Unsupervised Learning

- latent variable models
- variational autoencoders

Yee Whye Teh (Oxford Statistics & DeepMind)
https://www.stats.ox.ac.uk/~teh

https://github.com/OxfordAIML/uniqplus-aiml-2022

Unsupervised Learning via Probabilistic Models

- Unlabelled data $\{x_i\}_{i=1\dots n}$.
- Unsupervised learning from unlabelled data:
  - learning about structure and properties of the domain
  - learning to represent data in useful ways for downstream processing
- Dominant approach to unsupervised learning: probabilistic models
  - Model: Parameterised distribution $p_\theta(x)$ over the data space.
  - Learn by finding θ which maximises likelihood — the probability of the data:

$$\arg\max_\theta \prod_{i=1}^{n} p_\theta(x_i) = \arg\max_\theta \sum_{i=1}^{n} \log p_\theta(x_i)$$

  - In above, we assumed our data is identically and independently distributed according to the model $p_\theta(x)$.

Probabilistic Models

- Probabilistic models underpins much of statistics and probabilistic machine learning.
- In unsupervised learning, many probabilistic models are so-called **latent variable models**: they model observed data x using a joint probability distribution over x and latent variables z:

$$p_\theta(x) = \int p_\theta(x, z) dz$$

$$\sum_{i=1}^{n} \log p_\theta(x_i) = \sum_{i=1}^{n} \log \int p_\theta(x_i, z_i) dz_i$$

  - Latent variables make the model more expressive.
  - They also allow to capture useful properties of observed data.

Learning in Latent Variable Models

$$p_\theta(x) = \int p_\theta(x, z)dz$$

- Big problem: the marginal distribution $p_\theta(x)$ (sometimes called the evidence) is intractable to compute or to optimize.
- Variational learning: introduce a variational posterior $q_\phi(z|x)$.

$$\begin{aligned}\log p_\theta(x) &= \log \int p_\theta(x, z)dz \\ &= \log \int p_\theta(x|z)\frac{p_\theta(z)}{q_\phi(z|x)}q_\phi(z|x)dz \\ &\geq \int \log \left(p_\theta(x|z)\frac{p_\theta(z)}{q_\phi(z|x)} \right) q_\phi(z|x)dz\end{aligned}$$

- called the **evidence lower-bound (ELBO)**.

Learning in Latent Variable Models

$$\log p_\theta(x) \geq \int \Big(\log p_\theta(x\,|\,z) + \log p_\theta(z) - \log q_\phi(z\,|\,x) \Big)\, q_\phi(z\,|\,x) dz$$

- Now we have both θ and ϕ to optimise over, but the ELBO is fortunately a tractable objective function.
- Generalized Expectation-Maximization algorithm:
  - Alternate between optimising ϕ and optimising θ.
  - Optimising θ

$$\frac{d\text{ELBO}}{d\theta} = \int \big(\nabla_\theta \log p_\theta(x\,|\,z) + \nabla_\theta \log p_\theta(z) \big)\, q_\phi(z\,|\,x) dz$$

  - If we can draw samples $z \sim q_\phi(z\,|\,x)$, we can get unbiased gradient for θ.
  - Optimising ϕ is trickier: typically we can't take gradient through random variate generation $z \sim q_\phi(z\,|\,x)$.

Reparameterisation Trick

$$\log p_\theta(x) \geq \int \left(\log p_\theta(x \mid z) + \log p_\theta(z) - \log q_\phi(z \mid x) \right) q_\phi(z \mid x) dz$$

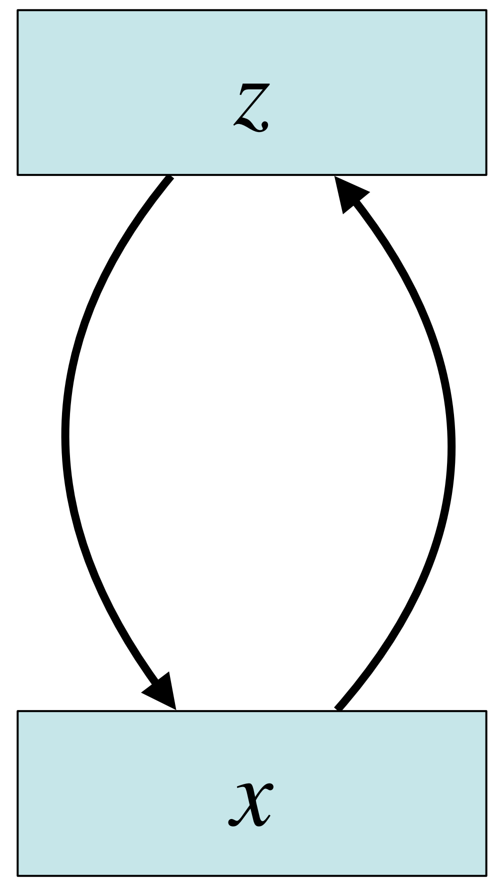
- Reparameterisation trick: suppose that $q_\phi(z \mid x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x))$ we can write $z_\phi(x) = \mu_\phi(x) + \sigma_\phi(x)\eta$ where $\eta \sim \mathcal{N}(0, I_d)$.

$$\log p_\theta(x) \geq \int \left(\log p_\theta(x \mid z_\phi(x)) + \log p_\theta(z_\phi(x)) - \log q_\phi(z_\phi(x) \mid x) \right) \mathcal{N}(\eta; 0, I_d) d\eta$$

- Now we can compute derivative wrt ϕ!

Variational Autoencoder

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



$$q_\phi(z \mid x) = \mathcal{N}(z; \mu_\phi(x), \sigma_\phi^2(x))$$

encoder

$$p_\theta(x \mid z) = \mathcal{N}(x; \mu_\theta(z), \sigma_\theta^2(z))$$

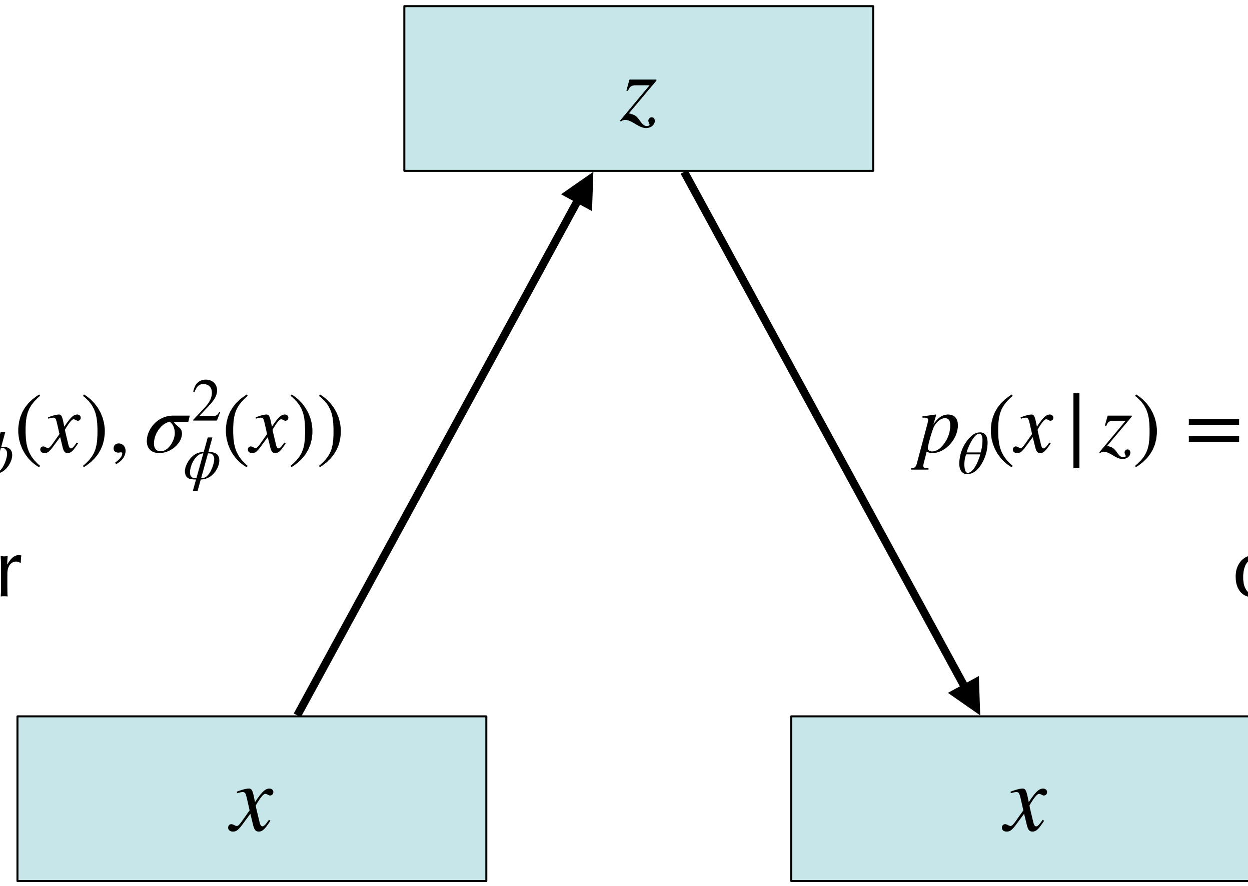
decoder

$$\log p_\theta(x) \geq \int \Big(\log p_\theta(x \mid z_\phi(x)) + \log p_\theta(z_\phi(x)) - \log q_\phi(z_\phi(x) \mid x) \Big) \mathcal{N}(\eta; 0, I_d) d\eta$$

Variational Autoencoder

Check out example colab: https://github.com/smartgeometry-ucl/dl4g/blob/master/variational_autoencoder.ipynb

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



$$q_\phi(z \mid x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x))$$

encoder

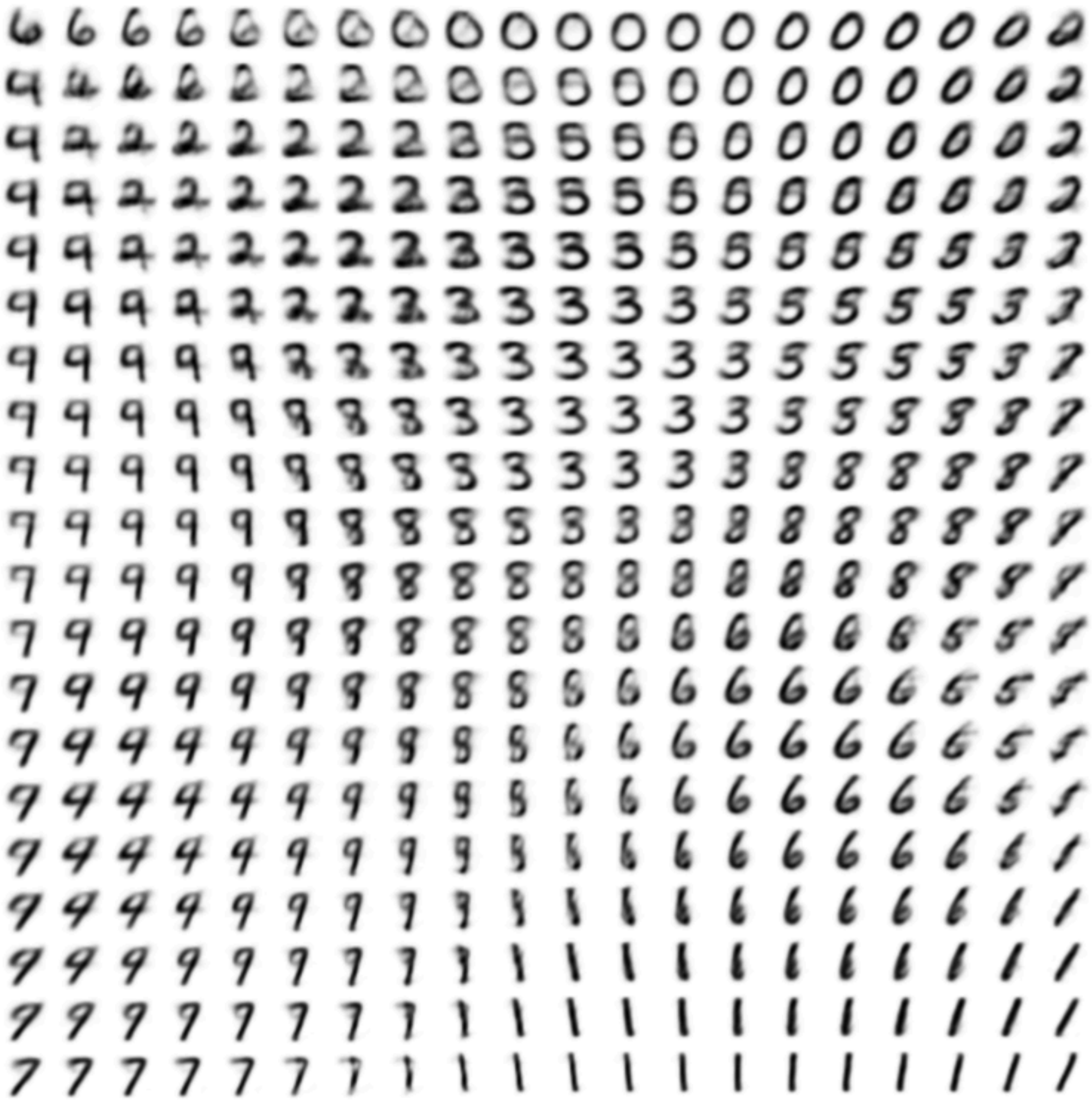
$$p_\theta(x \mid z) = \mathcal{N}(x; \mu_\theta(z), \sigma^2_\theta(z))$$

decoder

$$\log p_\theta(x) \geq \int \Big(\log p_\theta(x \mid z_\phi(x)) + \log p_\theta(z_\phi(x)) - \log q_\phi(z_\phi(x) \mid x) \Big) \mathcal{N}(\eta; 0, I_d) d\eta$$

$$= \int \Big(\log p_\theta(x \mid z_\phi(x)) \Big) \mathcal{N}(\eta; 0, I_d) d\eta - \mathrm{KL}(q_\phi(z \mid x) \| p(z))$$

Variational Autoencoder



Alternatives to Latent Variable Models

- Autoregressive models:
  - If x is multidimensional, say $x = [x[1], x[2], \ldots, x[d]]$, write the probability distribution in an "autoregressive" way:

$$\log p_\theta(x) = \sum_{j=1}^{d} \log p_\theta(x[j] \,|\, x[1 \ldots j-1])$$

  - Language models parameterise each of these conditional distributions using transformers.

Alternatives to Latent Variable Models

- Normalising flows:
  - Use a change of variables formula. Suppose $x = f_\theta(z)$ where f_θ is an **invertible and differentiable** function. Then:

$$p_\theta(x) = p\left(f_\theta^{-1}(x)\right) \left| \det \frac{df_\theta^{-1}(x)}{dx} \right|$$

  - We parameterise $f_\theta(z)$ using neural networks in a smart way making sure they are flexible and invertible.
  - We can construct a flexible class of functions as a composition of simpler invertible functions.

$$f_\theta(z) = f_\theta^{(l)} \circ f_\theta^{(l-1)} \circ \cdots \circ f_\theta^{(1)}(z)$$

Alternatives to Latent Variable Models

- Diffusion Models:

Forward SDE (data → noise)

$$\mathbf{x}(0) \longrightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x},t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \longrightarrow \mathbf{x}(T)$$

score function

$$\mathbf{x}(0) \longleftarrow \mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x},t) - g^2(t)\boxed{\nabla_{\mathbf{x}}\log p_t(\mathbf{x})}\right]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \longleftarrow \mathbf{x}(T)$$

Reverse SDE (noise → data)