Conditional KL-divergence in Hierarchical VAEs

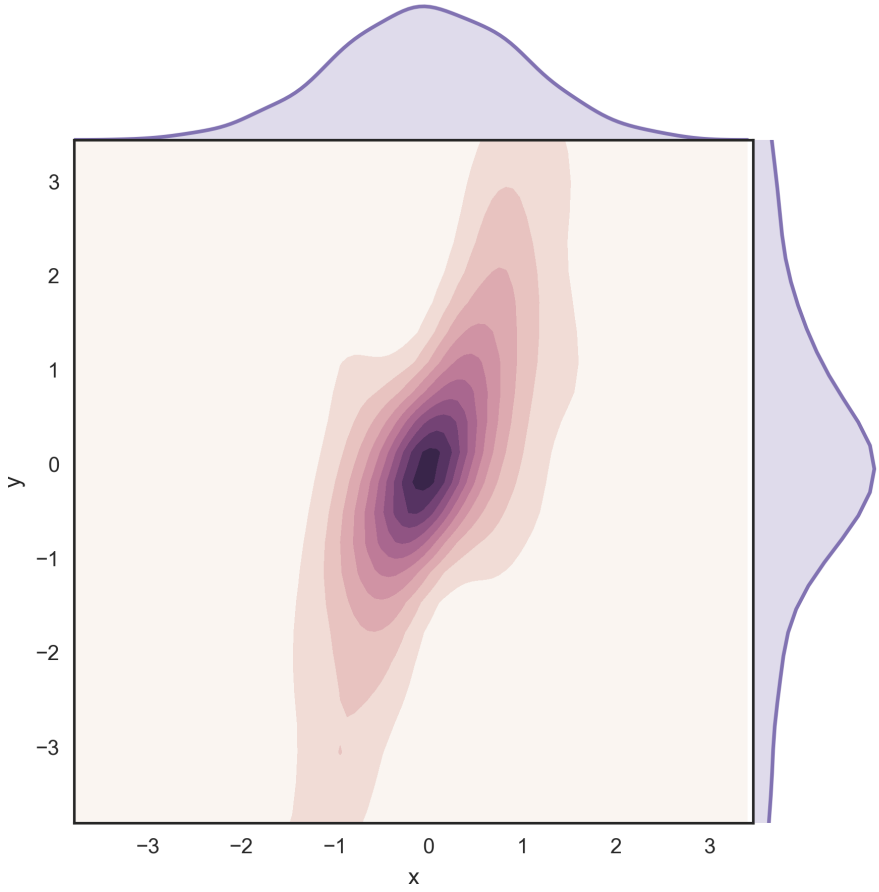
Sep 10, 2017

Inference is hard and often computationally expensive. Variational Autoencoders (VAE) lead to an efficient amortised inference scheme, where amortised means that once the model is trained (which can take a long time), the inference has constant computational complexity. Variational Autoencoders (VAE) learn the approximate posterior distribution $q(z \mid x)$ over some latent variables z by maximising a lower bound on the true data likelihood $p(x)$. This is useful, because the latent variables explain what we see (x), and often in a concise form.

One problem with VAEs is that we have to assume some functional form for q. While the majority of papers take the Gaussian distribution with a diagonal covariance matrix, it has been shown that more complex (*e.g.* multi-modal) approximate posterior distributions can improve the quality of the model, with a good example being the normalizing flows paper.

Normalizing flows take a simple probability distribution (here: a Gaussian) and apply a series of invertible transformations to get a more complicated distribution. While useful, the resulting distribution is limited by the form of the transforming functions, which in this case have to be invertible. Another way of achieving the same goal is to split the latent variables into two groups $z = \{u, v\}$, say, and express the joint distribution as $q(z) = q(u, v) = q(u \mid v)q(v)$ by using the product rule of probability. The conditional distribution $q(u \mid v)$ can depend on v in a highly non-linear fashion (it can be implemented as a neural net). Even though both the marginal $q(v)$ and the conditional $q(u \mid v)$ can be Gaussians, their joint might be highly non-Gaussian. Consider the below example and the resulting density plot (in the plot $x = v$ and $y = u$).

$$\begin{aligned} q(v) &= \mathcal{N}(v \mid 0, I) \\ q(u \mid v) &= \mathcal{N}(u \mid Fv, Fvv^T F^T + \beta I), \end{aligned} \tag{1}$$



The above density plot shows a highly non-Gaussian probability distribution. $x \sim q(v)$ is, in fact, a Gaussian random variable, but $y \sim q(u \mid v)$ is not, since its variance is not constant and depends on its mean: the variance increases with the increasing distance from the mean, resulting in heavy tails. In the above plot, $q(u \mid v)$ is obtained as a simple transformation of $v \sim q(v)$, which could be implemented by a one-layer neural net; see Importance Weighted Autoencoders for a more general example. This simple scheme results in a VAE with a hierarchy of latent variables and can lead to much more complicated posterior distributions, but it also leads to a more complicated (and ambiguous) optimisation procedures. Let me elaborate.

As part of the variational objective, the learning process is optimising the Kullback-Leibler divergence $KL[q \mid p]$ between the approximate posterior q and a prior over the latent variables p. KL is an asymmetric measure of similarity between two probability distributions q and p that is often used in machine learning. It can be interpreted as the information gain from using q instead of p, or in the context of coding theory, the extra number of bits to code samples from q by using p. You can read more about information measures in this cheat sheet. It is defined as

$$KL[q(z) \mid\mid p(z)] = \int q(z) \log \frac{q(z)}{p(z)} \mathrm{d}z. \tag{2}$$

If we split the random variable z into two disjoint sets $z = \{u, v\}$ as above, the KL factorises as

$$\begin{aligned} KL[q(u,v) \mid\mid p(u,v)] &= \iint q(u,v) \log \frac{q(u,v)}{p(u,v)} \mathrm{d}u\mathrm{d}v \\ &= \int q(v) \log \frac{q(v)}{p(v)} \mathrm{d}v + \int q(v) \int q(u \mid v) \log \frac{q(u \mid v)}{p(u \mid v))} \mathrm{d}u\mathrm{d}v \\ &= KL[q(v) \mid\mid p(v)] + KL[q(u \mid v) \mid\mid p(u \mid v)], \end{aligned} \tag{3}$$

where $KL[q(u \mid v) \mid\mid p(u \mid v)] = \mathbb{E}_{q(v)}\left[\tilde{KL}[q(u \mid v) \mid\mid p(u \mid v)]\right]$ is known as the conditional KL-divergence, with

$$\tilde{KL}[q(u \mid v) \mid\mid p(u \mid v) = \int q(u \mid v) \log \frac{q(u \mid v)}{p(u \mid v))} \mathrm{d}u. \tag{4}$$

The conditional KL-divergence amounts to the expected value of the KL-divergence between conditional distributions $q(u \mid v)$ and $p(u \mid v)$, where the expectation is taken with respect to $q(v)$. Since KL-divergence is non-negative, both terms are non-negative. KL is equal to zero only when both probability distributions are exactly equal. The conditional KL is equal to zero when both conditional distributions are exactly equal on the whole support defined by $q(v)$. This last bit makes it difficult to optimise with respect to the parameters of both distributions.

Let $q(z) = q_\psi(u, v) = q_\phi(u \mid v) q_\theta(v)$, such that the posterior is parametrised by $\psi = \begin{bmatrix} \phi \\ \theta \end{bmatrix}$. If we look at the gradient of the KL divergence, we have that

$$\nabla_\psi KL[q_\psi(u, v) \mid\mid p(u, v)] = \begin{bmatrix} \nabla_\phi KL[q_\psi(u, v) \mid\mid p(u, v)] \\ \nabla_\theta KL[q_\psi(u, v) \mid\mid p(u, v)] \end{bmatrix}, \tag{5}$$

with

$$\nabla_\phi KL[q_\psi(u, v) \mid\mid p(u, v)] = \nabla_\phi KL[q_\phi(u \mid v) \mid\mid p(u \mid v)], \tag{6}$$

$$\nabla_\theta KL[q_\psi(u, v) \mid\mid p(u, v)] = \nabla_\theta KL[q_\theta(v) \mid\mid p(v)] + \nabla_\theta KL[q_\phi(u \mid v) \mid\mid p(u \mid v)], \tag{7}$$

where the gradient with respect to the parameters of the lower-level distribution $q_\theta(v)$ is comprised of two components. The second component is problematic. Let's have a closer look:

$$\begin{aligned} \nabla_\theta KL[q_\phi(u \mid v) \mid\mid p(u \mid v)] &= \nabla_\theta \mathbb{E}_{q_\theta(v)}\left[\tilde{KL}[q_\phi(u \mid v) \mid\mid p(u \mid v)\right] \\ &= \mathbb{E}_{q_\theta(v)}\left[\tilde{KL}[q_\phi(u \mid v) \mid\mid p(u \mid v)] \nabla_\theta \log q_\theta(v)\right], \end{aligned} \tag{8}$$

where in the second line we used the log-derivative trick (suggested here by Max Soelch, thanks!). This score-function formulation makes it clear that following the (negative, as in SGD) gradient estimate maximises the probability of samples for which the conditional-KL divergence has the lowest values. In particular, it might be easier to change the support $q_\theta(v)$ to a volume where both conditionals have very small values instead of optimising $q_\phi(u \mid v)$. From my experience, it happens especially when the value of the conditional KL is much bigger than the value of the first KL term.

An alternative approach would be to optimise the conditional-KL only with respect to the parameters of the distribution inside the expectation: ϕ. That would result in the following gradient equation:

$$\nabla_\psi KL[q_\psi(u, v) \mid\mid p(u, v)] = \begin{bmatrix} 0 \\ \nabla_\theta KL[q_\theta(v) \mid\mid p(v)] \end{bmatrix} + \begin{bmatrix} \nabla_\phi KL[q_\phi(u \mid v) \mid\mid p(u \mid v)] \\ 0 \end{bmatrix} \tag{9}$$

This optimisation scheme resembles the Expectation-Maximisation (EM) algorithm. In the E step, we compute the expectations, while in the M step we fix the parameters with respect to which the expectations were computed and we maximise with respect to the functions inside the expectation. In EM we do this, because maximum-likelihood with latent variables often does not have closed-form solutions. The motivation here is to make the optimisation more stable.

I wrote this blog post, because I have no idea whether this *changed* optimisation procedure is justified in any way. What do you think? I would appreciate any comments.

Share this: Facebook Twitter Google+ LinkedIn

Adam Kosiorek
adamk@robots.ox.ac.uk
github
linkedin

Generative timeseries modelling, but also attention, memory and other cool tricks.