"Balancing Reconstruction Quality and Regularisation in Evidence Lower Bound for Variational Autoencoders" Notes

Feb 16

1 Notes:

1.1 High level:

Why: Variational autoencoders optimize two objectives: recreation quality and a continuous latent space. The first objective is done so via using a loss that penalizes bad recreations; the second is done so via urging the models' latent space to resemble a given (continuous) distribution.

These two losses commonly compete; purely optimizing one usually makes the other worse. As such, deciding which loss to weight more is a critical problem in variational inference. This is commonly done by an arbitrary weighting parameter, or by annealing the strength of one over time.

How: By representing the reconstruction loss proportional to the amount of variance in the data, one can reach a closed form solution for the weighting. If the variance of the data is high, weighting the reconstruction loss term high will merely lead to high loss with little further optimization. Instead of needlessly attempting to optimize this, we can instead attempt to develop a better latent space.

1.2 Medium level:

Competing learning objectives: The traditional ELBO loss of a variational autoencoder is:

$$\text{ELBO} := \underbrace{\mathbb{E}_{q_\phi(z|x_i)}\Big[\log p_\theta(\mathbf{x_i}|\mathbf{z})\Big]}_{\text{①Reconstruction likelihood}} - \underbrace{D_{\text{KL}}\Big[q_\phi(\mathbf{z}|\mathbf{x_i})||p(\mathbf{z})\Big]}_{\text{②Prior constraint}} \tag{1}$$

which is less than or equal to the true $\log p_\theta(\mathbf{x})$.

Naively optimizing these two objectives leads to competition between the two. If ② is 0, the posterior is entirely uninformative. If ① is the only thing optimized, the optimal solution is for the encoder to push the samples arbitrarily far apart. Despite the conflicting objectives, we wish to optimize both and get the best representation and recreation of the data as possible.

Dealing with it: In the case of neural networks as $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$, one solution is to anneal the weight of the reconstruction loss as training progresses. By allowing the variational autoencoder to select an expressive posterior at the start, then slowly easing it into a less expressive one, lets the encoder and decoder optimize θ and ϕ more efficiently.

However, this doesn't address the problem. This merely presents a solution for the symptoms of the problem. At its core, this approximates "caring" less about the variance in the data as training continues. The rate of annealing this loss weight is usually arbitrary, and a closed form solution would be more elegant and likely better.

1.3 Low level:

Example: Consider the case of a high-dimensional image dataset. There is a natural variance in the pixel values of each image, whether due to lighting, camera noise, or similar. As such, without perfectly overfitting to the training dataset, it is impossible to capture this variance. Continuing to attempt to optimize reconstruction quality will either lead to overfitting or no improvement at all. A better latent space is likely more useful.

Variance: Consider representing the by-pixel generative conditional distribution as:

$$p_\theta(\mathbf{x_i^k}|\mathbf{z}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(g_\theta^k(\mathbf{z}) - x_i^k)^2}{2\sigma^2} \right) \tag{2}$$

where σ^2 is common global variance parameter, reflecting the noise properties of the data; $g_\theta(\cdot)$ represents a nonlinear mapping from $\mathbf{z}$ to $\mathbf{x}$, and k represents the k-th dimension of $\mathbf{x}$.

Then assume

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{d} p_\theta(\mathbf{x_i^k}|\mathbf{z}), \tag{3}$$

where d is the dimension of $\mathbf{x}$. Therefore, $\log p_\theta(\mathbf{x}|\mathbf{z})$ can be computed as:

$$\log p_\theta(\mathbf{x}|\mathbf{z}) = -\frac{d}{2}\log(2\pi) - d\log\sigma - \frac{1}{2\sigma^2}\sum_{k=1}^{d}(g_\theta^k(\mathbf{z}) - \mathbf{x}_i^k)^2 \tag{4}$$

Notice that the terms inside the summation are element-wise square errors between the recreation and ground truth. The regularization term, $\frac{1}{2\sigma^2}$ appears naturally as a weighting factor of the sum-squared-error term. When $\log p_\theta(\mathbf{x}|\mathbf{z})$ is maximized, the $\log\sigma$ term regularizes σ from becoming too large, keeping the recreations from being arbitrarily bad.

Fixing the broken ELBO: The new ELBO is now:

$$\frac{d}{2}\log(2\pi)+d\log\sigma+\underbrace{\frac{1}{2\sigma^2}}_{\text{③ Weighting}}\mathbb{E}_{q_\phi(z|x_i)}\underbrace{\left[\sum_{k=1}^{d}(g_\theta^k(\mathbf{z})-\mathbf{x}_i^k)^2\right]}_{\text{①Reconstruction likelihood}}+\underbrace{D_{\mathrm{KL}}\Big[q_\phi(\mathbf{z}|\mathbf{x_i})||p(\mathbf{z})\Big]}_{\text{②Prior constraint}} \tag{5}$$

From (5), we can see that ③ represents the relative weighting between the KL and reconstruction loss. If we optimize the ELBO in (5) w.r.t. θ, ϕ, and σ, the model will automatically begin the weight the reconstruction error less as the variance increases. The optimal σ^2, denoted σ_*^2 can be interpreted as the amount of noise assumed to be in the data.

Finding σ_*^2: For fixed θ and ϕ, one can find σ_*^2 in closed form. Take the derivative of (5) w.r.t. σ and set it to 0.

$$\sigma_*^2=\frac{1}{d}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{k=1}^{d}(g_\theta^k(\mathbf{z})-\mathbf{x}_i^k)\right] \tag{6}$$

We can now iteratively update θ, ϕ, and σ for more stable learning.

Next steps: You might notice some strong assumptions in all of the above equations. Conditional independence of variance, same variance for all dimensions of $\mathbf{x}$, and Gaussian variance. The naive assumption for the same variance across all dimensions of $\mathbf{x}$ can be easily addressed, however, while the others must just be assumed to be true.

By replacing σ^2 in (2) by a variance estimation function, $\sigma_\theta^k(\mathbf{z})$, the corresponding $\log p_\theta(\mathbf{x}|\mathbf{z})$ function now becomes:

$$-\frac{d}{2}\log(2\pi)-\sum_{k=1}^{d}\left[\frac{1}{2(\sigma_\theta^k(\mathbf{x}))}(g_\theta^k(\mathbf{z})-\mathbf{x}_i^k)^2+\log\sigma_\theta^k(\mathbf{z})\right] \tag{7}$$

Which brings us to our final ELBO,

$$\frac{d}{2}\log(2\pi)+\mathbb{E}_{q_\phi(z|x_i)}\underbrace{\left[\sum_{k=1}^{d}\left[\frac{1}{2(\sigma_\theta^k(\mathbf{x}))}(g_\theta^k(\mathbf{z})-\mathbf{x}_i^k)^2+\log\sigma_\theta^k(\mathbf{z})\right]\right]}_{\text{①Reconstruction likelihood}}+\underbrace{D_{\mathrm{KL}}\Big[q_\phi(\mathbf{z}|\mathbf{x_i})||p(\mathbf{z})\Big]}_{\text{②Prior constraint}} \tag{8}$$

Optimizing (8) w.r.t. θ and ϕ optimizes reconstruction loss and latent space continuity dynamically and requires no further modification.

Empirically, estimating $\sigma_\theta^k(\mathbf{z})$ at the beginning of training leads to very poor results. It is recommended to first optimize the global variance then switch to the input dependent variance predictor.

2 Comments:

2.1 Aggregate posterior:

Even when optimizing the variance parameter, the learnt variational posterior may be significantly difference from the prior. In this case, it may be a good idea to estimate the aggregate posterior (AP) via Monte Carlo sampling, approximate it using a Gaussian-Mixture-Model (GMM), then sample from that.

In the case of artificial neural networks, this makes a large amount of empirical sense. The decoder, $p_\theta(\mathbf{x}|\mathbf{z})$, never receives any samples from the prior, $p(\mathbf{z})$. The decoder *only* receives samples from the encoder, which is assumed to be roughly equal to the prior. When this is not the case, using a GMM approximation of the AP leads to better results, although the new approximate AP is biased towards the distribution of the training data.

2.2 Results:

Lower (better) FID scores when compared to traditional VAEs, β-VAEs, DIP-VAE and WEAs. Most of the results seem to be had in low dimension (MNIST, fashionMNIST). Still improvement in medium dimension (celebA).