Variational AutoEncoder

Linda Wei (SJTU)

December 2022

1 Introduction

Variational Auto Encoder (VAE) is a generative modeled used in unsupervised learning, mainly to generate images which are similar to the original one. The mathematical theory background of VAE is variational inference. VAE is composed of an encoder, a sample layer and a decoder.

2 Theory Background

2.1 Problem Statement

We assume that there a dataset $\mathcal{X} = \{x^{(i)}\}, i = 1, 2, 3, \cdots n$, which follows the following generative process. Given the prior $p_\theta(z)$.

1. We sample $z^{(i)}$ from the prior $p_\theta(z)$.
2. We get $x^{(i)}$ from the conditional likelihood $p_\theta(x \mid z)$

the prior and the conditional likelihood is the function (neruo network) of parameter θ. z is not observable and x is observable.Then there are three problems:

1. Parameter Estimation: How can we get θ given $x^{(i)}$ (then we get the generative model)
2. Posterior Estimation: How can we get the posterior distribution $p_\theta(z \mid x)$,given certain $x^{(i)}$ (then we can do the dimensional compress for x, we can use z to express the information of x)
3. Marginal Distribution: How can we get $p_\theta(x^{(i)})$ (it can be suitable for any condition that needs to assume the prior of x)

2.2 Variational Inference Method

By Bayesian Theorem, we get:

$$\begin{aligned} p\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) &= \frac{p\left(\mathbf{z}, \mathbf{x}^{(i)}\right)}{p\left(\mathbf{x}^{(i)}\right)} \\ &= \frac{p\left(\mathbf{x}=\mathbf{x}^{(i)} \mid \mathbf{z}=\mathbf{z}^{(i)}\right) p\left(\mathbf{z}=\mathbf{z}^{(i)}\right)}{\int_{\mathbf{z}^{(i)}} p\left(\mathbf{x}=\mathbf{x}^{(i)} \mid \mathbf{z}=\mathbf{z}^{(i)}\right) p\left(\mathbf{z}=\mathbf{z}^{(i)}\right) d\mathbf{z}^{(i)}} \\ &= \frac{p\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right) p(\mathbf{z})}{\int_{\mathbf{z}} p\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right) p(\mathbf{z}) d\mathbf{z}} \end{aligned}$$

Note that the denominator is the joint marginal distribution, the high dimensional integral in the denominator is usually not solvable. In fact, we can use the MCNC(Markov Chain Montecarlo) such as Gibbs sampling we learned in STAT4510J to estimate the integral, however, as the dimension of the dataset is always very large in deep learning, it is hard for the Gibbs sampling method to find suitable PDF and it takes a lot of time. So, we must introduce new method, that is **Variational Inference Method.**

The variational inference method will first introduce another distribution $q_\phi(z \mid x)$ and we call the **recognition model**, actually from the perspective of Bayesian, it can be regarded as a posterior network.

The key idea of this method is that: it will use $q_\phi(z \mid x)$ to fit the real posterior $p_\theta(x \mid z)$, to make the two distribution more and more closer, thus will convert the calculation of posterior distribution to be a problem of parameter optimizing on $\phi.\theta$, the loss function, clearly, is directly related with the KL-Divergence of the two distributions.

$$\phi^*, \theta^* = \operatorname{argmin}_{\phi,\theta} KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p_\theta\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right)$$

From the further derivation, we get that:

$$\begin{aligned} KL\left(q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right) &= \int_{\mathbf{z}} q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \log \frac{q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)}{p\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)} d\mathbf{z} \\ &= \int_{\mathbf{z}} q(z \mid x^{(i)}) \log q(z \mid x^{(i)}) - q(z \mid x^{(i)}) \log p(z \mid x^{(i)}) d\mathbf{z} \\ &= \mathbb{E}_{\mathrm{q}}\left[\log q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] - \mathbb{E}_{\mathrm{q}}\left[\log p\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] \\ &= \mathbb{E}_{\mathrm{q}}\left[\log q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] - \mathbb{E}_{\mathrm{q}}\left[\log p\left(\mathbf{x}^{(i)}, \mathbf{z}\right)\right] + E_q\left[\log p\left(\mathbf{x}^{(i)}\right)\right] (***) \\ &= \mathbb{E}_{\mathrm{q}}\left[\log q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] - \mathbb{E}_{\mathrm{q}}\left[\log p\left(\mathbf{x}^{(i)}, \mathbf{z}\right)\right] + \log p\left(\mathbf{x}^{(i)}\right) \text{ not related with q} \\ &= -\operatorname{ELBO} + \log p\left(\mathbf{x}^{(i)}\right) \end{aligned}$$

Note the *** part is because: $p(x,y) = p(x) * p(y \mid x) \rightarrow p(x,y \mid u) = p(x \mid u) * p(y \mid u,x)$
The term $\log(p(x^{(i)})$ is a const, thus we omit it in the optimizing process.The utility function of this problem is just ELBO

$$\begin{aligned}
\text{ELBO} &= -\mathbb{E}_{\text{q}}\left[\log q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] + \mathbb{E}_{\text{q}}\left[\log p\left(\mathbf{x}^{(i)}, \mathbf{z}\right)\right] \\
&= -\mathbb{E}_{\text{q}}\left[\log q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right)\right] + \mathbb{E}_{\text{q}}[\log p(\mathbf{z})] + \mathbb{E}_{\text{q}}\left[\log p\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right] \\
&= -KL\left(q\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p(\mathbf{z})\right) + \mathbb{E}_{\text{q}}\left[\log p\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right]
\end{aligned}$$

thus finally, the optimization problem is:

$$\phi^*, \theta^* = \text{argmax}_{\phi,\theta}\{-KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p_\theta(\mathbf{z})\right) + \mathbb{E}_{q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right]\}$$

We write the utility function in the form of Loss Function:

$$\mathcal{L}(\phi, \theta, x^{(i)}) = KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p_\theta(\mathbf{z})\right) - \mathbb{E}_{q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right]$$

we apply samping method in MCNC(like Matropolis-Hasting Algorithm we learned in STAT4510J) to sample L $z^{(i,l)}$ from the distribution $q_\phi(z^{(i)} \mid x^{(i)})$ thus the expectation can be written as sample average:

$$\mathbb{E}_{q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right] = \frac{1}{L}\sum_{l=1}^{L} \log(p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}^{(\mathbf{i},\mathbf{l})}\right)$$

thus the loss function will be:

$$\mathcal{L}(\phi, \theta, x^{(i)}) = \mathcal{L}(\phi, \theta, x^{(i)}) = KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p_\theta(\mathbf{z})\right) - \frac{1}{L}\sum_{l=1}^{L} \log(p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}^{(\mathbf{i},\mathbf{l})}\right)$$

However, in real practice, we do not directly sample from $q_\phi(z \mid x^{(i)})$ using MCNC because it consume a lot of time in calculation, here we introduce another technique, called **reparameterazation**.We set:

$$\mathbf{z}^{(i,l)} = g_\phi\left(\epsilon^{(i;l)}; \mathbf{x}^{(i)}\right)$$

in which g is also a neruo network, ϵ is a random noise which is sampled from Standard NormalDistribution.
Another merit of reparameterazation is that: we consider the term $\frac{1}{L}\sum_{l=1}^{L} \log(p_\theta\left(\mathbf{x}^{(i)} \mid \mathbf{z}^{(\mathbf{i},\mathbf{l})}\right)$, we find that it is not differentiable respect to ϕ, then when we do the back propagation, we will find that we can not achieve our goal of "optimizing ϕ, θ together", since we could only use the gradient of KL divengence.

2.3 Conclusion

The learning process of VAE can be concluded as following 4 steps:

1. Get a minibatch M from the training set, M is the batchsize.
2. Calculate the batch loss $\frac{1}{M}\sum_{i=1}^{M}\mathcal{L}\left(\phi,\theta,\mathbf{x^{(i)}}\right)$
3. Back Propagation $\frac{1}{M}\sum_{i=1}^{M}\nabla_{\phi,\theta}\mathcal{L}\left(\phi,\theta,\mathbf{x^{(i)}}\right)$ and get new ϕ,θ
4. repeat to convergence.

3 VAE vs AE

3.1 Auto Encoder(AE)

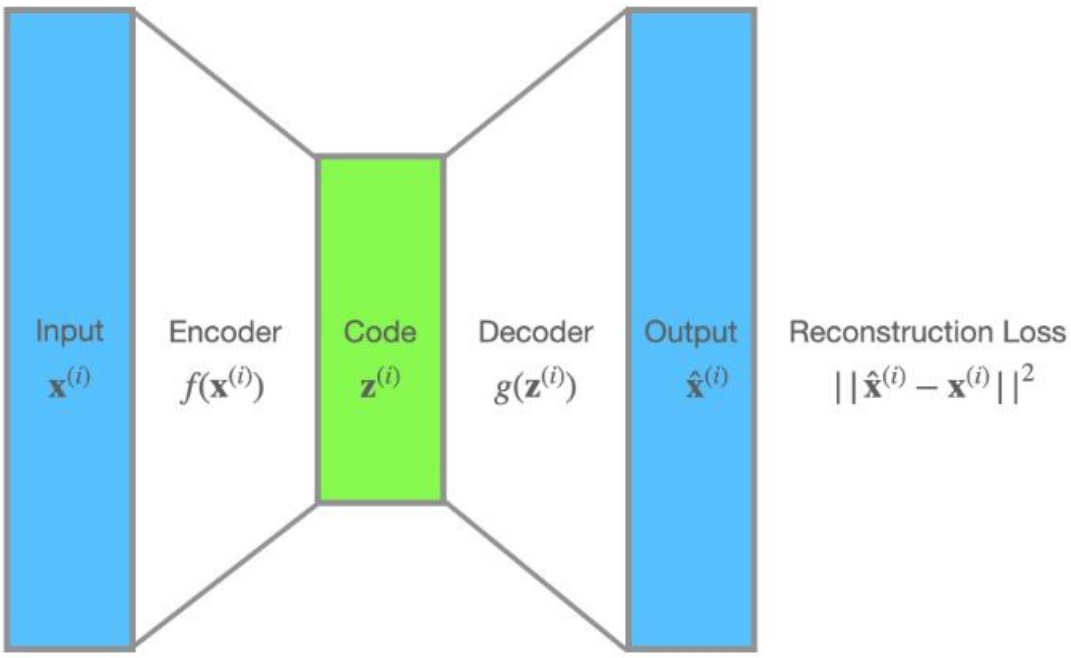

图 1: Structure of AE

This is the structure of AutoEncoder. The input x will be sent to the encoder and get its feature representaion z, and the decoder will accept z and reconstruct $\hat{x}$, then we use the l_2 norm $||\hat{x}-x||_2^2$ to measure the difference.

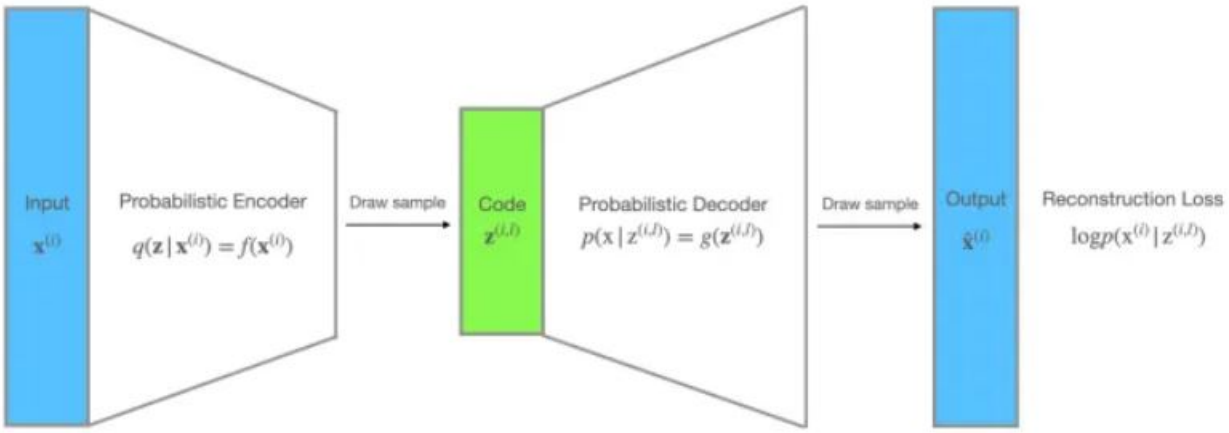

图 2: Structure of VAE

3.2 Variational Auto Encoder(VAE)

From the perspective of Auto Encoder, the output of encoder is no longer a certain point but a distribution, and the decoder also provide a distribution of reconstructed $\hat{x}$, if we want to get a certain point(image), we just need to sample from it, which solve the overfitting problem of AE. In practice, it means that we can get images with a larger variety.

4 Coding and Practice

In this part, I choose the dataset CelebA which is a dataset consists of many human faces RGB images.

4.1 Muti-Gaussian Encoder Realization

We always choose a multivariate Gaussian Distribution as the output of the Encoder, the parameter ϕ includes $\mathbf{W_h}, \mathbf{b_h}, \mathbf{W}_{\sigma^2}, \mathbf{b}_{\sigma^2}, \mathbf{W}_\mu, \mathbf{b}_\mu$

$$\begin{aligned}
\mathbf{h} &= relu\left(\mathbf{W_h x} + \mathbf{b_h}\right) \\
\mu &= \mathbf{W}_\mu \mathbf{h} + \mathbf{b}_\mu \\
\sigma^2 &= \exp\left(\mathbf{W}_{\sigma^2}\mathbf{h} + \mathbf{b}_{\sigma^2}\right) \\
q(\mathbf{z} \mid \mathbf{x}) &= \mathcal{N}\left(\mathbf{z}; \mu, \sigma^2\mathbf{I}\right)
\end{aligned}$$

4.2 Mutinomial Decoder Realization

The dataset we choose contains RGB image, so we must output Mutinomial distribution.We use softmax function to normalize the output. The complete code is on the github github/lullcant

5 Reference

1.https://zhuanlan.zhihu.com/p/574208925 2.https://zhuanlan.zhihu.com/p/519448634