Latent Variable Models: Priors, Decoders, and Intractable Posteriors

N-20251009

§1.1 From autoencoders to latent variable models

A deterministic autoencoder is a pair of maps

$$x \xrightarrow{\text{encoder}} h \xrightarrow{\text{decoder}} \widehat{x}.$$

The code h is a deterministic representation of x. A VAE keeps the encoder–decoder visual shape, but the mathematical object is different. The hidden representation is a random variable Z, and the decoder defines a conditional distribution, not merely a reconstruction function.

The generative story is

$$z \sim p(z), \qquad x \sim p_\theta(x \mid z).$$

Thus a VAE is best understood as a latent variable model equipped with an efficient approximate inference model.

The key distinction is:

deterministic autoencoder	VAE / latent variable model
$h = f_\phi(x)$	$z \sim q_\phi(z \mid x)$
$\widehat{x} = g_\theta(h)$	$x \sim p_\theta(x \mid z)$
reconstruction map	probabilistic generative model

§1.2 The generative story

The model begins with a prior distribution on the latent variable:

$$p(z).$$

In the basic VAE,

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

This prior says that latent variables should live in a simple, continuous, isotropic space.

Given z, the decoder defines the likelihood of an observation:

$$p_\theta(x \mid z).$$

The joint density is then

$$p_\theta(x, z) = p(z)p_\theta(x \mid z).$$

This formula is the mathematical core of the generative model.

The decoder network does not directly output a final sample. It outputs parameters of a distribution. For binary data, one often writes

$$p_\theta(x \mid z) = \text{Bernoulli}(x; \pi_\theta(z)).$$

Here $\pi_\theta(z)$ gives the success probabilities of the Bernoulli coordinates. For real-valued data, one may use

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

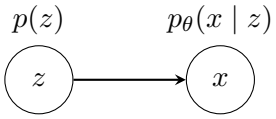
Here the decoder outputs a mean $\mu_\theta(z)$, and the variance is either fixed or parameterized separately.

§1.3 Graphical model notation

The generative model has one directed probabilistic arrow:

$$z \longrightarrow x.$$

A small diagram is:



For a dataset $x^{(1)}, \ldots, x^{(N)}$, the story repeats independently:

$$z^{(i)} \sim p(z), \qquad x^{(i)} \sim p_\theta(x \mid z^{(i)}).$$

The parameters θ are shared across all observations.

§1.4 Marginal likelihood

The model likelihood of one observation is obtained by integrating out the latent variable:

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

For a dataset, maximum likelihood would choose θ by maximizing

$$\sum_{i=1}^{N} \log p_\theta(x^{(i)}).$$

Equivalently, one wants the model distribution $p_\theta(x)$ to assign high probability density to the observed data.

The problem is that the integral

$$\int p(z) p_\theta(x \mid z)\, dz$$

is usually intractable. If $p_\theta(x \mid z)$ is parameterized by a nonlinear neural network, there is no closed-form expression for the integral over z.

This is the first reason variational methods enter.

§1.5 The posterior problem

The posterior distribution of the latent variable is

$$p_\theta(z \mid x) = \frac{p_\theta(x, z)}{p_\theta(x)} = \frac{p(z)p_\theta(x \mid z)}{\int p(z')p_\theta(x \mid z')\, dz'}.$$

The numerator is easy to evaluate up to normalization. The denominator is the marginal likelihood, which is the same difficult integral.

Therefore the model has a basic asymmetry:

sampling direction	inference direction
$z \sim p(z),\ x \sim p_\theta(x \mid z)$	$p_\theta(z \mid x)$ is hard

This is the central probabilistic problem behind VAEs:

$$\begin{gathered}\text{to train the model, we want } \log p_\theta(x),\\ \text{to understand } \log p_\theta(x), \text{ we need to reason about } z,\\ \text{but the true posterior } p_\theta(z \mid x) \text{ is intractable.}\end{gathered}$$

§1.6 Why naive Monte Carlo is not enough

One might try to estimate the marginal likelihood by drawing samples from the prior:

$$z^{(\ell)} \sim p(z), \qquad p_\theta(x) \approx \frac{1}{L}\sum_{\ell=1}^{L} p_\theta(x \mid z^{(\ell)}).$$

This is formally reasonable, but in high dimension it is usually inefficient. Most prior samples z do not explain a fixed observation x well. The average is then dominated by rare samples that land in useful regions of latent space.

For inference about a particular x, we would rather sample z values that are already plausible explanations of x. That is the motivation for introducing an approximate posterior.

§1.7 The recognition model

A VAE introduces a family

$$q_\phi(z \mid x)$$

to approximate the true posterior $p_\theta(z \mid x)$. This is often called the recognition model or encoder.

The standard Gaussian encoder has the form

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

The functions $\mu_\phi(x)$ and $\sigma_\phi(x)$ are outputs of a neural network. Mathematically, they parameterize a distribution over latent variables.

It is important not to confuse the two conditional distributions:

symbol	name	direction
$p_\theta(x \mid z)$	decoder likelihood	$z \to x$
$q_\phi(z \mid x)$	approximate posterior / encoder	$x \to z$

Only $p_\theta(x \mid z)$ belongs to the generative model. The encoder $q_\phi(z \mid x)$ is introduced to make inference and training feasible.

§1.8 Amortized inference

Classical variational inference might introduce separate variational parameters for each data point. For example, for every observation $x^{(i)}$, one could choose its own mean and variance:

$$q_i(z) = \mathcal{N}(z; \mu_i, \mathrm{diag}(\sigma_i^2)).$$

A VAE instead uses one shared inference network:

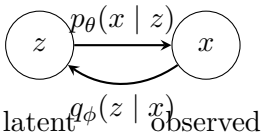
$$x \longmapsto \mu_\phi(x), \qquad x \longmapsto \sigma_\phi(x).$$

This is called amortized inference. The cost of inference is amortized across the dataset: after learning ϕ, the model can quickly produce an approximate posterior for a new x.

The mathematical tradeoff is that the family is restricted. We do not choose arbitrary variational parameters for each observation; we require them to be outputs of the same function.

§1.9 The VAE model diagram

The two directions can be displayed together:



The upper arrow is the probabilistic generative story. The lower arrow is the learned inference shortcut. The VAE objective will force these two directions to be compatible, but they play different roles.

§1.10 Remaining derivation

At this point, the problem is clear. We want to maximize

$$\log p_\theta(x),$$

but

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

is hard. We also want to use

$$p_\theta(z \mid x),$$

but it contains the same intractable marginal likelihood.

The solution is to introduce $q_\phi(z \mid x)$, then build a lower bound on $\log p_\theta(x)$ that can be optimized. That lower bound is the evidence lower bound, or ELBO.

The next note derives it from two facts:

$$D_{\mathrm{KL}}(q \| p) \geq 0, \qquad \log \mathbb{E}[Y] \geq \mathbb{E}[\log Y].$$