Variational Autoencoders

Handout Notes - Generative and Deep Learning (GDL)

Davide Bacciu - University of Pisa

Notation. Observed data are denoted by $\mathbf{X}$ and values by $\mathbf{x} \in \mathbb{R}^D$. Latent random variables are denoted by $\mathbf{Z}$ and values by $\mathbf{z} \in \mathbb{R}^K$. A dataset is $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ with $N = |\mathcal{D}|$. Model parameters are denoted by θ and variational parameters by ϕ. The decoder distribution is written $p_\theta(\mathbf{x} \mid \mathbf{z})$, the prior by $p(\mathbf{z})$, and the variational posterior by $q_\phi(\mathbf{z} \mid \mathbf{x})$. The log-likelihood is

$$\ell(\theta) = \log p_\theta(\mathcal{D}) = \sum_{n=1}^{N} \log p_\theta(\mathbf{x}^{(n)}).$$

The Kullback–Leibler divergence is written $\mathrm{KL}(q\|p)$.

1 From deterministic autoencoders to explicit generative models

A classical autoencoder learns an encoder

$$\mathbf{z} = f_\theta(\mathbf{x})$$

and a decoder

$$\tilde{\mathbf{x}} = g_\theta(\mathbf{z}),$$

by minimizing reconstruction error. This is useful for representation learning, but it does not by itself define a proper generative model of the data distribution. In particular, a deterministic autoencoder does not tell us how latent codes are distributed, nor does it provide a normalized density $p_\theta(\mathbf{x})$.

This limitation becomes especially clear if we ask a generative question:

Can we sample a new latent vector and decode it into a plausible new data point?

For an ordinary autoencoder, the answer is not principled. The latent space may be highly irregular, with valid encodings occupying only a complicated subset. Sampling an arbitrary latent point may decode to nonsense.

Variational autoencoders solve this by turning the autoencoder into a latent-variable probabilistic model. The key move is to define:

- a prior distribution over latent variables,
- a conditional decoder distribution from latent space to data space,
- an approximate posterior that makes learning tractable.

2 A latent-variable generative model

The VAE assumes that data are generated through latent variables. The generative story is:

1. sample a latent code

$$\mathbf{z} \sim p(\mathbf{z}),$$

typically with a simple prior such as

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

2. generate the observation from a decoder distribution

$$\mathbf{x} \sim p_\theta(\mathbf{x} \mid \mathbf{z}).$$

This defines the joint distribution

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

The corresponding marginal data density is

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{z})\, d\mathbf{z}.$$

This is the key point: unlike a deterministic autoencoder, the VAE explicitly defines a probabilistic model for the observed data. The challenge is that this marginalization over latent variables is usually intractable for nontrivial neural decoders.

Worked example — Why latent marginalization is difficult

Suppose the latent space is continuous and

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})\, d\mathbf{z}.$$

If $p_\theta(\mathbf{x} \mid \mathbf{z})$ is implemented by a neural network decoder, then this integral has no closed form in general. Even evaluating it numerically for every training point and every parameter update would be prohibitively expensive. Thus maximum-likelihood learning

$$\max_\theta \sum_{n=1}^{N} \log p_\theta(\mathbf{x}^{(n)})$$

is not directly tractable. This is the reason variational approximation enters the picture.

3 Probabilistic encoder and decoder

A VAE can be viewed as a probabilistic autoencoder. The decoder is no longer a deterministic map $\mathbf{z} \mapsto \tilde{\mathbf{x}}$, but a conditional distribution

$$p_\theta(\mathbf{x} \mid \mathbf{z}).$$

Similarly, the encoder becomes a probabilistic approximation to the posterior:

$$q_\phi(\mathbf{z} \mid \mathbf{x}).$$

In the most common continuous latent-space formulation, the encoder outputs the parameters of a Gaussian:

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\big(\boldsymbol{\mu}_\phi(\mathbf{x}), \operatorname{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x}))\big).$$

Thus the encoder network does not output one latent point, but a mean and variance:

$$\boldsymbol{\mu}(\mathbf{x}), \qquad \boldsymbol{\sigma}^2(\mathbf{x}).$$

The decoder then receives a sampled latent code

$$\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})$$

and defines a reconstruction distribution $p_\theta(\mathbf{x} \mid \mathbf{z})$.

This is the central architectural shift:

- deterministic encoding becomes stochastic inference,
- deterministic decoding becomes conditional generation,
- reconstruction becomes likelihood-based learning.

4 Maximum likelihood and the intractable posterior

Ideally, we would like to maximize the data likelihood

$$p_\theta(\mathcal{D}) = \prod_{n=1}^{N} p_\theta(\mathbf{x}^{(n)}),$$

or equivalently the log-likelihood

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

But each term involves

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})\, d\mathbf{z},$$

which is generally intractable.

A second intractable quantity is the true posterior

$$p_\theta(\mathbf{z} \mid \mathbf{x}) = \frac{p_\theta(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})}{p_\theta(\mathbf{x})}.$$

This posterior is exactly what we would need for EM-style latent-variable learning, but its denominator is the same intractable marginal likelihood.

The VAE therefore uses variational inference: instead of computing the true posterior, we introduce a tractable approximation

$$q_\phi(\mathbf{z} \mid \mathbf{x}).$$

5 Variational inference and the ELBO

The standard variational identity gives

$$\log p_\theta(\mathbf{x}) = \mathcal{L}(\mathbf{x}; \theta, \phi) + \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p_\theta(\mathbf{z} \mid \mathbf{x})\big),$$

where the evidence lower bound (ELBO) is

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})\big].$$

Since the KL divergence is nonnegative,

$$\mathcal{L}(\mathbf{x}; \theta, \phi) \le \log p_\theta(\mathbf{x}),$$

so maximizing the ELBO gives a tractable lower-bound optimization objective.

Expanding the joint term,

$$\log p_\theta(\mathbf{x}, \mathbf{z}) = \log p_\theta(\mathbf{x} \mid \mathbf{z}) + \log p(\mathbf{z}),$$

we obtain the familiar decomposition

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x} \mid \mathbf{z})\big] - \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big).$$

This is the core VAE objective.

6 Interpretation of the two ELBO terms

The ELBO

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x} \mid \mathbf{z})\big] - \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big)$$

has a very intuitive structure.

6.1 Reconstruction term

The first term,

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x} \mid \mathbf{z})\big],$$

encourages the decoder to reconstruct or explain the observed input well from the sampled latent code. It is the probabilistic analogue of the reconstruction loss in a classical autoencoder.

6.2 Regularization term

The second term,

$$\mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big),$$

encourages the variational posterior to stay close to the prior. This prevents the encoder from placing latent codes in arbitrary disconnected regions of latent space. It is precisely this regularization that makes latent-space sampling meaningful.

So the VAE simultaneously learns:

- a decoder that reconstructs data well from latent samples,
- an encoder whose output distributions remain compatible with a simple global prior.

7 Gaussian posterior and closed-form KL

The standard VAE assumes

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{x}), \mathrm{diag}(\boldsymbol{\sigma}^2(\mathbf{x}))\big), \qquad p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, I).$$

With this choice, the KL divergence has a closed form:

$$\mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big) = \frac{1}{2} \sum_{j=1}^{K} \left(\mu_j(\mathbf{x})^2 + \sigma_j(\mathbf{x})^2 - \log \sigma_j(\mathbf{x})^2 - 1\right).$$

This is computationally important because it means only the reconstruction expectation needs to be approximated stochastically.

Worked example — Closed-form KL for diagonal Gaussians

Let

$$q(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2)), \qquad p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, I).$$

Then for each latent dimension j,

$$\mathrm{KL}\big(\mathcal{N}(\mu_j, \sigma_j^2) \| \mathcal{N}(0, 1)\big) = \frac{1}{2}\left(\mu_j^2 + \sigma_j^2 - \log \sigma_j^2 - 1\right).$$

Summing over dimensions gives

$$\mathrm{KL}(q \| p) = \frac{1}{2} \sum_{j=1}^{K} \left(\mu_j^2 + \sigma_j^2 - \log \sigma_j^2 - 1\right).$$

This term is zero only when

$$\mu_j = 0, \qquad \sigma_j^2 = 1$$

for all j, meaning the variational posterior exactly matches the standard normal prior.

8 The reparameterization trick

At training time we need gradients of the ELBO with respect to both decoder parameters θ and encoder parameters ϕ. The reconstruction term depends on a sample

$$\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}),$$

and naive sampling is not differentiable with respect to ϕ.

The reparameterization trick solves this by expressing the random variable $\mathbf{z}$ as a deterministic function of:

- the encoder outputs $\boldsymbol{\mu}(\mathbf{x})$ and $\boldsymbol{\sigma}(\mathbf{x})$,
- an auxiliary noise variable independent of the parameters.

For a Gaussian posterior, write

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, I), \qquad \mathbf{z} = \boldsymbol{\mu}(\mathbf{x}) + \boldsymbol{\sigma}(\mathbf{x}) \odot \boldsymbol{\varepsilon}.$$

Now the randomness is entirely in $\boldsymbol{\varepsilon}$, while $\mathbf{z}$ becomes a differentiable function of the encoder outputs.

Worked example — Why reparameterization restores differentiability

Without reparameterization, the reconstruction term involves

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z})],$$

where the distribution of $\mathbf{z}$ depends on ϕ. With the Gaussian substitution

$$\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, I),$$

the expectation becomes

$$\mathbb{E}_{\boldsymbol{\varepsilon}\sim\mathcal{N}(\mathbf{0},I)} \left[\log p_\theta\big(\mathbf{x} \mid \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\varepsilon}\big)\right].$$

Now no expectation is taken with respect to a distribution that depends on ϕ. Therefore gradients with respect to ϕ can be pushed through the deterministic transformation from $(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon})$ to $\mathbf{z}$.

This is what makes VAE training by ordinary backpropagation possible.

9 The VAE training objective in practice

With reparameterization, the per-example objective becomes

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{\boldsymbol{\varepsilon}\sim\mathcal{N}(\mathbf{0},I)} \left[\log p_\theta\big(\mathbf{x} \mid \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\varepsilon}\big)\right] - \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big).$$

In stochastic optimization, the expectation is often approximated with one Monte Carlo sample of $\boldsymbol{\varepsilon}$ per data point per update. The full training objective is

$$\max_{\theta,\phi} \sum_{n=1}^{N} \mathcal{L}(\mathbf{x}^{(n)}; \theta, \phi).$$

Thus VAE training is simply backpropagation through a stochastic computation graph whose randomness has been reparameterized.

10 Information-theoretic view of the KL term

The KL regularizer

$$\mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\big)$$

can also be interpreted information-theoretically. It measures how many extra nats or bits are required to describe a latent sample from the input-dependent posterior rather than from the uninformed prior.

From this viewpoint:

- the reconstruction term rewards informative latent codes,
- the KL term penalizes excessive information being stored in the latent representation.

So the VAE learns under a compression trade-off:

> *encode enough information in* $\mathbf{z}$ *to reconstruct* $\mathbf{x}$*, but not so much that the latent representation ceases to resemble samples from the simple prior.*

This perspective is one reason VAEs are important for representation learning, not only for generation.

11 Sampling from a trained VAE

Once training is complete, the encoder is no longer needed for unconditional generation. To generate a new sample:

1. sample
$$\mathbf{z} \sim p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, I),$$
2. decode through
$$\mathbf{x} \sim p_\theta(\mathbf{x} \mid \mathbf{z}),$$
or take the decoder mean as the generated sample.

This is the crucial improvement over ordinary autoencoders: the latent space is explicitly regularized so that prior samples decode to plausible outputs.

Worked example — Why VAE sampling is principled whereas AE sampling is not

In a deterministic autoencoder, latent codes seen during training may occupy a highly irregular subset of latent space. Sampling a random $\mathbf{z}$ from, say, $\mathcal{N}(\mathbf{0}, I)$ has no particular reason to land in a meaningful region. In a VAE, however, training explicitly encourages

$$q_\phi(\mathbf{z} \mid \mathbf{x}) \approx p(\mathbf{z})$$

for all training inputs. Therefore the decoder is trained on latent samples drawn from distributions close to the same prior used at generation time. This is why random latent sampling followed by decoding is a coherent generative procedure in VAEs.

12 VAE versus denoising and contractive autoencoders

Denoising and contractive autoencoders learn useful geometry of the data distribution, especially near the training manifold. They can approximate the score or local vector field of the data density, but they do not provide an explicit global latent-variable density model.

The VAE differs in a fundamental way:

- it defines an explicit generative process with prior $p(\mathbf{z})$ and decoder $p_\theta(\mathbf{x} \mid \mathbf{z})$;

- it learns a variational posterior $q_\phi(\mathbf{z} \mid \mathbf{x})$ to make inference tractable;
- it supports principled sampling from latent space.

One way to summarize the contrast is:

> *regularized deterministic autoencoders learn local structure; VAEs learn a globally regularized latent-variable generative model.*

13 Conditional variational autoencoders

A conditional variational autoencoder (CVAE) extends the model so that generation is conditioned on an auxiliary variable $\mathbf{y}$. Instead of modeling $p_\theta(\mathbf{x})$, it models

$$p_\theta(\mathbf{x} \mid \mathbf{y}).$$

The generative process becomes

$$\mathbf{z} \sim p(\mathbf{z}), \qquad \mathbf{x} \sim p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{y}),$$

and the variational posterior becomes

$$q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y}).$$

The ELBO changes accordingly:

$$\mathcal{L}(\mathbf{x}, \mathbf{y}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}\big[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{y})\big] - \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y}) \| p(\mathbf{z})\big).$$

This allows controlled generation: for example, generate an image conditioned on a class label or an input attribute vector.

14 VAEs for representation learning

Although VAEs are generative models, they are also powerful representation learners. Because the latent space is regularized and continuous, it often acquires meaningful structure:

- nearby latent points decode to similar samples,
- interpolation in latent space yields smooth semantic transitions,
- some latent directions may correspond to interpretable factors of variation.

This explains why VAEs are widely used not only for sampling but also for:

- dimensionality reduction,
- latent-space interpolation,
- controllable generation,
- disentanglement-oriented representation learning.

Worked example — Latent interpolation in a VAE

Suppose two data points are encoded to latent means

$$\boldsymbol{\mu}_1, \qquad \boldsymbol{\mu}_2.$$

A linear interpolation in latent space is

$$\mathbf{z}(\alpha) = (1-\alpha)\boldsymbol{\mu}_1 + \alpha\boldsymbol{\mu}_2, \qquad \alpha \in [0,1].$$

Decoding the sequence

$$g_\theta(\mathbf{z}(0)),\ g_\theta(\mathbf{z}(0.1)),\ \ldots,\ g_\theta(\mathbf{z}(1))$$

often yields a smooth semantic transformation between the two original samples. This is possible because the KL term encourages the latent space to be organized and approximately filled according to a simple prior, rather than fragmented into disconnected code regions.

15 Disentanglement and factors of variation

An especially appealing goal in VAE-based representation learning is disentanglement: different latent coordinates should correspond to distinct underlying factors of variation, such as rotation, size, expression, or pose.

Standard VAEs do not guarantee disentanglement, but they provide the right starting point:

- a continuous latent space,
- a regularized prior,
- a reconstruction objective that ties latent variables to data variation.

This is why many later models for disentangled representation learning are modifications of the VAE objective.

16 Strengths and limitations of VAEs

VAEs have several important strengths:

- principled latent-variable probabilistic foundation,
- efficient end-to-end training by backpropagation,
- meaningful and smooth latent spaces,
- natural support for conditional generation and representation learning.

They also have limitations:

- the ELBO is only a lower bound on the true log-likelihood,
- approximate posterior families may be too restrictive,
- sample quality is often blurrier than that of adversarial or diffusion-based models, especially for images.

So VAEs are often strongest when latent structure and representation learning matter at least as much as raw perceptual sample sharpness.

17 Conceptual summary

Variational autoencoders are the probabilistic generative counterpart of classical autoencoders. Their key ideas are:

- introduce a latent-variable generative model

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x} \mid \mathbf{z})p(\mathbf{z});$$

- approximate the intractable posterior with an encoder distribution

$$q_\phi(\mathbf{z} \mid \mathbf{x});$$

- optimize the ELBO

$$\mathbb{E}_{q_\phi}\big[\log p_\theta(\mathbf{x} \mid \mathbf{z})\big] - \mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z}));$$

- use the reparameterization trick to train by standard backpropagation;
- generate new samples by drawing $\mathbf{z}$ from the prior and decoding.

Conceptually, the VAE solves the problem that deterministic autoencoders leave open: it turns representation learning into explicit density-based generative modeling while retaining the deep-learning flexibility of neural encoders and decoders.