Auto-Encoding Variational Bayes

LE TRAN Ngoc Tran
Universite de Paris
tran.le-tran-ngoc@etu.u-paris.fr

ZHENG Zhe
ENS Paris-Saclay
zzheng@ens-paris-saclay.fr

Abstract

In the presence of continuous latent variables with intractable posterior distributions and large datasets, the auto-encoding variational bayes proposes a reparameterization technique applied to the evidence lower bound, which yields a new lower bound estimator. With such estimator, the inference and learning can be especially efficient.

1 Introduction

Given a set of observed data, we are always interested in describing them with a probabilistic model. While assuming all data is random sample from a true distribution which is unknown, we try to approximate this unknown distribution with a chosen probabilistic model, say $p_\theta(\mathbf{x})$, which is parameterized by θ. We search for a value of the parameter θ such that the chosen model with this value can approximate the true distribution well. With fully observed model, this can be performed through maximum log-likelihood, while in the presence of latent variables $\mathbf{z}$, the marginal likelihood of the data $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})dz$ is usually intractable, and this is true for the posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$. Variational inference provide us a framework to deal with these intractable distributions, but traditional methods are usually computationally expensive, which makes them unsuitable for large dataset.

The paper [1] proposed a reparameterization technique leading to a lower bound estimator of the variational lower bound which can be optimized directly. Moreover, with the proposed lower bound estimator, the inference of the posterior can be make especially efficient.

2 Problem settings

Consider a dataset $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ consisting of $N \geq 1$ i.i.d. samples of some variable $\mathbf{x}$. We assume there is a latent variable $\mathbf{z}$, which is generated from some prior distribution $p_\theta(\mathbf{z})$, while the data x is generated from some conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$. The scenario we consider has a problem with tractability. Specifically, the marginal likelihood $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})dz = \int p_\theta(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z})dz$ does not have an analytic solution. Also, the posterior, $p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{x})}$ is intractable due to the marginal distribution in the denominator.

3 Method: Optimize the Approximate Posterior

To solve the intractable posterior inference and marginal likelihood estimating, this paper introduced an recognition model (also called encoder) $q_\phi(\mathbf{z}|\mathbf{x})$ parameterized by ϕ (variational parameter), which is an approximation to the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$. Note that this is actually under the variational inference framework [2] The aim is to optimize variational parameters ϕ, such that

$$q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x}) \tag{1}$$

3.1 Variational Lower bound

For any choice of the recognition model $q_\phi(\mathbf{z}|\mathbf{x})$, we have

$$\begin{aligned}\log(p_\theta(\mathbf{x})) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}))] = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(\frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})})] \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})})] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})})] \\ &= \mathcal{L}(\theta,\phi;\mathbf{x}) + D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})). \end{aligned} \tag{2}$$

The second term of (2) is the KL divergence which measures the distance from the approximate to the true posterior. Since this term is non-negative, the first term $\mathcal{L}(\theta,\phi;\mathbf{x})$ is called the variational lower bound (of the log likelihood of the data):

$$\mathcal{L}(\theta,\phi;\mathbf{x}) = \log(p_\theta(\mathbf{x})) - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) \le \log(p_\theta(\mathbf{x})). \tag{3}$$

Thus, by maximizing $\mathcal{L}(\theta,\phi;\mathbf{x})$ with respect to θ and ϕ, we are maximizing the marginal likelihood $p_\theta(\mathbf{x})$ and minimizing the KL divergence between the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ and the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$.

The above is the first step of Variational Inference which we have seen in the class. With this, we convert an inference problem to an optimization problem requiring minimizing the KL divergence, and this amounts to maximizing the ELBO. Before starting to optimize this term, let's show the objective again:

$$\mathcal{L}(\theta,\phi;\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})})] = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log(q_\phi(\mathbf{z}|\mathbf{x})) + \log(p_\theta(\mathbf{x},\mathbf{z}))] \tag{4}$$

Another way to express the ELBO can be derived as follows:

$$\begin{aligned}\mathcal{L}(\theta,\phi;\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{z})) - \log(q_\phi(\mathbf{z}|\mathbf{x}))] \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})),\end{aligned} \tag{5}$$

3.2 SGVB estimator and AEVB algorithm

Optimizing the ELBO requires computing its gradient w.r.t. θ and ϕ. Gradient of the ELBO w.r.t. the model parameter θ is simple to obtain:

$$\nabla_\theta \mathcal{L}(\theta,\phi;\mathbf{x}) = \nabla_\theta \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x},\mathbf{z}) - \log(q_\phi(\mathbf{z}|\mathbf{x}))] \tag{6}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\nabla_\theta(\log(p_\theta(\mathbf{x},\mathbf{z}) - \log(q_\phi(\mathbf{z}|\mathbf{x})))], \tag{7}$$

since the gradient and expectation are taken for different parameters. But the exchangeability may not hold when we treat parameter ϕ, i.e.

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x},\mathbf{z}) - \log(q_\phi(\mathbf{z}|\mathbf{x}))] \neq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\nabla_\phi(\log(p_\theta(\mathbf{x},\mathbf{z}) - \log(q_\phi(\mathbf{z}|\mathbf{x})))] \tag{8}$$

This paper proposed to use a reparameterization trick to compute an estimator. The essential reparameterzation trick is quite simple. Let the $z \sim q_\phi(\mathbf{z}|\mathbf{x})$ be a continuous random variable, and then it can be expressed as a differentiable transformation of an auxiliary variable ϵ given $\mathbf{x}$ and ϕ, i.e.

$$\mathbf{z} = g_\phi(\epsilon, \mathbf{x}),$$

where $\epsilon \sim p(\epsilon)$ is independent of $\mathbf{x}$ and ϕ.

With such a change of variable, expectations can be rewritten in terms of ϵ, i.e.,

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[f(\mathbf{z})] = \int q_\phi(\mathbf{z}|\mathbf{x}) f(\mathbf{z}) dz = \int p(\epsilon) f(g_\phi(\epsilon,\mathbf{x})) d\epsilon = \mathbb{E}_{p(\epsilon)}[f(g_\phi(\epsilon,\mathbf{x}))],$$

and this expectation can be estimated by a simple Monte Carlo estmate, i.e.

$$\mathbb{E}_{p(\epsilon)}[f(g_\phi(\epsilon, \mathbf{x}))] \simeq \frac{1}{L}\sum_{l=1}^{L} f(g_\phi(\epsilon^{(l)}, \mathbf{x})) \quad \text{where } \epsilon^{(l)} \sim p(\epsilon).$$

One simple example is given by the univariate Gaussian case: assume that r.v. $z \sim \mathcal{N}(\mu, \sigma^2)$. A valid reparameterization is $z = \mu + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$. Thus we have $\mathbb{E}_{\mathcal{N}(z;\mu,\sigma^2)}[f(z)] = \mathbb{E}_{\mathcal{N}(\epsilon;0,1)}[f(\mu + \sigma\epsilon),]$ and can be estimated by $\frac{1}{L}\sum_{l=1}^{L} f(\mu + \sigma\epsilon^{(l)})$, where $\epsilon^{(l)} \sim \mathcal{N}(0, 1)$.

Applying this technique to the ELBO (4) and using simple Monte Carlo estimate, we obtain the Stochastic gradient Variational Bayes estimator:

$$\tilde{\mathcal{L}}(\theta, \phi; x) = \frac{1}{L}\sum_{i=1}^{L}(\log(p_\theta(x, z^{(l)})) - \log(q_\phi(z^{(l)}|x)))$$
$$\text{where } z^{(l)} = g_\phi(\epsilon^{(l)}, x) \text{ and } \epsilon^{(l)} \sim p(\epsilon). \tag{9}$$

For the second version of the ELBO (5), the KL-divergence $D_{KL}(q_\phi(z|x)||p_\theta(z))$ can be integrated analytically for some cases (e.g. the Gaussian case) and only the first term needs estimate by sampling. Thus, we can propose a second version of the SGVB estimator:

$$\tilde{\mathcal{L}}(\theta, \phi; x) = \frac{1}{L}\sum_{i=1}^{L}\log(p_\theta(x|z^{(l)})) - D_{KL}(q_\phi(z|x)||p_\theta(z))$$
$$\text{where } z^{(l)} = g_\phi(\epsilon^{(l)}, x) \text{ and } \epsilon^{(l)} \sim p(\epsilon). \tag{10}$$

The obtained SGVB estimator yields a estimator of the gradient of the ELBO, $\nabla_{\theta,\phi}\tilde{\mathcal{L}}(\theta, \phi; x)$. This will be used to optimize the ELBO with SGD. With minibatches, the minibatch estimator is given by:

$$\mathcal{L}(\theta, \phi; X) \simeq \tilde{\mathcal{L}}^M(\theta, \phi; X^M) = \frac{N}{M}\sum_{i=1}^{M}\tilde{\mathcal{L}}(\theta, \phi; X^{(i)}),$$

where X^M represents the minibatch which is a randomly drawn sample of M datapoints from the full dataset X with N datapoints.

Algorithm 1 AEVB: Minibatch version of the Auto-Encoding Variational Bayes algorithm.

```
1: $(\theta, \phi) \leftarrow$ Initialize parameters
2: **repeat**
3:     $X^M \leftarrow$ minibatch of M datapoints, randomly drawn from the full dataset.
4:     $\epsilon \leftarrow$ random samples from noise distribution $p(\epsilon)$
5:     Compute $\tilde{\mathcal{L}}(\theta, \phi; X^M, \epsilon)$ and its gradient $\nabla_{\theta,\phi}\tilde{\mathcal{L}}(\theta, \phi; X^M, \epsilon)$
6:     Update $\theta$ and $\phi$
7: **until** convergence of parameters
8: **return** $\theta$ and $\phi$
```

3.3 Why reparameterization help

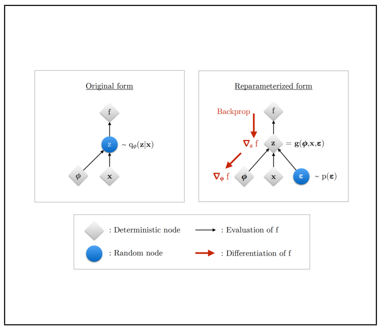

Figure 1: The reparameterization trick. Original from Kingma's slides.

Our objective function, the ELBO (4), is affected by the stochastic variable $\mathbf{z} \sim q_\phi(z|x)$, so we cannot take the derivative w.r.t. ϕ through $\mathbf{z}$. But with the reparameterization trick, we have the randomness on the newly introduced variable ϵ, and $\mathbf{z}$ is now deterministic function w.r.t. ϕ. Thus we can compute the gradient $\nabla_\phi \mathcal{L}$ directly.

4 Connection with auto-encoder and decoder

One important contribution of this paper is the proposal of the popular variational auto-encoders of which we will show an example later. An encoder means for each input data x, it can be encoded as a code z. The decoder means mapping the code z to an instance $\hat{x}$ in the original space.

Let's look at the second version of objective function (5). We sample ϵ from the distribution $p(\epsilon)$, and by a deterministic function, this sample together with the datapoint x is mapped to z. This is similar to the auto-encoder, which encodes a datapoint to a code, but note that these is uncertainty, which means it actually is obtained by sampling. Then this code is put into the function $\log(p_\theta(x|z)$, which gives a simulated datapoint given z. So the first term actually measures the reconstruction quality of encoding-decoding process. If this quantity is big, then we may think we have found a good process to represent our data. Since the true posterior is the best encoder of x, we can also think our approximate posterior $q_\phi(z|x)$ is a good approximation of the true posterior. The second term measures the distance between the approximate posterior $q_\phi(z|x)$ and the prior $p_\theta(z)$, which acts as a regularization of the objective, ensuring the obtained posterior has the property we want (which is put on the prior).

An important difference between this model and traditional encoder-decoder model is that, here we have uncertainty, since it didn't output the code directly but a distribution of it, and we need to sample from the distribution we have. This is also true for the decoding process. For this reason, this model can generate new datapoint according to the process we assumed at the beginning, and used as a generative model. When the encoder and decoder are parameterized by neural networks, the model is named as variational auto-encoder [1, 3].

5 Example: Variational Auto-Encoders

In this part, we give a detail example in which by using a neural network for the probabilistic encoder $q_\phi(\mathbf{z}|\mathbf{x})$, we approximate the posterior of the generative model $p_\theta(\mathbf{x}, \mathbf{z})$ and the parameters ϕ and θ are optimized jointly with the AEVB algorithm.

In this model, we used a MLP with Gaussian output for the encoder. In detail, both the prior $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$ and the posterior approximation $q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{z}; \mu^{(i)}, \sigma^{2(i)}\mathbf{I})$ are Gaussian, where $\mu^{(i)}$ and $\sigma^{(i)}$, are the outputs of the encoding MLP, i.e. nonlinear functions of datapoint $\mathbf{x}^{(i)}$ and the variational parameters ϕ.

As explained in subsection 3.2, we sample from the posterior $\mathbf{z}^{(i,l)} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$ using $\mathbf{z}^{(i,l)} = g_\phi(\mathbf{x}^{(i)}, \epsilon^{(l)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)}$ where $\epsilon^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ with $\odot$ represented for an element-wise product. From the equation (10), the estimator for this model and datapoint $\mathbf{x}^{(i)}$ is given:

$$\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) \simeq \frac{1}{2}\sum_{j=1}^{J}\Big(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\Big) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$$
$$\text{where} \quad \mathbf{z}^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)} \quad \text{and} \quad \epsilon^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{11}$$

where the decoding term $\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$ is defined as follow.

We let $p_\theta(\mathbf{x}|\mathbf{z})$ be a multivariate Bernoulli whose probabilities are computed from $\mathbf{z}$ with a MLP:

$$\log p(\mathbf{x}|\mathbf{z}) = \sum_{i=1}^{D} x_i \log y_i + (1 - x_i)\log(1 - y_i) \tag{12}$$
$$\text{where } \mathbf{y} = f_\sigma(\mathbf{W}_2 \tanh(\mathbf{W}_1\mathbf{z} + \mathbf{b}_1) + \mathbf{b}_2) \tag{13}$$

where $f_\sigma(\cdot)$ is the elementwise sigmoid activation function, and where $\theta = \{\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2\}$ are the weights and biases of the MLP.

6 Experiments

In this experiment part, we first implement the model variational auto-encoder described in section 5. We used the MNIST data to train the model, to make the diversity of the experiment, we trained the encoder and decoder not only with a MLP architecture but also with convolution networks. Then we can compare the performance of two different architectures on sampling and resontruction results. The detail of archietcture can be found at link github `https://github.com/ltngoctran/Auto-Encoding_Variational_Bayes`. Next, with a MLP, we change the dimension of latent space and plotted the coresponding generative $p_\theta(\mathbf{x}|\mathbf{z})$ with the learned parameters θ. Moreover, one of the most different and outstanding of variational auto-encoder compared to traditional auto-encoder decoder (we mentioned in section 4) is that the ability to generate new data when we get an arbitrary vector in the latent space $\mathbf{z}$, we call it as sampling. Inherited from that, the latent space of VAE and AE are plotted to see the difference of distribution of vectors in the latent space.

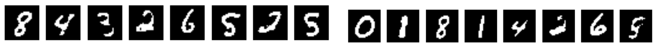
(a) Sampling MLP (b) Sampling CNN

Figure 2: The sampling images from MLP and CNN architecture

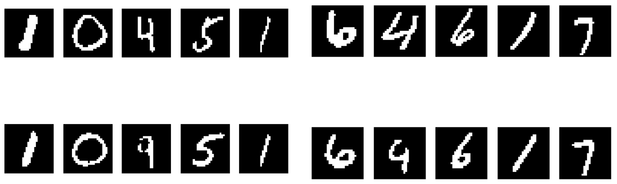
(a) Reconstruction MLP (b) Reconstruction CNN

Figure 3: The reconstruction images from MLP and CNN architecture

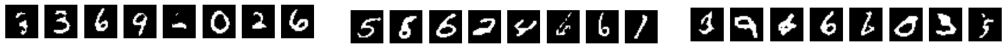
(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space

Figure 4: Random samples from learned generative models for different dimensionalities of latent space

After training, the cluster of latent space is given in figure 5. Observing the two figures, we can realize the total different structure of the latent space between VAE and AE. Vectors in latent space in figure 5a do not follow any distribution, meanwhile, the latent space of VAE more compact.

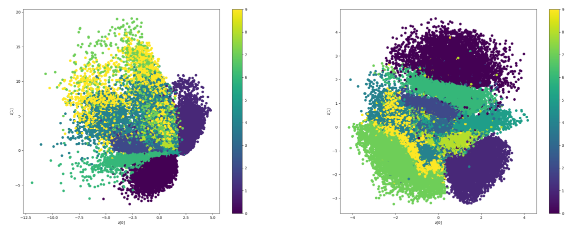
(a) Auto-encoder (b) Variational Auto-encoder

Figure 5: The label cluster of latent spaces.

7 Conclusion

In this report, we present a novel estimator of ELBO, Stochastic Gradient VB (SGVB), for efficient approximate inference in the presence of continuous latent variables, proposed in [1]. The reparameterization trick is essential for this efficient method. But we note that, 1. this technique is only applicable for continuous variable. how about discrete variable? 2. Authors mentioned that the naive estimator for gradient with score function has high variance [4], and the proposed estimator has small variance [1, 5] empirically, but it is of a lack of analyse. Via this report, we also obatin some interesting experiments. We can see how the performance changes when we training with a different dimension of latent spaces. And the higher the latent space, the much time to train data. When we compare the label cluster of latent space, the new characteristic of VAE compared to AE is illustrated.

References

[1] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[2] D. M. Blei, A. Kucukelbir, J. D. McAuliffe, Variational inference: A review for statisticians, Journal of the American statistical Association 112 (518) (2017) 859–877.

[3] C. Doersch, Tutorial on variational autoencoders, arXiv preprint arXiv:1606.05908 (2016).

[4] J. Paisley, D. M. Blei, M. I. Jordan, Variational bayesian inference with stochastic search, in: Proceedings of the 29th International Coference on International Conference on Machine Learning, 2012, pp. 1363–1370.

[5] D. Rezende, S. Mohamed, Variational inference with normalizing flows, in: International Conference on Machine Learning, PMLR, 2015, pp. 1530–1538.