CS182 Introduction to Machine Learning, Fall 2023 Discussion8

Zhan-Wang Mao

maozhw@shanghaitech.edu.cn

Outline

The Road From MLE to EM to VAE

- MLE Revisited
  - Difficulties of MLE
- Evidence Lower Bound (ELBO)
- Expectation-Maximization (EM) Algorithm
- Variational Auto-Encoder (VAE)
- The Reparameterization Trick

MLE Revisited

Suppose we have a latent variable model $p(x, z; \theta)$, where z is the latent variable and θ is the parameter. Given i.i.d. training set $\mathrm{X} = \{x^{(1)}, \ldots, x^{(n)}\}$:

- Maximum Likelihood Estimation (MLE):

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} \ \log p(\mathbf{X}; \theta)$$

$$\begin{aligned}\log p(\mathbf{X}; \theta) &= \log \prod_{i=1}^{n} \int_z p(x^{(i)}, z; \theta) \\ &= \sum_{i=1}^{n} \log \int_z p(x^{(i)}, z; \theta)\end{aligned}$$

Difficulties of MLE

From simple to complicated:

(1). The equation $\nabla_\theta \log p(\mathbf{X};\theta)$ has close-form solutions.

(2). Give θ, marginal likelihood $p(\mathbf{X};\theta)$ can be evaluated, which means $\int_z p(x,z;\theta)$ is tractable. Thus, we can perform **gradient ascent**:

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log p(\mathbf{X};\theta)$$

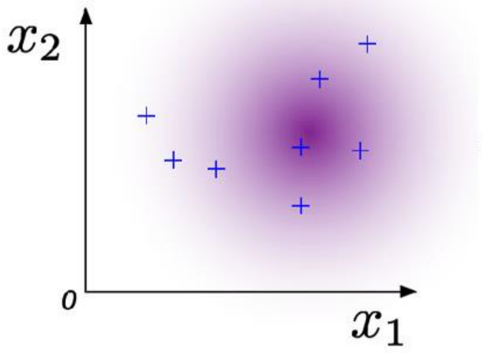
(3). The marginal likelihood $p(\mathbf{X};\theta)$ cannot be evaluated, because $\int_z p(x,z;\theta)$ is intractable. This often happens in the deep learning, where

$$p(x,z;\theta) = p(x \mid z;\theta) p_z(z;\theta)$$

$p(x \mid z;\theta)$ is modeled by a neural network.

Difficulties of MLE

(1) Guassian Model



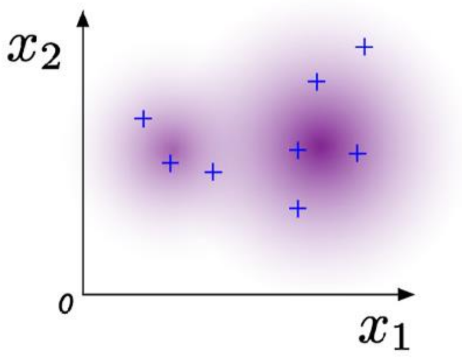
The hypothesized model is $\mathbf{x} \sim \mathcal{N}([\mu_1, \mu_2], \sigma^2 I)$, whose parameters $\theta = [\mu_1, \mu_2, \sigma^2]$.

$$\log p(\mathbf{X};\theta) = \sum_{i=1}^{N} \log p_{\mathsf{x}}(\mathbf{x}^{(i)};\theta)$$
$$= \sum_{i=1}^{N} (-\frac{1}{2}\log 2\pi - \log\sigma - \frac{(x_1^{(i)}-\mu_1)^2 + (x_2^{(i)}-\mu_2)^2}{2\sigma^2})$$

Let $\nabla_\theta \log p(\mathbf{X};\theta) = \mathbf{0}$,

$$\mu_1 = \frac{1}{N}\sum_{i=1}^{N} x_1^{(i)}, \mu_2 = \frac{1}{N}\sum_{i=1}^{N} x_2^{(i)},$$
$$\sigma^2 = \frac{\sum_{i=1}^{N}\left(\left(x_1^{(i)}-\mu_1\right)^2 + \left(x_2^{(i)}-\mu_2\right)^2\right)}{N}$$

(2) Guassian Mixture Model

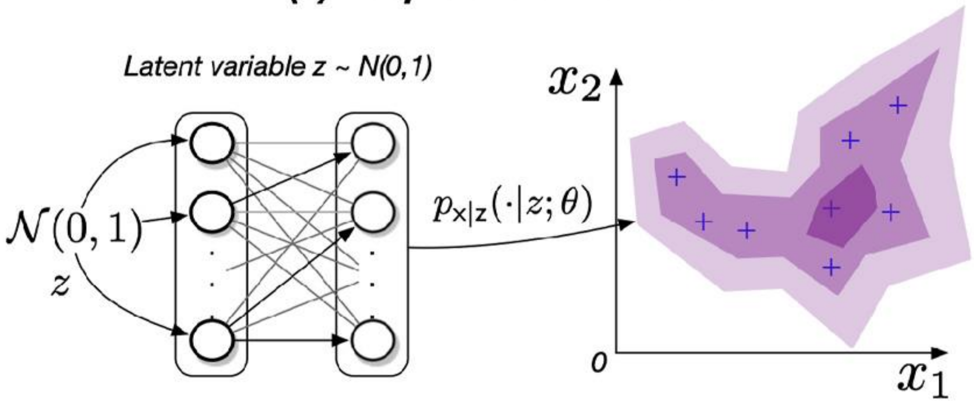


The data are supposed to be generated by two steps: first select a Gaussian component (latent variable) subject to a multinomial prior p_{z}, and then generate it by the z-th Gaussian $\mathcal{N}([\mu_{z1}, \mu_{z2}], \sigma_z^2 I)$.

$$\log p(\mathbf{X};\theta) = \sum_{i=1}^{N} \log \left(\sum_{z=1}^{Z} \frac{p_{\mathsf{z}}(z)}{\sqrt{2\pi}\sigma_z} e^{-\frac{(x_1^{(i)}-\mu_{z1})^2 + (x_2^{(i)}-\mu_{z2})^2}{2\sigma_z^2}}\right).$$

The equation $\nabla_\theta \log p(\mathbf{X};\theta) = \mathbf{0}$ actually has no close-form solution. But EM and SGD are still applicable, because we can easily compute the log-likelihood for any given $\theta = [\theta_1, ..., \theta_Z]$.

(3) Deep Generative Model



An example of deep generative model, where the latent variable z is generated from standard Gaussian prior. Each z is then transformed into a distribution $p_{\mathsf{x}|\mathsf{z}}(\cdot|z;\theta)$ by a deep neural networks parameterized by θ. In many cases, the distribution is a Gaussian with center from the NeuralNet,

$$p_{\mathsf{x}|\mathsf{z}}(\cdot|z;\theta) = \mathcal{N}(\mathrm{NeuralNet}(z;\theta), \mathbf{I}).$$

Then the log-likelihood $\log p(\mathbf{X};\theta)$ becomes

$$\sum_{i=1}^{N} \log \int_{-\infty}^{\infty} \frac{1}{2\pi} e^{-\frac{1}{2}\|\mathbf{x}^{(i)} - \mathrm{NeuralNet}(z;\theta)\|^2} p_{\mathbf{z}}(z)dz$$
$$= \sum_{i=1}^{N} \log \int_{-\infty}^{\infty} (2\pi)^{-\frac{2}{3}} e^{-\frac{1}{2}z^2\|\mathbf{x}^{(i)} - \mathrm{NeuralNet}(z;\theta)\|^2} dz.$$

We cannot integrate over the continous $\mathbf{z}$ with the function containing a neural network, so $\log p(\mathbf{X};\theta)$ cannot even be evaluated with known θ.

Evidence Lower Bound (ELBO)

- Consider optimizing of the likelihood for a single sample x.
- Introducing an extra distribution $q(z)$
- Construct a lower bound of $\log p(x;\theta)$

$$
\begin{aligned}
\log p(x;\theta) &= \log \int_z p(x,z;\theta) \\
&= \log \int_z \frac{p(x,z;\theta)}{q(z)} q(z) \\
&\geq \underbrace{\int_z q(z) \log \frac{p(x,z;\theta)}{q(z)}}_{\mathrm{ELBO}(x;q,\theta)} = -D_{KL}(q(\cdot) \parallel p(x,\cdot;\theta))
\end{aligned}
$$

- The last line follows from Jensen's Inequality.

Evidence Lower Bound (ELBO)

- Choose $q(z)$ to make the lower bound tight for current guess θ.
- Recall it is sufficient for Jensen's Inequality hold with equality when

$$q(z) \propto p(x, z; \theta)$$

- Normalize $p(x, z; \theta)$ we have:

$$q(z) = \frac{p(x, z; \theta)}{\int_z p(x, z; \theta)} = \frac{p(x, z; \theta)}{p(x; \theta)} = p(z \mid x; \theta)$$

- For training set $\mathrm{X} = \{x^{(1)}, \dots, x^{(n)}\}$:

$$\ell(\theta) = \log p(\mathbf{X}; \theta) \geq \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; q_i, \theta)$$

Expectation-Maximization (EM) Algorithm

- Take initial guess $\theta^{(0)}$.
- Alternating following steps, until convergence.
- **(E-step):** For each i, update

$$q_i^{(t+1)}(z^{(i)}) = p(z^{(i)} \mid x^{(i)}; \theta^{(t)})$$

- **(M-step):** Update

$$\theta^{(t+1)} = \arg\max_{\theta} \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; q_i^{(t+1)}, \theta)$$

Convergence of EM Algorithm

- Prove $\ell(\theta^{(t)}) \leq \ell(\theta^{(t+1)})$

$$\begin{aligned}
\ell(\theta^{(t+1)}) &\geq \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; q_i^{(t)}, \theta^{(t+1)}) \\
&\geq \sum_{i=1}^{n} \mathrm{ELBO}(x^{(i)}; q_i^{(t)}, \theta^{(t)}) \\
&= \ell(\theta^{(t)})
\end{aligned}$$

- EM always monotonically improves the log-likelihood.

https://cs229.stanford.edu/notes2020spring/cs229-notes8.pdf

Decompositions of ELBO

- We can rewrite ELBO to several forms:

$$
\begin{aligned}
\mathrm{ELBO}(x; q, \theta) &= \mathbb{E}_{z\sim q}[\log p(x, z; \theta)] - \mathbb{E}_{z\sim q}[\log q(z)] \\
&\overset{(1)}{=} \log p_x(x) - D_{KL}(q \parallel p_{z|x}) \\
&\overset{(2)}{=} \mathbb{E}_{z\sim q}[\log p(x \mid z; \theta)] - D_{KL}(q \parallel p_z)
\end{aligned}
$$

- (1) is corresponding to E-step.
- (2) is corresponding to M-step.

Variational Auto-Encoder

- Parameterization of $p(x, z; \theta)$ by a neural network (suppose $\boldsymbol{\sigma_x}$ is known).

$$z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$x \mid z \sim \mathcal{N}(\text{Decoder}(z; \theta), \sigma_x^2 \mathbf{I})$$

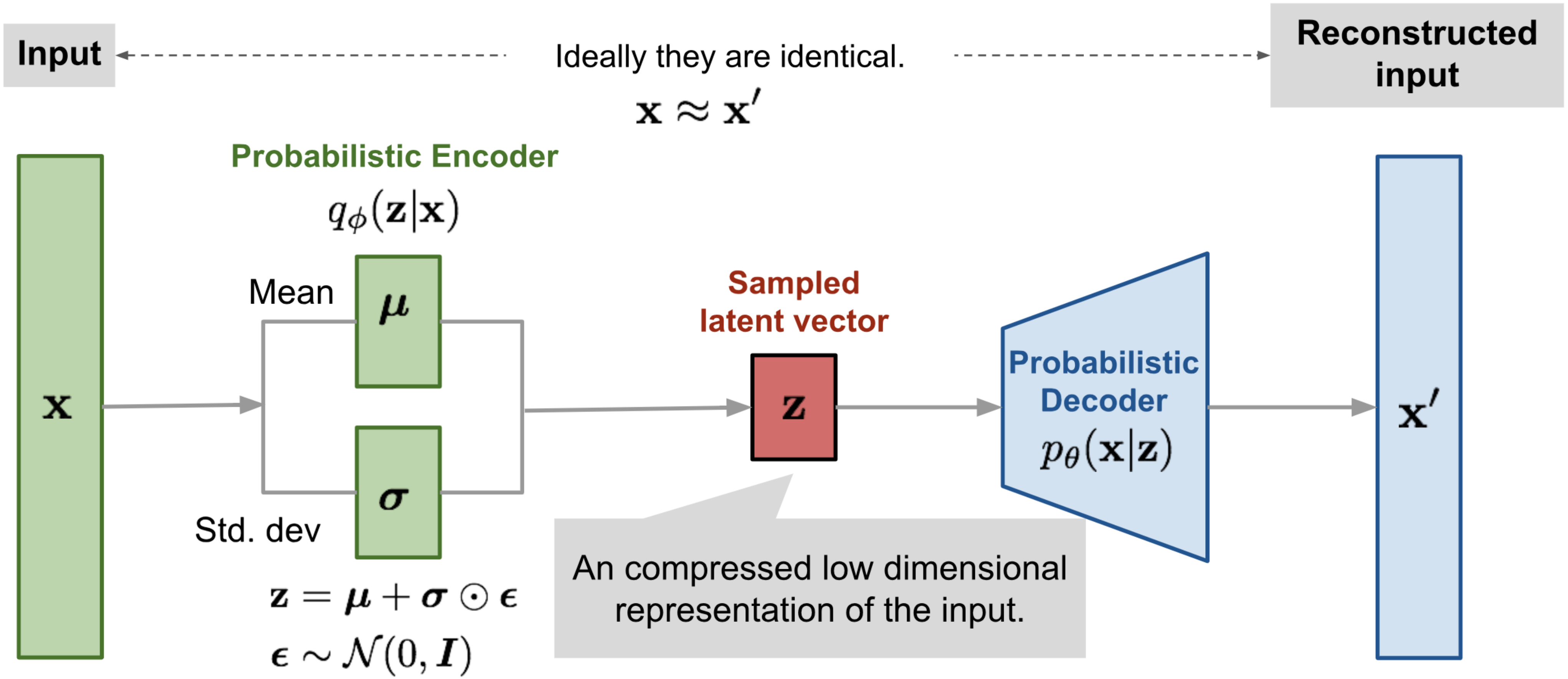
- Intractable to compute the exact posterior distribution $q(z) = (z \mid x; \theta)$
- VAE limits $q(z)$ to the family of isotropic Gaussian distribution $\mathcal{Q}$ which keep ELBO easy to compute.

$$q(z) = \mathcal{N}(\mu, \text{diag}(\sigma)^2)$$

$$\mu,\ \sigma = \text{Encoder}(x; \phi)$$

- Note in traditional EM, we should find a networks for each data point. VAE uses Amortized Variational Inference (AVI), which shares parameters ϕ.

Variational Auto-Encoder



https://lilianweng.github.io/posts/2018-08-12-vae/

Variational Auto-Encoder

- Recall the decomposition of ELBO:

$$\begin{aligned}
\mathrm{ELBO}(x; q_\phi, \theta) &= \mathbb{E}_{z\sim q_\phi}[\log p(x, z; \theta)] - \mathbb{E}_{z\sim q_\phi}[\log q_\phi(z)] \\
&= \mathbb{E}_{z\sim q_\phi}[\log p(x \mid z; \theta)] - D_{KL}(q_\phi \parallel p_z) \\
&= \underbrace{\mathbb{E}_{z\sim q_\phi}[\log p(x \mid z; \theta)]}_{\text{Reconstruction Loss}} - \underbrace{D_{KL}(q_\phi \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))}_{KL \text{ between } q_\phi \text{ and prior}}
\end{aligned}$$

- Maximizing ELBO by EM via apply stochastic gradient ascent to θ and ϕ.

$$\max_\phi \max_\theta \ \mathrm{ELBO}(x; q_\phi, \theta)$$

Variational Auto-Encoder

- M-step:

$$
\begin{aligned}
\nabla_\theta \ \mathrm{ELBO}(x; q_\phi, \theta) &= \nabla_\theta \ \mathbb{E}_{z \sim q_\phi}[\log p(x \mid z; \theta)] \\
&\approx \nabla_\theta \ \frac{1}{n} \sum_{i=1}^{n} \log p(x \mid z^{(i)}; \theta) \\
&\propto -\nabla_\theta \ \frac{1}{2n} \sum_{i=1}^{n} \|x - \mathrm{Decoder}(z^{(i)}; \theta)\|^2
\end{aligned}
$$

- E-step:

$$
\nabla_\phi \ \mathrm{ELBO}(x; q_\phi, \theta) = \nabla_\phi \left(\mathbb{E}_{z \sim q_\phi}[\log p(x \mid z; \theta)] - D_{KL}(q_\phi \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) \right)
$$

The Reparameterization Trick

- The sampling distribution q_ϕ depends on ϕ.
- Recall the property of Gaussian distribution:

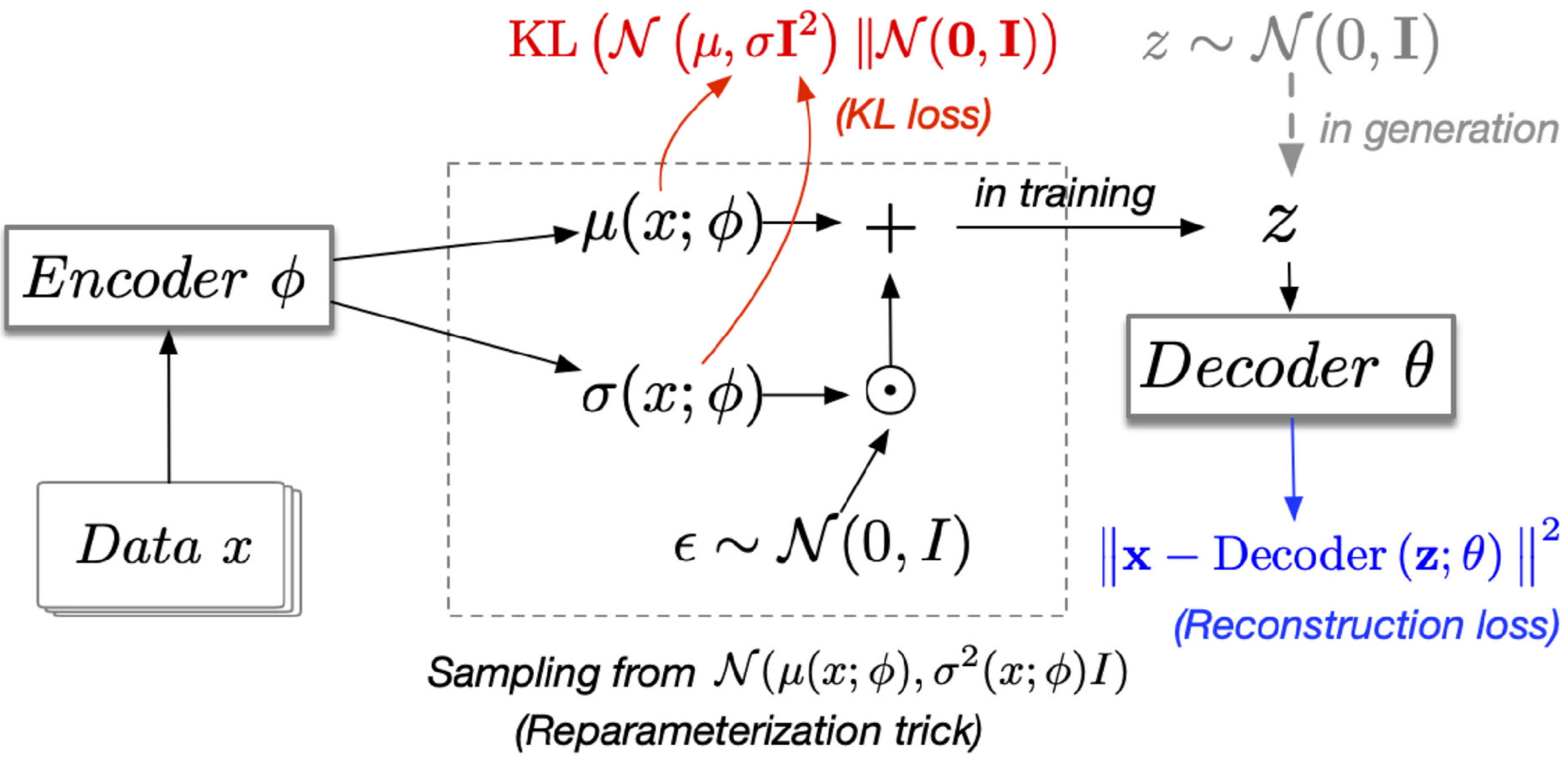
$$z \sim q_\phi = \mathcal{N}(\mu(x;\phi), \mathrm{diag}(\sigma(x;\phi))^2)$$

$$\Longleftrightarrow z = \mu(x;\phi) + \sigma(x;\phi) \odot \epsilon, \ \ \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

- E-step:

$$\begin{aligned} \nabla_\phi \ \mathbb{E}_{z \sim q_\phi}[\log p(x \mid z;\theta)] &= \nabla_\phi \ \mathbb{E}_{\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})}[\log p(x \mid \mu(x;\phi) + \sigma(x;\phi) \odot \epsilon;\theta)] \\ &= \mathbb{E}_{\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})}[\nabla_\phi \ \log p(x \mid \mu(x;\phi) + \sigma(x;\phi) \odot \epsilon;\theta)] \end{aligned}$$

Summary


$\mathrm{KL}\left(\mathcal{N}\left(\mu, \sigma\mathbf{I}^2\right) \| \mathcal{N}(\mathbf{0}, \mathbf{I})\right)$
(KL loss)
$z \sim \mathcal{N}(0, \mathbf{I})$
in generation
in training
$Encoder\ \phi$
$\mu(x; \phi)$
$+$
z
$\sigma(x; \phi)$
$\odot$
$Decoder\ \theta$
$Data\ x$
$\epsilon \sim \mathcal{N}(0, I)$
$\left\|\mathbf{x} - \mathrm{Decoder}\left(\mathbf{z}; \theta\right)\right\|^2$
(Reconstruction loss)
Sampling from $\mathcal{N}(\mu(x; \phi), \sigma^2(x; \phi)I)$
(Reparameterization trick)

References

[1] Ming Ding. The road from MLE to EM to VAE: A brief tutorial. AI Open, 3:29-34, 2022.

[2] Lilian Weng. From Autoencoder to Beta-VAE. 2018. https://lilianweng.github.io/posts/2018-08-12-vae/.

[3] Tengyu Ma and Andrew Ng. CS229 Lecture notes. 13 May 2019. https://cs229.stanford.edu/notes2020spring/cs229-notes8.pdf.