Variational Auto Encoders

Nils Hallerfelt

October 2022

General setting

From a probability model perspective, a Variational Autoencoder (VAE) is some model that first sample from a latent space, then samples from the output space on the conditional probability of the sample from the latent space:

$$z_i \sim p(z) \quad x_i \sim p(x|z)$$

where $p(z)$ is the prior, and $p(x|z)$ is the conditional likelihood of getting x given z. The joint probability of the model can be expressed as:

$$p(x,z) = p(x|z)p(z)$$

We can now consider inference in the model. By Bayes Theorem we get that:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Unless the latent space is small so that z only takes on a few values, the evidence $p(x)$ is intractable making the whole expression hard to compute. We also don't really know how to sample from this. To work around this issue we make use of variational inference. We pick some family of distributions $q_\phi(z|x)$ that is parameterized by ϕ, and try to find a good approximation of $p(z|x)$ in $q_\phi(z|x)$. To measure the difference between $q_\phi(z|x)$ and $p(z|x)$ we can use Kullback-Leibler divergence, which has some nice properties that come in handy (such as non-negativity and the use of logarithms to determine the divergence).

$$\begin{aligned}
&D_{KL}(q_\phi(z|x)||p(z|x)) \\
=&\mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x) - \log p(z|x)] \\
=&\mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x) - \log p(x,z)] + \log p(x)
\end{aligned}$$

From this we can get some insight. By moving the expectation to the other side of the equality we get:

$$\begin{aligned}
&\log p(x) \\
=&\mathbb{E}_{q_\phi(z|x)}[-\log q_\phi(z|x) + \log p(x,z)] + D_{KL}(q_\phi(z|x)||p(z|x)) \\
=&\text{ELBO}(\phi) + D_{KL}(q_\phi(z|x)||p(z|x))
\end{aligned}$$

Since the KL-divergence is non-negative, we have that the expectation term is a lower bound on $\log p(x)$. Also, we can see that maximizing the lower bound is equivalent to minimizing the KL-divergence! This is a good thing, computing the KL-divergence between the approximate posterior and the exact posterior would be intractable, because of the evidence. Since maximizing the ELBO(ϕ) seems like a good idea, we put it alone on the LHS. After some simple rewriting the intractable $p(x)$ can be cancelled, giving us the following:

$$\text{ELBO}(\phi) = \mathbb{E}_{q_\phi(z|x)}[\log p(x|z)] - D_{KL}(q_\phi(z|x)||p(z))$$

Introducing Neural Networks

From a neural network perspective, we can view the VAE as neural network consisting of an encoder and one decoder. The encoder takes some observed data as input and gives some parameters ϕ as output. The network is constructed so that the parameters ϕ parameterize the likelihood of getting z in the latent space, given some input data point x. The decoder samples from $q_\phi(z|x)$ and reconstructs it back to the original space, defining the parameters θ.

$$\begin{aligned}
&x \xrightarrow{ENCODER} \phi \\
&z \sim p(z|x) \\
&z \xrightarrow{DECODER} \theta
\end{aligned}$$

We thus let the encoder define the parameterized distribution $q_\phi(z|x)$. In this setting we can write the ELBO as function over the parameters we want to learn:

$$\text{ELBO}(\phi, \theta) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z))$$

The expectation term describes the reconstruction loss, and the KL-divergence is a regularization term that enforces the proposed posterior distribution to be similar to the prior distribution enforcing some structure on the latent space. We know that the ELBO is a lower bound that will not overshoot as was shown in the previous section. However, in implementations we would instead like to formaulate this as a loss function that we want to minimize:

$$L(\phi, \theta) = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + D_{KL}(q_\phi(z|x)||p(z)) \tag{1}$$

In the coming sections we will go through how this can be optimized in two different scenarios.

VAE with continuous latent space and binary observed space

We will now consider an example of when the input data is binary, and the latent space is continuous and standard multivariate Gaussian. We thus fix $p(z) = N(0, 1)$ for each coordinate in the latent space. We let the encoder parameterize the conditional $q_\phi(z|x)$ and has as output the mean μ and the logarithmic variance $\log \sigma^2$ (to ensure positive variance). The decoder samples from the parameterized latent space and runs the sampled points from the latent space through its network that parameterize the conditional $p_\theta(x|z)$. The goal is to minimize (1), so we want to back-propagate through the network using stochastic gradient descent (SGD) to update the ϕ and θ parameters. That means that we must be able to take the gradient with respect to both ϕ and θ.

Optimization

We want to take the gradient of the (1), however this is not possible straight away. We write (1) in terms of expectations:

$$L(\phi, \theta) = \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x) - \log p(z) - \log p_\theta(x|z)]$$

Getting the gradient over θ is not problematic, so we can already back propagate through the decoder. However it is not possible to directly take the gradient with respect to ϕ, as we do the sampling over ϕ which makes it "disappear". To be able to take the gradient we use the reparametrization trick over Gaussian distributions:

$$z = \mu + \sigma \cdot \epsilon, \quad \epsilon \sim N(0, 1)$$

The sampled z is now deterministic with respect to ϕ, and the variation comes in with the sampling of ϵ.

$$\nabla_\phi L(\phi, \theta) = \nabla_\phi \mathbb{E}_{\epsilon \sim N(0,1)}[\log q_\phi(z|x)]$$

Which will allow back propagation through the decoder.

VAE with binary latent space and binary observed space

The main difference now is that we have a Bernoulli distributed latent space: $p(z) = Ber(0.5)$. Thus any sample from the latent space will be binary. We let the decoder parameterize the posterior conditional $q_\phi(z|x)$, and let the decoder parameterize the conditional $p_\theta(x|z)$. The decoder will now output an m-dimensional parameterization for each input. Since parameters that are output from the encoder should describe some Bernoulli probability, they are made to be in the intervall (0, 1) by taking the sigmoid. Once again we wish to backpropagate through the network by using SGD.

1 Optimization

Once again there is no real issue in dealing with the back propagation of the decoder, so we focus on how we can get a gradient over ϕ when the latent space is binary. Its not possible to apply any standard reparametrization trick (from my understanding, however I cannot motivate why this would not be possible). Instead we use likelihood ratio gradient. The calculations are a bit long...

$$\begin{aligned}
&\nabla_\phi L(\phi,\theta)\\
=&\nabla_\phi \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x) - \log p(z) - \log p_\theta(x|z)]\\
=&\nabla_\phi \sum_z q_\phi(z|x)(\log q_\phi(z|x) - \log p(z) - \log p_\theta(x|z)\\
=&\nabla_\phi \sum_z q_\phi(z|x)\log q_\phi(z|x) - \nabla_\phi \sum_z q_\phi(z|x)(\log p(z) + \log p_\theta(x|z))\\
=&\mathbb{E}_{q_\phi(z|x)}[\frac{\nabla_\phi q_\phi(z|x)}{q_\phi(z|x)}] - \sum_z \frac{q_\phi(z|x)}{q_\phi(z|x)}\nabla_\phi q_\phi(z|x)(\log p(z) + \log p_\theta(x|z))\\
=&\nabla_\phi \sum_z q_\phi(z|x) - \mathbb{E}_{q_\phi(z|x)}[\frac{\nabla_\phi q_\phi(z|x)}{q_\phi(z|x)}(\log p(z) + \log p_\theta(x|z))]\\
=&0 - \mathbb{E}_{q_\phi(z|x)}[\nabla_\phi \log q_\phi(z|x)(\log p(z) + \log p_\theta(x|z))]
\end{aligned}$$

Which we can sample and estimate. This way we are able to back propagate through the network.