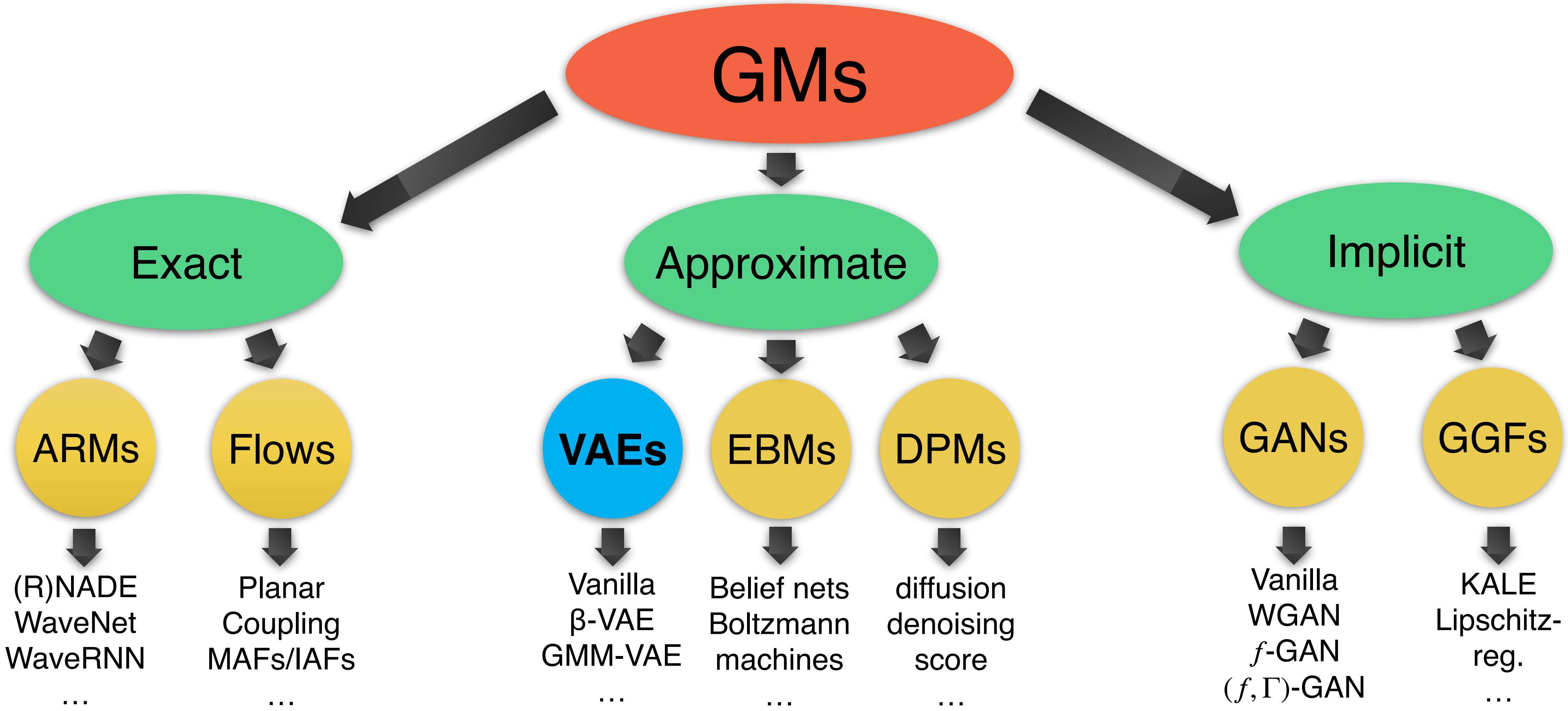
Introduction to Deep Generative Modeling

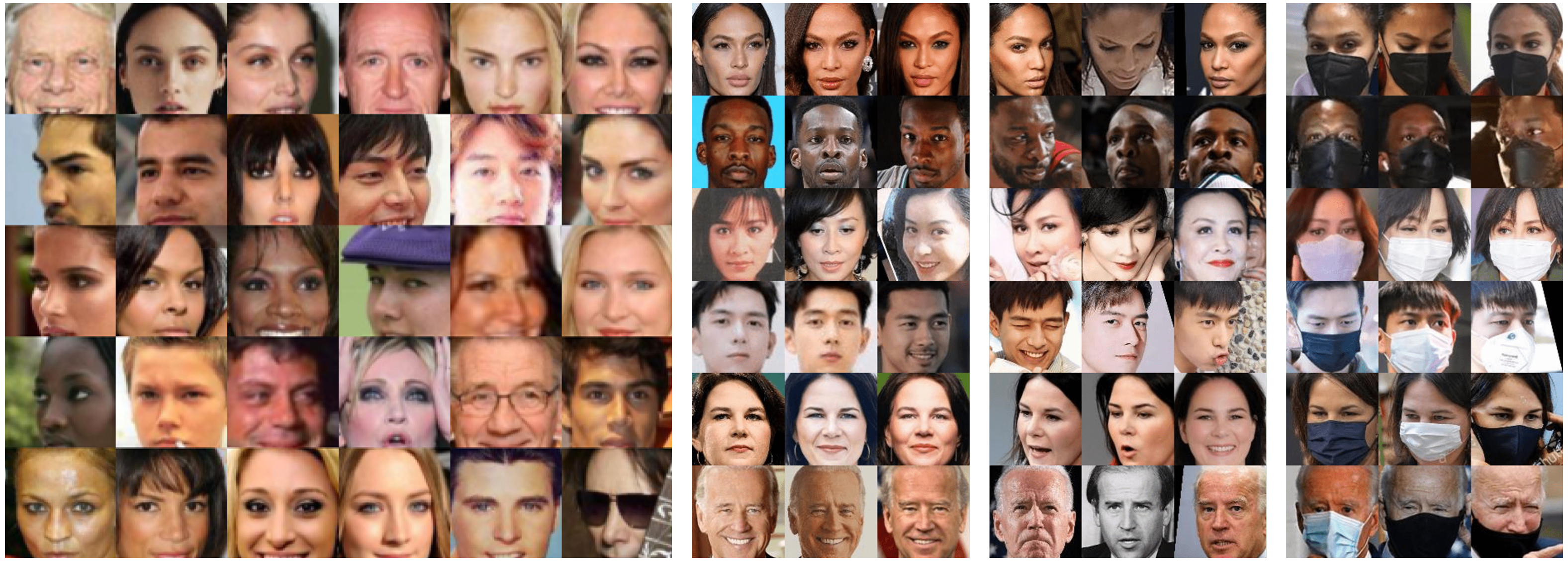
Lecture #10

HY-673 – Computer Science Dep., University of Crete
Professors: Yannis Pantazis, Yannis Stylianou
Teaching Assistant: Michail Raptakis

Taxonomy of GMs



Latent Variable Models: Motivation

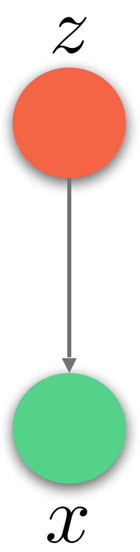


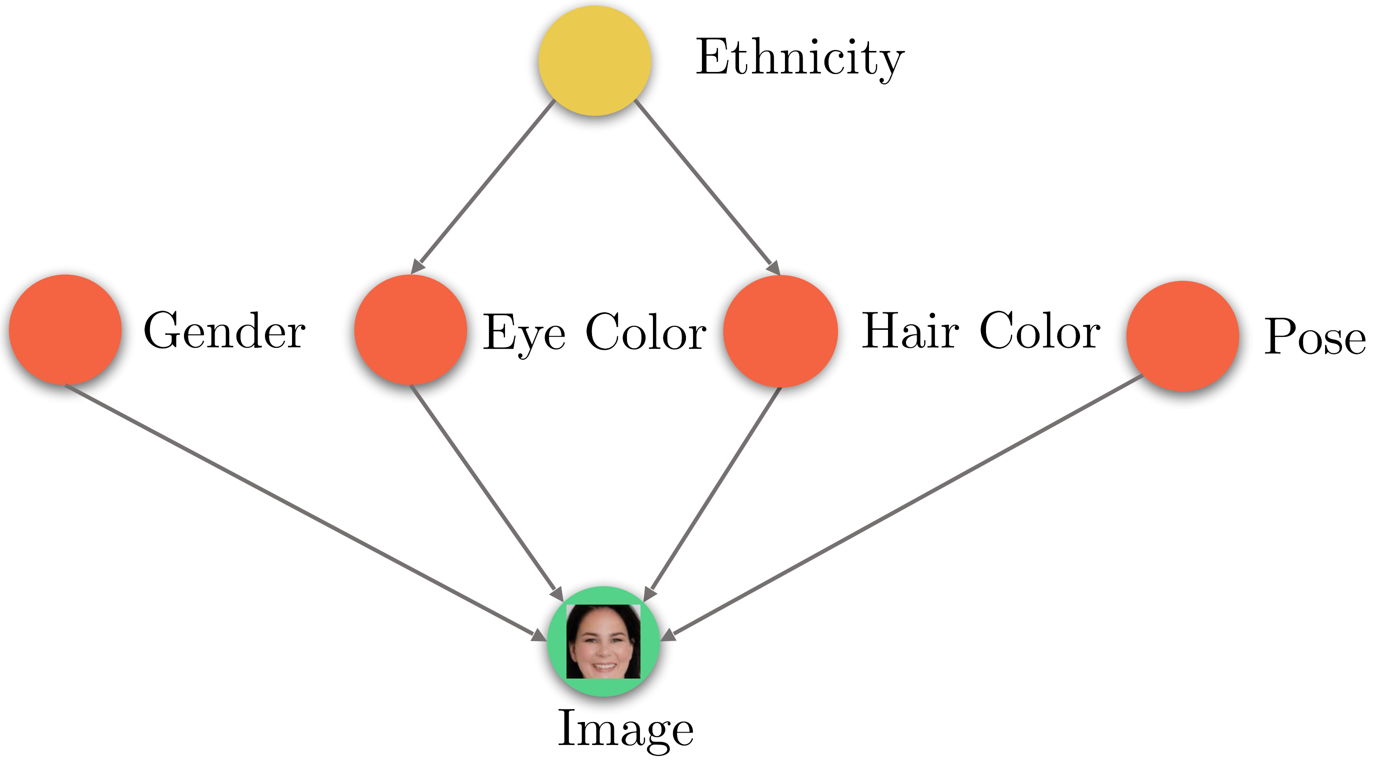
- Lots of variability in images x due to gender, eye color, hair color, pose, etc. However, unless images are annotated, these factors of variation are not explicitly available (**latent**).
- **Idea:** Explicitly model these factors using latent variables z

Latent Variable Models: Motivation

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





Deep Latent Variable Models

- Use neural networks to model the conditionals (deep latent variable models):

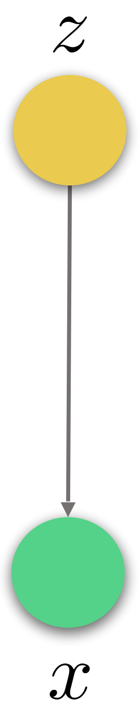
    1. $z \sim \mathcal{N}(0, I)$
    2. $p(x|z) = \mathcal{N}(\mu_\theta(z), \Sigma_\theta(z))$ where $\mu_\theta, \Sigma_\theta$ are the output of a neural network

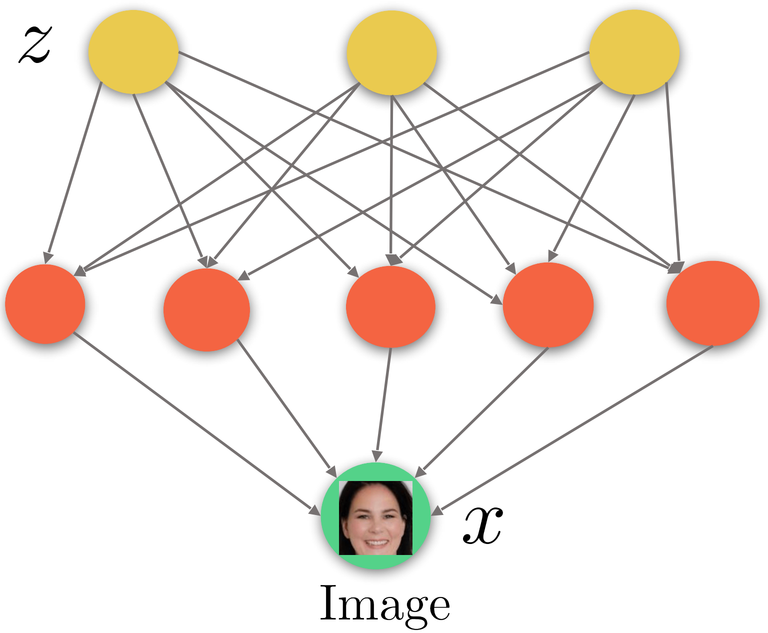
- Hope that after training, z will correspond to meaningful latent factors of variation (features).
  $\longrightarrow$ A type of *Unsupervised representation learning*.

- Features can be computed via $p(z|x)$.

Deep Latent Variable Models

Lecture #10

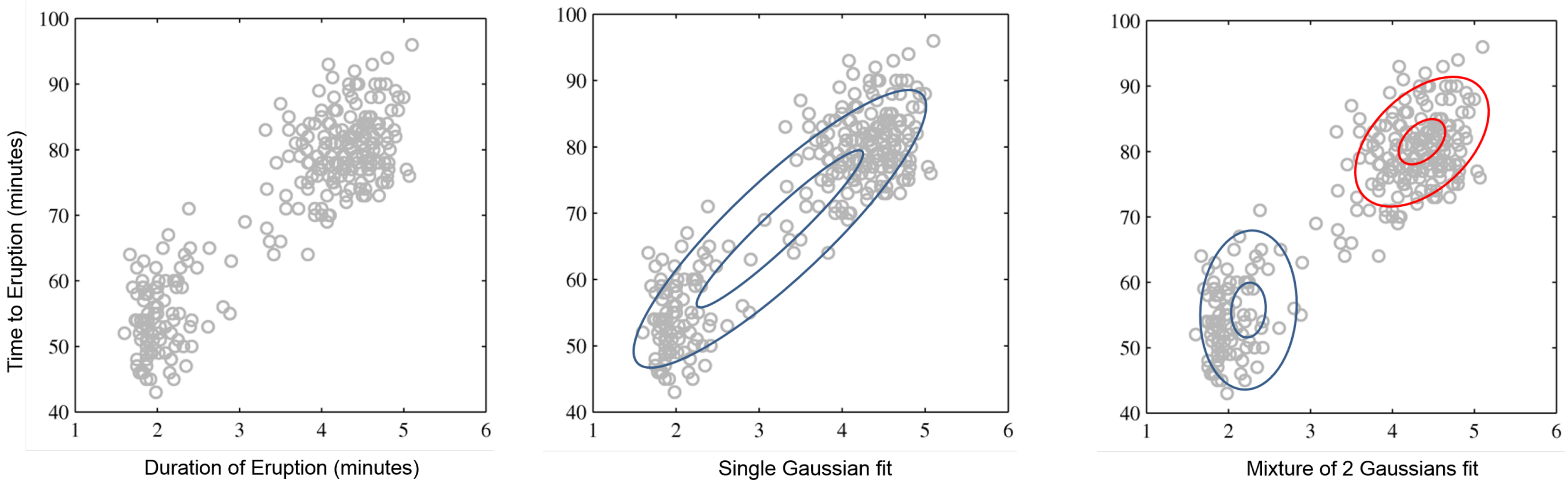




Mixture of Gaussians: A Shallow Latent Variable Model

Mixture of Gaussians:

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

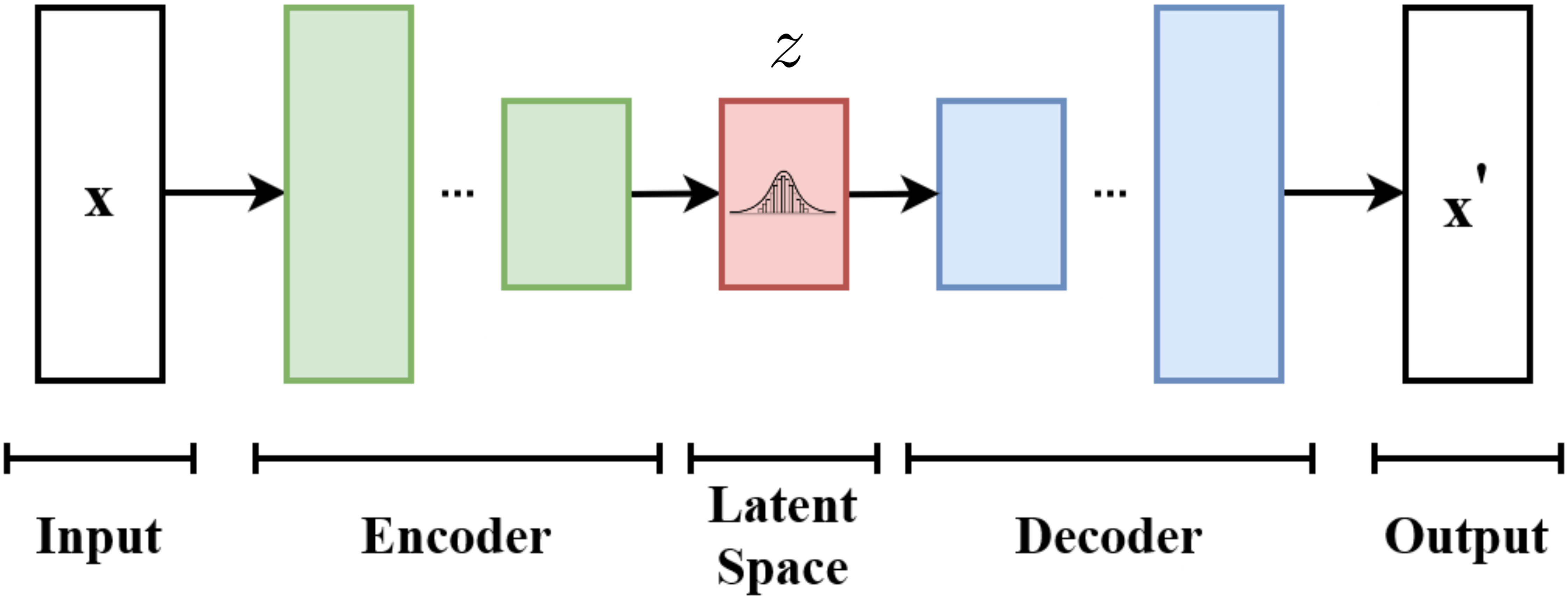


- **Clustering:** Posterior $p(z|x)$ identifies mixture component.
- **Unsupervised learning:** Hope to learn from unlabeled data (ill-posed).

Till now…

- 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)



Variational Autoencoder (VAE)

- $p_\theta(x, z) = p_\theta(x|z)p_\theta(z)$: 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.

Variational Autoencoder (VAE)

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 Guassians then $p_\theta(x)$ is an infinite mixture of Gaussians.
- Consequently, the posterior distribution $p_\theta(z|x)$ is also intractable.
- Our aim is:

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

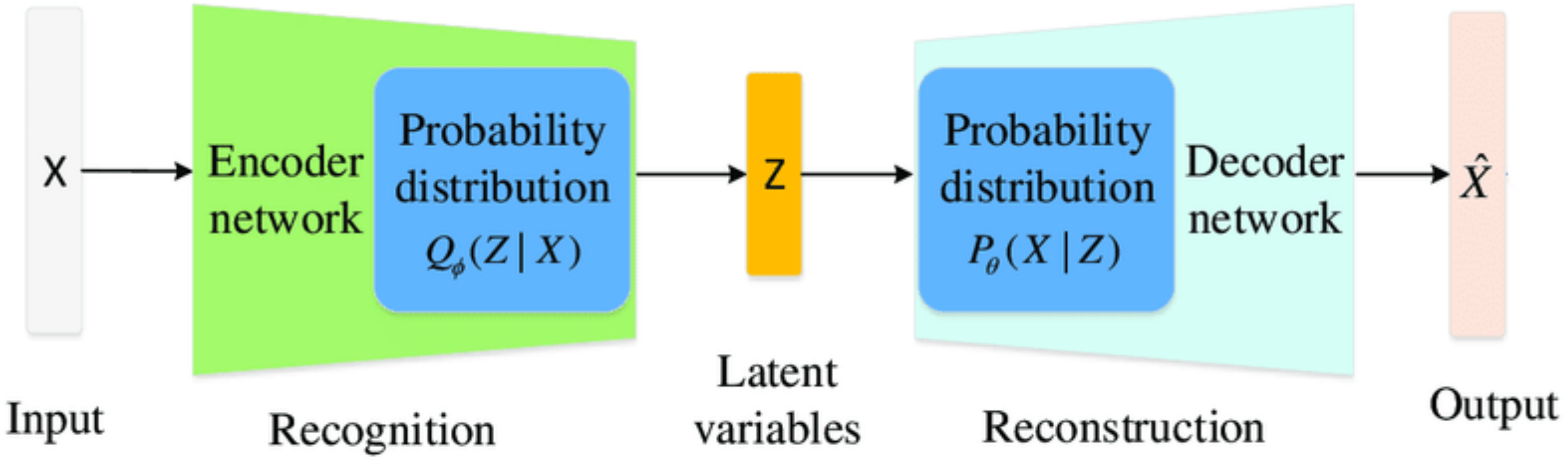
Variational Inference

- Key idea of variatioanl 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



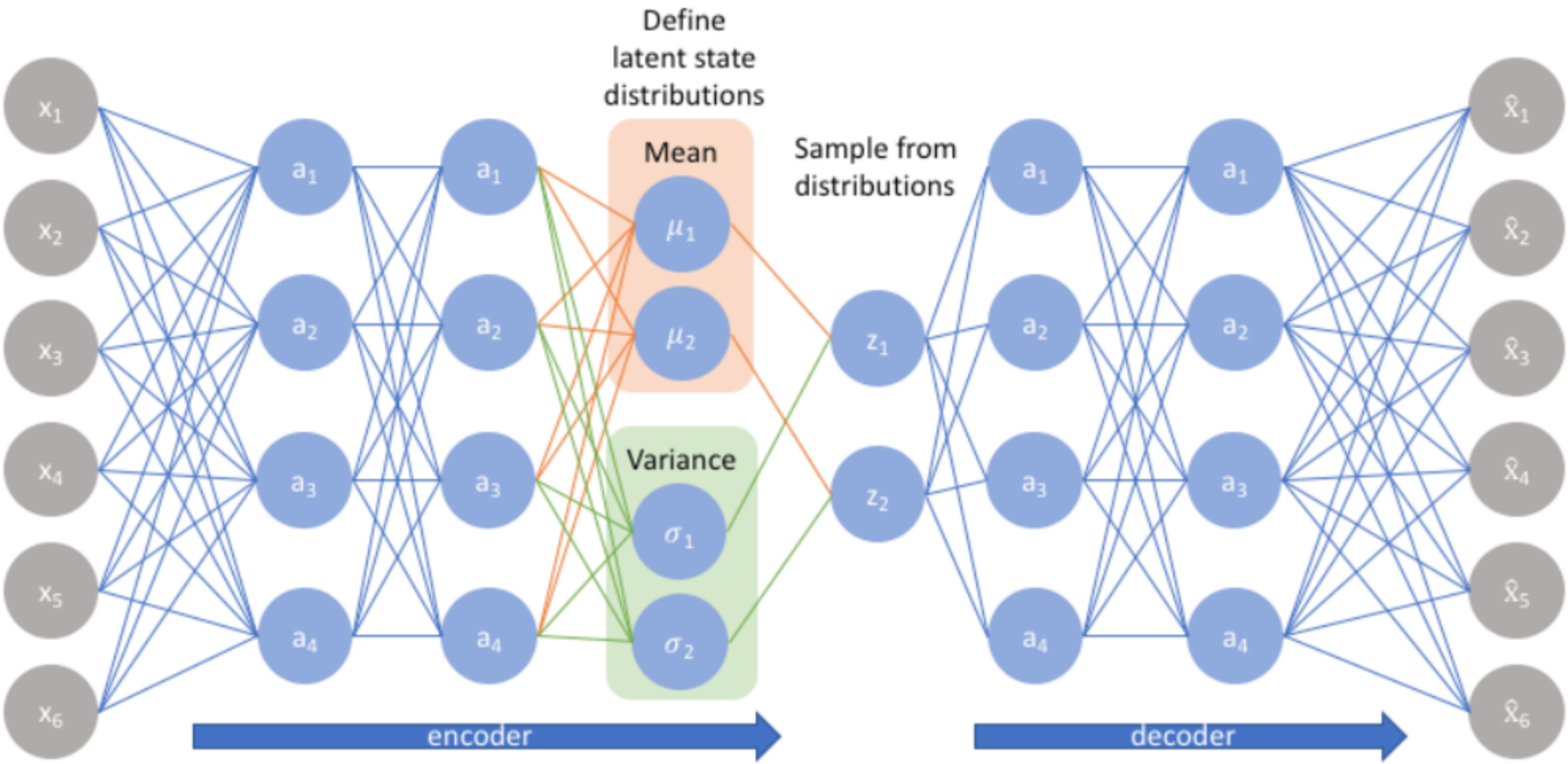
A Concrete VAE

1. Prior distribution is isotropic/spherical Gaussian:
   $p(z) = \mathcal{N}(0, I)$.

2. Stochastic decoder is Gaussian:
   $p_\theta(x|z) = \mathcal{N}\left(\mu_\theta(z), \text{diag}(\sigma_\theta(z))\right)$ where $\mu_\theta, \sigma_\theta$ are neural networks.

3. Stochastic encoder (i.e., inference or recognition model) is Gaussian:
   $q_\phi(z|x) = \mathcal{N}\left(\mu_\phi(x), \text{diag}(\sigma_\phi(x))\right)$ where μ_ϕ, σ_ϕ are also neural networks.

A Concrete VAE



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

How to train a VAE model?

We will emlpoy 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)

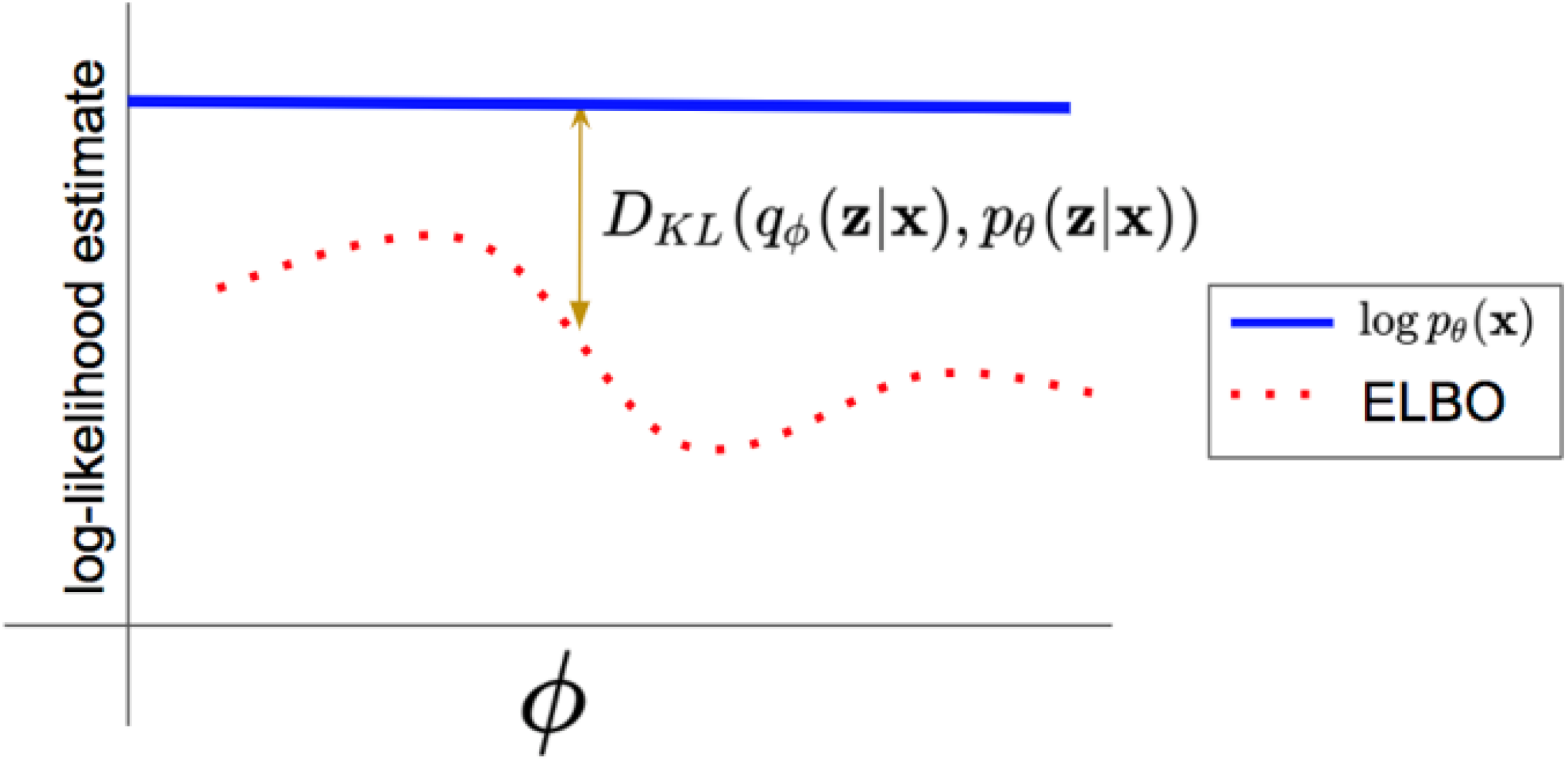
- Evidence or the (marginal) likelihood for a single data x equals to

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

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

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

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 θ and ϕ to maximize the ELBO over a dataset.

ELBO Derivation – To Be Filled

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

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

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

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

ELBO Applied to the Entire Dataset

- 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):

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

- Therefore:

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

Gradient Estimation

- Recall

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

- The gradient with respect to θ (easy):

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

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

Reparametrization Trick

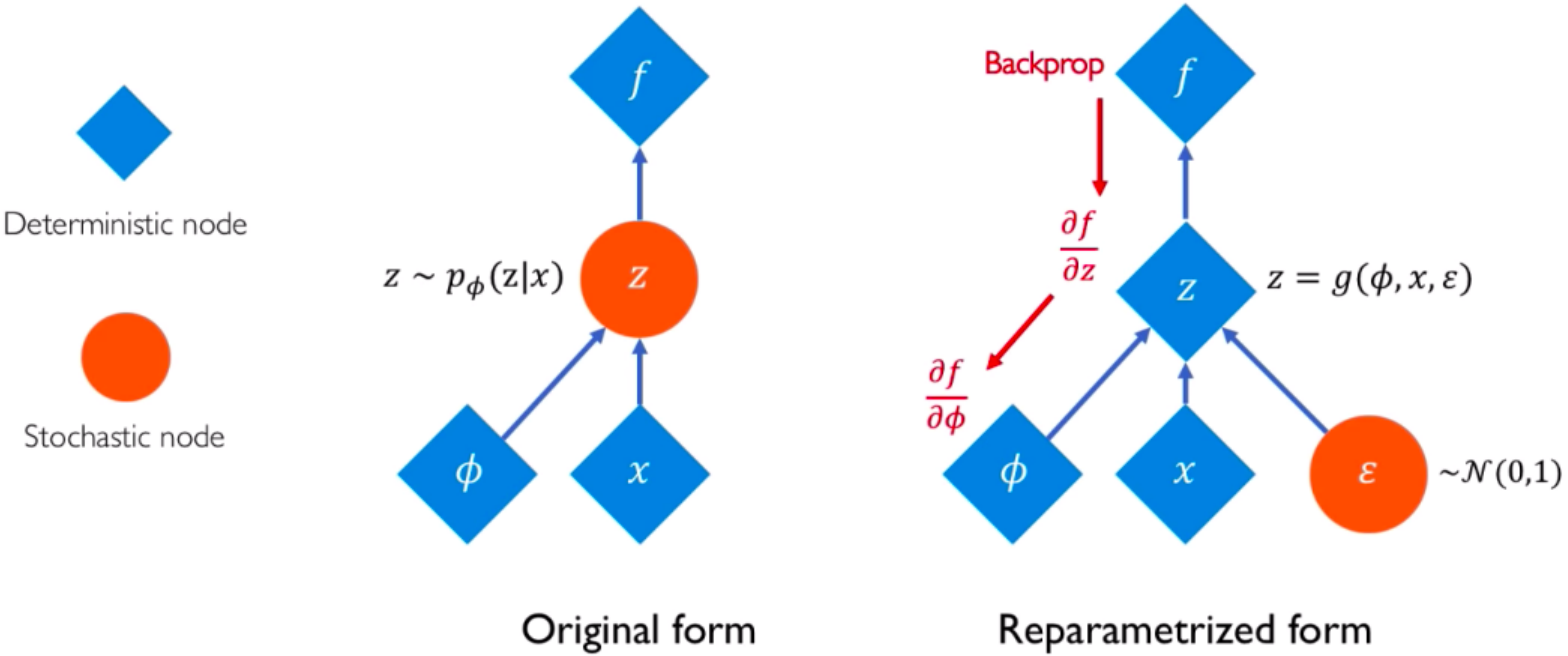
- Want to compute a gradient with respect to ϕ of:

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

- Suppose $q_\phi(z|x) = \mathcal{N}\left(\mu_\phi(x), \operatorname{diag}(\sigma_\phi^2(x))\right)$ 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:

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

Reparametrization Trick



Reparametrization Trick in VAEs

- Thus, the gradient w.r.t. ϕ becomes

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

$\longrightarrow$ Easy to estimate Monte Carlo if f and g are differentiable w.r.t. ϕ and ϵ is easy to sample from.

- In VAEs

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

VAE's Training Algorithm

Data:

$\mathcal{D}$: Dataset

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

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

Result:

$\boldsymbol{\theta}, \boldsymbol{\phi}$: Learned parameters

$(\boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow$ Initialize parameters

while *SGD not converged* **do**

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

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

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

Update $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ using SGD optimizer

end

Summary on Latent Variable Models

- 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.

References

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 introduction 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`

Introduction to Deep Generative Modeling

Lecture #10

HY-673 – Computer Science Dep., University of Crete
Professors: Yannis Pantazis, Yannis Stylianou
Teaching Assistant: Michail Raptakis