Structured Model Learning
Variational Autoencoders

Boris Flach
Czech Technical University in Prague

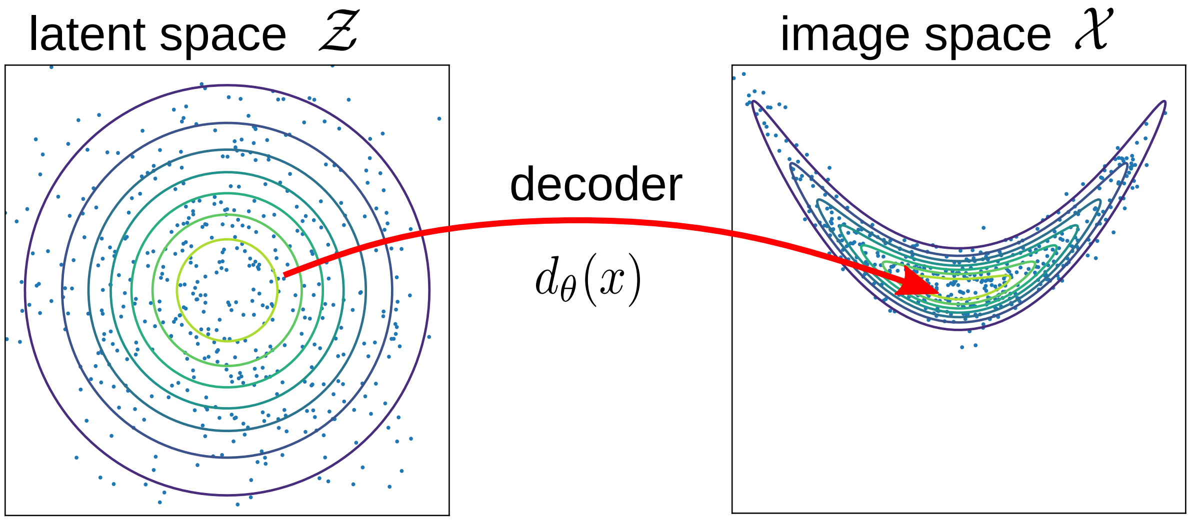
- Variational autoencoders (VAE)
- VAE approximation errors
- Hierarchical VAEs

Generative models

Generative models: Given training data $\mathcal{T} = \{x_j \mid j = 1, \ldots, \ell\}$ drawn i.i.d. from an unknown distribution $p_d(x)$, the goal is to learn a DNN model that allows to generate random instances of x similar to $x \sim p_d(x)$.

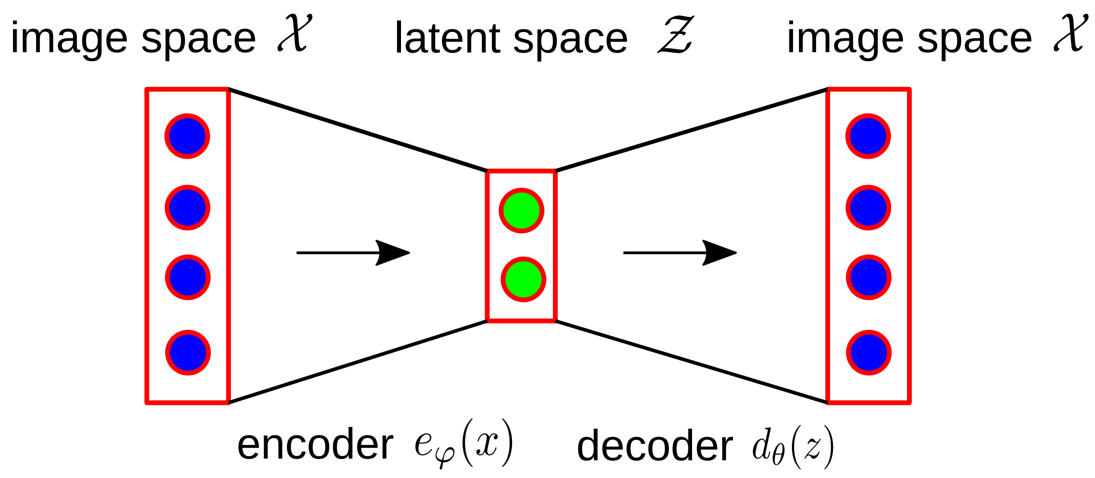
Approach this task by using *latent variable models:*

- fix a latent noise space $\mathcal{Z}$ and a distribution $p(z)$ on it,
- design a neural network d_θ that maps $\mathcal{Z}$ to the feature space $\mathcal{X}$,
- learn its parameters θ so that the resulting distribution $p_\theta(x)$ "reproduces" the data distribution.



Generative models

Classical autoencoder networks



e.g. with learning criterion $\mathbb{E}_{\mathcal{T}}\|x - d_\theta \circ e_\varphi(x)\|^2$. However,

- the distribution in the latent space is beyond our control,
- the model can not be used for sampling/generating x instances.

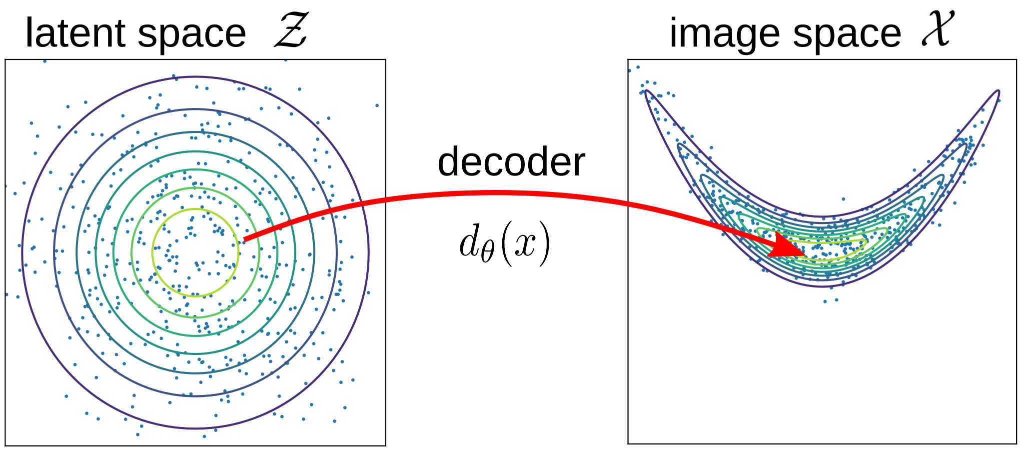
(Gaussian) Variational Autoencoders

- latent space $\mathcal{Z} = \mathbb{R}^m$, prior distribution $p(z)\colon \mathcal{N}(0, \mathbb{I})$
- image space $\mathcal{X} = \mathbb{R}^n$, conditional distribution $p_\theta(x \mid z)\colon \mathcal{N}(\mu_\theta(z), \sigma^2\mathbb{I})$

  The mapping $\mathcal{Z} \ni z \mapsto \mu_\theta \in \mathcal{X}$ is modelled in terms of a (deep, convolutional) *decoder network* $d_\theta\colon \mathcal{Z} \to \mathcal{X}$.
- Learning goal: maximise data log-likelihood

$$L(\theta; \mathcal{T}) = \mathbb{E}_{\mathcal{T}} \log p_\theta(x) = \mathbb{E}_{\mathcal{T}} \log \int_{\mathcal{Z}} dz\, p_\theta(x \mid z) p(z)$$

Computing $L(\theta)$ or $\nabla_\theta L(\theta)$ is not tractable! It would require to integrate the decoder mapping $d_\theta(z)$ over the latent space $\mathcal{Z}$:



(Gaussian) Variational Autoencoders

Use ELBO, i.e. a lower bound of the data log-likelihood

$$L(\theta) \geqslant L_B(\theta, q) = \mathbb{E}_{\mathcal{T}} \mathbb{E}_{q(z \,|\, x)} \Big[\log p_\theta(x \,|\, z) - \log \frac{q(z \,|\, x)}{p(z)} \Big]$$

May be we can apply the *EM algorithm* directly?

EM-algorithm corresponds to block-coordinate ascent of $L_B(\theta, q)$ w.r.t. θ and q

E-step fix θ_t, set $q_t(z \,|\, x) = \arg\max_q L(\theta_t, q) \Rightarrow q_t(z \,|\, x) = p_{\theta_t}(z \,|\, x)$

M-step fix $q_t(z \,|\, x)$, maximise $\theta_{t+1} = \arg\max_\theta \mathbb{E}_{\mathcal{T}} \mathbb{E}_{q_t(z \,|\, x)} \log p_\theta(x \,|\, z)$

No, it is not feasible because computing

$$p_{\theta_t}(z \,|\, x) = \frac{p_{\theta_t}(x \,|\, z) p(z)}{\int dz' \, p_{\theta_t}(x \,|\, z') p(z')}$$

would require to integrate the decoder mapping.

(Gaussian) Variational Autoencoders

Way out: choose a class of *amortized inference* models $q_\varphi(z \,|\, x)$

$$z \,|\, x \sim \mathcal{N}\big(\mu_\varphi(x), \operatorname{diag}(\sigma^2_\varphi(x))\big)$$

The mapping $x \mapsto \mu_\varphi(x), \sigma_\varphi(x)$ is modelled in terms of a (deep, convolutional) *encoder network* $e_\varphi(x) = \big(\mu_\varphi(x), \sigma_\varphi(x)\big)$.

The ELBO criterion reads now

$$L_B(\theta, \varphi) = \mathbb{E}_{\mathcal{T}}\Big[\mathbb{E}_{q_\varphi(z \,|\, x)} \log p_\theta(x \,|\, z) - D_{KL}(q_\varphi(z \,|\, x) \parallel p(z))\Big]$$

Can we maximise it by gradient ascent w.r.t. θ and φ?

- $\mathbb{E}_{\mathcal{T}}$: SGD with mini-batches ✓
- $D_{KL}(q_\varphi(z \,|\, x) \parallel p(z))$: both Gaussians factorise and the KL-divergence decomposes into a sum over components $\sum_{i=1}^m D_{KL}(q_\varphi(z_i \,|\, x) \parallel p(z_i))$. The KL-divergence of univariate Gaussian distributions can be computed in closed form! ✓

(Gaussian) Variational Autoencoders

$$L_B(\theta, \varphi) = \mathbb{E}_{\mathcal{T}}\Big[\mathbb{E}_{q_\varphi(z \,|\, x)} \log p_\theta(x \,|\, z) - D_{KL}(q_\varphi(z \,|\, x) \parallel p(z))\Big]$$

- $\nabla_\theta \mathbb{E}_{q_\varphi(z \,|\, x)} \log p_\theta(x \,|\, z)$: use SGD by sampling $z \sim q_\varphi(z \,|\, x)$. ✓
- $\nabla_\varphi \mathbb{E}_{q_\varphi(z \,|\, x)} \log p_\theta(x \,|\, z)$: this gradient is *critical*.
  We can not replace $\mathbb{E}_{q_\varphi(z \,|\, x)}$ by a sample $z \sim q_\varphi(z \,|\, x)$, because it will depend on φ!

Re-parametrisation trick: Simple solution for Gaussians:

$$z \sim \mathcal{N}(\mu, \sigma^2) \iff \epsilon \sim \mathcal{N}(0,1) \text{ and } z = \sigma\epsilon + \mu$$

Now, if μ and σ depend on φ:

$$\nabla_\varphi \mathbb{E}_{z \sim \mathcal{N}(\mu_\varphi, \sigma_\varphi^2)}[f(z)] = \mathbb{E}_{z\epsilon \sim \mathcal{N}(0,1)}\big[\nabla_\varphi f(\sigma_\varphi \epsilon + \mu_\varphi)\big]$$

(Gaussian) Variational Autoencoders

Overall, the learning step for a (Gaussian) VAE is pretty simple:

Fetch a mini-batch x from training data

1. apply the encoder network $e_\varphi(x) \mapsto \mu_\varphi(x), \sigma_\varphi(x)$ and compute $q_\varphi(z \,|\, x)$
2. compute the KL-divergence $D_{KL}(q_\varphi(z \,|\, x) \,\|\, p(z))$
3. sample a batch $z \sim q_\varphi(z \,|\, x)$ with reparametrisation
4. apply the decoder network $d_\theta(z) \mapsto \mu_\theta(z)$ and compute $\log p_\theta(x \,|\, z)$
5. combine the ELBO terms and let PyTorch compute the derivatives and make an SGD step.

Strengths and weaknesses of VAEs

- concise model, simple objective (ELBO), can be optimised by SGD ✓
- local optima, *posterior collapse*: some latent components collapse to $q_\varphi(z_i \,|\, x) = p(z_i)$, i.e. they carry no information. ✗
- amortized inference models $q_\varphi(z \,|\, x)$ may have not enough expressive power to close the gap between $L(\theta)$ and $L_B(\theta, \varphi)$. ✗

VAE approximation errors

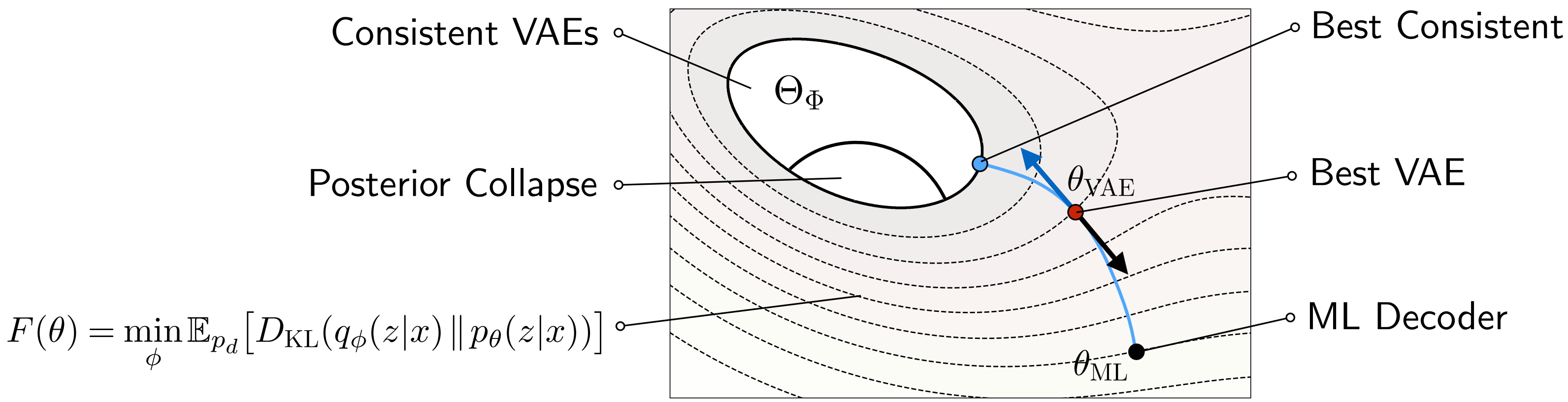
The ELBO objective can be written in two equivalent forms

$$L_B(\theta, \varphi) = \mathbb{E}_{p_d}\big[\mathbb{E}_{q_\varphi} \log p_\theta(x \,|\, z) - D_{KL}(q_\varphi(z \,|\, x) \,\|\, p(z))\big]$$
$$= L(\theta) - \mathbb{E}_{p_d}\big[D_{KL}(q_\varphi(z \,|\, x) \,\|\, p_\theta(z \,|\, x))\big].$$

The second one shows that the lower bound is tight if and only if $q_\varphi(z \,|\, x) \equiv p_\theta(z \,|\, x)$. Define the *consistent set* $\Theta_\Phi \subseteq \Theta$ as the subset of distributions $p_\theta(x, z)$ whose posteriors are in $\mathcal{Q}_\Phi$, i.e.,

$$\Theta_\Phi = \big\{\theta \in \Theta \;\big|\; \exists \varphi \in \Phi : q_\varphi(z \,|\, x) \equiv p_\theta(z \,|\, x)\big\}. \tag{1}$$

The KL-divergence in the ELBO objective can vanish only if $\theta \in \Theta_\Phi$. If the likelihood maximizer θ_{ML} is not contained in Θ_Φ, then this KL-divergence pulls the optimizer towards Θ_Φ and away from θ_{ML}.



VAE approximation errors

Let us assume that the encoder and the decoder are exponential families

$$p_\theta(x \mid z) = h(x) \exp\big[\langle \nu(x), d_\theta(z)\rangle - A(d_\theta(z))\big]$$
$$q_\varphi(z \mid x) = h'(z) \exp\big[\langle \psi(z), e_\varphi(x)\rangle - A(e_\varphi(x))\big],$$

where $\nu(x)$, $\psi(z)$ are the corresponding sufficient statistsics.

Theorem 1. *The consistent set Θ_Φ of an exponential family VAE is given by decoders (and encoders) of the form*

$$p(x \mid z) = h(x) \exp\big[\langle \nu(x), W\psi(z)\rangle + \langle \nu(x), u\rangle - A(z)\big],$$
$$q(z \mid x) = h'(z) \exp\big[\langle \psi(z), W^T \nu(x)\rangle + \langle \psi(z), v\rangle - B(x)\big],$$

where W is a $n \times m$ matrix and $u \in \mathbb{R}^n$, $v \in \mathbb{R}^m$ are vectors.

The corresponding joint probability distribution $p(x, z)$ takes the form of an EF Harmonium:

$$p(x, z) \propto h(x) h'(z) \exp\big(\langle \nu(x), W\psi(z)\rangle + \langle \nu(x), u\rangle + \langle \psi(z), v\rangle\big).$$

The subset Θ_Φ of consistent models can not be enlarged by considering more complex encoder networks $g(x)$, provided that the affine family $W^\mathsf{T}\nu(x)$ can already be represented.

Hierarchical VAEs

Hierarchical decoder (Sønderby et al., 2016)

$$p_\theta(z) = p(z_0) \prod_{i=1}^{m} p_\theta(z_i \mid z_{i-1}) \text{ and } p_\theta(x \mid z_m)$$

HMM + EM algorithm view: Compute pairwise marginals of $p(z \mid x)$ for each $x \in \mathcal{T}^\ell$ in the E-step. Here instead, sample from it (notice that $p(z \mid x)$ is a Markov model). We have

$$p(z_i \mid z_{i-1}, x) \propto p(z_i \mid z_{i-1}) p(x \mid z_i).$$

In HMMs with small finite state spaces, the probabilities $p(x \mid z_i)$ are computed by the backward algorithm with iteration

$$p(x \mid z_{i-1}) = \sum_{z_i} p(z_i \mid z_{i-1}) p(x \mid z_i).$$

This is however not possible for hierarchical VAEs, because their latent variables z_i are usually high dimensional vectors. The computation of the $p(x \mid z_i)$ is therefore approximated by the encoder $q(z \mid x)$.

Hierarchical VAEs

We assume for simplicity binary valued latent vectors $z_i \in \mathcal{B}^{n_i}$. To approximate the values $p(x \,|\, z_i)$, the encoder uses a deterministic deep network which (in the simplest case) computes

$$a_i = W_i f(a_{i+1})$$

starting from $a_m = W_m x$. Notice that we denote the non-linear activation function of this network by f. Finally, the log-probabilities $\log p(x \,|\, z_i)$ are approximated by a_i. This gives

$$p(z_i \,|\, z_{i-1}, x) \propto \exp\langle z_i, d_i(z_{i-1}) + a_i(x)\rangle,$$

where $d_i(z_{i-1})$ is the natural parameter vector of the distribution $p(z_i \,|\, z_{i-1})$.

ELBO learning for such models requires

- Computing KL-divergence between $p(z_i \,|\, z_{i-1}, x)$ and $p(z_i \,|\, z_{i-1})$ ✓
- Differentiating a sample w.r.t. parameters of the distribution that generates it. Gaussian case: re-parmaterisation, Bernoulli case: e.g. *straight through gradient estimator*.

Hierarchical Variational Autoencoders

Advanced VAEs with strong encoders can generate very good images. A. Vahdat et al., NeurIPS 2020: A Deep Hierarchical VAE trained on CelebA data.