

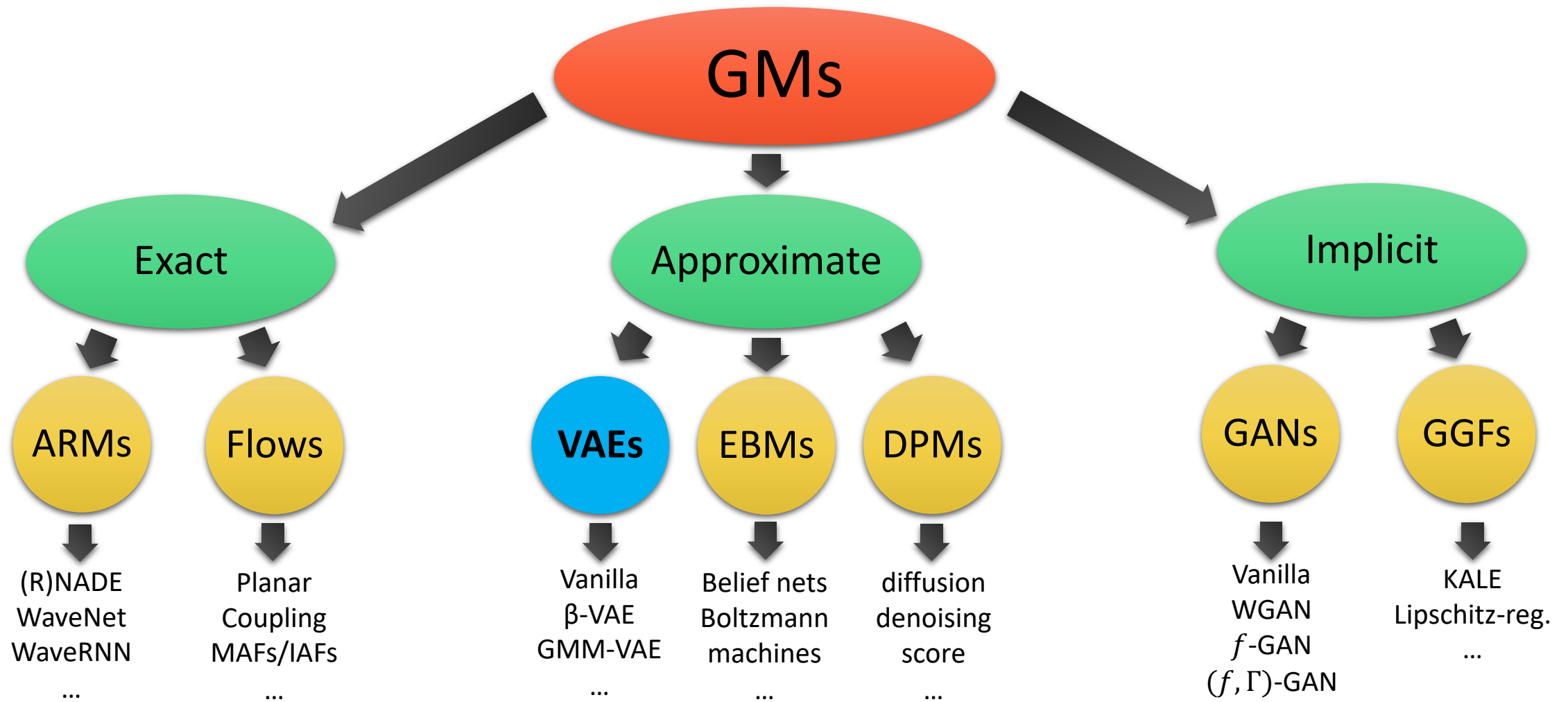
# Introduction to Deep Generative Modeling

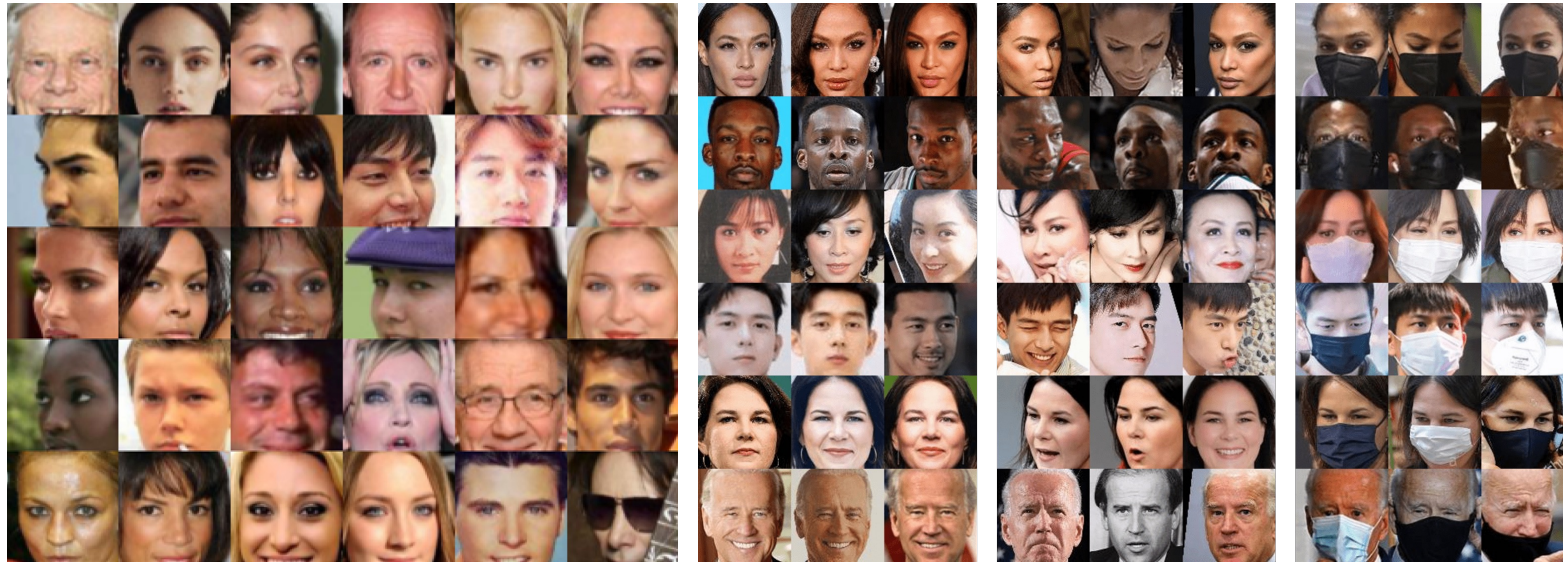
Lecture #9

**HY-673** – Computer Science Dep., University of Crete

Professors: Yannis Pantazis, Yannis Stylianos

Teaching Assistant: Michail Raptakis

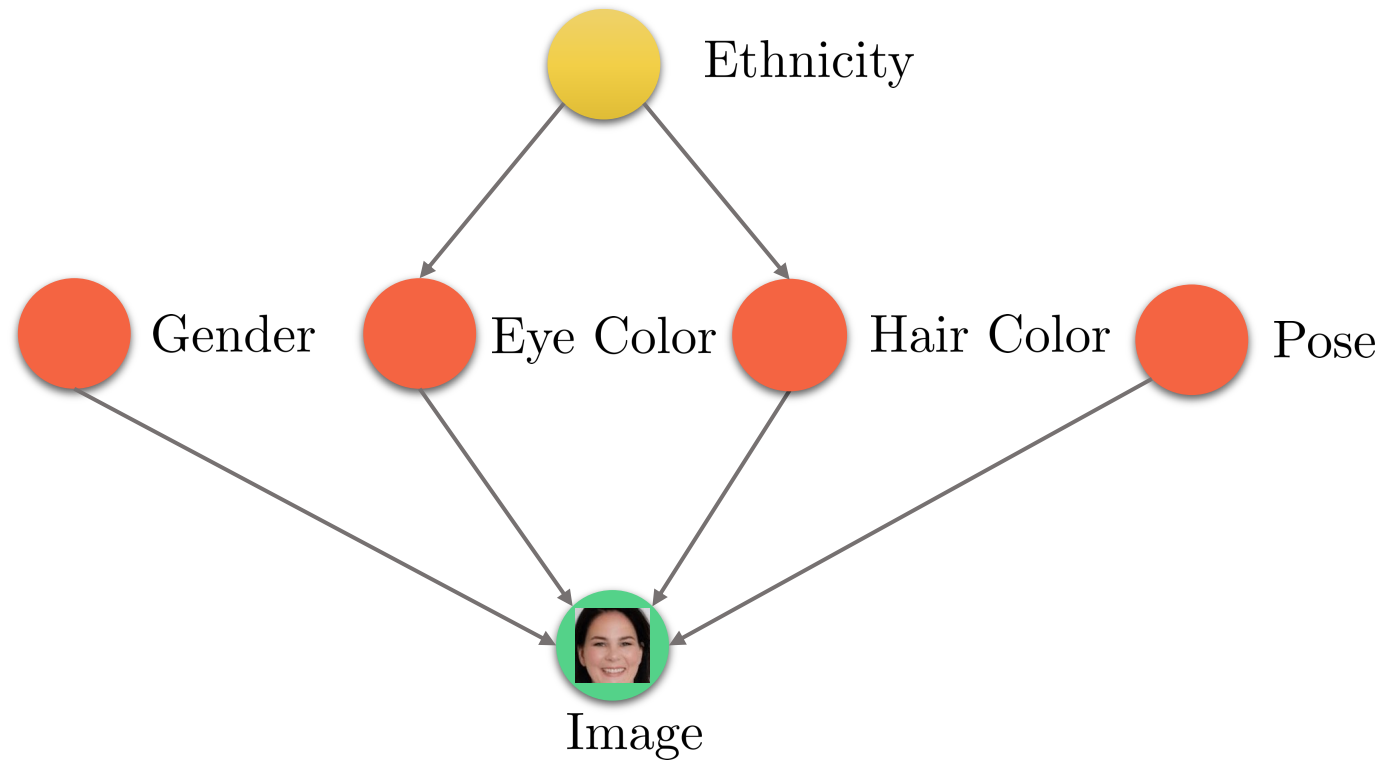
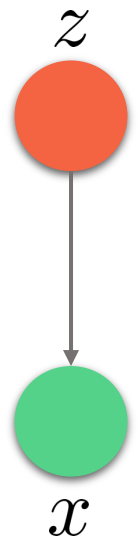




- Lots of variability in images  $x$  due to gender, eye color, hair color, pose, etc. However, unless images are annotated, these factors of variation are not explicitly available (**latent**).
- **Idea:** Explicitly model these factors using latent variables  $z$ .

1. Only variable  $x$  is observed (pixel values).
2. Latent variable  $z$  correspond to high level features.
  - If  $z$  is chosen properly,  $p(x|z)$  could be much simpler than  $p(x)$ .
  - If we had trained this model, then we could identify features via  $p(z|x)$ , e.g.,  $p(\text{EyeColor} = \text{Blue}|x)$ .
3. **Challenge:** Very difficult to specify these conditionals by hand.

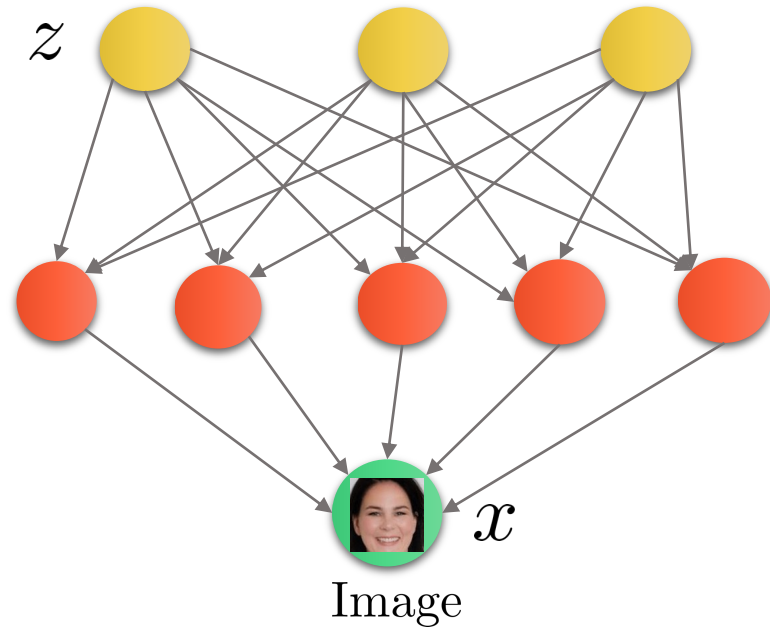
# Latent Variable Models: Motivation



- Use neural networks to model the conditionals (deep latent variable models):
  1.  $z \sim \mathcal{N}(0, I)$
  2.  $p(x|z) = \mathcal{N}(\mu_\theta(z), \Sigma_\theta(z))$  where  $\mu_\theta, \Sigma_\theta$  are the output of a neural network
- Hope that after training,  $z$  will correspond to meaningful latent factors of variation (features).  
→ A type of Unsupervised representation learning.
- Features can be computed via  $p(z|x)$ .

# Deep Latent Variable Models

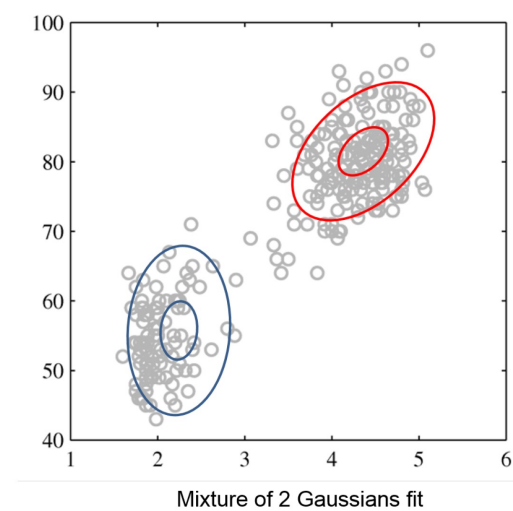
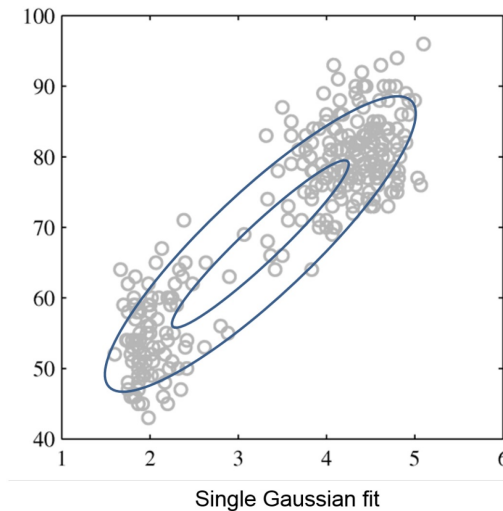
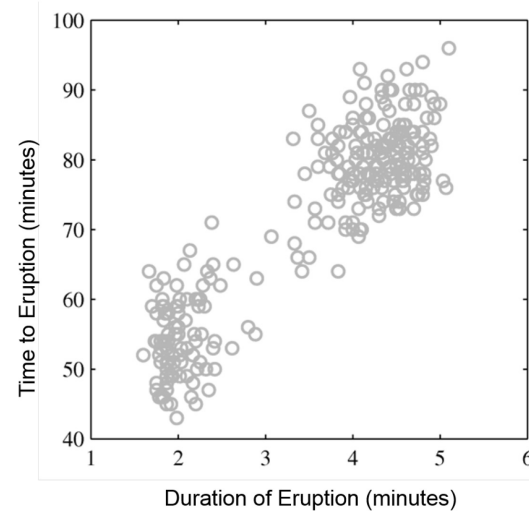
Lecture #9



# Mixture of Gaussians: A Shallow Latent Variable Model

Mixture of Gaussians:

1.  $z \sim \text{Categorical}(1, \dots, K)$ .
2.  $p(x|z = k) = \mathcal{N}(\mu_k, \Sigma_k)$ .



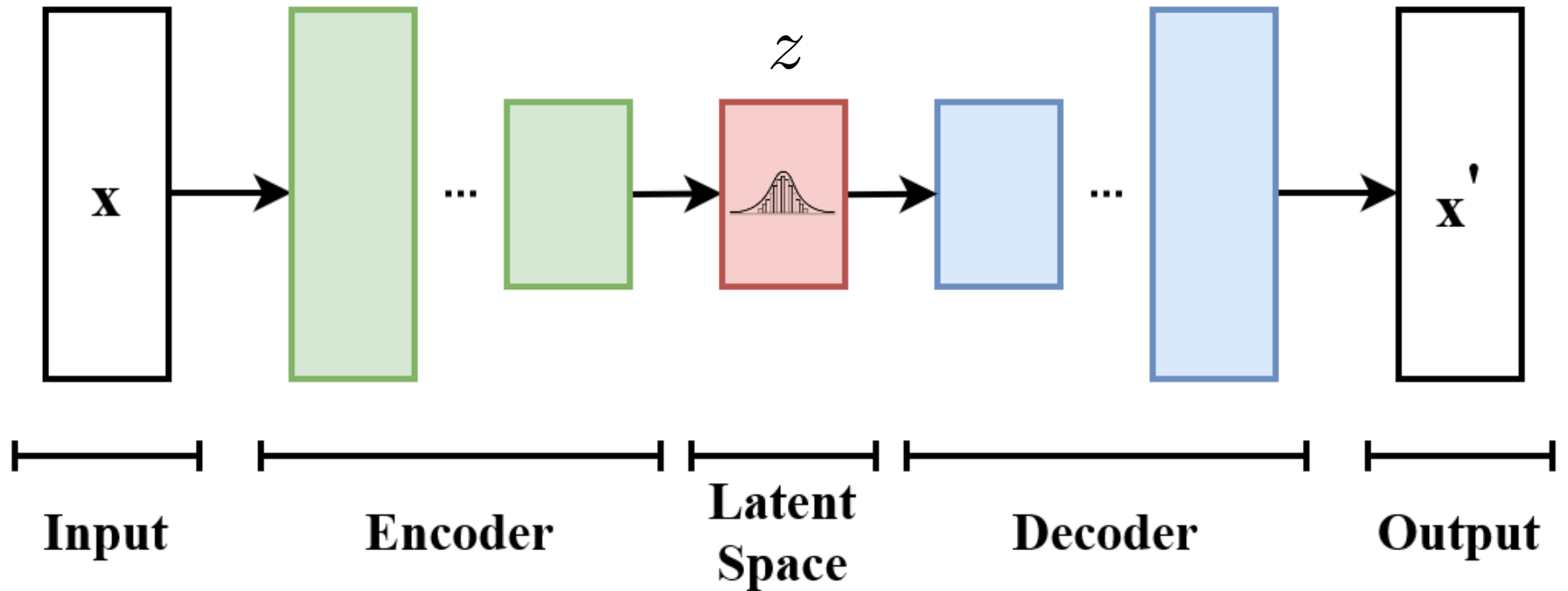
- **Clustering:** Posterior  $p(z|x)$  identifies mixture component.
- **Unsupervised learning:** Hope to learn from unlabeled data (ill-posed).



- Latent Variable Models:
  - Allow us to define complex models  $p(x)$  in terms of simpler building blocks  $p(x|z)$ .
  - Natural for unsupervised learning tasks (clustering, unsupervised representation learning, etc.)
  - No free lunch: much more difficult to learn compared to fully observed, autoregressive models.

# Variational Autoencoder (VAE)

Lecture #9



- $p_{\theta}(x, z) = p_{\theta}(x|z)p_{\theta}(z)$ : joint generative distribution.
  - $p_{\theta}(z)$ : prior distribution.
  - $p_{\theta}(x|z)$ : likelihood of the (stochastic) decoder.
- $p_{\theta}(x, z) = p_{\theta}(z|x)p_{\theta}(x)$  where
  - $p_{\theta}(x)$ : marginal likelihood or model evidence.
  - $p_{\theta}(z|x)$ : posterior distribution.

By design:

- It is easy to sample from  $p_{\theta}(x, z) = p_{\theta}(x|z)p_{\theta}(z)$ .
- Marginal  $p_{\theta}(x) = \int p_{\theta}(x, z)dz$  is very complex/flexible and unfortunately intractable.
  - If both  $p_{\theta}(z)$  and  $p_{\theta}(x|z)$  are Gaussians then  $p_{\theta}(x)$  is an infinite mixture of Gaussians.
- Consequently, the posterior distribution  $p_{\theta}(z|x)$  is also intractable.
- Our aim is:

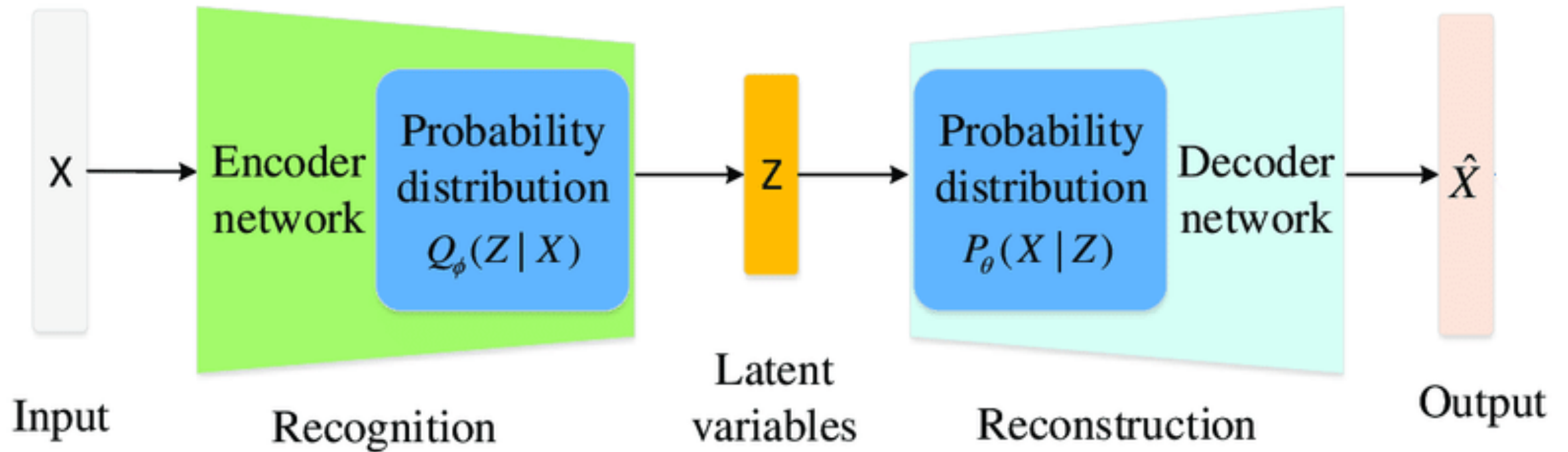
$$p_{\theta}(x) \approx p_d(x)$$

- Key idea of variational inference: approximate the intractable posterior with a (parametric) inference model.
- Mathematically, we introduce  $q_\phi(z|x)$  such that

$$q_\phi(z|x) \approx p_\theta(z|x)$$

- Why is called variational?  
Simply because we optimize w.r.t. a **function** (of  $z$  conditioned on  $x$ ).

# Variational Inference



1. Prior distribution is isotropic/spherical Gaussian:

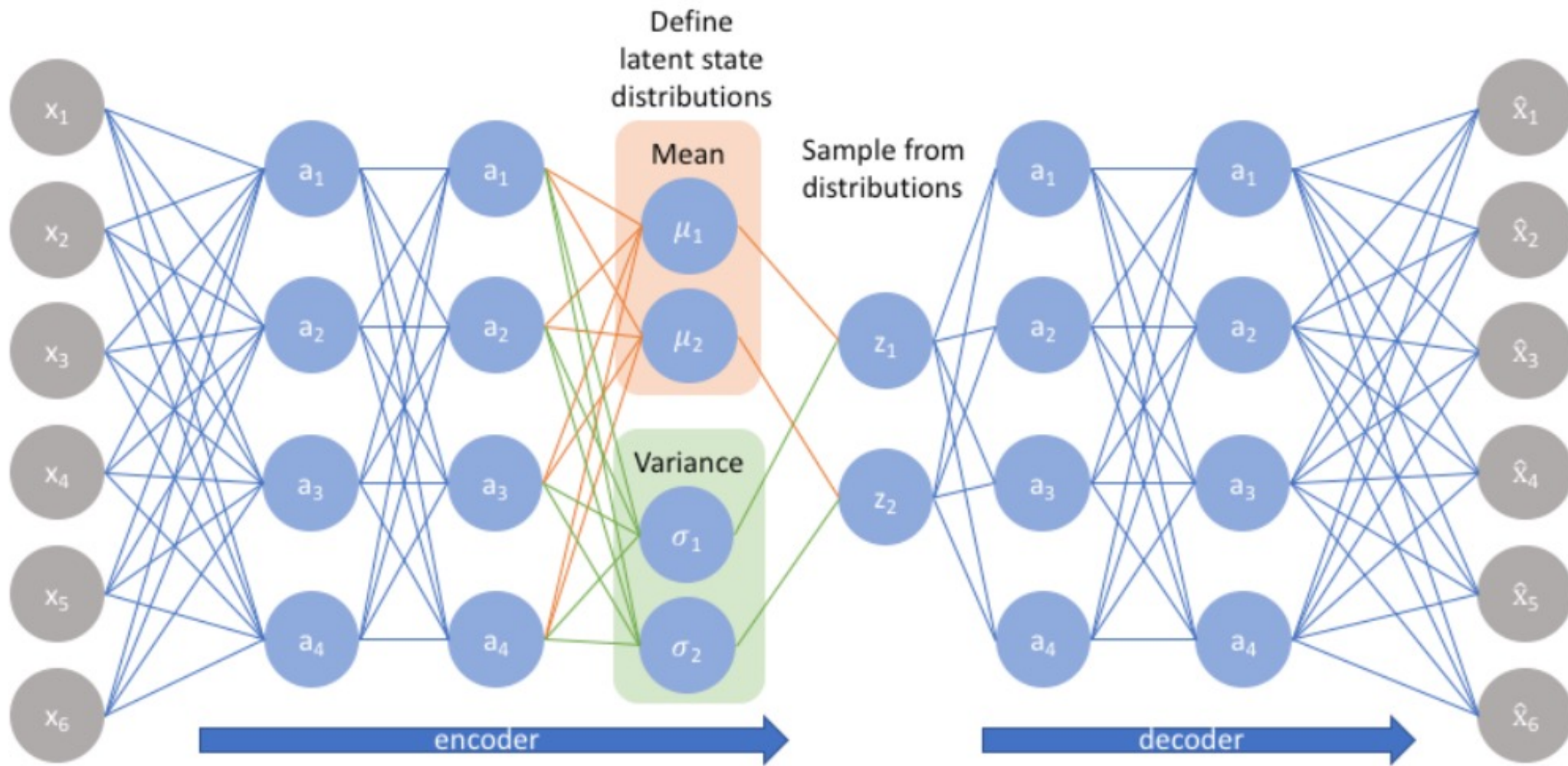
$$p(z) = \mathcal{N}(0, I).$$

2. Stochastic decoder is Gaussian:

$$p_{\theta}(x|z) = \mathcal{N}(\mu_{\theta}(z), \text{diag}(\sigma_{\theta}(z))) \text{ where } \mu_{\theta}, \sigma_{\theta} \text{ are neural networks.}$$

3. Stochastic encoder (i.e., inference or recognition model) is Gaussian:

$$q_{\phi}(z|x) = \mathcal{N}(\mu_{\phi}(x), \text{diag}(\sigma_{\phi}(x))) \text{ where } \mu_{\phi}, \sigma_{\phi} \text{ are also neural networks.}$$



- VAE resembles an autoencoder when  $\dim(z) < \dim(x)$ .



# How to train a VAE model?

We will employ two tricks:

1. Approximate the model evidence with a lower bound called **ELBO** (from Evidence Lower Bound) and maximize ELBO instead of the evidence.
2. Reparametrization trick for efficient gradient estimation.

# The Evidence Lower Bound (ELBO)

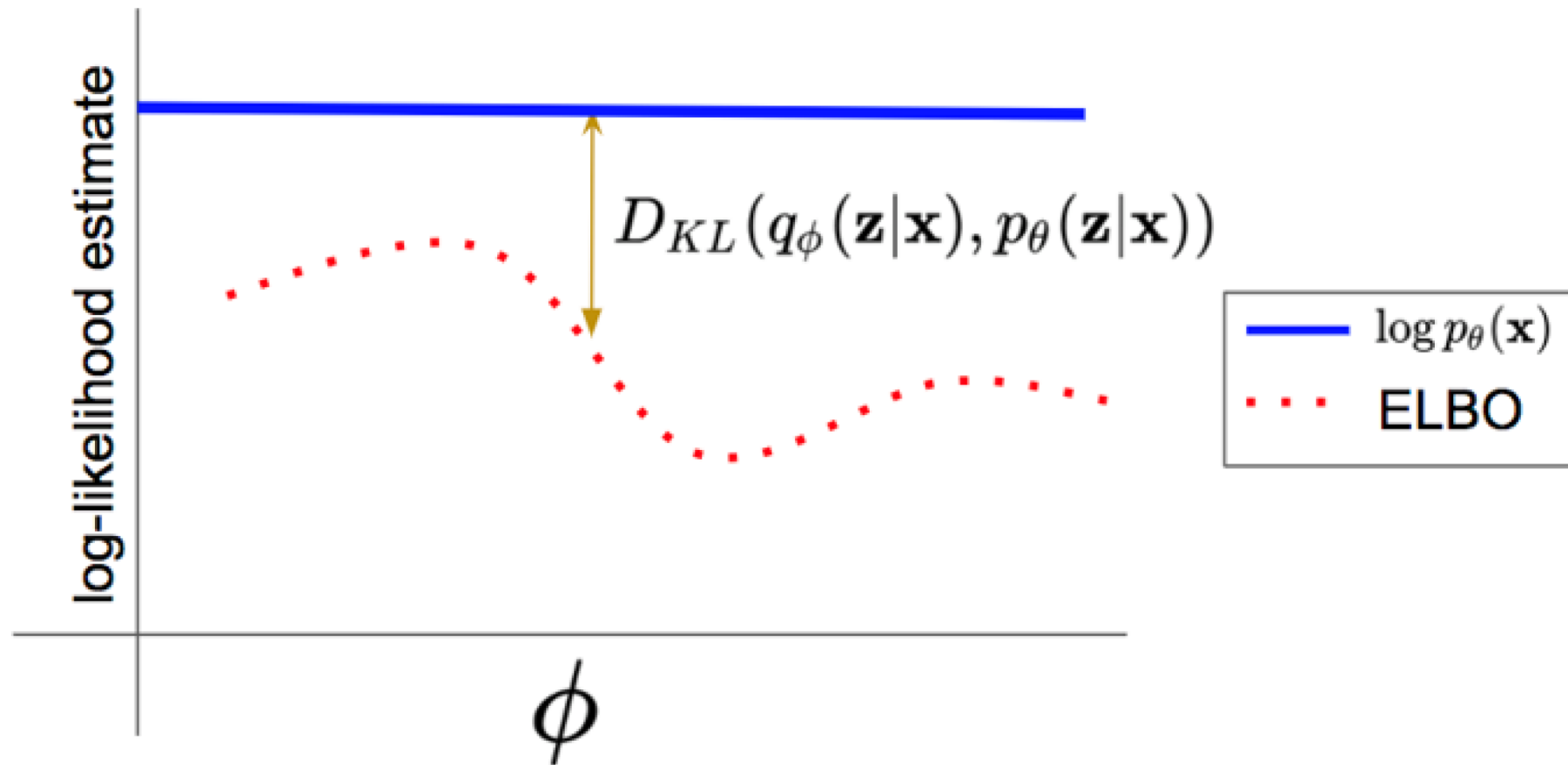
- Evidence or the (marginal) likelihood for a single data  $x$  equals to

$$\begin{aligned} \log p_{\theta}(x) &= \underbrace{\mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right]}_{= \mathcal{L}_{\theta, \phi}(x) \text{ (ELBO)}} + \underbrace{\mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right]}_{= D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))} \\ &= \mathcal{L}_{\theta, \phi}(x) \text{ (ELBO)} \quad = D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x)) \end{aligned}$$

- Since  $D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x)) \geq 0$ , it holds that

$$\log p_{\theta}(x) \geq \mathcal{L}_{\theta, \phi}(x)$$

# The Evidence Lower Bound (ELBO)



The better  $q_{\phi}(z|x)$  can approximate the posterior  $p_{\theta}(z|x)$ , the smaller  $D_{KL}(q_{\phi}(z|x)||p_{\theta}(z|x))$  we can achieve, thus, the closer ELBO will be to  $\log p_{\theta}(x)$ .

Next: Jointly optimizer over  $\theta$  and  $\phi$  to maximize the ELBO over a dataset.

- Evidence or the (marginal) likelihood for a single data  $x$  equals to

$$\begin{aligned} \log p_{\theta}(x) &= \underbrace{\mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right]}_{= \mathcal{L}_{\theta, \phi}(x) \text{ (ELBO)}} + \underbrace{\mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right]}_{= D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))} \end{aligned}$$

- Since  $D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x)) \geq 0$ , it holds that

$$\log p_{\theta}(x) \geq \mathcal{L}_{\theta, \phi}(x)$$

- ELBO holds for any  $q_\phi(z|x)$ :

$$\log p_\theta(x) \geq \mathcal{L}_{\theta,\phi}(x).$$

- Maximum likelihood learning (over the entire dataset):

$$\ell(\theta; \mathcal{D}) = \sum_{x_i \in \mathcal{D}} \log p_\theta(x_i) \geq \sum_{x_i \in \mathcal{D}} \mathcal{L}_{\theta,\phi}(x).$$

- Therefore:

$$\max_{\theta} \ell(\theta; \mathcal{D}) \geq \max_{\theta,\phi} \sum_{x_i \in \mathcal{D}} \mathcal{L}_{\theta,\phi}(x_i).$$

- Recall

$$\mathcal{L}_{\theta, \phi}(x) = \mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right] = \mathbb{E}_{q_{\phi}(z|x)} [\log p_{\theta}(x, z)] - \mathbb{E}_{q_{\phi}(z|x)} [\log q_{\phi}(z|x)]$$

- The gradient with respect to  $\theta$  (easy):

$$\nabla_{\theta} \mathcal{L}_{\theta, \phi}(x) = \mathbb{E}_{q_{\phi}(z|x)} [\nabla_{\theta} \log p_{\theta}(x, z)] \approx \nabla_{\theta} \log p_{\theta}(x_i, z_i)$$

- The gradient with respect to  $\phi$  requires a trick:  $z_i \sim q_{\phi}(z|x_i)$

- Want to compute a gradient with respect to  $\phi$  of:

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

- Suppose  $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \text{diag}(\sigma_\phi^2(x)))$  is a Gaussian with  $\mu_\phi(x), \sigma_\phi(x)$  be neural nets. These are equivalent ways of sampling:

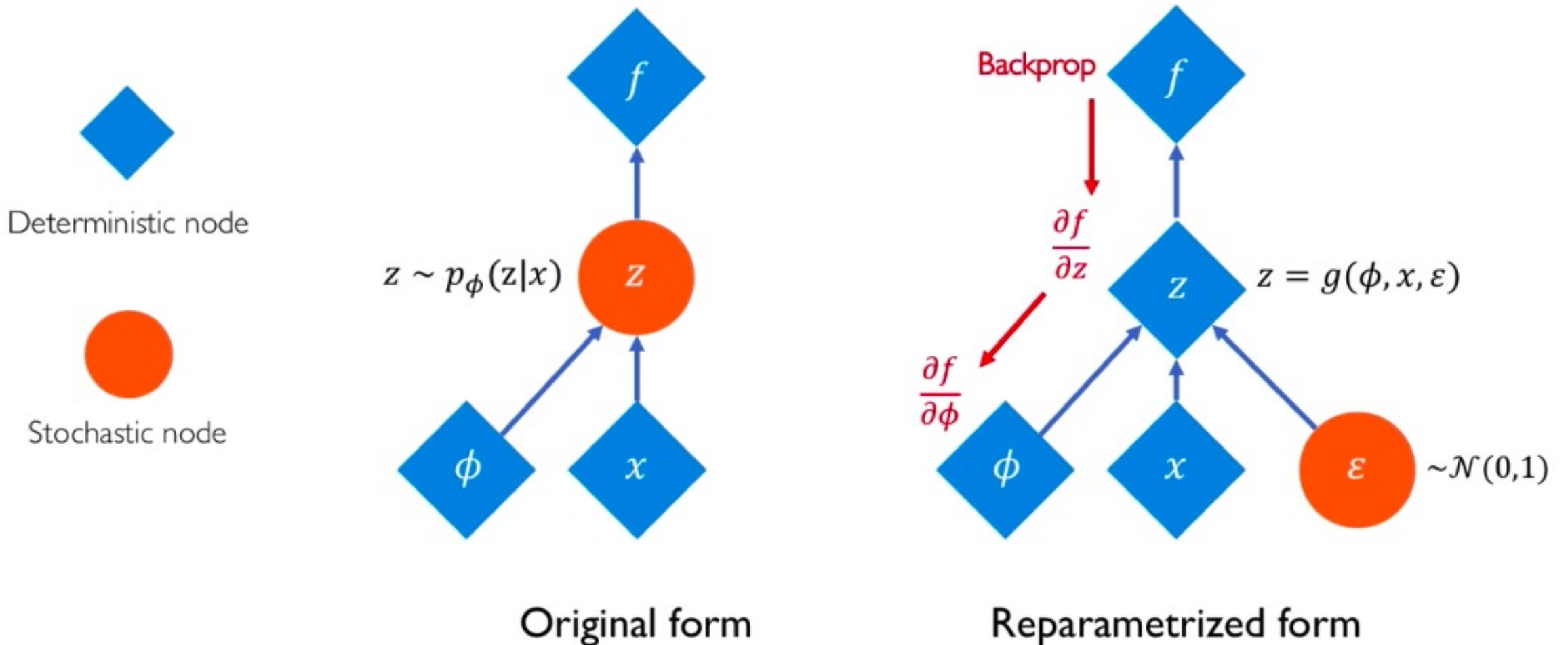
- Sample  $z \sim q_\phi(z|x)$ .

- Sample  $\epsilon \sim \mathcal{N}(0, I) =: p(\epsilon), z = \mu_\phi(x) + \sigma_\phi(x)\epsilon =: g_\phi(\epsilon, x)$ .

- Therefore:

$$\mathbb{E}_{q_\phi(z|x)}[f(z)] = \mathbb{E}_{\epsilon \sim p(\epsilon)}[f(g_\phi(\epsilon, x))] := \int f(\mu_\phi(x) + \sigma_\phi(x)\epsilon)p(\epsilon)d\epsilon.$$

# Reparametrization Trick





- Thus, the gradient w.r.t.  $\phi$  becomes

$$\nabla_{\phi} \mathbb{E}_{q_{\phi}(z|x)} [f(z)] = \nabla_{\phi} \mathbb{E}_{p(\epsilon)} [f(g(\epsilon, \phi, x))] = \mathbb{E}_{p(\epsilon)} [\nabla_{\phi} f(g(\epsilon, \phi, x))] \approx \nabla_{\phi} f(g(\epsilon_i, \phi, x_i)).$$

→ Easy to estimate Monte Carlo if  $f$  and  $g$  are differentiable w.r.t.  $\phi$  and  $\epsilon$  is easy to sample from.

- In VAEs

$$f(g(\epsilon, \phi, x)) = \log p_{\theta}(x, z) - \log q_{\phi}(z|x).$$

---

**Data:**

$\mathcal{D}$ : Dataset

$q_\phi(\mathbf{z}|\mathbf{x})$ : Inference model

$p_\theta(\mathbf{x}, \mathbf{z})$ : Generative model

**Result:**

$\theta, \phi$ : Learned parameters

$(\theta, \phi) \leftarrow$  Initialize parameters

**while** *SGD not converged* **do**

$\mathcal{M} \sim \mathcal{D}$  (Random minibatch of data)

$\epsilon \sim p(\epsilon)$  (Random noise for every datapoint in  $\mathcal{M}$ )

    Compute  $\tilde{\mathcal{L}}_{\theta, \phi}(\mathcal{M}, \epsilon)$  and its gradients  $\nabla_{\theta, \phi} \tilde{\mathcal{L}}_{\theta, \phi}(\mathcal{M}, \epsilon)$

    Update  $\theta$  and  $\phi$  using SGD optimizer

**end**

---

- Latent Variable Models Pros:
  - Easy to build flexible models.
  - Suitable for unsupervised learning.
- Latent Variable Models Cons:
  - Hard to evaluate likelihoods.
  - Hard to train via maximum-likelihood.
  - Fundamentally, the challenge is that posterior distribution  $p_{\theta}(z|x)$  is hard. Typically requires variational approximations.

1. Probabilistic Machine Learning: Advanced Topics (Chapter 20)  
Kevin P Murphy, The MIT Press (2023)
2. An Introduction to Variational Autoencoders, D. Kingma & M. Welling,  
Foundations and Trends in ML, 2019. (A coherent and accessible intro-  
duction to variational autoencoders - Highly recommended read!)  
<https://arxiv.org/abs/1906.02691>
3. Auto-Encoding Variational Bayes, D. Kingma & M. Welling, ICLR, 2014.
4. <https://github.com/matthewvowels1/Awesome-VAEs>