

ENM53 I: Data-driven modeling and probabilistic scientific computing

Lecture #25: Auto-encoders

Paris Perdikaris
April 27, 2021



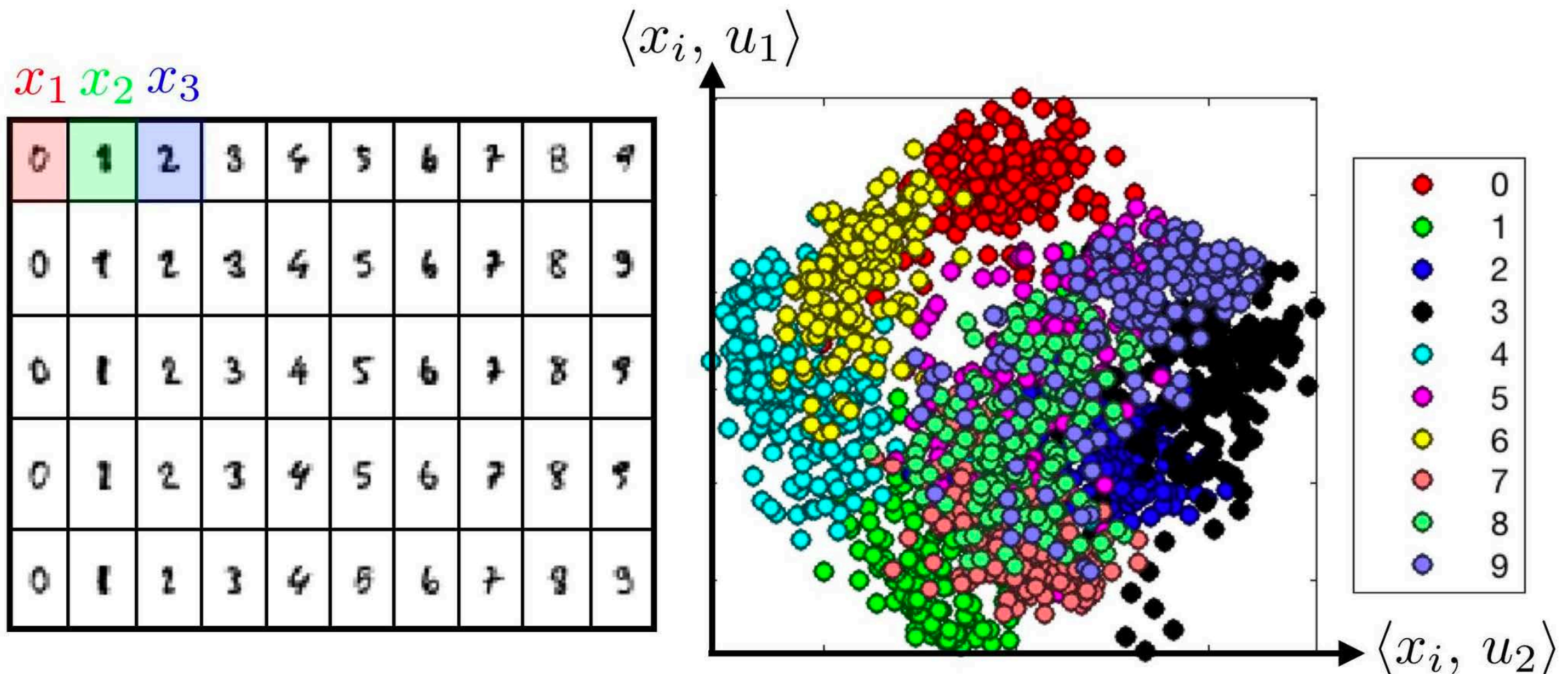
Principal component analysis

Input data: $X = (x_i)_{i=1}^n \in \mathbb{R}^{n \times p}, x_i \in \mathbb{R}^p$

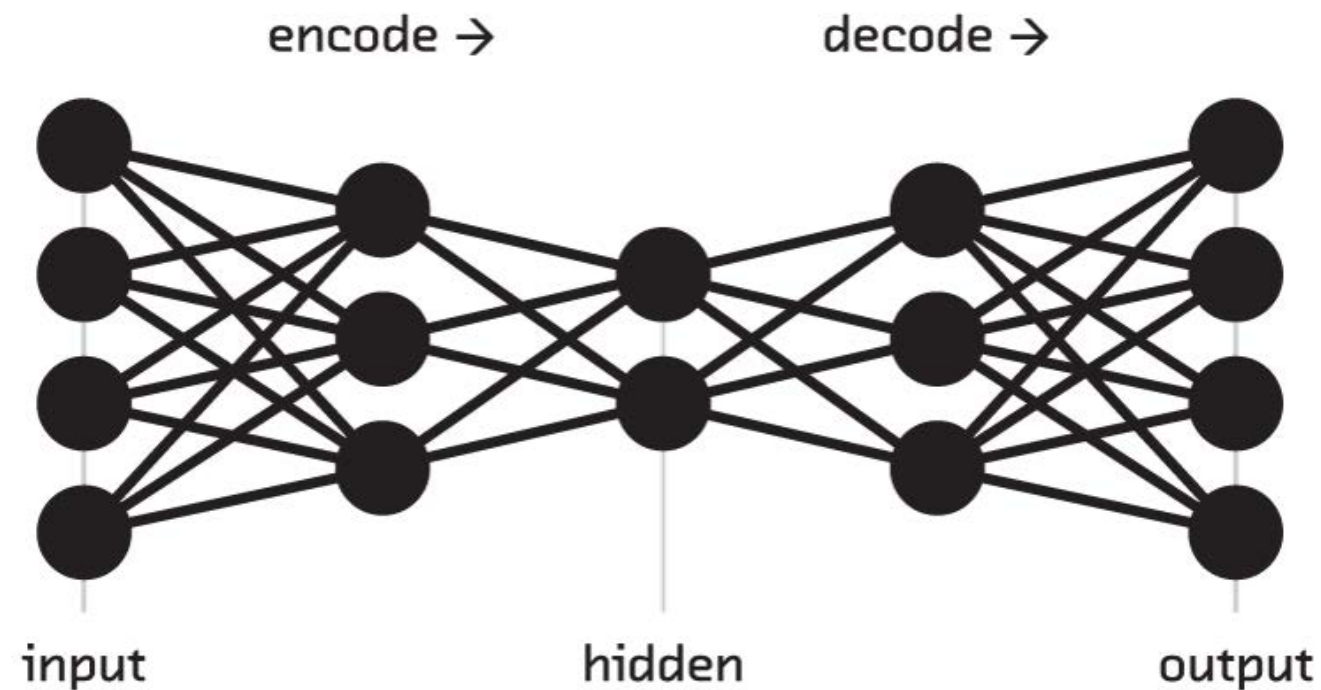
Remove mean: $x_i \leftarrow x_i - \frac{1}{n} \sum_j x_j$

Covariance: $C \stackrel{\text{def.}}{=} \frac{1}{n} X^\top X \in \mathbb{R}^{p \times p}$

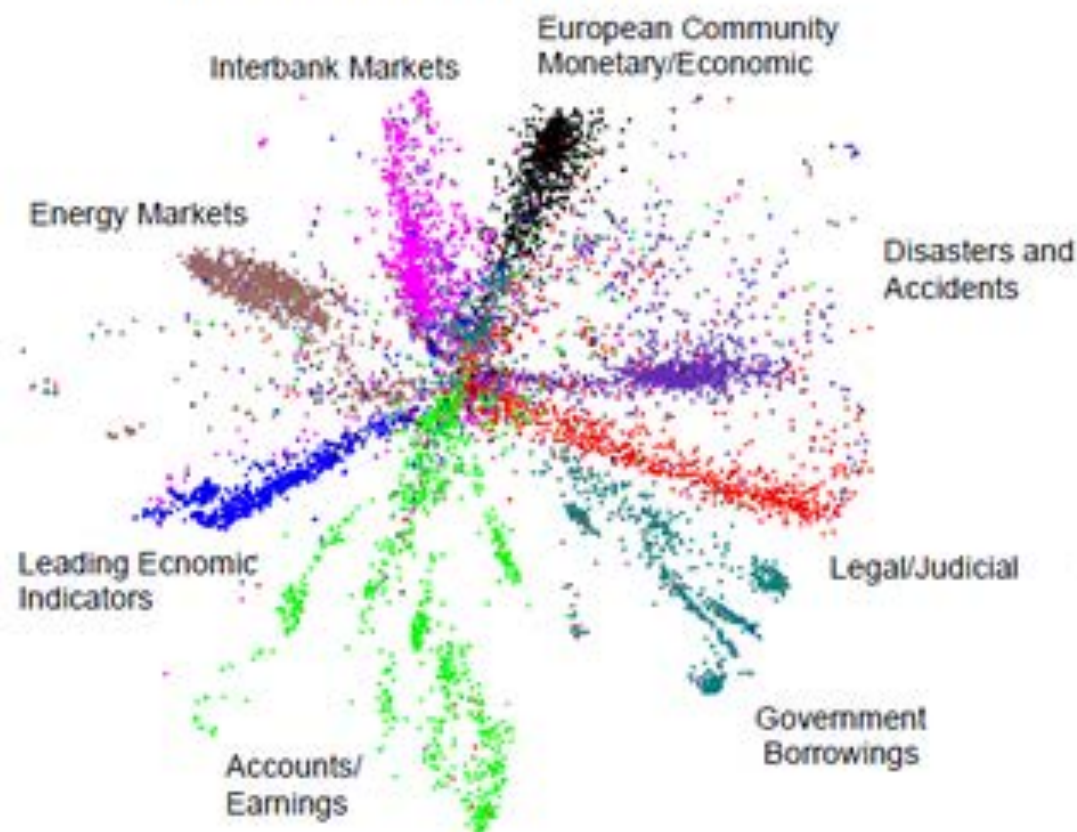
Eigen-decomposition: $C = U \text{diag}(\sigma_k^2) U^\top, U = (u_k)_{k=1}^p$



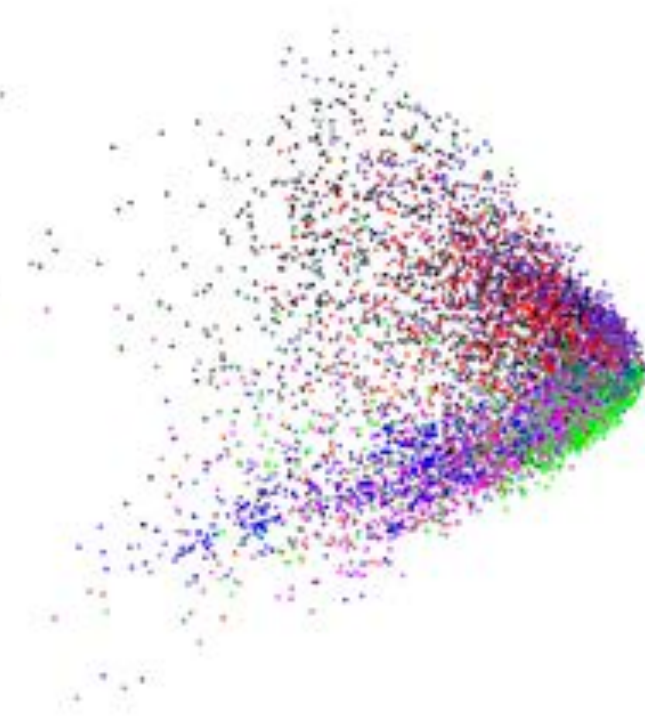
Neural network auto-encoders



Autoencoder 2-D Topic Space



LSA 2-D Topic Space



Tricks of the trade

- Variational bounds
- Density re-parametrizations
- Density ratio estimation
- Variational optimization/evolution strategies
- Adversarial games

Variational bounds

Typical problem:

My loss function $f(\theta)$ is intractable to compute, typically because it involves intractable marginalization. I can't evaluate it let alone minimize it.

Solution:

Let's construct a family of - typically differentiable - upper-bounds:

$$f(\theta) \leq \inf_{\psi} g(\theta, \psi),$$

and solve the optimization problem

$$\theta^*, \psi^* \leftarrow \operatorname{argmin}_{\theta, \psi} g(\theta, \psi)$$

instead. Technically, once optimization is finished, you can discard the auxiliary parameter ψ^* - although often turns out to be meaningful and useful in itself, often for approximate inference such as the recognition model of VAEs.

Tricks of the trade:

Jensen's inequality: The mean value of a convex function is never lower than the value of the convex function applied to the mean. Generally appears in some variant of the standard evidence lower bound (ELBO) derivation below:

$$\begin{aligned} -\log p(x) &= -\log \int p(x, y) dy \\ &= -\log \int q(y|x) \frac{p(y, x)}{q(y|x)} dy \\ &\leq -\int q(y|x) \log \frac{p(y, x)}{q(y|x)} dy \end{aligned}$$

The re-parametrization trick

One oft-encountered problem is computing the gradient of an expectation of a smooth function f :

$$\nabla_{\theta} \mathbb{E}_{p(z;\theta)}[f(z)] = \nabla_{\theta} \int p(z; \theta) f(z) dz$$

This is a recurring task in machine learning, needed for posterior computation in **variational inference**, value function and policy learning in **reinforcement learning**, derivative pricing in **computational finance**, and **inventory control** in operations research, amongst many others. This gradient is often difficult to compute because the integral is typically unknown and the parameters θ , with respect to which we are computing the gradient, are of the distribution $p(z; \theta)$. But where a random variable z appears we can try our random variable reparameterisation trick, which in this case allows us to compute the gradient in a more amenable way:

$$\nabla_{\theta} \mathbb{E}_{p(z;\theta)}[f(z)] = \mathbb{E}_{p(\epsilon)}[\nabla_{\theta} f(g(\epsilon, \theta))]$$

The re-parametrization trick

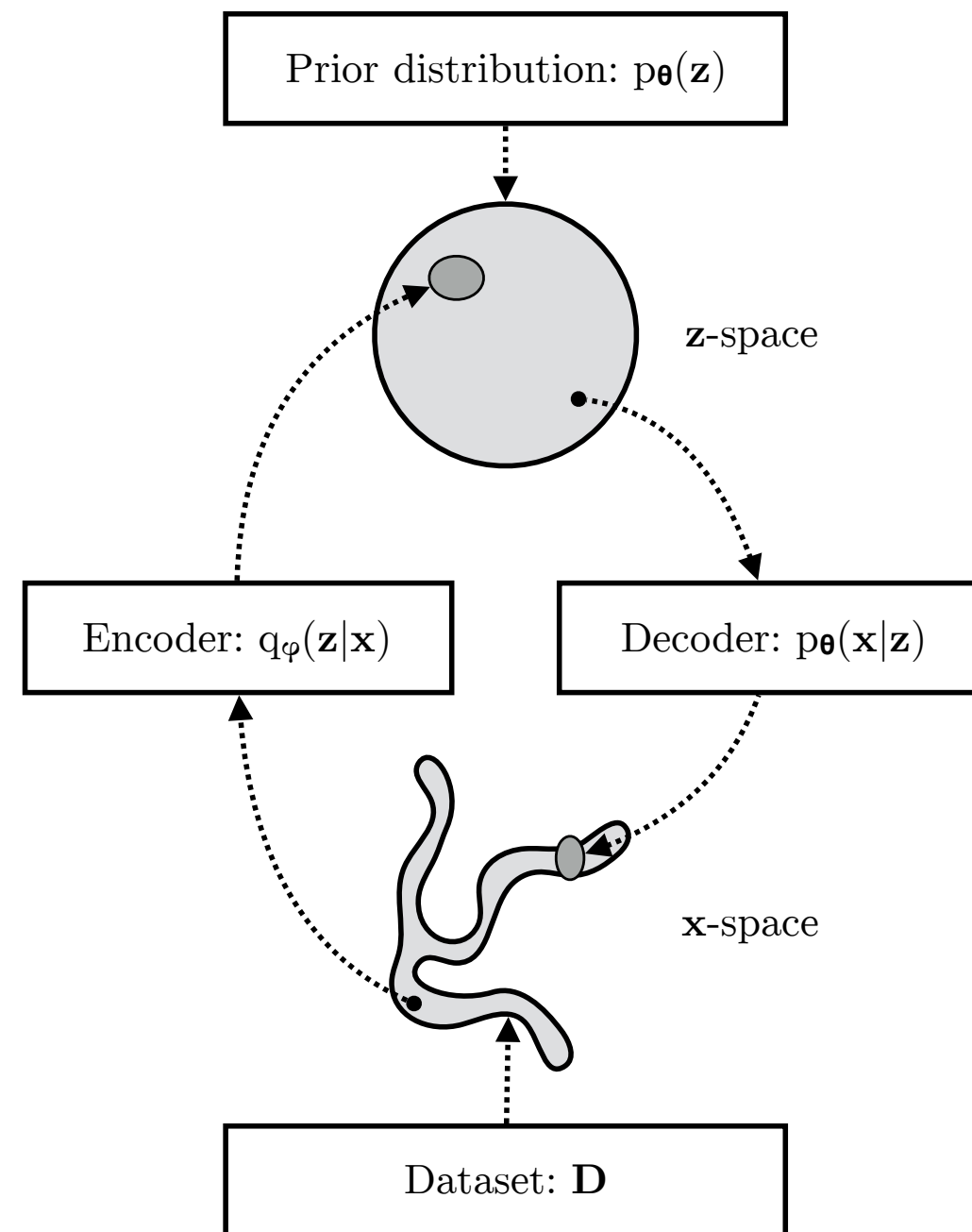
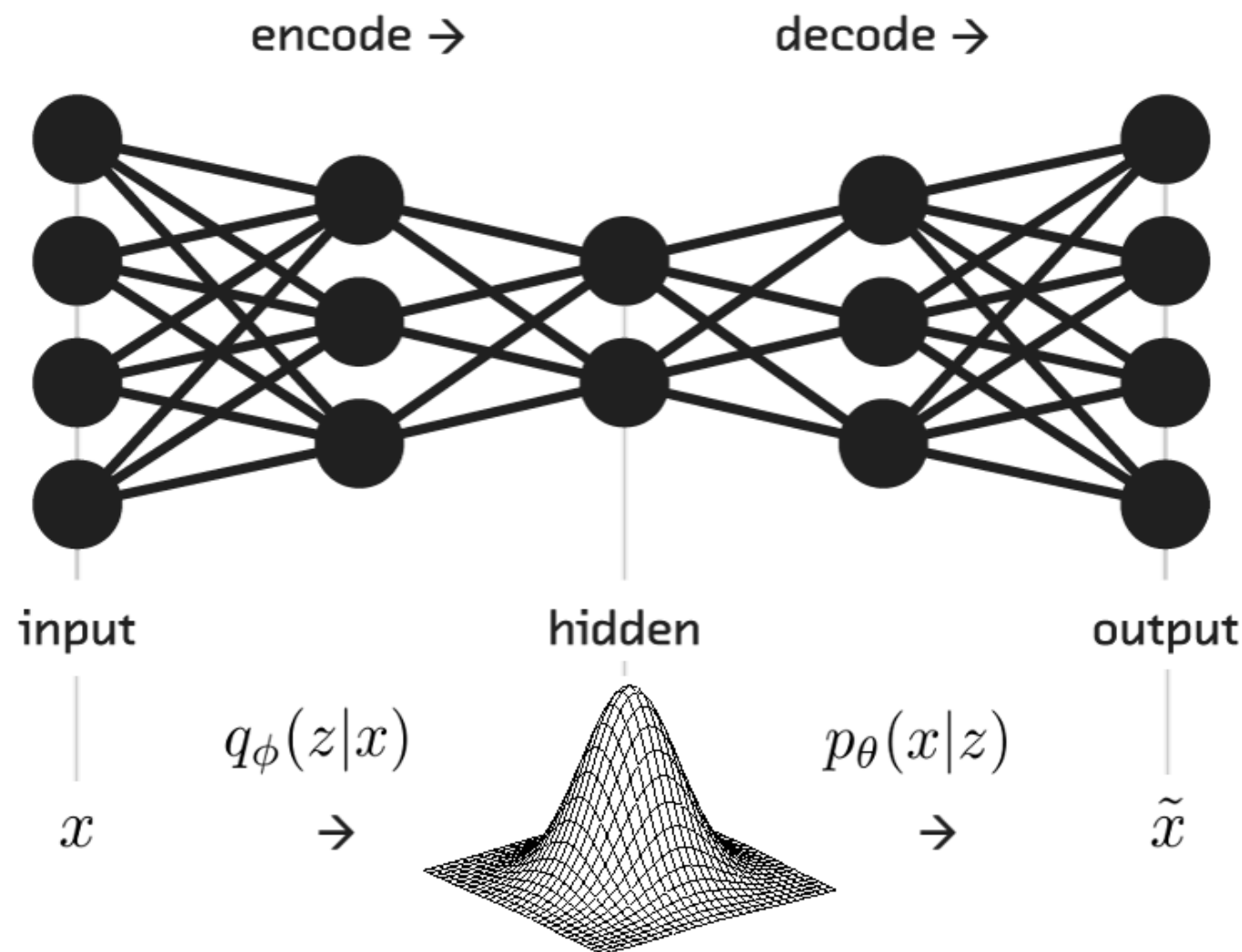
Let's derive this expression and explore the implications of it for our optimisation problem. One-liners give us a transformation from a distribution $p(\epsilon)$ to another $p(z)$, thus the differential area (mass of the distribution) is invariant under the **change of variables**. This property implies that:

$$p(z) = \left| \frac{d\epsilon}{dz} \right| p(\epsilon) \implies |p(z)dz| = |p(\epsilon)d\epsilon|$$

Re-expressing the troublesome stochastic optimisation problem using random variate reparameterisation, we find:

$$\begin{aligned} \nabla_{\theta} \mathbb{E}_{p(z;\theta)}[f(z)] &= \nabla_{\theta} \int p(z; \theta) f(z) dz \\ &= \nabla_{\theta} \int p(\epsilon) f(z) d\epsilon = \nabla_{\theta} \int p(\epsilon) f(g(\epsilon, \theta)) d\epsilon \\ &= \nabla_{\theta} \mathbb{E}_{p(\epsilon)}[f(g(\epsilon, \theta))] = \mathbb{E}_{p(\epsilon)}[\nabla_{\theta} f(g(\epsilon, \theta))] \end{aligned}$$

Variational auto-encoders

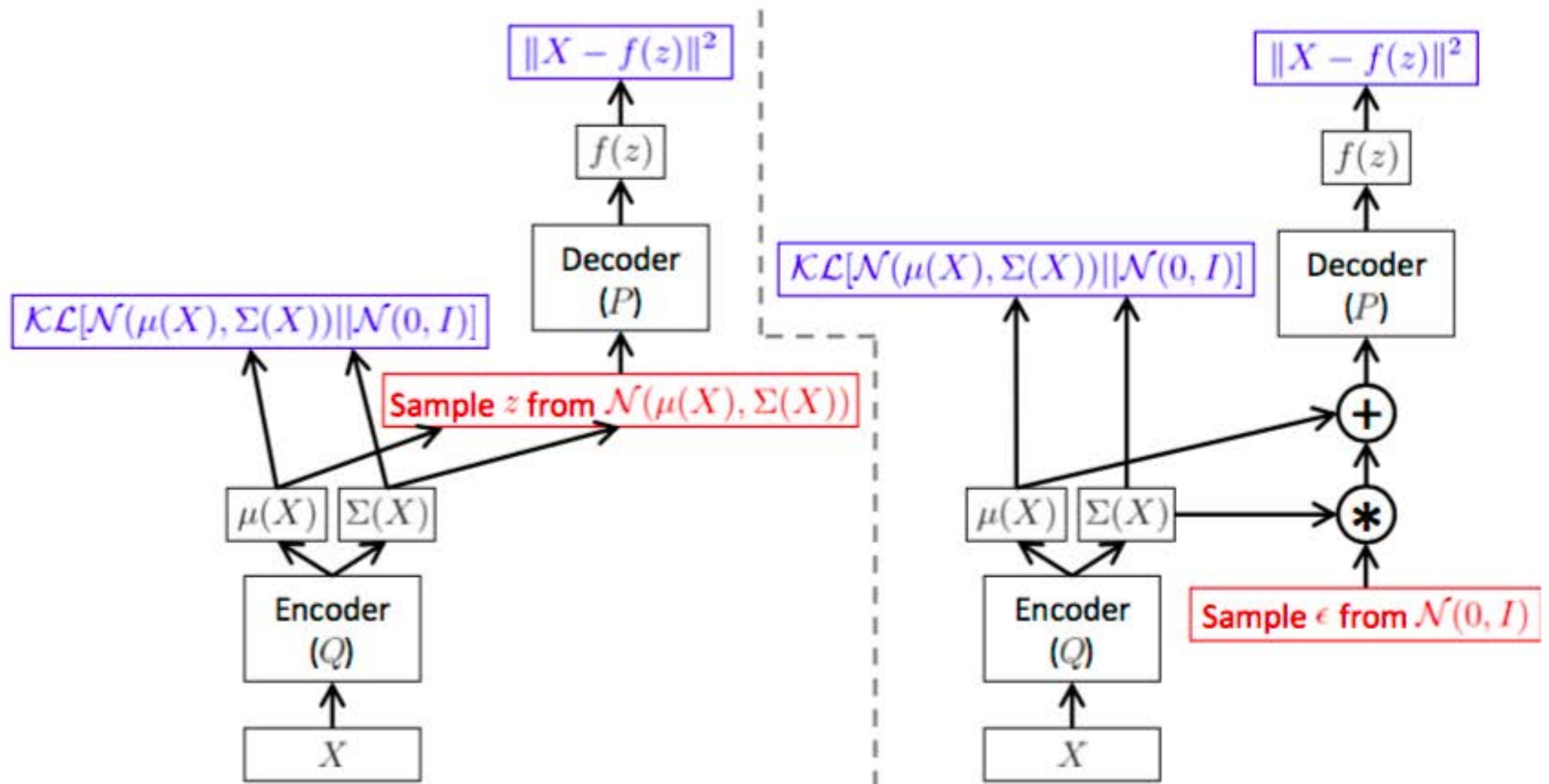


A VAE learns stochastic mappings between the observed \mathbf{x} -space, whose empirical distribution $q_D(\mathbf{x})$ is typically complicated, and a latent \mathbf{z} -space, whose distribution can be relatively simple (such as spherical, as in this figure). The generative model learns a joint distribution $p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})$, factorized into a prior distribution over latent space, $p_\theta(\mathbf{z})$, and a stochastic decoder $p_\theta(\mathbf{x}|\mathbf{z})$. The stochastic encoder $q_\phi(\mathbf{z}|\mathbf{x})$, also called *inference model*, approximates the true but intractable posterior $p_\theta(\mathbf{z}|\mathbf{x})$ of the generative model.

Variational auto-encoders

Before re-parametrization

After re-parametrization



Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Doersch, C. (2016). Tutorial on variational autoencoders. arXiv preprint arXiv:1606.05908.

Generative models

BOLD

A A A A A A A A
A A A A A A A

A A A A A A A A
A A A A A A A

ITALIC

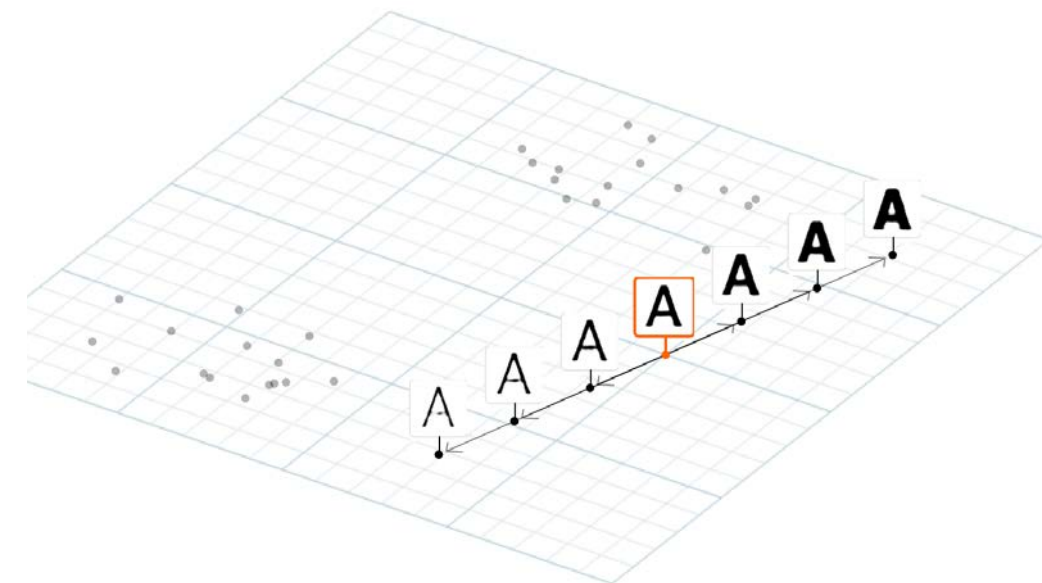
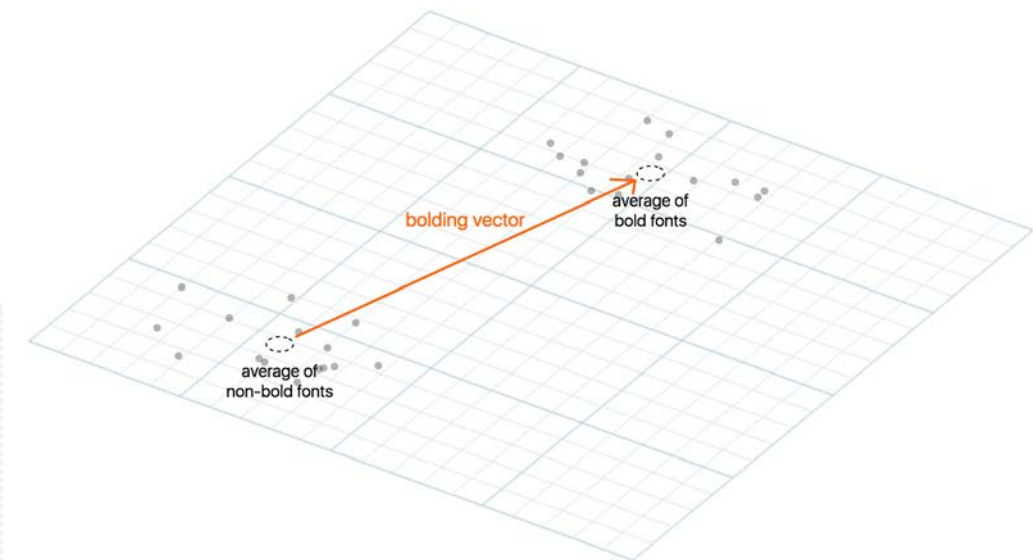
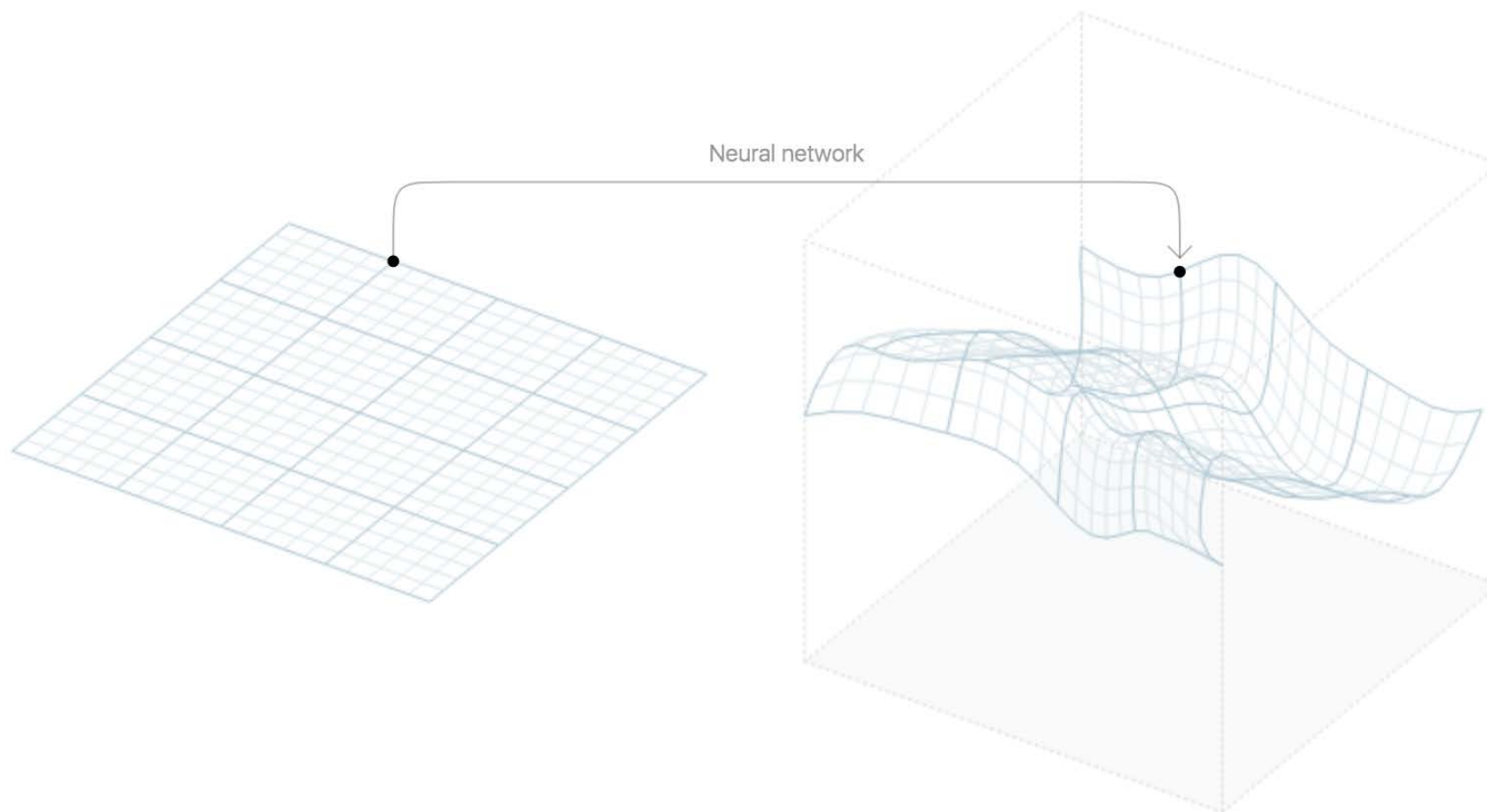
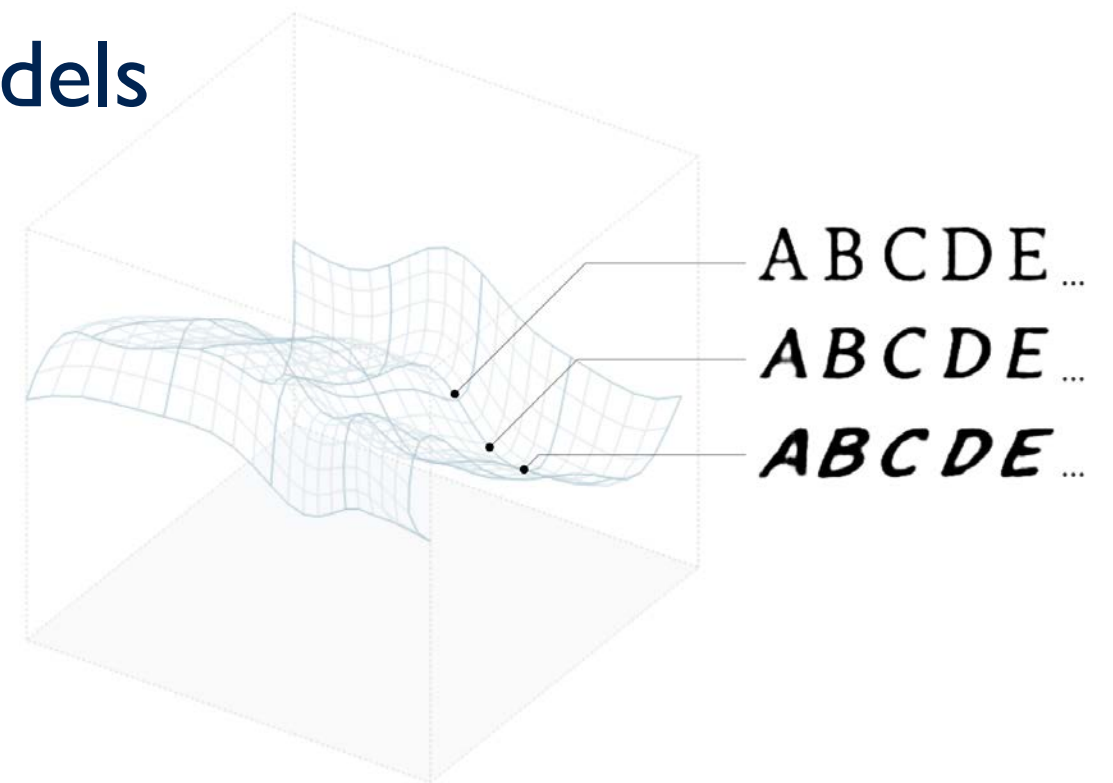
A A A A A A A A
A A A A A A A

A A A A A A A A
A A A A A A A

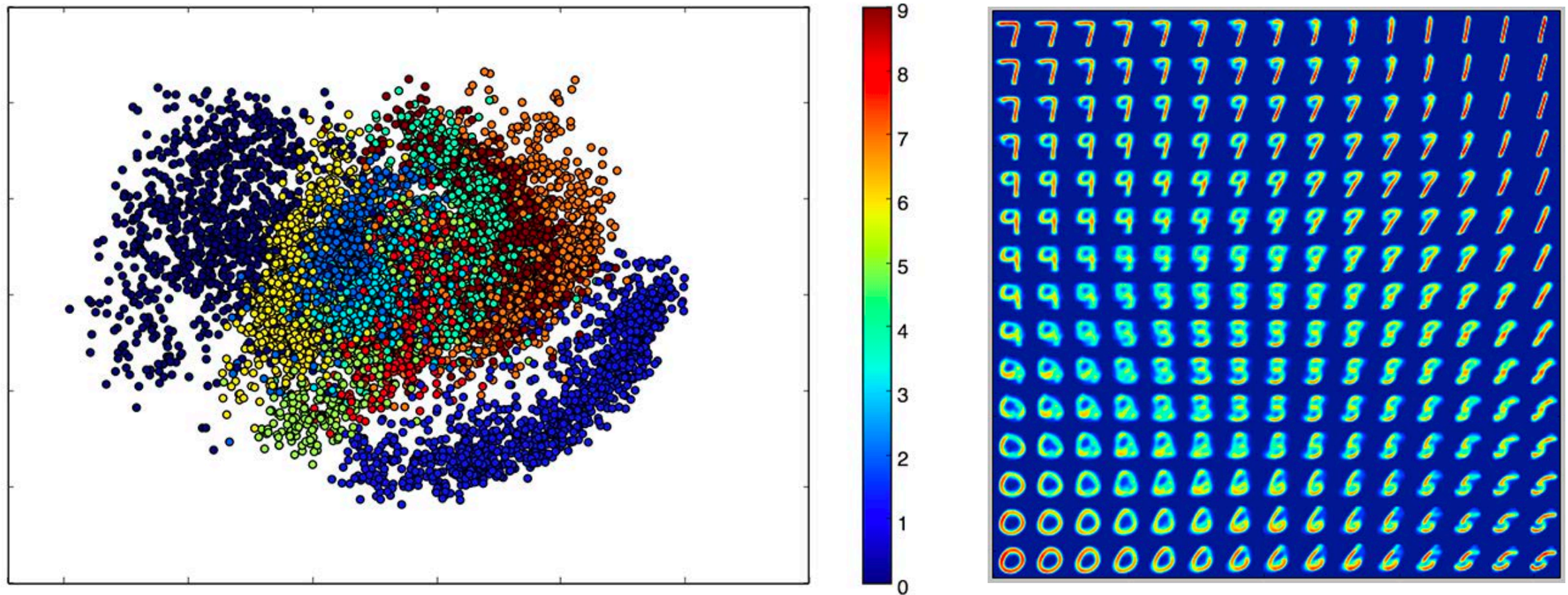
CONDENSED

A A A A A A A A
A A A A A A A

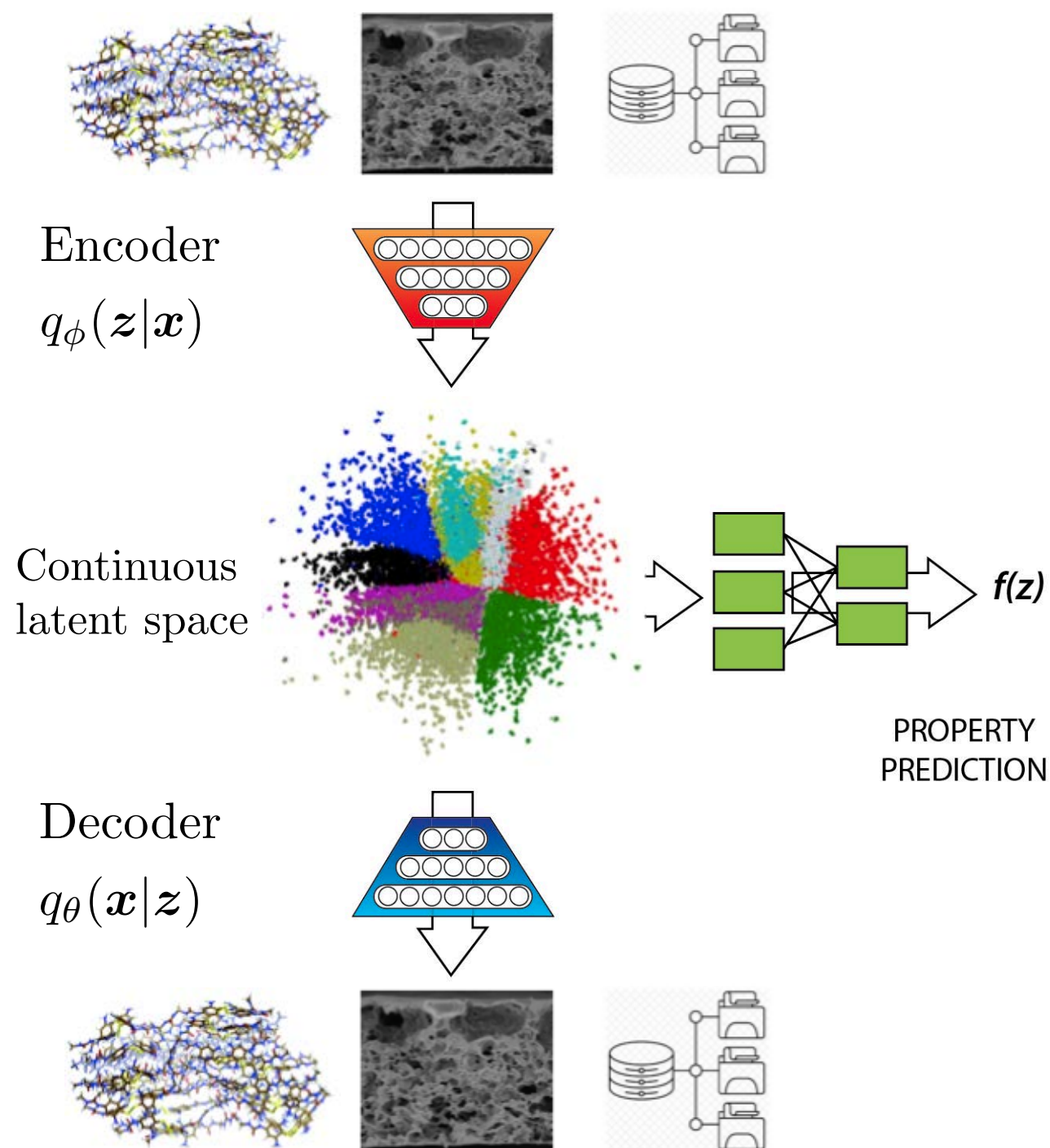
A A A A A A A A
A A A A A A A



The latent space of hand-written digits



Generative models



Bayesian optimization

