

# Deep Generative Models

## Lecture 3

Roman Isachenko



AI Masters

Autumn, 2022

# Recap of previous lecture

## MLE problem for autoregressive model

$$\theta^* = \arg \max_{\theta} p(\mathbf{X}|\theta) = \arg \max_{\theta} \sum_{i=1}^n \sum_{j=1}^m \log p(x_{ij}|\mathbf{x}_{i,1:j-1}, \theta).$$

## Sampling

$$\hat{x}_1 \sim p(x_1|\theta), \quad \hat{x}_2 \sim p(x_2|\hat{x}_1, \theta), \quad \dots, \quad \hat{x}_m \sim p(x_m|\hat{\mathbf{x}}_{1:m-1}, \theta)$$

New generated object is  $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$ .

Masking helps to make neural network autoregressive.

- ▶ **MADE** - masked autoencoder (MLP).
- ▶ **WaveNet** - masked 1D convolutions.
- ▶ **PixelCNN** - masked 2D convolutions.

# Recap of previous lecture

## Posterior distribution

$$p(\boldsymbol{\theta}|\mathbf{X}) = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})} = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

## Bayesian inference

$$p(\mathbf{x}|\mathbf{X}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{X})d\boldsymbol{\theta}$$

## Maximum a posteriori (MAP) estimation

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) = \arg \max_{\boldsymbol{\theta}} (\log p(\mathbf{X}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}))$$

## MAP inference

$$p(\mathbf{x}|\mathbf{X}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{X})d\boldsymbol{\theta} \approx p(\mathbf{x}|\boldsymbol{\theta}^*).$$

# Recap of previous lecture

## Latent variable models (LVM)

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})d\mathbf{z} = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z})d\mathbf{z}.$$

## MLE problem for LVM

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{X}|\boldsymbol{\theta}) = \arg \max_{\boldsymbol{\theta}} \sum_{i=1}^n \log p(\mathbf{x}_i|\boldsymbol{\theta}) = \\ &= \arg \max_{\boldsymbol{\theta}} \log \sum_{i=1}^n \int p(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\theta})p(\mathbf{z}_i)d\mathbf{z}_i.\end{aligned}$$

## Naive Monte-Carlo estimation

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z})d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})}p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) \approx \frac{1}{K} \sum_{k=1}^K p(\mathbf{x}|\mathbf{z}_k, \boldsymbol{\theta}),$$

where  $\mathbf{z}_k \sim p(\mathbf{z})$ .

# Recap of previous lecture

## ELBO derivation 1 (inequality)

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} \geq \mathbb{E}_q \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} = \mathcal{L}(q, \boldsymbol{\theta})$$

## ELBO derivation 2 (equality)

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\theta}) &= \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) p(\mathbf{x}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \\ &= \log p(\mathbf{x}|\boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \end{aligned}$$

## Variational decomposition

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(q, \boldsymbol{\theta}).$$

# Outline

1. EM-algorithm, amortized inference
2. ELBO gradients, reparametrization trick
3. Variational autoencoder (VAE)

# Outline

1. EM-algorithm, amortized inference
2. ELBO gradients, reparametrization trick
3. Variational autoencoder (VAE)

## Variational lower bound (ELBO)

$$\begin{aligned}\mathcal{L}(q, \theta) &= \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z} | \theta)}{q(\mathbf{z})} d\mathbf{z} = \\ &= \int q(\mathbf{z}) \log p(\mathbf{x} | \mathbf{z}, \theta) d\mathbf{z} + \int q(\mathbf{z}) \log \frac{p(\mathbf{z})}{q(\mathbf{z})} d\mathbf{z} \\ &= \mathbb{E}_q \log p(\mathbf{x} | \mathbf{z}, \theta) - KL(q(\mathbf{z}) || p(\mathbf{z}))\end{aligned}$$

### Log-likelihood decomposition

$$\begin{aligned}\log p(\mathbf{x} | \theta) &= \mathcal{L}(q, \theta) + KL(q(\mathbf{z}) || p(\mathbf{z} | \mathbf{x}, \theta)) \\ &= \mathbb{E}_q \log p(\mathbf{x} | \mathbf{z}, \theta) - KL(q(\mathbf{z}) || p(\mathbf{z})) + KL(q(\mathbf{z}) || p(\mathbf{z} | \mathbf{x}, \theta)).\end{aligned}$$

- Instead of maximizing incomplete likelihood, maximize ELBO

$$\max_{\theta} p(\mathbf{x} | \theta) \quad \rightarrow \quad \max_{q, \theta} \mathcal{L}(q, \theta)$$

- Maximization of ELBO by **variational** distribution  $q$  is equivalent to minimization of KL

$$\arg \max_q \mathcal{L}(q, \theta) \equiv \arg \min_q KL(q(\mathbf{z}) || p(\mathbf{z} | \mathbf{x}, \theta)).$$



# EM-algorithm

$$\begin{aligned}\mathcal{L}(q, \theta) &= \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \theta) - KL(q(\mathbf{z})||p(\mathbf{z})) = \\ &= \mathbb{E}_q \left[ \log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z})}{p(\mathbf{z})} \right] d\mathbf{z} \rightarrow \max_{q, \theta}.\end{aligned}$$

## Block-coordinate optimization

- ▶ Initialize  $\theta^*$ ;
- ▶ **E-step** ( $\mathcal{L}(q, \theta) \rightarrow \max_q$ )

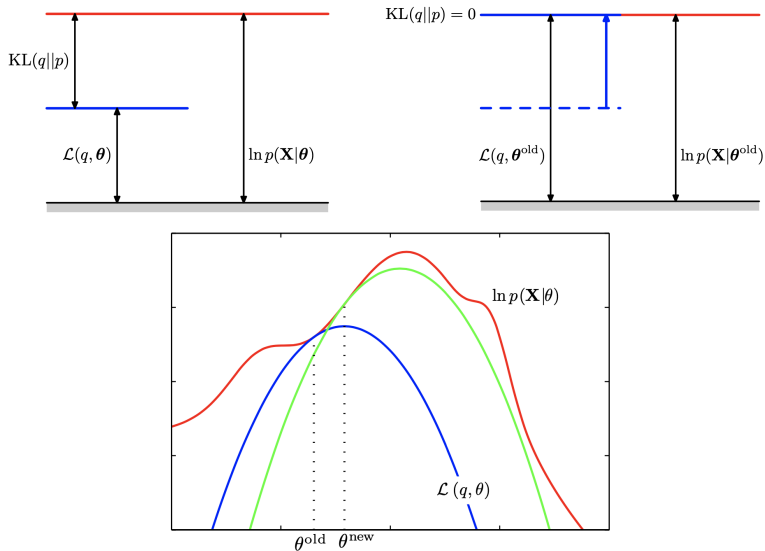
$$\begin{aligned}q^*(\mathbf{z}) &= \arg \max_q \mathcal{L}(q, \theta^*) = \\ &= \arg \min_q KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \theta^*)) = p(\mathbf{z}|\mathbf{x}, \theta^*);\end{aligned}$$

- ▶ **M-step** ( $\mathcal{L}(q, \theta) \rightarrow \max_\theta$ )

$$\theta^* = \arg \max_\theta \mathcal{L}(q^*, \theta);$$

- ▶ Repeat E-step and M-step until convergence.

# EM-algorithm illustration



# Amortized variational inference

## E-step

$$q(\mathbf{z}) = \arg \max_q \mathcal{L}(q, \theta^*) = \arg \min_q KL(q||p) = p(\mathbf{z}|\mathbf{x}, \theta^*).$$

- ▶  $q(\mathbf{z})$  approximates true posterior distribution  $p(\mathbf{z}|\mathbf{x}, \theta^*)$ , that is why it is called **variational posterior**;
- ▶  $p(\mathbf{z}|\mathbf{x}, \theta^*)$  could be **intractable**;
- ▶  $q(\mathbf{z})$  is different for each object  $\mathbf{x}$ .

## Idea

Restrict a family of all possible distributions  $q(\mathbf{z})$  to a parametric class  $q(\mathbf{z}|\mathbf{x}, \phi)$  conditioned on samples  $\mathbf{x}$  with parameters  $\phi$ .

## Variational Bayes

- ▶ E-step

$$\phi_k = \phi_{k-1} + \eta \nabla_{\phi} \mathcal{L}(\phi, \theta_{k-1})|_{\phi=\phi_{k-1}}$$

- ▶ M-step

$$\theta_k = \theta_{k-1} + \eta \nabla_{\theta} \mathcal{L}(\phi_k, \theta)|_{\theta=\theta_{k-1}}$$

# Variational EM-algorithm

## ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) + KL(q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}).$$

### ► E-step

$$\boldsymbol{\phi}_k = \boldsymbol{\phi}_{k-1} + \eta \nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}_{k-1})|_{\boldsymbol{\phi}=\boldsymbol{\phi}_{k-1}},$$

where  $\boldsymbol{\phi}$  – parameters of variational posterior distribution  $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ .

### ► M-step

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}_k, \boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_{k-1}},$$

where  $\boldsymbol{\theta}$  – parameters of the generative distribution  $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ .

Now all we have to do is to obtain two gradients  $\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$ ,  $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$ .

**Challenge:** Number of samples  $n$  could be huge (we need to derive unbiased stochastic gradients).

# Outline

1. EM-algorithm, amortized inference
2. ELBO gradients, reparametrization trick
3. Variational autoencoder (VAE)

## ELBO gradients, (M-step, $\nabla_{\theta} \mathcal{L}(\phi, \theta)$ )

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \phi)} \left[ \log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] \rightarrow \max_{\phi, \theta}.$$

M-step:  $\nabla_{\theta} \mathcal{L}(\phi, \theta)$

$$\begin{aligned} \nabla_{\theta} \mathcal{L}(\phi, \theta) &= \int q(\mathbf{z}|\mathbf{x}, \phi) \nabla_{\theta} \log p(\mathbf{x}|\mathbf{z}, \theta) d\mathbf{z} \approx \\ &\approx \nabla_{\theta} \log p(\mathbf{x}|\mathbf{z}^*, \theta), \quad \mathbf{z}^* \sim q(\mathbf{z}|\mathbf{x}, \phi). \end{aligned}$$

Naive Monte-Carlo estimation

$$p(\mathbf{x}|\theta) = \int p(\mathbf{x}|\mathbf{z}, \theta) p(\mathbf{z}) d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})} p(\mathbf{x}|\mathbf{z}, \theta) \approx \frac{1}{K} \sum_{k=1}^K p(\mathbf{x}|\mathbf{z}_k, \theta),$$

where  $\mathbf{z}_k \sim p(\mathbf{z})$ .

The variational posterior  $q(\mathbf{z}|\mathbf{x}, \phi)$  assigns typically more probability mass in a smaller region than the prior  $p(\mathbf{z})$ .

image credit: [https://jmtomczak.github.io/blog/4/4\\_VAE.html](https://jmtomczak.github.io/blog/4/4_VAE.html)

## ELBO gradients, (E-step, $\nabla_{\phi} \mathcal{L}(\phi, \theta)$ )

### E-step: $\nabla_{\phi} \mathcal{L}(\phi, \theta)$

Difference from M-step: density function  $q(\mathbf{z}|\mathbf{x}, \phi)$  depends on the parameters  $\phi$ , it is impossible to use the Monte-Carlo estimation:

$$\begin{aligned}\nabla_{\phi} \mathcal{L}(\phi, \theta) &= \nabla_{\phi} \int q(\mathbf{z}|\mathbf{x}, \phi) \left[ \log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] d\mathbf{z} \\ &\neq \int q(\mathbf{z}|\mathbf{x}, \phi) \nabla_{\phi} \left[ \log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] d\mathbf{z}\end{aligned}$$

### Reparametrization trick (LOTUS trick)

- ▶  $r(x) = \mathcal{N}(x|0, 1)$ ,  $y = \sigma \cdot x + \mu$ ,  $p_Y(y|\theta) = \mathcal{N}(y|\mu, \sigma^2)$ ,  $\theta = [\mu, \sigma]$ .
- ▶  $\epsilon^* \sim r(\epsilon)$ ,  $\mathbf{z} = g(\mathbf{x}, \epsilon, \phi)$ ,  $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}, \phi)$

$$\begin{aligned}\nabla_{\phi} \int q(\mathbf{z}|\mathbf{x}, \phi) f(\mathbf{z}) d\mathbf{z} &= \nabla_{\phi} \int r(\epsilon) f(\mathbf{z}) d\epsilon \\ &= \int r(\epsilon) \nabla_{\phi} f(g(\mathbf{x}, \epsilon, \phi)) d\epsilon \approx \nabla_{\phi} f(g(\mathbf{x}, \epsilon^*, \phi))\end{aligned}$$

## ELBO gradient (E-step, $\nabla_{\phi} \mathcal{L}(\phi, \theta)$ )

$$\begin{aligned}\nabla_{\phi} \mathcal{L}(\phi, \theta) &= \nabla_{\phi} \int q(\mathbf{z}|\mathbf{x}, \phi) \log p(\mathbf{x}|\mathbf{z}, \theta) d\mathbf{z} - \nabla_{\phi} \text{KL}(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z})) \\ &= \int r(\epsilon) \nabla_{\phi} \log p(\mathbf{x}|g(\mathbf{x}, \epsilon, \phi), \theta) d\epsilon - \nabla_{\phi} \text{KL}(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z})) \\ &\approx \nabla_{\phi} \log p(\mathbf{x}|g(\mathbf{x}, \epsilon^*, \phi), \theta) - \nabla_{\phi} \text{KL}(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z}))\end{aligned}$$

### Variational assumption

$$r(\epsilon) = \mathcal{N}(0, \mathbf{I}); \quad q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mu_{\phi}(\mathbf{x}), \sigma_{\phi}^2(\mathbf{x})).$$

$$\mathbf{z} = g(\mathbf{x}, \epsilon, \phi) = \sigma_{\phi}(\mathbf{x}) \cdot \epsilon + \mu_{\phi}(\mathbf{x}).$$

Here  $\mu_{\phi}(\cdot), \sigma_{\phi}(\cdot)$  are parameterized functions (outputs of neural network).

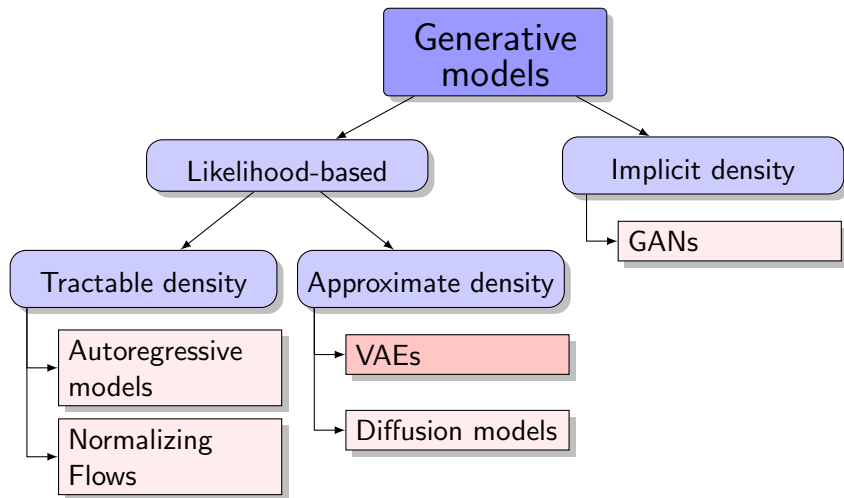
- ▶  $p(\mathbf{z})$  – prior distribution on latent variables  $\mathbf{z}$ . We could specify any distribution that we want. Let say  $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ .
- ▶  $p(\mathbf{x}|\mathbf{z}, \theta)$  – generative distribution. Since it is a parameterized function let it be neural network with parameters  $\theta$ .



# Outline

1. EM-algorithm, amortized inference
2. ELBO gradients, reparametrization trick
3. Variational autoencoder (VAE)

# Generative models zoo



# Variational autoencoder (VAE)

## Final EM-algorithm

- ▶ pick random sample  $\mathbf{x}_i, i \sim U[1, n]$ .
- ▶ compute the objective:

$$\epsilon^* \sim r(\epsilon); \quad \mathbf{z}^* = g(\mathbf{x}, \epsilon^*, \phi);$$

$$\mathcal{L}(\phi, \theta) \approx \log p(\mathbf{x}|\mathbf{z}^*, \theta) - KL(q(\mathbf{z}^*|\mathbf{x}, \phi)||p(\mathbf{z}^*)).$$

- ▶ compute a stochastic gradients w.r.t.  $\phi$  and  $\theta$

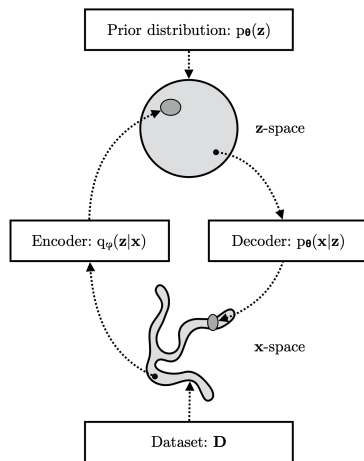
$$\begin{aligned}\nabla_{\phi} \mathcal{L}(\phi, \theta) &\approx \nabla_{\phi} \log p(\mathbf{x}|g(\mathbf{x}, \epsilon^*, \phi), \theta) - \nabla_{\phi} KL(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z})); \\ \nabla_{\theta} \mathcal{L}(\phi, \theta) &\approx \nabla_{\theta} \log p(\mathbf{x}|\mathbf{z}^*, \theta).\end{aligned}$$

- ▶ update  $\theta, \phi$  according to the selected optimization method (SGD, Adam, RMSProp):

$$\begin{aligned}\phi &:= \phi + \eta \nabla_{\phi} \mathcal{L}(\phi, \theta), \\ \theta &:= \theta + \eta \nabla_{\theta} \mathcal{L}(\phi, \theta).\end{aligned}$$

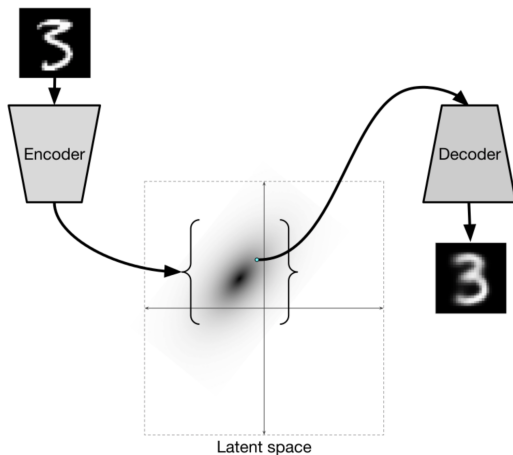
# Variational autoencoder (VAE)

- ▶ VAE learns stochastic mapping between  $\mathbf{x}$ -space, from complicated distribution  $\pi(\mathbf{x})$ , and a latent  $\mathbf{z}$ -space, with simple distribution.
- ▶ The generative model learns a joint distribution  $p(\mathbf{x}, \mathbf{z}|\theta) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z}, \theta)$ , with a prior distribution  $p(\mathbf{z})$ , and a stochastic decoder  $p(\mathbf{x}|\mathbf{z}, \theta)$ .
- ▶ The stochastic encoder  $q(\mathbf{z}|\mathbf{x}, \phi)$  (inference model), approximates the true but intractable posterior  $p(\mathbf{z}|\mathbf{x}, \theta)$  of the generative model.



# Variational Autoencoder

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \phi)} \left[ \log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] \rightarrow \max_{\phi, \theta}.$$



# Variational autoencoder (VAE)

- ▶ Encoder  $q(\mathbf{z}|\mathbf{x}, \phi) = \text{NN}_e(\mathbf{x}, \phi)$  outputs  $\mu_\phi(\mathbf{x})$  and  $\sigma_\phi(\mathbf{x})$ .
- ▶ Decoder  $p(\mathbf{x}|\mathbf{z}, \theta) = \text{NN}_d(\mathbf{z}, \theta)$  outputs parameters of the sample distribution.

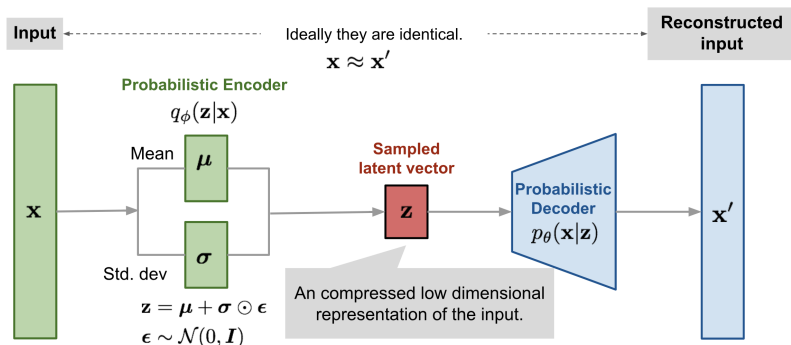


image credit:

<https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html>

# Summary

- ▶ The general variational EM algorithm maximizes ELBO objective for LVM model to find MLE for parameters  $\theta$ .
- ▶ Amortized variational inference allows to efficiently compute the stochastic gradients for ELBO using Monte-Carlo estimation.
- ▶ The reparametrization trick gets unbiased gradients w.r.t to the variational posterior distribution  $q(\mathbf{z}|\mathbf{x}, \phi)$ .
- ▶ The VAE model is an LVM with two neural network: stochastic encoder  $q(\mathbf{z}|\mathbf{x}, \phi)$  and stochastic decoder  $p(\mathbf{x}|\mathbf{z}, \theta)$ .