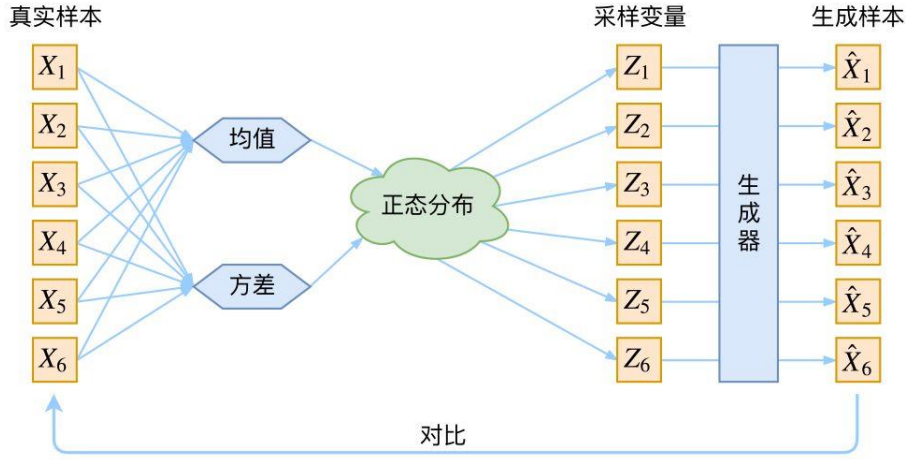
生成模型---变分自编码器

目　录

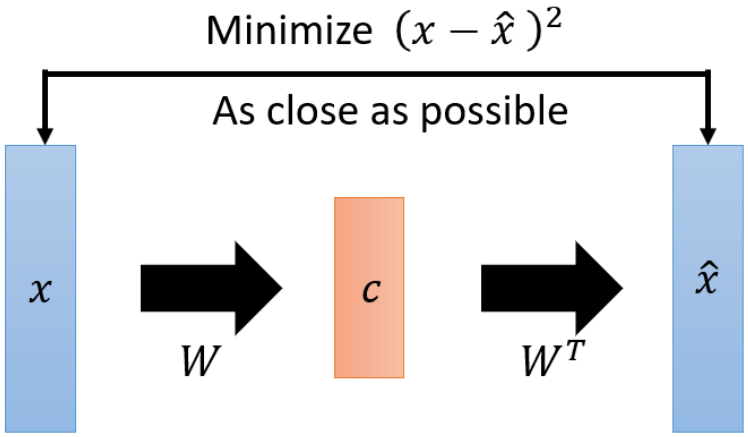
先简单介绍一下 VAE，VAE 作为一个生成模型，其基本思路是很容易理解的：把一堆真实样本通过编码器网络变换成一个理想的数据分布，然后这个数据分布再传递给一个解码器网络，得到一堆生成样本，生成样本与真实样本足够接近的话，就训练出了一个自编码器模型。那 VAE(变分自编码器)就是在自编码器模型上做进一步变分处理，使得编码器的输出结果能对应到目标分布的均值和方差，如下图所示，具体的方法和思想在后文会介绍：



1.VAE 的设计思路

VAE 最想解决的问题是什么？当然是如何构造编码器和解码器，使得图片能够编码成易于表示的形态，并且这一形态能够尽可能无损地解码回原真实图像。

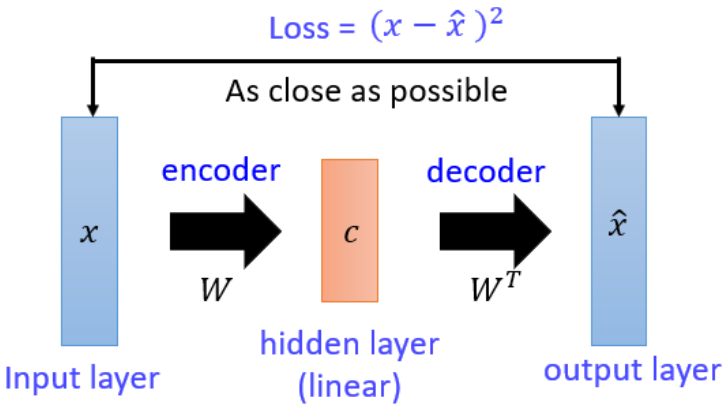
这似乎听起来与 PCA（主成分分析）有些相似，而 PCA 本身是用来做矩阵降维的：



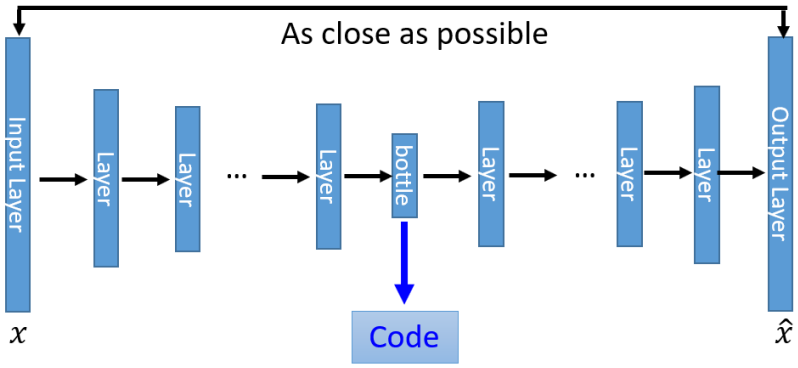
PCA

如图，X 本身是一个矩阵，通过一个变换 W 变成了一个低维矩阵 c，因为这一过程是线性的，所以再通过一个W^T变换就能还原出一个$\hat{X}$，现在我们要找到一种变换 W，使得矩阵 X 与$\hat{X}$能够尽可能地一致，这就是 PCA 做的事情。在 PCA 中找这个变换 W 用到的方法是 SVD（奇异值分解）算法，这是一个纯数学方法，不再细述，因为在 VAE 中不再需要使用 SVD，直接用神经网络代替。

回顾上述介绍，我们会发现 PCA 与我们想要构造的自编码器的相似之处是在于，如果把矩阵 X 视作输入图像，W 视作一个编码器，低维矩阵 c 视作图像的编码，然后W^T和$\hat{X}$分别视作解码器和生成图像，PCA 就变成了一个自编码器网络模型的雏形。

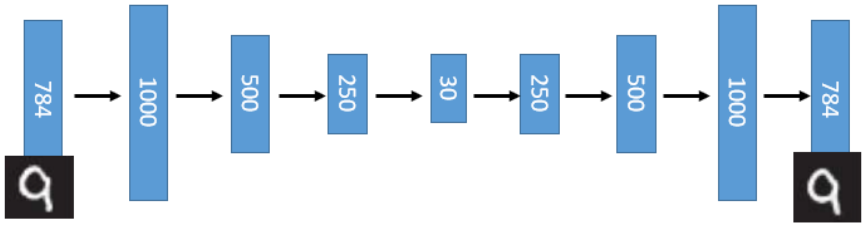


现在我们需要对这一雏形进行改进。首先一个最明显能改进的地方是用神经网络代替W变换和W^T变换，就得到了如下 Deep Auto-Encoder 模型：



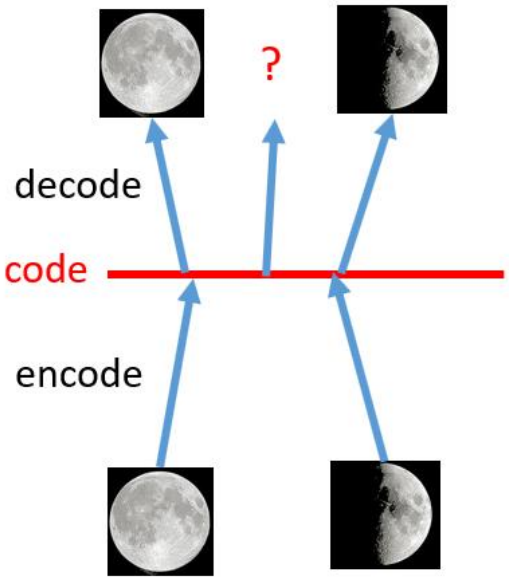
Deep Auto-Encoder 模型

这一替换的明显好处是，引入了神经网络强大的拟合能力，使得编码（Code）的维度能够比原始图像（X）的维度低非常多。在一个手写数字图像的生成模型中，Deep Auto-Encoder 能够把一个 784 维的向量（28*28 图像）压缩到只有 30 维，并且解码回的图像具备清楚的辨认度（如下图）。



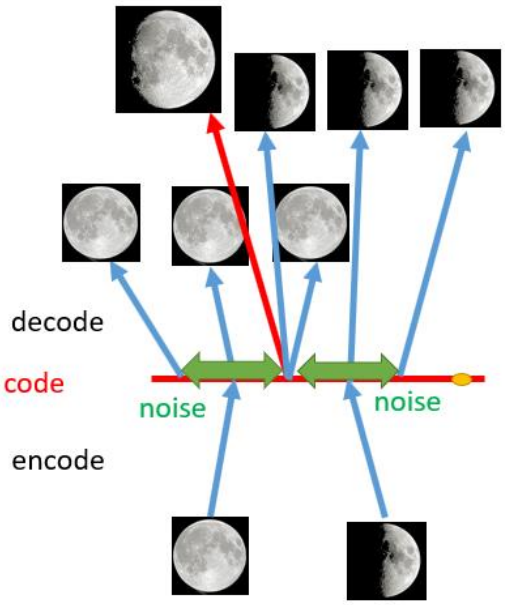
至此我们构造出了一个重构图像比较清晰的自编码模型，但是这并没有达到我们真正想要构造的生成模型的标准，因为，对于一个生成模型而言，解码器部分应该是单独能够提取出来的，并且对于在规定维度下任意采样的一个编码，都应该能通过解码器产生一张清晰且真实的图片。

我们先来分析一下现有模型无法达到这一标准的原因。



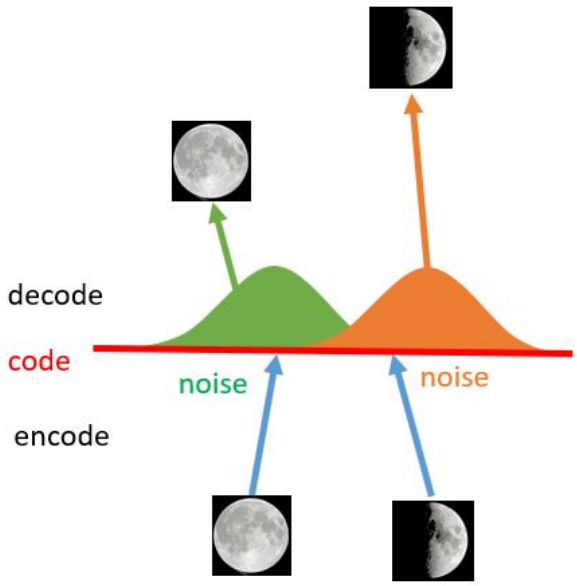
如上图所示，假设有两张训练图片，一张是全月图，一张是半月图，经过训练我们的自编码器模型已经能无损地还原这两张图片。接下来，我们在 code 空间上，两张图片的编码点中间处取一点，然后将这一点交给解码器，我们希望新的生成图片是一张清晰的图片（类似 3/4 全月的样子）。但是，实际的结果是，生成图片是模糊且无法辨认的乱码图。一个比较合理的解释是，因为编码和解码的过程使用了深度神经网络，这是一个非线性的变换过程，所以在 code 空间上点与点之间的迁移是非常没有规律的。

如何解决这个问题呢？我们可以引入噪声，使得图片的编码区域得到扩大，从而掩盖掉失真的空白编码点。



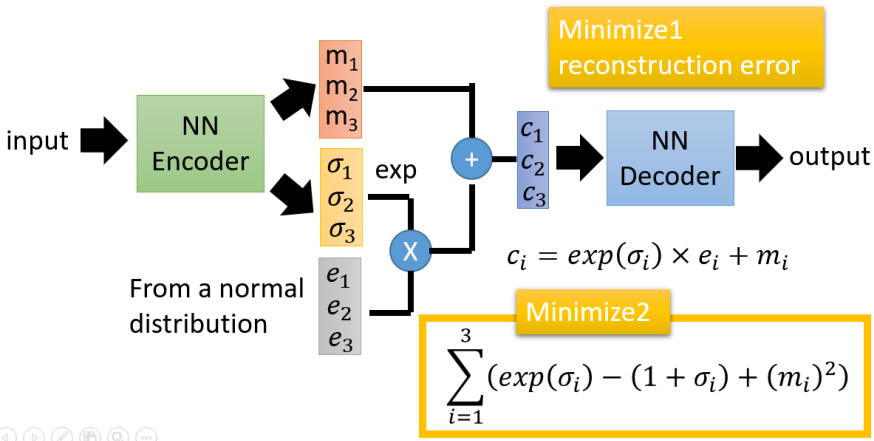
如上图所示，现在在给两张图片编码的时候加上一点噪音，使得每张图片的编码点出现在绿色箭头所示范围内，于是在训练模型的时候，绿色箭头范围内的点都有可能被采样到，这样解码器在训练时会把绿色范围内的点都尽可能还原成和原图相似的图片。然后我们可以关注之前那个失真点，现在它处于全月图和半月图编码的交界上，于是解码器希望它既要尽量相似于全月图，又要尽量相似于半月图，于是它的还原结果就是两种图的折中（3/4 全月图）。

由此我们发现，给编码器增添一些噪音，可以有效覆盖失真区域。不过这还并不充分，因为在上图的距离训练区域很远的黄色点处，它依然不会被覆盖到，仍是个失真点。为了解决这个问题，我们可以试图把噪音无限拉长，使得对于每一个样本，它的编码会覆盖整个编码空间，不过我们得保证，在原编码附近编码的概率最高，离原编码点越远，编码概率越低。在这种情况下，图像的编码就由原先离散的编码点变成了一条连续的编码分布曲线，如下图所示。



那么上述的这种将图像编码由离散变为连续的方法，就是变分自编码的核心思想。下面就会介绍 VAE 的模型架构，以及解释 VAE 是如何实现上述构思的。

2.VAE 的模型架构



上面这张图就是 VAE 的模型架构，我们先粗略地领会一下这个模型的设计思想。

在 auto-encoder 中，编码器是直接产生一个编码的，但是在 VAE 中，为了给编码添加合适的噪音，编码器会输出两个编码，一个是原有编码(m_1, m_2, m_3)，另外一个是控制噪音干扰程度的编码$(\sigma_1, \sigma_2, \sigma_3)$，第二个编码其实很好理解，就是为随机噪音码$(e_1, e_2, e_3)$分配权重，然后加上 $exp(\sigma_i)$的目的是为了保证这个分配的权重是个正值，最后将原编码与噪音编码相加，就得到了 VAE 在 code 层的输出结果(c_1, c_2, c_3)。其它网络架构都与 Deep Auto-encoder 无异。

损失函数方面，除了必要的重构损失外，VAE 还增添了一个损失函数（见上图 Minimize2 内容），这同样是必要的部分，因为如果不加的话，整个模型就会出现问题：为了保证生成图片的质量越高，编码器肯定希望噪音对自身生成图片的干扰越小，于是分配给噪音的权重越小，这样只需要将$(\sigma_1, \sigma_2, \sigma_3)$赋为接近负无穷大的值就好了。所以，第二个损失函数就有限制编码器走这样极端路径的作用，这也从直观上就能看出来，$\exp(\sigma_i)-(1+\sigma_i)$在$\sigma_i=0$ 处取得最小值，于是$(\sigma_1, \sigma_2, \sigma_3)$就会避免被赋值为负无穷大。

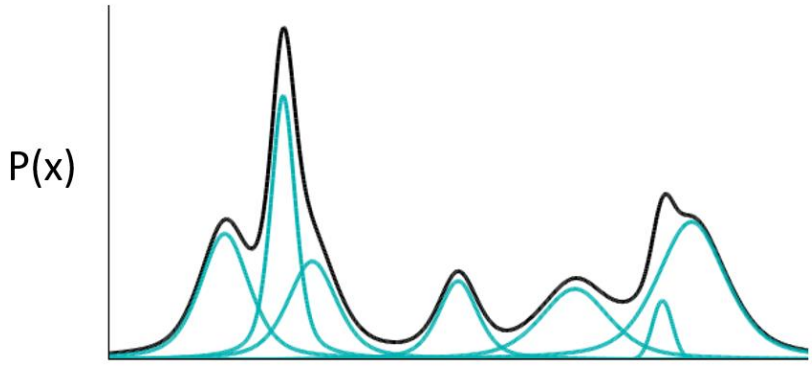
上述我们只是粗略地理解了 VAE 的构造机理，但是还有一些更深的原理需要挖掘，例如第二个损失函数为何选用这样的表达式，以及 VAE 是否真的能实现我们的预期设想，即“图片能够编码成易于表示的形态，并且这一形态能够尽可能无损地解码回原真实图像”，是否有相应的理论依据。

下面我们会从理论上深入地分析一下 VAE 的构造依据以及作用原理。

3.VAE 的作用原理

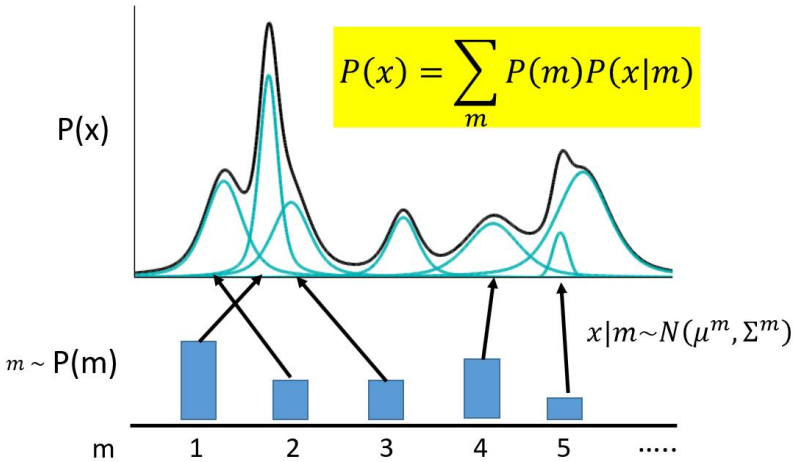
我们知道，对于生成模型而言，主流的理论模型可以分为隐马尔可夫模型 HMM、朴素贝叶斯模型 NB 和高斯混合模型 GMM，而 VAE 的理论基础就是高斯混合模型。

什么是高斯混合模型呢？就是说，任何一个数据的分布，都可以看作是若干高斯分布的叠加。



如图所示，如果 P(X)代表一种分布的话，存在一种拆分方法能让它表示成图中若干浅蓝色曲线对应的高斯分布的叠加。有意思的是，这种拆分方法已经证明出，当拆分的数量达到 512 时，其叠加的分布相对于原始分布而言，误差是非常非常小的了。

于是我们可以利用这一理论模型去考虑如何给数据进行编码。一种最直接的思路是，直接用每一组高斯分布的参数作为一个编码值实现编码。

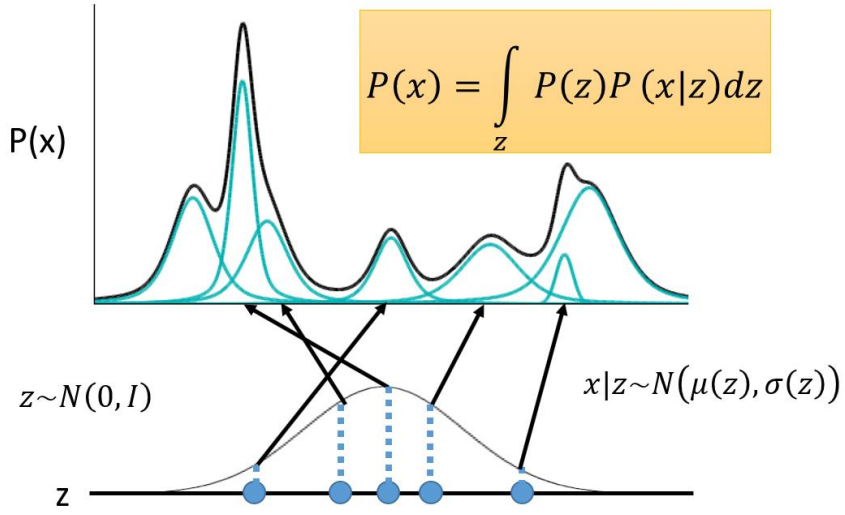


如上图所示，m 代表着编码维度上的编号，譬如实现一个 512 维的编码，m 的取值范围就是 1,2,3……512。m 会服从于一个概率分布 P(m)（多项式分布）。现在编码的对应关系是，每采样一个 m，其对应到一个小的高斯分布$N(\mu^m, \Sigma^m)$，P(X)就可以等价为所有的这些高斯分布的叠加，即：

$$P(x) = \sum_m P(m)P(x|m)$$

其中，$m \sim P(m)$，$x|m \sim N(\mu^m, \Sigma^m)$。

上述的这种编码方式是非常简单粗暴的，它对应的是我们之前提到的离散的、有大量失真区域的编码方式。于是我们需要对目前的编码方式进行改进，使得它成为连续有效的编码。



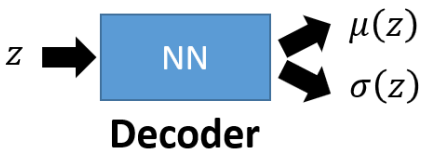
现在我们的编码换成一个连续变量 z，我们规定 z 服从正态分布$N(0,1)$（实际上并不一定要选用$N(0,1)$，其他的连续分布都是可行的）。每对于一个采样 z，会有两个函数μ和σ，分别决定 z 对应到的高斯分布的均值和方差，然后在积分域上所有的高斯分布的累加就成为了原始分布 P(X),即：

$$P(x) = \int_z P(z)P(x|z)dz$$

其中$z \sim N(0,1)$，$x|z \sim N(\mu(z), \sigma(z))$。

接下来就可以求解这个式子。由于$P(z)$是已知的，$P(x|z)$未知，而$x|z \sim N(\mu(z), \sigma(z))$，于是我们真正需要求解的，是$\mu$和$\sigma$两个函数的表达式。又因为 P$(x)$通常非常复杂，导致$\mu$和$\sigma$难以计算，我们需要引入两个神经网络来帮助我们求解。

第一个神经网络叫做 Decoder，它求解的是μ和σ两个函数，这等价于求解$P(x|z)$。



第二个神经网络叫做 Encoder，它求解的结果是$q(z|x)$，q可以代表任何分布。

值得注意的是，这儿引入第二个神经网路 Encoder 的目的是，辅助第一个 Decoder 求解$\mathrm{P}(x|z)$，这也是整个 VAE 理论中最精妙的部分，下面我会详细地解释其中的奥妙。

我们先回到最开始要求解的目标式。

$$P(x) = \int_z P(z)P\,(x|z)dz$$

我们希望$P(x)$越大越好，这等价于求解：

$$Maximum\ L = \sum_x logP(x)$$

注意到：

$$logP(x) = \int_z q(z|x)logP(x)dz \qquad (\mathrm{q}(z|x)\text{可以是任何分布})$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\right)dz + \int_z q(z|x)log\left(\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\right)dz + KL\big(q(z|x)||P(z|x)\big)$$

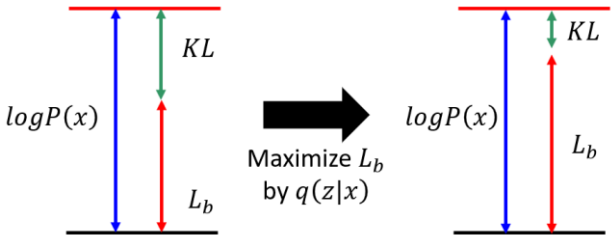
上式的第二项是一个大于等于 0 的值，于是我们就找到了一个$logP(x)$的下界：

$$logP(x) \geq \int_z q(z|x)log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz$$

我们把这个下界记作$L_b = \int_z q(z|x)log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz$。

于是原式化为：$logP(x) = L_b + KL\big(q(z|x)||P(z|x)\big)$。

接下类，VAE 思维的巧妙设计就体现出来了。原本，我们需要求$\mathrm{P}(x|z)$使得$logP(x)$最大，现在引入了一个$q(z|x)$，变成了同时求$\mathrm{P}(x|z)$和$q(z|x)$使得$logP(x)$最大。不妨观察一下$logP(x)$和L_b的关系：



一个有趣的现象是，当我们固定住$\mathrm{P}(x|z)$时，因为$logP(x)$只与$\mathrm{P}(x|z)$有关，所以$logP(x)$的值是会不变的，此时我们去调节$q(z|x)$，使得L_b越来越高，同时 KL 散度越来越小，当我们调节到$q(z|x)$与$P(z|x)$完全一致时，KL 散度就消失为 0，L_b与$logP(x)$完全一致。由此可以得出，不论$logP(x)$的值如何，我们总能够通过调节使得L_b等于$logP(x)$，又因为L_b是$logP(x)$的下界，所以求解$Maximum\ logP(x)$等价为求解$Maximum\ L_b$。

这个现象从宏观上来看也是很有意思，调节P$(x|z)$就是在调节Decoder，调节$q(z|x)$就是在调节Encoder。于是，VAE 的训练逻辑就变成了，Decoder每前进一步，Encoder就调节成与其一致的样子，并且站在那拿“枪”顶住Decoder，这样Decoder在下次训练的时候就只能前进，不能退步了。

上述便是 VAE 的巧妙设计之处。再回到我们之前的步骤上，现在需求解$Maximum\ L_b$。

注意到：

$$L_b = \int_z q(z|x) log\left(\frac{P(z,x)}{q(z|x)}\right) dz$$

$$= \int_z q(z|x) log\left(\frac{P(x|z)P(z)}{q(z|x)}\right) dz$$

$$= \int_z q(z|x) log\left(\frac{P(z)}{q(z|x)}\right) dz + \int_z q(z|x) log P(x|z) dz$$

$$= -KL\big(q(z|x)||P(z)\big) + \int_z q(z|x) log P(x|z) dz$$

所以，求解$Maximum\ L_b$，等价于求解$KL\big(q(z|x)||P(z)\big)$的最小值和$\int_z q(z|x) log P(x|z) dz$的最大值。

我们先来求第一项，其实$-KL\big(q(z|x)||P(z)\big)$的展开式刚好等于：

$$\sum_{i=1}^{J} (exp(\sigma_i) - (1 + \sigma_i) + (m_i)^2)$$

具体的展开计算过程可以参阅《Auto-Encoding Variational Bayes》的 Appendix B。

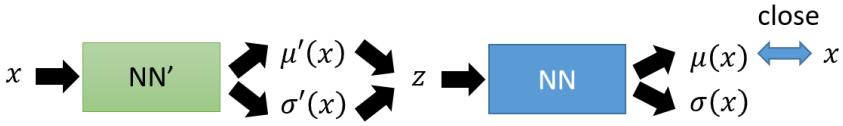
于是，第一项式子就是第二节 VAE 模型架构中第二个损失函数的由来。

接下来求第二项，注意到

$$Maximum \int_z q(z|x) log P(x|z) dz$$

$$= Maximum\ E_{q(z|x)}[log P(x|z)]$$

上述的这个期望，也就是表明在给定$q(z|x)$（编码器输出）的情况下$P(x|z)$（解码器输出）的值尽可能高，这其实就是一个类似于 Auto-Encoder 的损失函数（方差忽略不计的话）：



因此，第二项式子就是第二节 VAE 模型架构中第一个损失函数的由来。

综上，关于 VAE 模型架构中的理论证明部分至此全部介绍完毕。

4.致谢及引用

很感谢李宏毅老师的教程视频，讲得实在是简单通透，视频地址如下：
https://www.bilibili.com/video/av15889450/?p=33

课程资源地址如下：
http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML17_2.html