Auto-Encoding Variational Bayes

Dillon Laird

Introduction

We implement an Auto-Encoding Variational Bayes model and try and reproduce some of the results in (Kingma et. al., 2013). We start with an overview of the auto-encoding variational Bayes model and then describe experimental results and technical details of our implementation.

In our problem setting we are given some dataset $X = \{x^{(i)}\}_{i=1}^N$ of N i.i.d. samlpes of some random variable. We assume this data was generated from some latent random variabel z where $z \sim p_{\theta^*}(z)$ some prior distribution, and $x \sim p_{\theta^*}(x|z)$ some conditional distribution.

In order to model this process in a tractable way we use variational inference. We introduce a new distribution $q_\phi(z|x)$, with variational parameters ϕ, to approximate the true posterior $p_\theta(z|x)$ and call this the encoder. We also refer to $p_\theta(x|z)$ as the decoder. We then minimize the evidence lower bound:

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(\theta, \phi; x^{(i)})$$

$$\mathcal{L}(\theta, \phi; x^{(i)}) = -D_{KL}(q_\phi(z|x^{(i)}) \| p_\theta(z)) + \mathbb{E}_{q_\phi(z|x^{(i)})}[\log p_\theta(x^{(i)}|z)]$$

Taking derivatives with respect to ϕ can be tricky so we reparamaterize z using a differentiable transform $g_\phi(\epsilon, x)$ where $\epsilon \sim p(\epsilon)$. $g_\phi(\epsilon, x)$ and $p(\epsilon)$ will change depending on what distribution $q_\phi(z|x)$ is. We describe the details of our implementation in the details section.

Details

For our model we used a Gaussian encoder and a Bernoulli decoder. Since our encoder is Gaussian we have:

$$\log q_\phi(z|x^{(i)}) = \log \mathcal{N}(z; \mu^{(i)}, \sigma^{2^{(i)}} I)$$

$$z^{(i)} \sim q_\phi(z|x^{(i)}),\ \epsilon \sim \mathcal{N}(0, I)$$

$$z^{(i)} = g_\phi(\epsilon, x^{(i)}) = \mu^{(i)} + \sigma^{(i)} \odot \epsilon$$

Our encoding step is then:

$$\begin{aligned} h &= \tanh(W_3 x + b_3) \\ \mu &= W_4 h + b_4 \\ \log \sigma^2 &= W_5 h + b_5 \\ z &= \mu + \sigma \odot \epsilon \qquad\qquad \text{where } \epsilon \sim \mathcal{N}(0, I) \end{aligned}$$

And our decoding step is:

$$\begin{aligned} y &= \sigma(W_2 \tanh(W_1 z + b_1) + b_2) \\ \log p_\theta(x|z) &= \sum_{i=1}^{D} x_i \log y_i + (1 - x_i) \log(1 - y_i) \end{aligned}$$

Where $\theta = \{W_1, W_2, b_1, b_2\}$ and $\phi = \{W_3, W_4, W_5, b_3, b_4, b_5\}$. We can use these to minimize an estimated evidence lower bound:

$$\mathcal{L}(\theta, \phi, x^{(i)}) \approx \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \log p_\theta(x^{(i)}, z^{(i)})$$

Where J is the dimension of the latent variable z. The details of the KL divergence term can be found in the appendix of (Kingma et. al., 2013).

Experiments

For our experiments we used the MNIST data set. This gave us a dimension of 784 for our data, x. We used a dimesion of 3 for our latent variable z. We used a hidden dimension of 512 for h. We trained for 2000 epochs or roughly 110 million images with a batch size of 128.

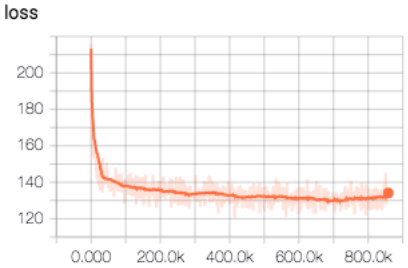

Figure 1: Training Loss

Figure 1 shows the negative log likelihood for the training data over iterations. We were able to achieve a negative log likelihood of 136 on the test set which is about the same as (Kingma et. al., 2013).

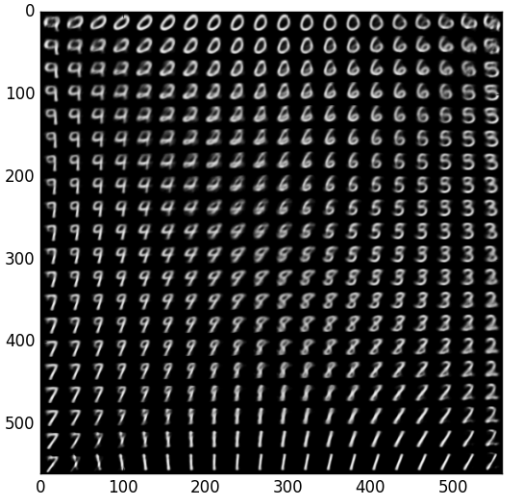

Figure 2: MNIST Manifold

In Figure 2 we trained a model with z dimension 2 and sampled a unit square from the latent variable and produce an image similar to Figure 4 in (Kingma et. al., 2013).

References

[Kingma et. al. 2013] Diederik P. Kingma, Max Welling. 2013. Auto-Encoding Variational Bayes.