Lecture notes on variational auto-encoders (VAEs)

1.1 Prerequisites

Divergence minimisation

Given a set of probability distributions $\mathcal{P}$ on a random variable X, a divergence is defined as a function $\mathrm{D}[\cdot||\cdot] : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ such that $\mathrm{D}[P||Q] \geq 0$ for all $P, Q \in \mathcal{P}$, and $\mathrm{D}[P||Q] = 0$ iff. $P = Q$.

The definition of divergence is much weaker than that for a *distance* such as the ℓ_2-norm, since it does not need to satisfy either symmetry in arguments or the triangle inequality. There exist many available divergences to use, some of them will be introduced throughout this course.

In this course we assume the probability distributions/measures in $\mathcal{P}$ are dominated by the Lebesgue measure of the underlying Eucledean space, so that we can work with *probability density functions* (PDFs)

$$p(\boldsymbol{x}) = \frac{dP}{d\boldsymbol{x}}, \forall P \in \mathcal{P}. \tag{1}$$

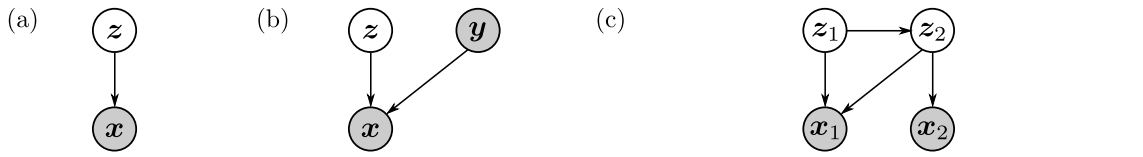
In the following we will also write $\mathcal{P}$ as the set of PDFs w.l.o.g., and use the terms probability distribution and probability density functions interchangeably (unless specifically mentioned).

Probabilistic graphical models

In machine learning tasks we may define the model as a distribution on a set of random variables with particular dependency structures. Some of the variables might be unobserved as well. Probabilistic graphical models are powerful models that use graphs to describe the dependency structure of the random variables. In particular we consider direct acyclic graphs (DAGs) which are graphs with directed edges and without directed cycles. By assuming Markov properties, DAGs can be used to describe the factorisation structure of the joint distribution. Interested readers are referred to e.g. Chapter 8 of Bishop [2007] for a formal introduction of probabilistic graphical models. For this course we only introduce the principles for reading joint distributions from a DAG (and vice versa). Assuming we are interested in the distribution $p(\boldsymbol{x}_1, ..., \boldsymbol{x}_D)$ for a given DAG with nodes $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_D\}$ and directed edges between them, then the joint distribution is:

$$p(\boldsymbol{x}_1, ..., \boldsymbol{x}_D) = \prod_{i=1}^{D} p(\boldsymbol{x}_i | pa(\boldsymbol{x}_i)), \tag{2}$$

where $pa(\boldsymbol{x}_i) \subset \{\boldsymbol{x}_1, ..., \boldsymbol{x}_D\}$ represents the parent nodes of $\boldsymbol{x}_i$ in the DAG. For a DAG there always exists root node(s) that have no parents (i.e. $pa(\boldsymbol{x}_i) = \emptyset$), and in such case $p(\boldsymbol{x}_i|pa(\boldsymbol{x}_i)) = p(\boldsymbol{x}_i)$. Conversely, given a joint distribution in the form of (2), we can also draw the corresponding DAG by adding arrows from nodes in $pa(\boldsymbol{x}_i)$ to $\boldsymbol{x}_i$. A number of examples are visualised in Figure 1.



$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$ $p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{y})p(\boldsymbol{z})p(\boldsymbol{y})$ $p(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{z}_1, \boldsymbol{z}_2) = p(\boldsymbol{x}_1|\boldsymbol{z}_1, \boldsymbol{z}_2)p(\boldsymbol{x}_2|\boldsymbol{z}_2)p(\boldsymbol{z}_2|\boldsymbol{z}_1)p(\boldsymbol{z}_1)$

Figure 1: Examples of probabilistic graphical models: graphs & the corresponding factorisations of the joint distributions. Shaded nodes represent observed variables and the other nodes represent unobserved/latent variables. Example (a) corresponds to the latent variable model used in VAEs & GANs, and example (b) corresponds to to the latent variable model used in conditional VAEs & GANs where $\mathbf{y}$ represents additional information that the generative model is conditioned on.

Jensen's inequality

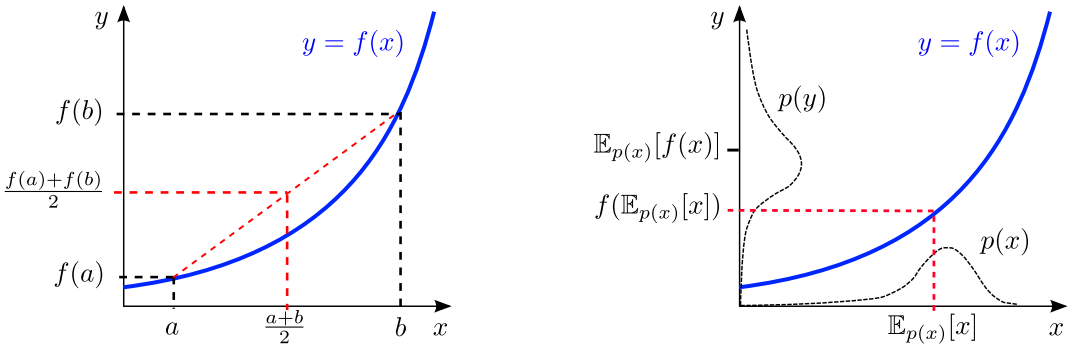
Below we introduce Jensen's inequality as a prerequisite for later discussions on divergences.

Proposition 1. *(Jensen's inequality) If $f : \mathbb{R} \to \mathbb{R}$ is a convex function, then for any distribution $p(x)$,*

$$\mathbb{E}_{p(x)}[f(x)] \geq f(\mathbb{E}_{p(x)}[x]),$$

with equality holds iff. f is linear or $p(x)$ is a delta measure.

A visual proof is provided in the below figures.



(a) for discrete distributions

(b) for continuous distributions

Jensen's inequality can be generalised to functions formed by compositions of functions. To see this, we first introduce the *law of the unconscious statisticians* (LOTUS) rule:

Proposition 2. *(LOTUS) Given a distribution $p_X(\boldsymbol{x})$ and a function $\mathbf{y} = g(\boldsymbol{x})$ such that $\mathbb{E}_{p_X(\boldsymbol{x})}[g(\boldsymbol{x})] < +\infty$, the random variable $Y = g(X)$ has its distribution $p_Y(\mathbf{y})$ satisfying $\mathbb{E}_{p_Y(\mathbf{y})}[\mathbf{y}] = \mathbb{E}_{p_X(\boldsymbol{x})}[g(\boldsymbol{x})]$.*

Then a generalised version of Jensen's inequality reads as follows.

Proposition 3. *(Generalised Jensen's inequality) If a function $g(\boldsymbol{x})$ maps inputs to scalar outputs in $\mathbb{R}$ and $f : \mathbb{R} \to \mathbb{R}$ is a convex function, then for any distribution $p_X(\boldsymbol{x})$,*

$$\mathbb{E}_{p_X(\boldsymbol{x})}[f(g(\boldsymbol{x}))] \geq f(\mathbb{E}_{p_X(\boldsymbol{x})}[g(\boldsymbol{x})]),$$

with equality holds iff. f is linear or $p_X(\boldsymbol{x})$ is a delta measure.

Proof.

$$\begin{aligned}
\mathbb{E}_{p_X(\boldsymbol{x})}[f(g(\boldsymbol{x}))] &= \mathbb{E}_{p_Y(y)}[f(y)] && \text{(LOTUS applied to } y = g(\boldsymbol{x})) \\
&\geq f[\mathbb{E}_{p_Y(y)}[y]] && \text{(Jensen's inequality)} \\
&= f(\mathbb{E}_{p_X(\boldsymbol{x})}[g(\boldsymbol{x})]). && \text{(LOTUS applied to } y = g(\boldsymbol{x}))
\end{aligned}$$

□

Kullback-Leibler (KL) divergence

Kullback-Leibler divergence [Kullback and Leibler, 1951; Kullback, 1959], or *KL divergence*, is arguably one of the most widely used divergence measures in machine learning, statistics, and information theory.

Definition 1. *(Kullback-Leibler Divergence) The Kullback-Leibler (KL) divergence on $\mathcal{P}$ is defined as a function $\mathrm{KL}[\cdot||\cdot] : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ with the following form*

$$\mathrm{KL}[p||q] = \int p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x}, \quad p, q \in \mathcal{P}, \tag{3}$$

where log *is the natural logarithm (to base e).*

One can easily check that indeed the above definition is a valid divergence: define $f(x) = -\log x$ (which is convex) and $g(\boldsymbol{x}) = q(\boldsymbol{x})/p(\boldsymbol{x})$, we have

$$\begin{aligned}\mathrm{KL}[p||q] &= \mathbb{E}_{p(\boldsymbol{x})}[-\log g(\boldsymbol{x})] \\ &\geq -\log \mathbb{E}_{p(\boldsymbol{x})}[g(\boldsymbol{x})] \qquad \text{(Jensen's inequality)} \\ &= -\log \int p(\boldsymbol{x}) \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} d\boldsymbol{x} = -\log 1 = 0,\end{aligned}$$

and the equality holds iff. $p(\boldsymbol{x}) = q(\boldsymbol{x})$.[1] This means one can minimise the KL divergence in order to fit a distribution to a target one. Also notice that the KL divergence is asymmetric, i.e. $\mathrm{KL}[p||q] \neq \mathrm{KL}[q||p]$ in general.

Maximum likelihood estimation (MLE)

Given a dataset $\{(\boldsymbol{x}_n)\}_{n=1}^N \sim p_{\text{data}}(\boldsymbol{x})$, we would like to fit to it a generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ with parameter $\boldsymbol{\theta}$. Since the KL divergence can be used to measure the closeness of the model to the underlying data distribution, it makes sense to find the optimal parameters by minimising the KL divergence:

$$\boldsymbol{\theta}^* = \arg\min \mathrm{KL}[p_{\text{data}}(\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{x})]. \tag{4}$$

Expanding the above objective and re-arranging terms, we have

$$\mathrm{KL}[p_{\text{data}}(\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{x})] = \underbrace{\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\log p_{\text{data}}(\boldsymbol{x})]}_{\text{constant w.r.t. } \boldsymbol{\theta}} - \underbrace{\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]}_{\text{dependent on } \boldsymbol{\theta}}.$$

This means we can ignore the constant terms w.r.t. $\boldsymbol{\theta}$ and instead work with the following *maximum likelihood* objective:

$$\boldsymbol{\theta}^* = \arg\max \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]. \tag{5}$$

The obtained optimal parameters $\boldsymbol{\theta}^*$ is called the *maximum likelihood estimate* (MLE) of the parameters. In practice the data distribution is approximated by the empirical distribution on the dataset $\{\boldsymbol{x}_n\}_{n=1}^N \sim p_{\text{data}}(\boldsymbol{x})$, leading to

$$\boldsymbol{\theta}^* = \arg\max \frac{1}{N} \sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_n). \tag{6}$$

1.2 Variational inference

We are interested in fitting the following latent variable model (LVM) to the data:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}. \tag{7}$$

See Figure 1 (a) for a visualisation of the graphical model. In deep generative modelling context, this LVM is often constructed as (for continuous data)

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I}), \quad p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; G_{\boldsymbol{\theta}}(\boldsymbol{z}), \sigma^2 \mathbf{I}), \tag{8}$$

with $G_{\boldsymbol{\theta}}(\cdot)$ define as a neural network transform that is parameterised by weights $\boldsymbol{\theta}$. For discrete variables $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is usually defined as a categorical distribution with a neural network generator in use accordingly. Now to fit $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ to $p_{\text{data}}(\boldsymbol{x})$ we optimise the MLE objective (5) w.r.t. $\boldsymbol{\theta}$, which involves computing the integral (7). This is intractable as it involves computing the non-linear transformation $G_{\boldsymbol{\theta}}(\boldsymbol{z})$ for every single configuration of $\boldsymbol{z}$ within the support of the Gaussian prior $p(\boldsymbol{z})$, which is the full space $\boldsymbol{z} \in \mathbb{R}^d$.

[1] Technically speaking: $p(\boldsymbol{x}) = q(\boldsymbol{x})$ almost everywhere.

Variational inference provides a variational lower-bound of $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ as an approximation to it. For any distribution $q(\boldsymbol{z})$ satisfying $q(\boldsymbol{z}) > 0$ whenever $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) > 0$, we have

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \int q(\boldsymbol{z})\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \\
&\geq \int q(\boldsymbol{z}) \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \qquad \text{(Jensen's inequality)} \\
&= \mathbb{E}_{q(\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}[q(\boldsymbol{z})||p(\boldsymbol{z})] := \mathcal{L}(\boldsymbol{x}, q, \boldsymbol{\theta}).
\end{aligned} \tag{9}$$

With suitable choice of $q(\boldsymbol{z})$ and tricks that will be introduced later, this varitational lower-bound can be used as a tractable approximation to the marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$.

The choice of the $q(\boldsymbol{z})$ distribution is crucial to the quality of the approximation (or the tightness of the lower-bound). To see this, note that

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{x})}, \qquad \text{(Bayes' rule)} \tag{10}$$

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \mathrm{KL}[q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})] &= \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{q(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}\right] \\
&= \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) + \mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{x})}\right] \qquad \text{(Bayes' rule)} \\
&= \mathbb{E}_{q(\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}[q(\boldsymbol{z})||p(\boldsymbol{z})] = \mathcal{L}(\boldsymbol{x}, q, \boldsymbol{\theta}).
\end{aligned} \tag{11}$$

This means the gap (or the approximation error) between the variational lower-bound $\mathcal{L}(\boldsymbol{x}, q, \boldsymbol{\theta})$ and the marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is the KL divergence $\mathrm{KL}[q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})]$. Therefore the lower-bound improves as $q(\boldsymbol{z})$ approaches to the exact posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$. It also motivates the optimisation of the variational lower-bound w.r.t. the q distribution to obtain an approxiate posterior: since $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is constant w.r.t. q, maximising $\mathcal{L}(\boldsymbol{x}, q, \boldsymbol{\theta})$ is equivalent to minimising $\mathrm{KL}[q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})]$.

1.3 Variational auto-encoders

As discussed so far, we wish to fit the generative model (7) to the data by maximum likelihood (5), and variational inference provides a useful approximation $\mathcal{L}(\boldsymbol{x}, q, \boldsymbol{\theta}) \leq \log p_{\boldsymbol{\theta}}(x)$ for a given datum $\boldsymbol{x}$. Since this approximation is required for every datapoint in $\{\boldsymbol{x}_n\}_{n=1}^N$, having N separated q distributions $q_1(\boldsymbol{z}_1), ..., q_N(\boldsymbol{z}_n)$ to pair with $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$ can be memory inefficient. However, notice that the exact posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ depends on the input $\boldsymbol{x}$, and the variational lower-bound is tight when $q_n(\boldsymbol{z}_n) \approx p_{\boldsymbol{\theta}}(\boldsymbol{z}_n|\boldsymbol{x}_n)$. This motivates the *variational auto-encoder* (VAE) approach [Kingma and Welling, 2014; Rezende et al., 2014] which defines the q distribution as $q(\boldsymbol{z}) := q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, with the distribution often defined by a neural network:

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), diag(\sigma^2_{\boldsymbol{\phi}}(\boldsymbol{x}))), \quad \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \log \sigma_{\boldsymbol{\phi}}(\boldsymbol{x}) = \mathrm{NN}_{\boldsymbol{\phi}}(\boldsymbol{x}). \tag{12}$$

This allows us to define the VAE optimisation objective:

$$\boldsymbol{\phi}^*, \boldsymbol{\theta}^* = \arg\max \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}), \quad \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})]}_{:=\mathcal{L}(\boldsymbol{x}, \boldsymbol{\phi}, \boldsymbol{\theta})}]. \tag{13}$$

Analytic KL between factorised Gaussians

Given that both $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{z})$ are all factorised Gaussian distributions, the KL divergence term in (13) has an analytic form (assuming $\boldsymbol{z} \in \mathbb{R}^d$):

$$\mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})] = \frac{1}{2}\left(||\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})||_2^2 + ||\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x})||_2^2 - 2\langle \log \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}), \mathbf{1}\rangle - d\right). \tag{14}$$

To see this, let us assume two factorised distributions $p(\boldsymbol{z}) = \prod_{i=1}^{d} p(z_i)$ and $q(\boldsymbol{z}) = \prod_{i=1}^{d} q(z_i)$. Then the KL divergence from q to p can be written as a sum of KL divergences:

$$\begin{aligned}\mathrm{KL}[q(\boldsymbol{z})||p(\boldsymbol{z})] &= \mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{\prod_{i=1}^{d} q(z_i)}{\prod_{i=1}^{d} p(z_i)}\right] = \mathbb{E}_{q(\boldsymbol{z})}\left[\sum_{i=1}^{d} \log \frac{q(z_i)}{p(z_i)}\right] \\ &= \sum_{i=1}^{d} \mathbb{E}_{q(z_i)}\left[\log \frac{q(z_i)}{p(z_i)}\right] = \sum_{i=1}^{d} \mathrm{KL}[q(z_i)||p(z_i)].\end{aligned} \tag{15}$$

Then, assuming each $q(z_i)$ and $p(z_i)$ distributions are Gaussians: $q(z_i) = \mathcal{N}(z_i; \mu_i, \sigma_i^2)$, $p(z_i) = \mathcal{N}(z_i; 0, 1)$, we have the KL divergence as:

$$\begin{aligned}\mathrm{KL}[q(z_i)||p(z_i)] &= \mathbb{E}_{q(z_i)}\left[\log \frac{\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp[-\frac{1}{2\sigma_i^2}(z_i - \mu_i)^2]}{\frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2} z_i^2]}\right] \\ &= \mathbb{E}_{q(z_i)}\left[-\log \sigma_i - \frac{1}{2\sigma_i^2}(z_i - \mu_i)^2 + \frac{1}{2} z_i^2\right] \\ &= -\log \sigma_i - \frac{1}{2\sigma_i^2}\mathbb{E}_{q(z_i)}\left[(z_i - \mu_i)^2\right] + \frac{1}{2}\mathbb{E}_{q(z_i)}\left[(z_i - \mu_i)^2 - \mu_i^2 + 2\mu_i z_i\right] \\ &= -\log \sigma_i - \frac{1}{2} + \frac{1}{2}[\sigma_i^2 + \mu_i^2].\end{aligned} \tag{16}$$

Writing $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}) = [\mu_1, ..., \mu_d]$, $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}) = [\sigma_1, ..., \sigma_d]$, we can sum up the KL divergence (16) over $i = 1, ..., d$ and write the resulting $\mathrm{KL}[q(\boldsymbol{z})||p(\boldsymbol{z})]$ as (14). This is done by noticing e.g. $||\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})||_2^2 = \sum_{i=1}^{d} \mu_i^2$ and $\sum_{i=1}^{d} \log \sigma_i = \langle \log \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}), \mathbf{1} \rangle$.

Monte Carlo estimation

The VAE objective $\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$ in (13) is still intractable since the expectation computation $\mathbb{E}_{q_{\boldsymbol{\phi}}}[\cdot]$ requires evaluating neural network transformations for all possible $\boldsymbol{z}$. Monte Carlo (MC) estimation comes into rescue, as we can replace the expectation with MC approximations:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \approx \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}), \quad \boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}). \tag{17}$$

By doing so, the gradient of the objective w.r.t. $\boldsymbol{\theta}$ can be estimated as

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\phi}, \boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}), \quad \boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}). \tag{18}$$

It remains to compute the gradient of the objective w.r.t. $\boldsymbol{\phi}$

$$\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\phi}, \boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \nabla_{\boldsymbol{\phi}} \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})]. \tag{19}$$

While the gradient w.r.t. the KL term tractable (by differentiate eq. (14) w.r.t. $\boldsymbol{\phi}$), MC approximation is still required for the first term in (19).

Reparameterisation trick [2]

The MC approximation to $\nabla_{\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})]$ is further assisted by the reparameterisation trick [Kingma and Welling, 2014; Rezende et al., 2014]. Note that the sampling procedure of a Gaussian variable is the following:

$$\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \quad \Leftrightarrow \quad \boldsymbol{z} = \boldsymbol{\mu}_{\boldsymbol{\phi}} + \boldsymbol{\sigma}_{\boldsymbol{\phi}} \odot \boldsymbol{\epsilon}, \ \boldsymbol{\epsilon} \sim \mathcal{N}(\epsilon; \mathbf{0}, \mathbf{I}), \tag{20}$$

[2] Please note that the reparameterisation trick is not the only method to enable MC estimation of VAE gradients w.r.t. ϕ even when we use Gaussian q distributions. *If interested, see e.g., Section 2.2.3 of this note.

with $\odot$ denoting element-wise product. Writing $\pi(\boldsymbol{\epsilon}) := \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$ and $T_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}) := \boldsymbol{\mu}_{\boldsymbol{\phi}} + \boldsymbol{\sigma}_{\boldsymbol{\phi}} \odot \boldsymbol{\epsilon}$, we have, by LOTUS,

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] = \mathbb{E}_{\pi(\boldsymbol{\epsilon})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|T_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}))], \tag{21}$$

$$\nabla_{\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] = \mathbb{E}_{\pi(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\phi}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|T_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}))] = \mathbb{E}_{\pi(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\phi}} \boldsymbol{z} \nabla_{\boldsymbol{z}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})|_{\boldsymbol{z}=T_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{\epsilon})}]. \tag{22}$$

Then with MC estimation:

$$\mathbb{E}_{\pi(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\phi}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|T_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon}))] \approx \nabla_{\boldsymbol{\phi}} \boldsymbol{z} \nabla_{\boldsymbol{z}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})|_{\boldsymbol{z}=T_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{\epsilon})}, \ \boldsymbol{\epsilon} \sim \pi(\boldsymbol{\epsilon}). \tag{23}$$

Combined with eq. (18) and mini-batch training, one can compute an MC estimation of the VAE objective (13) as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) \approx \frac{1}{M} \sum_{m=1}^{M} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_m|T_{\boldsymbol{\phi}}(\boldsymbol{x}_m, \boldsymbol{\epsilon}_m)) - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}_m|\boldsymbol{x}_m)||p(\boldsymbol{z}_m)], \\ \boldsymbol{x}_1, ..., \boldsymbol{x}_m \sim \{\boldsymbol{x}_n\}^M, \ \boldsymbol{\epsilon}_1, ..., \boldsymbol{\epsilon}_M \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \end{aligned} \tag{24}$$

and apply e.g. automatic differentiation to obtain the (MC estimation of) gradient of the VAE objective w.r.t. parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

1.4 Conditional VAE

For conditional generative models, the goal is to generate data (e.g. images) conditioned on additional information. Such additional information can be class labels (which is discrete) or the viewing angle for the image (which is continuous). Mathematically, this corresponds to learning a generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y})$ which approximates the data distribution $p_{\mathrm{data}}(\boldsymbol{x}|\mathbf{y})$. Here $\boldsymbol{x}$ is the random variable for data (e.g. images) and $\mathbf{y}$ is the random variable corresponding to the additional information (e.g. label or viewing angle).

For the design of the generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y})$, we use a conditional LVM as follows:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, \mathbf{y}) p(\boldsymbol{z}) d\boldsymbol{z}, \tag{25}$$

See Figure 1 (b) for a visualisation of the graphical model. Often we set $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathrm{I})$. If $\boldsymbol{x}$ is continuous, then we can define e.g.

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, \mathbf{y}) = \mathcal{N}(\boldsymbol{x}; G_{\boldsymbol{\theta}}(\boldsymbol{z}, \mathbf{y}), \sigma^2 \mathrm{I}), \tag{26}$$

with $G_{\boldsymbol{\theta}}(\boldsymbol{z}, \mathbf{y})$ defined by a neural network that takes both $\boldsymbol{z}$ and $\mathbf{y}$ as inputs. Similar to VAEs, learning is done by maximising a variational lower-bound:

$$\boldsymbol{\phi}^*, \boldsymbol{\theta}^* = \arg\max \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}), \ \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{p_{\mathrm{data}}(\boldsymbol{x},\mathbf{y})}[\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, \mathbf{y})] - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, \mathbf{y})||p(\boldsymbol{z})]], \tag{27}$$

$$\mathbb{E}_{p_{\mathrm{data}}(\boldsymbol{x},\mathbf{y})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y})] \geq \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}). \tag{28}$$

Although in principle the choice of the q distribution is flexible (since the variational lower-bound holds for almost any q distribution satisfying mild conditions, see Section 1.2), using $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, \mathbf{y})$ and parameterising it with flexible neural networks would return the best posterior approximation. Using Bayes' rule

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}, \mathbf{y}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, \mathbf{y}) p(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y})}, \tag{29}$$

we can show that maximising variational lower-bound w.r.t. q is also equivalent to minimising the KL divergence $\mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})]$:

$$\begin{aligned}
&\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y}) - \left(\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z},\mathbf{y})] - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})||p(\boldsymbol{z})]\right) \\
=&\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\mathbf{y})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z},\mathbf{y})p(\boldsymbol{z})}\right] \\
=&\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}\left[\log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})}\right] = \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})].
\end{aligned} \tag{30}$$

Therefore if we were to replace $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},\mathbf{y})$ with $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, then the optimal solution does not return the exact posterior approximation, unless the learned generator is degenerate: $G_{\boldsymbol{\theta}}(\boldsymbol{z},\mathbf{y}) = G_{\boldsymbol{\theta}}(\boldsymbol{z})$. In such case the $\mathbf{y}$ information is ignored (i.e. $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z},\mathbf{y}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$) and the model is no longer a conditional generative model.

1.5 *Practical interpretations & KL annealing

Comparisons with auto-encoders

Looking at the likelihood part of the VAE objective (13), under Gaussian likelihood assumption we have (by using the reparam. trick)

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[-\frac{1}{2\sigma^2}||\boldsymbol{x} - G_{\boldsymbol{\theta}}(T_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{\epsilon}))||_2^2\right] + \text{const.} \tag{31}$$

On the other hand, an auto-encoder contains a pair of encoder $E_{\boldsymbol{\phi}}(\cdot)$ and decoder $D_{\boldsymbol{\theta}}(\cdot)$ which are trained using e.g. ℓ_2 reconstruction loss:

$$\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[||\boldsymbol{x} - D_{\boldsymbol{\theta}}(E_{\boldsymbol{\phi}}(\boldsymbol{x}))||_2^2]. \tag{32}$$

Comparing the reconstruction loss of the auto-encoder training objective to (31), we see that VAEs can be viewed from a viewpoint of *stochastic* auto-encoder. Architecture-wise, the main difference is the usage of stochastic encoder $T_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{\epsilon})$ that injects random noise $\boldsymbol{\epsilon}$ to the encoding of $\boldsymbol{x}$. Training objective-wise, the VAE objective has the extra $\mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})]$ term which regularises the q distribution towards the prior $p(\boldsymbol{z})$. When $p(\boldsymbol{z})$ is non-degenerate (e.g. $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z};\mathbf{0},\mathbf{I})$) the resulting $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ at optimum is non-degenerate as well, i.e. $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}) > \mathbf{0}$.

KL annealing

Practitioners sometimes find that training VAEs with the original variational lower-bound objective (13) leads to under-fitting issues, in such case often the reconstructed images using the model are blurry. A practical strategy to alleviate this is to introduce a "KL annealing" coefficient β and optimise the $\boldsymbol{\theta},\boldsymbol{\phi}$ parameters using the following objective:

$$\boldsymbol{\phi}^*,\boldsymbol{\theta}^* = \arg\max \mathcal{L}(\boldsymbol{\phi},\boldsymbol{\theta},\beta), \quad \mathcal{L}(\boldsymbol{\phi},\boldsymbol{\theta},\beta) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}[\underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \beta\mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})]}_{:=\mathcal{L}(\boldsymbol{x},\boldsymbol{\phi},\boldsymbol{\theta},\beta)}]. \tag{33}$$

If using $0 < \beta < 1$, this objective introduces less regularisation for the $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to be close to the prior $p(\boldsymbol{z})$. In particular when $\beta = 0$, it results in a stochastic auto-encoder which is trained by the reconstruction loss only. Since stochasticity in $\boldsymbol{z}$ naturally degrades the quality of reconstruction, training with reconstruction loss only will drive $\boldsymbol{\phi}$ towards making $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}) \to \mathbf{0}$ for any $\boldsymbol{x}$, which also means $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \to \delta(\boldsymbol{z} = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))$. In such case the resulting model is simply an auto-encoder which cannot be used directly as a generative model for new images.

We should also emphasise that for $\beta < 1$, $\mathcal{L}(\boldsymbol{x},\boldsymbol{\phi},\boldsymbol{\theta},\beta)$ is no longer a lower-bound for $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$, and the training objective (33) cannot be well justified using (approximate) MLE for learning $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx$

$p_{\text{data}}(\boldsymbol{x})$. In fact for small β the learned generative distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}$ can be very different from $p_{\text{data}}(\boldsymbol{x})$, again explaining why generation quality can be worse when using such small β values. Therefore β needs to be carefully chosen to achieve the trade of between good reconstruction & good generation. Another strategy is to use different β_t values for different training epochs $t = 0, ..., T$; a recommended recipe is to select increasing values $0 \leq \beta_1 \leq ... \leq \beta_T$.

Sometimes $\beta > 1$ values are also used but for a different purpose. Although there is no theoretical guarantee, existing research shows that empirically, with factorised prior $p(\boldsymbol{z})$ and $\beta > 1$, one can train a VAE to obtain a *disentangled representation*, so that controlled generation can be achieved by varying different dimensions of the $\boldsymbol{z}$ variable [Higgins et al., 2017].

References

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representations*.

Kullback, S. (1959). *Information theory and statistics*. John Wiley & Sons.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1278–1286.