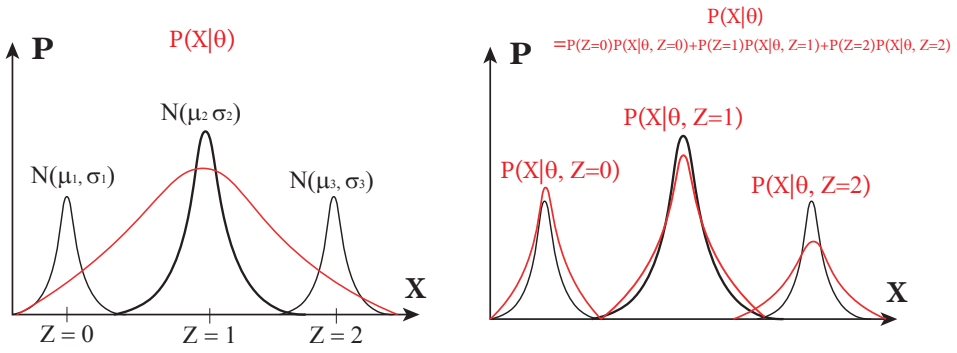
Basics VAE

June 27, 2022

Useful papers - [Doe16]; [KW13]

1 Introduction

Let $x_1, ..., x_n$ - independent and identically distributed (i.i.d) random variables from distribution $P_{data}(X)$ (let's say this would be training dataset). Goal of any generative model is to reproduce the distribution $P_{data}(X)$ (or at least be able to make samples from it). VAE works with latent variables Z from some distribution $P(Z)$. The main idea behind latent distribution is to reproduce initial object (X variable) using hidden description (Z). Note that we also assume that variable Z can be easy sampled from $P(Z)$. Lets discuss the motivation of this idea in more detail. It is quite logical to assume the following fact: if we have some additional information about variable X, then the conditional distribution $P_{data}(X|Z)$ is simpler than initial $P_{data}(X)$. Lets show this with an example of mixture of gaussians (Figure 1). In this example we have



mix of three gaussians. If we will approximate it directly using normal distribution as $P(X|\Theta)$, then we obtain bad approximation (as can be seen from figure). However, if we add latent description Z that define each gaussian, we can consider conditional distribution as gaussian. After that, using marginalization, we obtain $P(X|\Theta)$ with higher approximation accuracy to $P_{data}(X)$. Thus, conditional distribution much simpler than $P(X|\Theta)$ and final approximation quite better than non-latent approach.

So, if we have latent description, we need to build parameterized function $f : Z \times \Theta \to X$ to display hidden variable to the initial. In this way we create samples from

conditional distribution $P(X|Z,\Theta)$. We can obtain marginal distribution as

$$\begin{aligned} P(X|\Theta) = \int_Z P(X,Z|\Theta)dz = \\ = [P(X,Z|\Theta) = P(Z,X|\Theta) = P(X|Z,\Theta)P(Z|\Theta), P(Z|\Theta) = P(Z)] = \\ = \int_Z P(X|Z,\Theta)P(Z)dz = \mathbb{E}_{z\sim P}\, P(X|Z,\Theta) \approx \frac{1}{m}\sum_{i=1}^{m} P(X|z_i,\Theta) \end{aligned} \tag{1}$$

Here, we used Monte-Carlo approximation of the integral, thus $z_i \sim P(Z)$. This marginal will be an approximation of our data distribution $P_{data}(X)$. VAE using following assumptions: $P(X|Z,\Theta) = \mathbf{N}(X|f_\Theta(Z), \sigma I)$, $P(Z) = \mathbf{N}(Z|0,I)$. Usually, if we have some function that want to approximate, lets use universal neural network for this purpose. That is, $f_\Theta(Z)$ is neural network. We can use maximum marginal likelihood estimation for approximation the data distribution:

$$\begin{aligned} \Theta^* = \operatorname*{argmax}_\Theta P(X|\Theta) = \operatorname*{argmax}_\Theta \mathbb{E}_{z\sim P}\, P(X|Z,\Theta) \\ = \operatorname*{argmax}_\Theta \int dZ P(Z)P(X|Z,\Theta) = \operatorname*{argmax}_\Theta \prod_j \int dz_j P(z_j)P(x_j|z_j,\Theta) \\ \approx \operatorname*{argmax}_\Theta \frac{1}{m}\sum_{i=1}^{m}\prod_{j=1}^{n} P(x_j|z_{j_i},\Theta) \\ P_{data}(X) \approx P(X|\Theta^*) \end{aligned} \tag{2}$$

A question may arise: why this optimization problem gives the approximation of the $P_{data}(X)$. Lets show it. From the properties of the KL-divergence (in Section 2 details are provided): if $\mathbb{KL}(P_{data}(X), P(X|\Theta)) = 0$, then $P(X|\Theta) = P_{data}(X)$. We need to show that minimization of the KL-divergence is equivalence to maximum likelihood estimation.

$$\begin{aligned} \operatorname*{argmin}_\Theta \mathbb{KL}(P_{data}(X), P(X|\Theta)) = \operatorname*{argmin}_\Theta \int dX P_{data}(X)\log\frac{P_{data}(X)}{P(X|\Theta)} = \\ \operatorname*{argmin}_\Theta \left[\int dX P_{data}(X)\log P_{data}(X) - \int dX P_{data}(X)\log P(X|\Theta)\right] = \\ = \operatorname*{argmax}_\Theta \int dX P_{data}(X)\log P(X|\Theta) = \operatorname*{argmax}_\Theta \mathbb{E}_{X\sim P_{data}}\log P(X|\Theta) \end{aligned} \tag{3}$$

So, we show it for logarithm. But the maximum of $\log P(X|\Theta)$ is equal to maximum of $P(X|\Theta)$ due to monotic properties of the logarithm. However, we also have expectation on samples from data distribution $\mathbb{E}_{X\sim P_{data}}$. This expectation is sometimes called an experimental distribution, and it is discrete, not continuous, as I wrote in (3) (but im not sure :)). Then,

$$\begin{aligned} \operatorname*{argmax}_\Theta \mathbb{E}_{X\sim P_{data}}\log P(X|\Theta) = \operatorname*{argmax}_\Theta \sum_{x_i} P_{data}(X=x_i)\log P(X=x_i|\Theta) = \\ = \operatorname*{argmax}_\Theta \frac{1}{n}\sum_i \log P(x_i|\Theta) = \operatorname*{argmax}_\Theta \log P(X|\Theta) \end{aligned} \tag{4}$$

Here, x_i - sample from dataset. We used that experimental distribution is uniform over the number of dataset samples, i.e. $i \sim \mathbb{U}(1, ..., n)$. Hence, we have shown the equivalence between KL-divergence and log-likelihood maximization. Then, we can approximate data distribution using (2).

So, as was said, we can use log-likelihood maximization. But in (2) we have a product of distribution without logarithm. It is so hard to take gradients from product, this procedure is ineffective and resource-consuming. To avoid a product lets change it to logarithm

$$\Theta^* = \arg\max_{\Theta} \log P(X|\Theta) = \arg\max_{\Theta} \log \mathbb{E}_{z\sim P} P(X|Z,\Theta) \tag{5}$$

Unfortunately, we have met a problem. We cant forward logarithm through expectation ($\log \mathbb{E} \neq \mathbb{E} \log$), thus cant avoid a product. For clarity lets prove $\log \mathbb{E} \neq \mathbb{E} \log$. As we know, logarithm is concave function. Function f calls concave if

$$f(y) \leq f(x) + f'(x)(y - x), \ \forall y \tag{6}$$

It means that the tangent at any point lies above the function, x is a random but fixed point. Let y and x be a random variables. If we can choose x the way we want, let $x = \mathbb{E}\, y$. Taking expectations of both parts:

$$\begin{gathered} \mathbb{E}\, f(y)\, \mathbb{E} \leq f(\mathbb{E}\, x) + \mathbb{E}\, f'(x)(y - x) \\ \mathbb{E}\, f(y) \leq f(\mathbb{E}\, y) + f'(x)(\mathbb{E}\, y - x) = f(\mathbb{E}\, y) \end{gathered} \tag{7}$$

It calls Jensen inequality, lets do it for (5)

$$\arg\max_{\Theta} \log \mathbb{E}_{z\sim P} P(X|Z,\Theta) \geq \arg\max_{\Theta} \mathbb{E}_{z\sim P} \log P(X|Z,\Theta) \tag{8}$$

Finally, we did everything and ready for computational optimization!... No. So, we can maximize $\mathbb{E}_{z\sim P} \log P(X|Z,\Theta)$, but the gap between $\mathbb{E}_{z\sim P} \log P(X|Z,\Theta)$ and $\log \mathbb{E}_{z\sim P} P(X|Z,\Theta)$ can be very large, thus we obtain bad parameter Θ. We have to consider another approach. More precisely, we need something that decrease mentioned gap. This is where VAE comes in.

VAE introduce new distribution - $Q(Z)$. As we will further, this $Q(Z)$ will help to decrease the gap. But, now we need to connect $Q(Z)$ and $P(X|\Theta)$.

$$\begin{gathered} \log P(X|\Theta) = \int_Z Q(Z) \log P(X|\Theta) dZ = [P(X|\Theta) = \frac{P(X,Z|\Theta)}{P(Z|X,\Theta)}] = \\ = \int_Z Q(Z) \log \frac{P(X,Z|\Theta)}{P(Z|X,\Theta)} dZ = \int_Z Q(Z) \log \frac{P(X,Z|\Theta)Q(Z)}{P(Z|X,\Theta)Q(Z)} dZ = \\ = \int_Z Q(Z) \log \frac{P(X,Z|\Theta)}{Q(Z)} dZ + \int_Z Q(Z) \log \frac{Q(Z)}{P(Z|X,\Theta)} dZ = \\ = \int_Z Q(Z) \log P(X|Z,\Theta) dZ + \int_Z Q(Z) \log \frac{P(Z)}{Q(Z)} dZ + \int_Z Q(Z) \log \frac{Q(Z)}{P(Z|X,\Theta)} dZ \end{gathered} \tag{9}$$

Finally, we have

$$\log P(X|\Theta) - \mathbb{KL}(Q(Z)||P(Z|X)) = \mathbb{E}_{z\sim Q} \log P(X|Z,\Theta) - \mathbb{KL}(Q(Z)||P(Z)) = \mathcal{L}(\Theta, Q)$$

$$\log P(X|\Theta) = \mathcal{L}(\Theta, Q) + \mathbb{KL}(Q(Z)||P(Z|X,\Theta)) \tag{10}$$

$\mathcal{L}(\Theta, Q)$ calls evidence lower bound (ELBO).

Definition 1 *Variational lower bound*

Function g(x, y(x)) calls lower bound for function f(x) if and only if
1. $\forall x, y \quad g(x, y(x)) \leq f(x)$
2. $\exists x_0 : g(x_0, y(x_0) = f(x_0)$

As can be seen $\mathcal{L}(\Theta, Q)$ satisfy conditionals of definition:

$$\begin{gathered} 1. \log P(X|\Theta) \geq \mathcal{L}(\Theta, Q); \\ \mathbb{KL}(Q(Z)||P(Z|X)) \geq 0, \forall \Theta, \Phi \end{gathered} \tag{11}$$

$$2.\ \exists Q : Q(Z) = P(Z|X,\Theta) \Rightarrow \log P(X|\Theta) = \mathcal{L}(\Theta, Q)$$

Lets look at (10) more precisely. We found "something" that can decrease the gap, it is Q. Indeed, if we maximize $\mathcal{L}(\Theta, Q)$ with respect to Q then $\mathbb{KL}(Q(Z)||P(Z|X,\Theta)$ should decrease, because $\log P(X|\Theta)$ does not depend on Q. Thus, maximization of $\mathcal{L}(\Theta, Q)$ with respect to Q decrease the gap between $\log P(X|\Theta)$ and $\mathcal{L}(\Theta, Q)$. Also, maximization by Θ moves $P(X|\Theta)$ towards $P_{data}(X)$. So, ELBO optimization is a very nice one!

$$\log P(X|\Theta) = \sum_{i=1}^{n} \log p(x_i, \Theta) \geq \sum_{i=1}^{n} \left[\int_{z_i} Q(z_i) \log p(x_i|z_i,\Theta) dz_i - \mathbb{KL}(Q(z_i)||p(z_i)) \right] \tag{12}$$

Where n is the number of training data. So, the main idea of VAE is to maximize ELBO instead of maximzing likelihood directly, because $\mathbb{KL}(Q(Z)||P(Z|X,\Theta))$ is intractable, because we dont know $P(Z|X,\Theta)$. Also, it is hard to optimizing by function. Thus, lets constrain Q by parameterical class $Q(z_i|\phi_i) = \mathbf{N}(z_i|\mu(\phi_i), \sigma(\phi_i))$ and consider $P(z_i) = \mathbf{N}(z_i|0, I)$. Here ϕ_i depends on the number of object, because for every object we want to get distribution of latent variable. We have two more problems here: 1) Using stochastic gradient descent, we will not update all parameters at the end of the epoch, thus it leads to slow convergence; 2) If we want to calculate latent distribution for a new object, we have to retrain our network, because we have no parameters ϕ_i for a new object. To solve both problems we can make dependencies on X and no dependecies on i, i.e. $Q(z_i|x_i,\phi)$. So, now ELBO looks like

$$\mathcal{L}(\Theta, \phi) = \sum_{i=1}^{n} \left[\int_{z_i} Q(z_i|x_i,\phi) \log p(x_i|z_i,\Theta) dz_i - \mathbb{KL}(Q(z_i|x_i,\phi)||p(z_i)) \right] \tag{13}$$

With assumptions:

$$\begin{gathered} Q(z_i|x_i,\phi) = \mathbf{N}(z_i|\mu(\phi,x_i),\sigma(\phi,x_i)) - \text{Encoder} \\ p(z_i) = \mathbf{N}(z_i|0,I) - \text{Prior latent distribution} \\ p(x_i|z_i,\Theta) = \mathbf{N}(x_i|f(z_i,\Theta),s^2I) - \text{Decoder} \\ x_i, f(z_i,\Theta) \in \mathbb{R}^D;\ \mu,\sigma,z_i \in \mathbb{R}^d \\ \phi,\Theta - \text{Parameters of neural network} \\ (x_1...,x_n) - \text{Training data} \end{gathered} \tag{14}$$

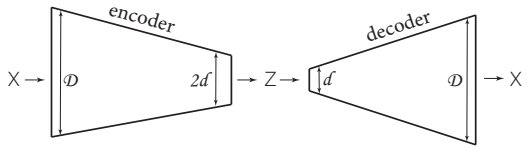
Lets calculate the second term of ELBO - $\mathbb{KL}(Q(z_i|x_i,\phi)||P(z_i))$ (will be tough)

$$\begin{gathered} \mathbb{KL}(\mathbf{N}(z_i|\mu(x_i,\phi),\sigma(x_i,\phi))||\mathbf{N}(z_i|0,I)) = \int\limits_{z_i} \mathbf{N}(z_i|\mu,\sigma)\log\frac{\mathbf{N}(z_i|\mu,\sigma)}{\mathbf{N}(z_i|0,I)}dz_i = \\ = \int\limits_{z_i} \mathbf{N}(z_i|\mu,\sigma)\log\mathbf{N}(z_i|\mu,\sigma)dz_i - \int\limits_{z_i} \mathbf{N}(z_i|\mu,\sigma)\log\mathbf{N}(z_i|0,I)dz_i = \mathbf{I}_1 - \mathbf{I}_2 \end{gathered} \tag{15}$$

Note that $\mathbf{cov}(z_i) = diag(\sigma)$

$$\begin{aligned} \mathbf{N}(z_i|\mu,\sigma) &= \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbf{cov}|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(z_i-\mu)^T\mathbf{cov}^{-1}(z_i-\mu)\right) = \\ &= \frac{1}{(2\pi)^{\frac{d}{2}}\prod_{k=1}^d\sigma_k}\exp\left(-\frac{1}{2}\sum_{j=1}^d(z_i^j-\mu_j)^2\frac{1}{\sigma_j^2}\right) \\ \mathbf{N}(z_i|0,I) &= \frac{1}{(2\pi)^{\frac{d}{2}}}\exp\left(-\frac{1}{2}z_i^Tz_i\right) = \frac{1}{(2\pi)^{\frac{d}{2}}}\exp\left(-\frac{1}{2}\sum_{j=1}^d(z_i^j)^2\right) \end{aligned} \tag{16}$$

Let's consider $\mathbf{I}_2$



$$\mathbf{I}_2 = \frac{1}{(2\pi)^{\frac{d}{2}} \prod_{k=1}^{d} \sigma_k} *$$

$$\int\limits_{z_i} \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}} \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j)^2\right)\right) dz_i =$$

$$= \frac{1}{(2\pi)^{\frac{d}{2}} \prod_{k=1}^{d} \sigma_k}$$

$$* \left[\int\limits_{z_i} \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}}\right) dz_i - \frac{1}{2}\int\limits_{z_i} \sum_{j=1}^{d}(z_i^j)^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) dz_i\right] =$$

$$= \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}}\right) - \frac{1}{2(2\pi)^{\frac{d}{2}} \prod_{k=1}^{d} \sigma_k} \mathbf{J} \tag{17}$$

$$\mathbf{J} = \int\limits_{z_i} \sum_{j=1}^{d}(z_i^j)^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) dz_i = \int\limits_{z_i} (z_i^1)^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) dz_i + ... +$$

$$+ \int\limits_{z_i} (z_i^d)^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{d}(z_i^j - \mu_j)^2 \frac{1}{\sigma_j^2}\right) dz_i = \mathbf{J_1} + ... + \mathbf{J_n} \tag{18}$$

$$\mathbf{J_1} = \int\limits_{-\infty}^{\infty} (z_i^1)^2 \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1 ... \int\limits_{-\infty}^{\infty} \exp\left(-\frac{1}{2}(z_i^d - \mu_d)^2 \frac{1}{\sigma_d^2}\right) dz_i^d =$$

$$= \int\limits_{-\infty}^{\infty} (z_i^1)^2 \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1 (2\pi)^{\frac{d-1}{2}} \prod_{k=2}^{d} \sigma_k =$$

$$= (2\pi)^{\frac{d-1}{2}} \prod_{k=2}^{d} \sigma_k \int\limits_{-\infty}^{\infty} ((z_i^1 - \mu_1)^2 + 2\mu_1 z_i^1 - \mu_1^2) \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1 =$$

$$= (2\pi)^{\frac{d-1}{2}} \prod_{k=2}^{d} \sigma_k \left[\int\limits_{-\infty}^{\infty} (z_i^1 - \mu_1)^2 \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1 + 2\mu_1 \int\limits_{-\infty}^{\infty} z_i^1 \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1\right] -$$

$$-(2\pi)^{\frac{d-1}{2}} \prod_{k=2}^{d} \sigma_k \mu_1^2 \int\limits_{-\infty}^{\infty} \exp\left(-\frac{1}{2}(z_i^1 - \mu_1)^2 \frac{1}{\sigma_1^2}\right) dz_i^1 = (2\pi)^{\frac{d}{2}} \prod_{k=1}^{d} \sigma_k (\sigma_1^2 + \mu_1^2)$$

$$\mathbf{J} = (2\pi)^{\frac{d}{2}} \prod_{k=1}^{d} \sigma_k \sum_{j=1}^{d} (\sigma_j^2 + \mu_j^2) \tag{19}$$

$$\mathbf{I}_2 = \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}}\right) - \frac{1}{2}\sum_{j=1}^{d}(\sigma_j^2 + \mu_j^2) \tag{20}$$

Analogue

$$\mathbf{I}_1 = \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}}\right) - \frac{1}{2}\sum_{j=1}^{d}(1+\log\sigma_j^2) \tag{21}$$

Finally

$$\mathbb{KL}(\mathbf{N}(z_i|\mu(x_i,\phi),\sigma(x_i,\phi))||\mathbf{N}(z_i|0,I)) = -\frac{1}{2}\sum_{j=1}^{d}(1+\log\sigma_j(x_i,\phi)^2 - \sigma_j(x_i,\phi)^2 - \mu_j(x_i,\phi)^2) \tag{22}$$

Our goal is to find a gradient for maximizing ELBO

$$\begin{gathered}\nabla_{\Theta,\phi}\mathcal{L}(\Theta,\phi) \\ = \nabla_{\Theta,\phi}\sum_{i=1}^{n}\left[\int\limits_{z_i} Q(z_i|x_i,\phi)\log p(x_i|z_i,\Theta)dz_i + \frac{1}{2}\sum_{j=1}^{d}(1+\log\sigma_j(x_i,\phi)^2 - \sigma_j(x_i,\phi)^2 - \mu_j(x_i,\phi)^2)\right] = \\ = \sum_{i=1}^{n}\nabla_{\Theta,\phi}\mathcal{L}_i(\Theta,\phi) - ?\end{gathered} \tag{23}$$

We will use stochastic gradient for gradient estimation

$$\nabla_{\Theta,\phi}\mathcal{L}(\Theta,\phi) \approx \frac{n}{m}\sum_{i=1}^{m}\nabla_{\Theta,\phi}\mathcal{L}_i(\Theta,\phi),\ \ i\sim\mathbf{U}(1,...,n) \tag{24}$$

We need to take gradient from intergral in ELBO, we cant just estimate this integral by Monte-Carlo. Lets make reparameterization trick

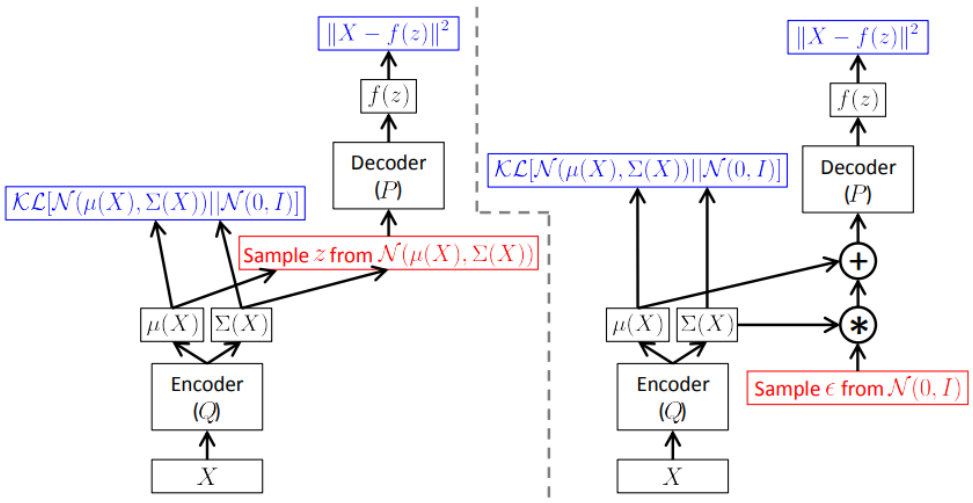
$$\begin{gathered}\int\limits_{z_i} Q(z_i|x_i,\phi)\log p(x_i|z_i,\Theta)dz_i = \int\limits_{\epsilon} p(\epsilon)\log p(x_i|\epsilon\sigma(x_i,\phi)+\mu(x_i,\phi),\Theta)d\epsilon \approx \\ \approx \frac{1}{V}\sum_{v=1}^{V}\log p(x_i|\epsilon_v\sigma(x_i,\phi)+\mu(x_i,\phi),\Theta),\ \ \epsilon_v\sim\mathbf{N}(\epsilon_v|0,I) \\ p(x_i|\epsilon_v\sigma(x_i,\phi)+\mu(x_i,\phi),\Theta) = \mathbf{N}(x_i|f_\Theta(\epsilon_v\sigma(x_i,\phi)+\mu(x_i,\phi)),s^2I)\end{gathered} \tag{25}$$

Now everything is done, we are ready to write final optimization problem

$$\begin{gathered}\nabla_{\Theta,\phi}\mathcal{L}(\Theta,\phi) \\ \approx \frac{n}{m}\sum_{i=1}^{m}\nabla_{\Theta,\phi}[-\frac{1}{V}\sum_{v=1}^{V}||f_\Theta(\epsilon_v\sigma(x_i,\phi)+\mu(x_i,\phi)) - x_i||^2 \\ +\frac{1}{2}\sum_{j=1}^{d}(1+\log\sigma_j(x_i,\phi)^2 - \sigma_j(x_i,\phi)^2 - \mu_j(x_i,\phi)^2)]\end{gathered}$$

$$\begin{gathered}V - \text{number of sampling for Monte-Carlo estimation}; m - \text{batch size}; n - \text{training size}; \\ \Theta,\phi - \text{parameters of NN}; \\ f_\Theta, x_i \in \mathbb{R}^D\ (D - \text{data dimension});\ \ \sigma_\phi,\mu_\phi\in\mathbb{R}^d\ (d - \text{latent dimension})\end{gathered} \tag{26}$$

This procedure calls double stochastic derivation or variational Bayesian derivation. Schematically it's looks like Figure 1



2 Information theory

Here some details about information theory needed in generative models will be discussed. First of all, lets consider an entropy of random variable X with distribution $P(X)$ (everything will be considered for discrete variables):

$$\mathbb{H}(X) = -\sum_i P(x_i) \log P(x_i) \tag{27}$$

Here x_i is a realization of the random variable X. This is called a measure of uncertainty. Thus, entropy has a maximum in cases when we do not know anything about variable, so $P = 0.5$. Lets show it in an example of random variable with two possible realization:

$$\mathbb{H}(X) = -P(x_1) \log P(x_1) - (1 - P(x_1)) \log(1 - P(x_1)) \tag{28}$$

Lets take a derivative (we cant do it because of the discreteness, but who cares:))

$$\frac{\partial \mathbb{H}}{\partial P}(X) = -\left[\log P(x_1) - \log(1 - P(x_1))\right] = 0 \tag{29}$$

We have following extremes: $P = 0.5, 0, 1$ and 0.5 is a maximum.

Also, we can calculate a KL-divergence between two probability distributions to measure the dissimilarity between them

$$\mathbb{KL}(p||q) = \sum_i p_i \log \frac{p_i}{q_i} = \sum_i p_i \log p_i - \sum_i p_i \log q_i = -\mathbb{H}(p) + \mathbb{H}(p, q) \tag{30}$$

where $\mathbb{H}(p, q)$ - cross entropy. This can be interpreted as the average number of bits needed to encode sample coming from p when we use samples from q. So, if samples from both distributions are same, then the number of bits is equal. Then the regular entropy is the expected number of bits to encode sample if we use the true model. Thus, the KL-divergence is the average number of extra bits needed to encode the data, due to the fact that we used distribution q to encode the data instead of the true distribution p. If these distributions are same, then extra bits is equal to zero.

References

[Doe16] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[KW13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.