Variational Autoencoder notes

Eugene Francisco

January 2026

We have big data x and we assume that x has some small dimensional latent representation as z. Typically our goal will be to generate examples of x given a sample of z. For example, we might first sample z from $p(z)$ (which we typically fix in advance as Gaussian) and then sample x from $p_\theta(x|z)$. In the first place, we want to maximize the log-likelihood of $p(x)$, which can be rewritten as

$$p(x) = \int p(x|z)p(z)dz$$

but we don't know what $p(x|z)$. **VAE is one technique to learn** $p_\theta(x|z) \approx p(x|z)$ and to maximize the likelihood above.

Motivation:

Suppose we want to sample from $q(x)$. We have an easy way to sample from q but we cannot actually compute $q(x)$ for any particular x. We wish however to estimate what q is, so we introduct a distribution p_θ which we control. **We want $p_\theta(x)$ to be as close to $q(x)$ as possible.**

One way to do this is to minimize $\mathrm{KL}(q(x) \parallel p_\theta(x))$. We can rewrite the KL as

$$\begin{aligned}\mathrm{KL}(q(x) \parallel p_\theta(x)) &= \mathbb{E}_{x\sim q}\left[\log \frac{q(x)}{p_\theta(x)}\right] \\ &= -\mathbb{E}_{x\sim q}[\log p_\theta(x)] - H(q).\end{aligned}$$

This tells us that, to minimize the $\mathrm{KL}(q \parallel p_\theta)$, we can sample $x \sim q(x)$, calculate the log-likelihood of x under p_θ and use this to maximize blue. In particular, this means that **to minimize $\mathrm{KL}(q \parallel p_\theta)$, we need to be able to sample from q and we need to be able to compute $p_\theta(x)$.**

Connection to VAE:

In VAE, we still want $p_\theta(x)$ to be close to $q(x)$. Except that we introduce a latent variable z. In particular, we assume that z is related to x according to Figure 1. It may seem like we have made the problem harder by introducing the new latent. Indeed now we can no longer compute $p_\theta(x)$ because we would have to marginalize over Z. However, there are two things here which matter a lot and wil help us:

(i) The joint $p_\theta(x, z)$ is easy to compute and the joint $q_\phi(x, z)$ is easy to sample from.

(ii) The KL divergence between $q_\phi(x, z)$ and $p_\theta(x, z)$ is an upper bound for the KL between $q(x)$ and $p_\theta(x)$.

Item (i) means that we can easily minimize $\mathrm{KL}(q_\phi(x, z) \parallel p_\theta(x, z))$. Item (ii) means that minimizing $\mathrm{KL}(q_\phi(x, z) \parallel p_\theta(x, z))$ means (hopefully) that we will minimize $\mathrm{KL}(q(x) \parallel p_\theta(x))$. More on part (ii), the exact gap between the joint KL and the marginal KL is

$$\mathrm{KL}(q_\phi(x, z) \parallel p_\theta(x, z)) = \mathrm{KL}(q(x) \parallel p_\theta(x)) + \mathbb{E}_{x\sim q(x)}[\mathrm{KL}(q_\phi(z|x) \parallel p_\theta(z|x))]. \qquad (\star)$$

The blue is what we are practically capable of minimizing. The red is what we want to minimize. And the orange is the gap between the two, also called the *ELBO* gap. **When we minimize , we want to make sure we are minimizing red on the RHS.** We will perform exactly this minimization point-wise on a particular x, and the ELBO will allow us to do so while minimizing blue in $(\star)$.

$$\log p_\theta(x) = \mathbb{E}_{z\sim q(z|x)}[\log p_\theta(x|z)] + \mathrm{KL}(q_\phi(z|x) \parallel p(z)) - \mathrm{KL}(q_\phi(z|x)\|p_\theta(z|x)).$$

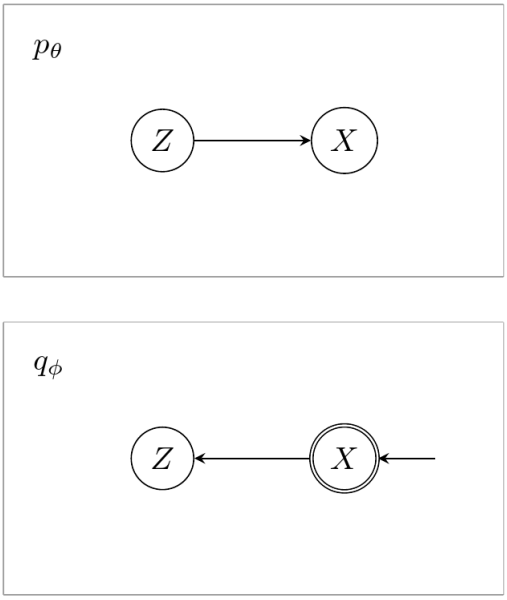

Figure 1: Image credit Jack Hseih.

Steps to VAE:

0.1 ELBO

This is difficult in practice so instead we maximize the a lower bound on the log-likelihood (the algebra to show this is a lower bound is some light algebra and Jensen's inequality)

$$\log p_\theta(x) \geq \mathrm{ELBO}(\theta, \phi) = \mathbb{E}_{z \sim q_\phi(z|x)}\big(\log p_\theta(x|z)\big) - \mathrm{KL}\big(q_\phi(z|x) \parallel p(z)\big). \tag{0.1.1}$$

This ELBO works for any q and any θ, ϕ. How is this ELBO estimated in practice? Pick x from your dataset. Sample a $z \sim q_\phi(z|x)$. For the first term of the ELBO (called the *reconstruction*) compute the log-likelihood of x given z. For the second term, compute $\log(q(z|x)/p(z))$ (in practice the KL of two Gaussians has closed form expression). Repeat this many times over all the x in your dataset.

The ELBO can also be rewritten using some algebra as

$$\log p_\theta(x) - \mathrm{KL}\big(q_\phi(z|x) \parallel p_\theta(z|x)\big)$$

which makes it clearer that in maximizing this lower bound, we are making $q_\phi(z|x)$ as close as possible to $p_\theta(z|x)$. (This formulation isn't very useful because the first $\log$ term can't be computed).

We typically will assume that $p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), I)$ (though can be different for discrete) and that $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma^2_\phi(x))$ (which is pretty much always the case) ; this will be very convenient later on when we have to take gradients w.r.t ϕ.

0.2 Choose family of conditionals

Choose a family $\mathcal{Q} = \{q_\phi(z) | \phi \in \mathbb{R}^Q\}$, a family of distributions of z. Recall from section (0.1) that we want to pick q_{ϕ^*} here which minimizes $\mathrm{KL}(q \parallel p_\theta(z|x))$.

0.3 Amortizing

Suppose that we are given dataset $\{x_n\}_{n=1}^N$. In the olden days, we may train one λ_i per x_i so that $q_{\lambda_i}(x_i) \approx p_\theta(z|x)$. But this doesn't generalize to out-of-sample xs and means the number of parameters scales with the dataset.

Amortizing means that we train the parameters ϕ so that $q_\phi(x) \approx p_\theta(z|x)$. In other words, we are choosing the q_{ϕ^*} we want for the previous two sections but ϕ^* learns to distinguish between ys too. Recalling our assumption that $q_\phi(z|x) \sim \mathcal{N}(\mu_\phi(x), \sigma^2_\phi(x))$, this means that, for example, ϕ could be the parameters to a neural network that output the mean and covariance matrix of z.

0.4 Taking Gradients and Reparametrizing Trick

Recall we wish to maximize (0.1.1). This means we need to take its gradient w.r.t θ and ϕ. For θ, this is easy enough, since

$$\nabla_\theta(\text{ELBO})(\theta, \phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\big(\nabla_\theta \log p_\theta(x|z)\big).$$

So to get an unbiased estimate of the gradient at x (one of our datapoints), just sample $z \sim q_\theta(z|x)$, then get the gradient of $\log p_\theta(x|z)$ w.r.t. θ.

It is not so easy for ϕ because ϕ shows up in the expectation. But recall that we assumed $q_\phi(z|x) \overset{d}{=} \mathcal{N}(\mu_\phi(x), \sigma^2_\phi(x))$ (the last entry is a diagonal covariance matrix). This means we can write

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon$$

where $\epsilon \sim \mathcal{N}(0, I)$ and now the randomness comes from ϵ instead of ϕ. Doing so allows us to write the gradient now as

$$\begin{aligned}\nabla_\phi(\text{ELBO})(\theta, \phi) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}(\nabla_\phi \log p_\theta(x|\mu_\phi(x) + \sigma_\phi(x) \odot \epsilon)) \\ - \nabla_\phi \text{KL}\big(q_\phi(z|x) \parallel p(z)\big).\end{aligned}$$