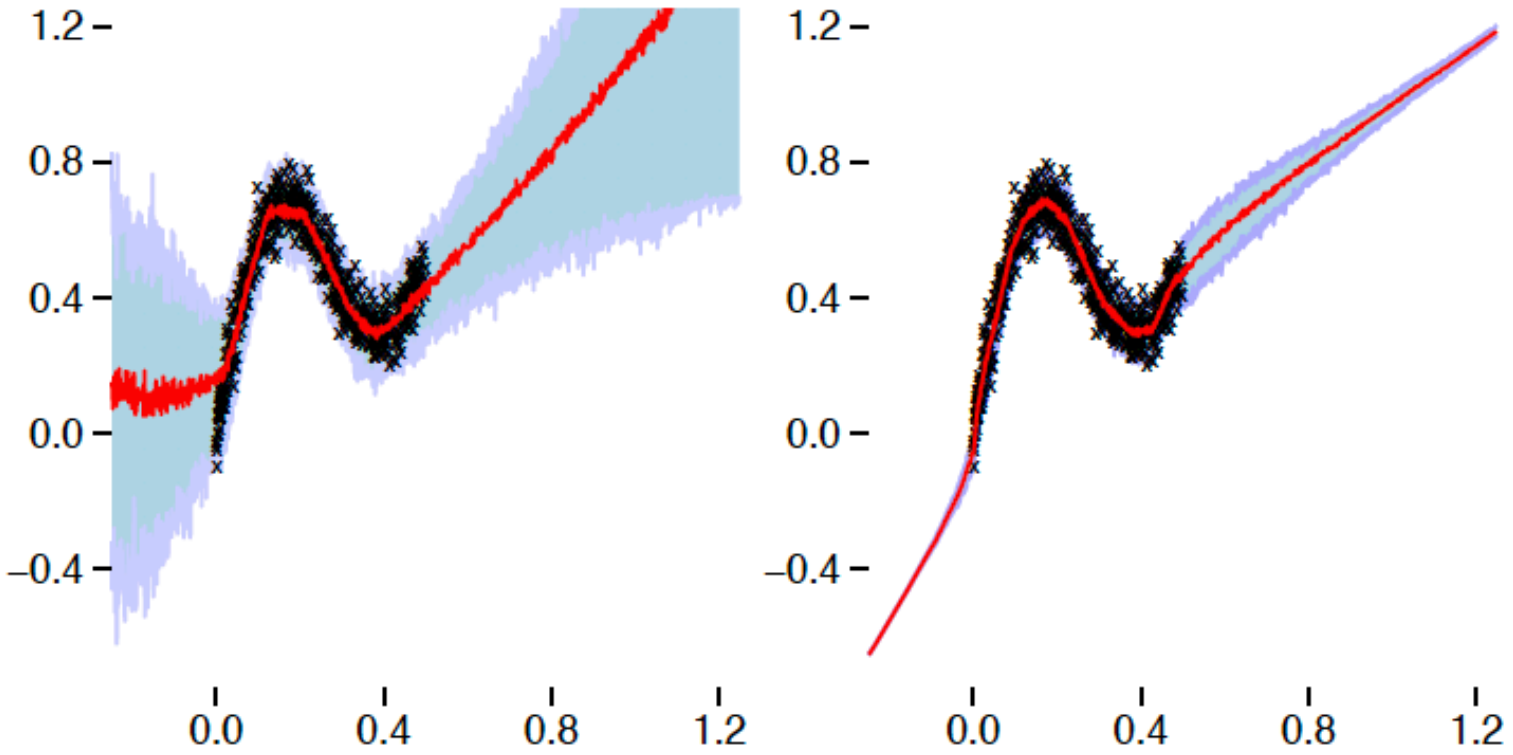
$$\Delta_\rho = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} \frac{\epsilon}{1 + \exp(-\rho)} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \rho}. \tag{4}$$

7. Update the variational parameters:

$$\mu \leftarrow \mu - \alpha \Delta_\mu \tag{5}$$

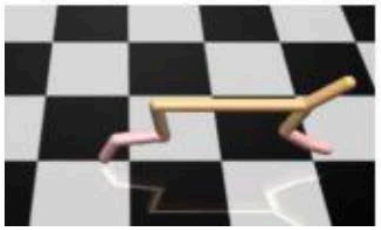
$$\rho \leftarrow \rho - \alpha \Delta_\rho. \tag{6}$$

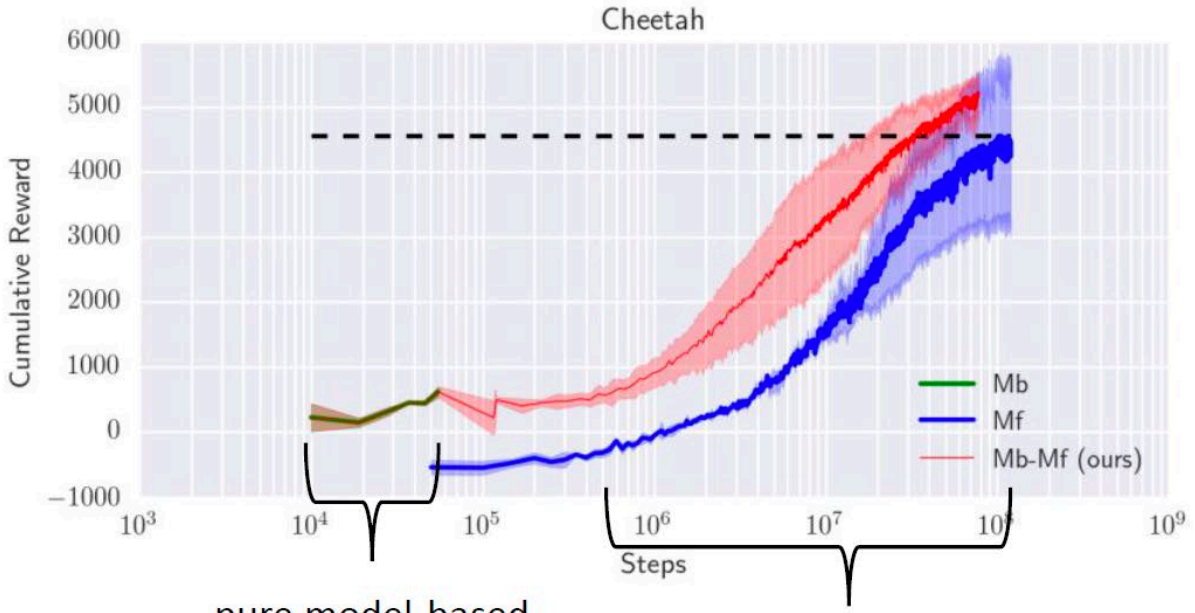
Bayesian Neural Networks (BNN)



Weight Uncertainty in Neural Networks. https://arxiv.org/abs/1505.05424

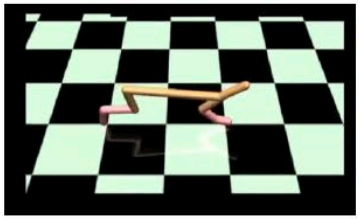
Uncertainty in Model-Based RL

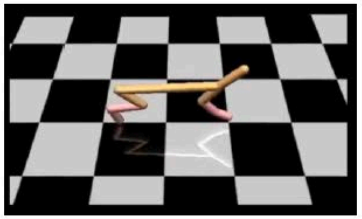




pure model-based
(about 10 minutes real time)

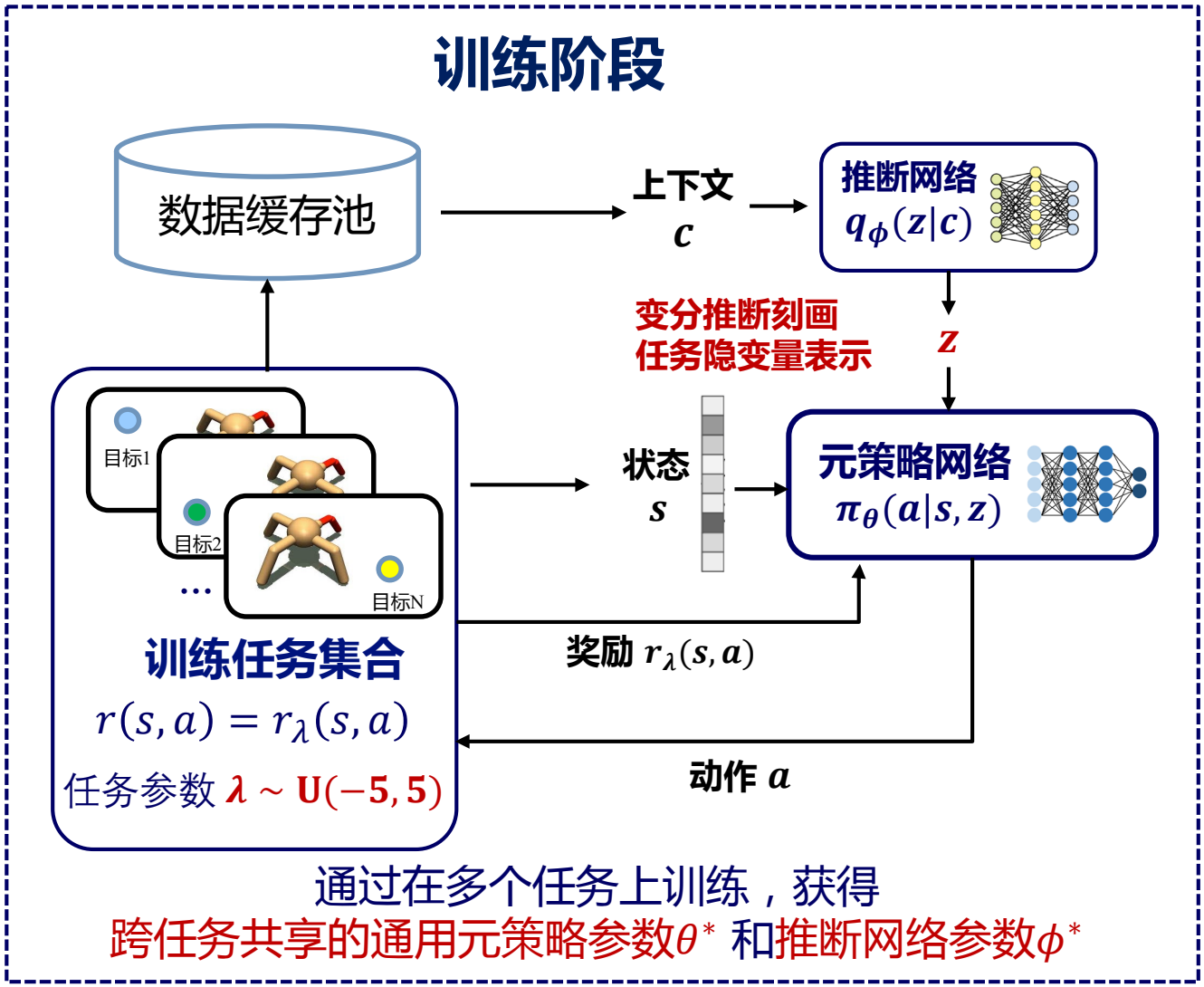
model-free training
(about 10 days…)





Nagabandi, Kahn, Fearing, L. ICRA 2018

Meta-Reinforcement Learning



Meta-Reinforcement Learning