Notes on Variational Autoencoder

https://hustwj.github.io/notes/
https://github.com/hustwj/notes

ELOB

$$\begin{aligned}\ln p(\mathbf{X};\theta) &= \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z};\theta)}{q(\mathbf{Z})} d\mathbf{Z} + \int q(\mathbf{Z}) \ln \frac{q(\mathbf{Z})}{p(\mathbf{Z}|\mathbf{X};\theta)} d\mathbf{Z} \\ &= \mathbf{ELOB} + KL(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X};\theta))\end{aligned}$$

Expectation Maximization (EM)以及Variational Inference (VI)中核心问题都是求解 $\mathbf{ELOB}$ 的最大化。只不过EM中将θ作为参数来进行点估计，而VI中将θ也看作random variable，需要考虑其prior分布，所以通常将其合并到latent variable $\mathbf{Z}$中处理。关于EM和VI的详细说明参见之前的相关Notes。

上面的公式是为了方便表示采取的compact形式，$\mathbf{X} = \{x^{(n)} : 1 \le n \le N\}$是观察到数据点的集合，$x^{(n)}$是其中第$n$个数据点。$\mathbf{Z} = \{z^{(n)} : 1 \le n \le N\}$是每个数据点对应的latent variable的集合。

对于每个数据点和对应的latent variable

$$\begin{aligned}\ln p(x^{(n)};\theta) &= \int q(z^{(n)}) \ln \frac{p(x^{(n)}, z^{(n)};\theta)}{q(z^{(n)})} dz^{(n)} + \int q(z^{(n)}) \ln \frac{q(z^{(n)})}{p(z^{(n)}|x^{(n)};\theta)} dz^{(n)} \\ &= \mathbf{ELOB}^{(n)} + KL(q(z^{(n)})||p(z^{(n)}|x^{(n)};\theta))\end{aligned}$$

Notes: 了解为何可以用上面compact方式来表示，可以参考之前的**Notes on Compact representation of EM**，compact表示形式和用单个数据点表示之间关系如下：

$$\ln p(\mathbf{X};\theta) = \sum_{n=1}^{N} \ln p(x^{(n)}, \theta)$$

$$\mathbf{ELOB} = \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z};\theta)}{q(\mathbf{Z})} d\mathbf{Z} = \sum_{n=1}^{N} \int \ln \frac{p(x^{(n)}, z^{(n)};\theta)}{q(z^{(n)})} q(z^{(n)}) dz^{(n)}$$

$$KL(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X};\theta)) = \sum_{n=1}^{N} KL(q(z^{(n)})||p(z^{(n)}|x^{(n)};\theta))$$

虽然Variational Autoencoder (VAE)也可以支持full bayesian模型，也就是将θ也作为random variable。不过最初论文给出的VAE的基本例子是将θ做参数来做点估计，所以这里我们说明中我们将θ做参数处理。

$\mathbf{ELOB}$ 也可以表示以下的形式：

$$\begin{aligned}
\mathbf{ELOB} &= \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z};\theta)}{q(\mathbf{Z})} d\mathbf{Z} \\
&= \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X}|\mathbf{Z};\theta)p(\mathbf{Z};\theta)}{q(\mathbf{Z})} d\mathbf{Z} \\
&= \int q(\mathbf{Z}) \ln p(\mathbf{X}|\mathbf{Z};\theta) d\mathbf{Z} - \int q(\mathbf{Z}) \ln \frac{q(\mathbf{Z})}{p(\mathbf{Z};\theta)} d\mathbf{Z} \\
&= E_{q(\mathbf{Z})}[\ln p(\mathbf{X}|\mathbf{Z};\theta)] - E_{q(\mathbf{Z})}[\ln \frac{q(\mathbf{Z})}{p(\mathbf{Z};\theta)}] \\
&= E_{q(\mathbf{Z})}[\ln p(\mathbf{X}|\mathbf{Z};\theta)] - KL(q(\mathbf{Z})||p(\mathbf{Z};\theta))
\end{aligned}$$

到目前为止$\mathbf{ELOB}$的表示都是根据基本的Bayes公式推导而来的通用公式，并不限于**VAE**。

此外，上面$\mathbf{ELOB}$公式中的$q(\mathbf{Z})$只是表示random variable $\mathbf{Z}$的某一个概率分布，对于$\mathbf{Z}$的任何一种分布上面的公式都成立。

上面公式中有两个途径来最大化$\mathbf{ELOB}$，也就是分别通过调整θ和$q(\mathbf{Z})$。在EM中设定了$q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X};\theta)$，所以实际是不断调整$\theta$的值来逐渐最大化$\mathbf{ELOB}$。在VI中，$\theta$也

被吸收到$\mathbf{Z}$中，所以实际是通常通过变分法来不断将$q(\mathbf{Z})$调整为新的分布来逐渐最大化$\mathbf{ELOB}$。

根据**VAE**的应用场景，我们可以选择一个特定的$\mathbf{Z}$分布：$q(\mathbf{Z}|\mathbf{X};\phi)$。这里选择的分布是$\mathbf{Z}$在给定$\mathbf{X}$之后的条件分布，分布对应的参数为$\phi$。将上面$\mathbf{ELOB}$公式中的$q(\mathbf{Z})$替换成$q(\mathbf{Z}|\mathbf{X};\phi)$，整个公式依然是成立的。

Notes：**There are many values of the latent variables that don't matter in practice – by conditioning on the observed variables, we emphasize the latent variable values we actually care about: the ones mostlikely given the observations.**

We would like to be able to encode our data into the latent variable space. This conditional weighting distribution enables that encoding.

$$
\begin{aligned}
\mathbf{ELOB} &= \int q(\mathbf{Z}|\mathbf{X};\phi)\ln\frac{p(\mathbf{X},\mathbf{Z};\theta)}{q(\mathbf{Z}|\mathbf{X};\phi)}d\mathbf{Z} \\
&= \int q(\mathbf{Z}|\mathbf{X};\phi)\ln\frac{p(\mathbf{X}|\mathbf{Z};\theta)p(\mathbf{Z};\theta)}{q(\mathbf{Z}|\mathbf{X};\phi)}d\mathbf{Z} \\
&= \int q(\mathbf{Z}|\mathbf{X};\phi)\ln p(\mathbf{X}|\mathbf{Z};\theta)d\mathbf{Z} - \int q(\mathbf{Z}|\mathbf{X};\phi)\ln\frac{q(\mathbf{Z}|\mathbf{X};\phi)}{p(\mathbf{Z};\theta)}d\mathbf{Z} \\
&= E_{q(\mathbf{Z}|\mathbf{X};\phi)}[\ln p(\mathbf{X}|\mathbf{Z};\theta)] - E_{q(\mathbf{Z}|\mathbf{X};\phi)}[\ln\frac{q(\mathbf{Z}|\mathbf{X};\phi)}{p(\mathbf{Z};\theta)}] \\
&= E_{q(\mathbf{Z}|\mathbf{X};\phi)}[\ln p(\mathbf{X}|\mathbf{Z};\theta)] - KL(q(\mathbf{Z}|\mathbf{X};\phi))||p(\mathbf{Z};\theta))
\end{aligned}
$$

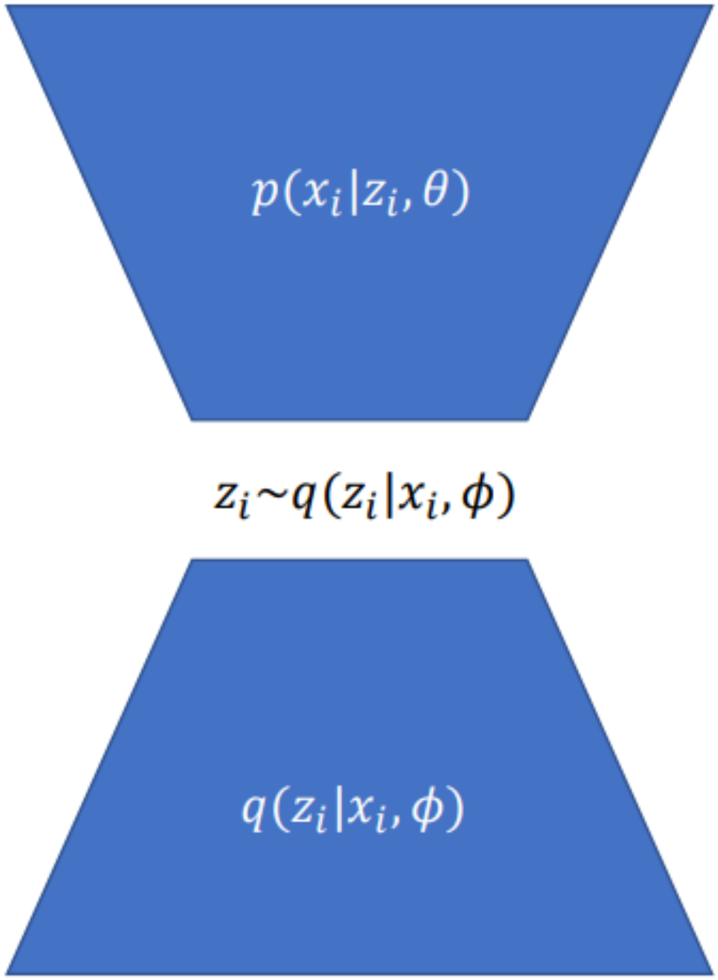
同样对每个数据点，可以表示如下

$$
\mathbf{ELOB}^{(n)} = E_{q(z^{(n)}|x^{(n)};\phi)}[\ln p(x^{(n)}|z^{(n)};\theta)] - KL(q(z^{(n)}|x^{(n)};\phi)||p(z^{(n)};\theta))
$$

Implement $p(x^{(n)}|z^{(n)};\theta)$ as a neural network, this can also be seen as a probabilistic decoder, and implement $q(\mathbf{Z}|\mathbf{X};\phi)$ as a neural network, this can also be seen as a probabilistic encoder。Sample $z^{(n)}$ from $q(\mathbf{Z}|\mathbf{X};\phi)$ in the middle.

Notes:
The basic idea is that when feed data base of $\mathbf{X}$ to encoder, the corresponding $\mathbf{Z}$ are

“forced into” to form a distribution, so that a new sample $\mathbf{z}'$ randomly drawn from this distribution creates a reasonable data.



所以原始论文中给出的基本VAE和常规的EM和VI有些不同，VAE既要对θ做点估计，同时也对$q(\mathbf{Z}|\mathbf{X};\phi)$做变分（当然最终还是体现在对参数$\phi$做点估计）。通过一系列的近似简化，最后把对$\theta$和$\phi$的优化求解问题，转化为一个常规的neural network的通过backpropagation来进行优化的问题。

在VAE中假设$p(z^{(n)};\theta)$是标准多元高斯分布

$$p(z^{(n)};\theta)) \sim \mathcal{N}(\mathbf{0},\mathbf{I})$$

同时假设其对应的posterior approximation $q(z^{(n)}|x^{(n)};\phi)$也是多元高斯分布

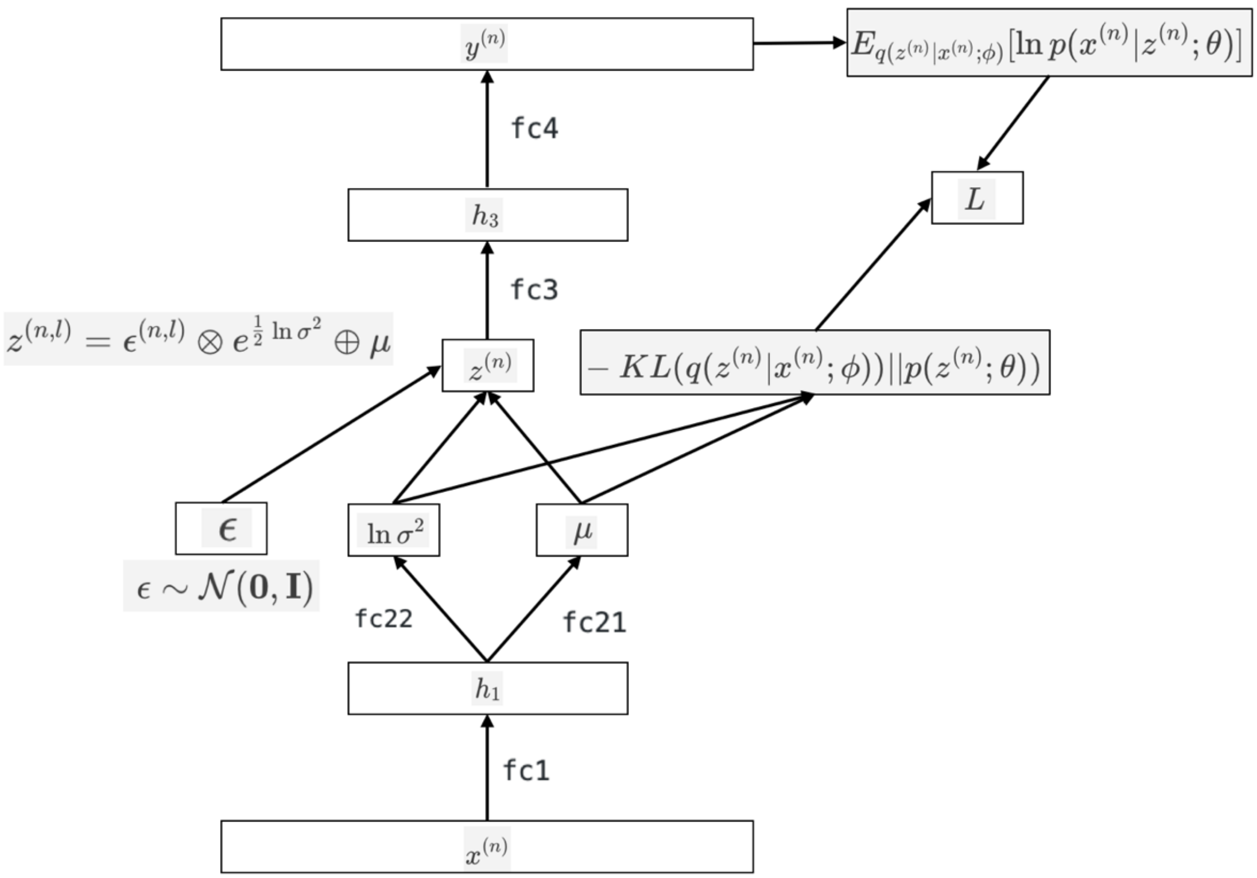
$$q(z^{(n)}|x^{(n)};\phi) \sim \mathcal{N}(\mu, \Sigma)$$

μ is a J-dimension vector, and let μ_j simply denotes the j-th element of the mean μ.

Σ 也是一个对角矩阵，所有的对角元素组成一个 J-dimension vector σ, and let σ_j simply denote the j-th element.

Notes: 实际实现时，neural network中计算用的是$\ln \sigma^2$，而不是σ。

下图是pytorch的示例代码采用的NN结构。



这里我们将 parameters of the networks, fc1, fc22 and fc21 collectively as ϕ。根据fc1, fc22 and fc21计算得到的μ和$\ln \sigma^2$作为概率分布$q(z^{(n)}|x^{(n)};\phi)$的直接参数。 $z^{(n)}$ depends in a complicated, non-linear way on $x^{(n)}$。

这里我们将 parameters of the networks, fc3 and fc4 collectively as θ。

$$
\begin{aligned}
&\int q(z^{(n)}|x^{(n)};\phi)p(z^{(n)};\theta)dz^{(n)} \\
=&\int \mathcal{N}(\mu,\Sigma)\mathcal{N}(\mathbf{0},\mathbf{I})dz^{(n)} \\
=&-\frac{J}{2}\ln(2\pi)-\frac{1}{2}\sum_{j=1}^{J}(\mu_j^2+\sigma_j^2)
\end{aligned}
$$

And

$$
\begin{aligned}
&\int q(z^{(n)}|x^{(n)};\phi)q(z^{(n)}|x^{(n)};\phi)dz^{(n)} \\
=&\int \mathcal{N}(\mu,\Sigma)\mathcal{N}(\mu,\Sigma)dz^{(n)} \\
=&-\frac{J}{2}\ln(2\pi)-\frac{1}{2}\sum_{j=1}^{J}(1+\ln\sigma_j^2)
\end{aligned}
$$

所以当$p(z^{(n)};\theta)$和$q(z^{(n)}|x^{(n)};\phi)$都是高斯分布的时候，前面公式的第二项 （常被称为KL Regularization）can be integrated analytically，也就是能通过解析得到结果。

$$
\begin{aligned}
&-KL(q(z^{(n)}|x^{(n)};\phi)||p(z^{(n)};\theta)) \\
=&\frac{1}{2}\sum_{j=1}^{J}(1+\ln\sigma_j^2-\mu_j^2-\sigma_j^2)
\end{aligned}
$$

前面公式的第一项（被称为Reconstruction error），可以用sampling的方法来近似计算。

$$
\begin{aligned}
&E_{q(z^{(n)}|x^{(n)};\phi)}[\ln p(x^{(n)}|z^{(n)};\theta)] \\
=&\frac{1}{L}\sum_{l=1}^{L}\ln p(x^{(n)}|z^{(n,l)};\theta)
\end{aligned}
$$

这里$z^{(n,l)}=\epsilon^{(n,l)}\otimes e^{\frac{1}{2}\ln\sigma^2}\oplus\mu,\ \ \epsilon^{(n,l)}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$

$\otimes$表示两个向量element-wise的相乘，而$\oplus$表示两个向量element-wise的相加。

VAE的原始论文中提到：

According to the experiments in the original VAE paper, the number of samples L per datapoint can be set to 1 as long as the minibatch size M was large enough, e.g. $M = 100$.

所以将ELOB的负值作为Loss function可以近似简化为

$$L^{(n)} = -\ln p(x^{(n)}|z^{(n,l)};\theta) - \frac{1}{2}\sum_{j=1}^{J}(1 + \ln\sigma_j^2 - \mu_j^2 - \sigma_j^2)$$

对于autoencoder，$-\ln p(x^{(n)}|z^{(n)};\theta)$的具体计算方法取决于$x^{(n)}$的值，原始论文对此做了简单的说明，而更详细的说明参见Hugo Larochelle的讲课视频**Neural networks [6.2] : Autoencoder - loss function**。

数据中$x^{(n)}$每一维的值都是0或1，或者取值都在0和1之间的时候，分别计算每一维输出的cross entropy，然后将所有维计算结果都相加就得到了$-\ln p(x^{(n)}|z^{(n)};\theta)$。pytorch的VAE示例代码中的$x^{(n)}$是先normalize为0和1之间的值了。

当数据中$x^{(n)}$每一维的值是real values的时候，可以将$p(x^{(n)}|z^{(n)};\theta)$看作服从高斯分布（均值为decoder的输出结果, 方差为常数），则可以采用MSE来计算$-\ln p(x^{(n)}|z^{(n)};\theta)$。这里类似Linear regression的loss function，具体可以参见之前的Linear regression的loss function的概率解释的notes： **Linear regression with regularization**。

VAE中用到了一个被称为Reparameterization trick的方法，在encoder中，如果直接根据$q(z^{(n)}|x^{(n)};\phi) \sim \mathcal{N}(\mu, \sigma^2)$来sampling来产生$z^{(n)}$的话，因$z^{(n)}$是sampling产生而不是一个确定的计算过程得到的结果，会导致backpropagation在这里被截断，进而无法计算gradient来进行优化。这里采取一个变通的方法，就是先从一个标准的高斯分布$\mathcal{N}(\mathbf{0}, \mathbf{I})$采样得到一个$\epsilon^{(n)}$，然后通过公式$z^{(n)} = \epsilon^{(n)} \times e^{\frac{1}{2}\ln\sigma^2} + \mu$计算得到$z^{(n)}$的值。这样采样得到的$\epsilon^{(n)}$就和$x^{(n)}$一样都是作为NN的输入了，而$z^{(n)}$是确定性的计算方法得到的，不会妨碍backpropagation进行优化。

Reference

- Auto-Encoding Variational Bayes
  https://arxiv.org/abs/1312.6114

- Neural networks [6.2] : Autoencoder - loss function
  https://www.youtube.com/watch?v=xTU79Zs4XKY

- Basic VAE example codes in pytorch
  https://github.com/pytorch/examples/blob/master/vae/main.py