一.VAE 算法介绍以及为什么引入 VAE 算法

1.VAE 算法（显式密度估计）

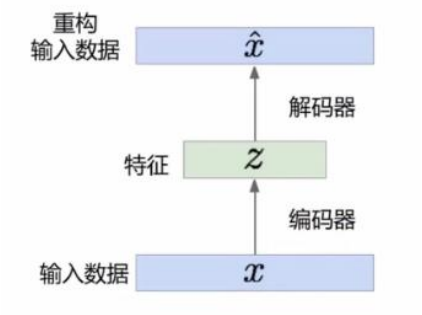
显式密度估计：显式的定义并求解分布 Pmodel(x)，分布的方程是可以定义并写出的。可以算出特征空间中选取的点生成样本的概率值等。

2.AE 算法：

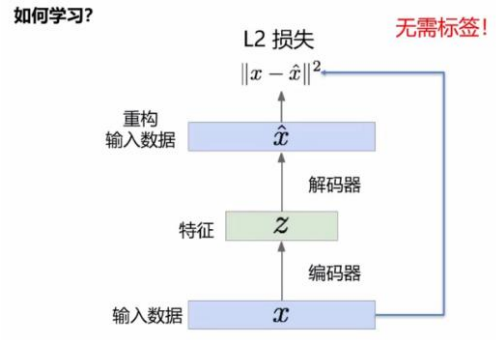
目标：无监督学习，其目标是利用无标签的数据找到一个有效地低维的特征提取器。

特征降维的作用：希望降维后的特征仅保留数据中有用的信息。

思路：把输入数据 x 输入到编码器产生 z 编码，再由解码器产生重构数据$\hat{x}$，如果 x 和$\hat{x}$越接近，说明 z 包含 x 的信息特征越多。如果训练的很好的话，则输入数据和重构输入数据应该会很相似，特征码 z 也是我们所希望得到的东西。

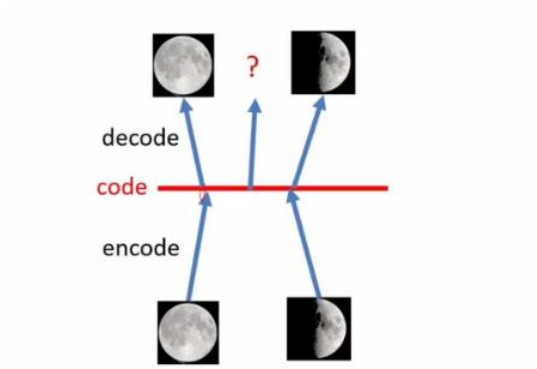


如何学习？



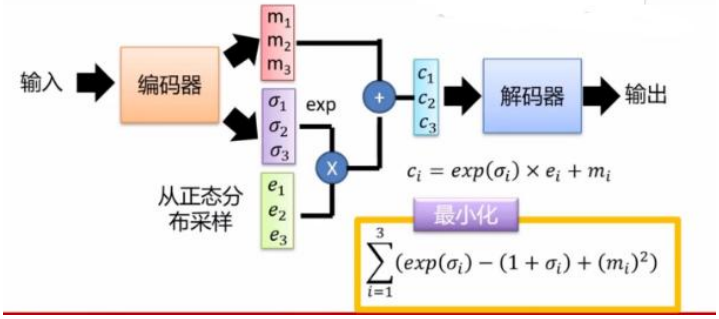
通过 L2 损失作为损失函数，调整编码器和解码器中的参数，在这一过程中是不需要标签的。学好之后，我们便可以单独用解码器和编码器做一些特定的需求。比如我们可以用编码器做分类，已训练的编码器可以作为有监督学习的初始特征提取模型。解码器可以通过输入相应的码 z 得到重构的输入数据。比如生成 MNIST 图像。

3.引入 VAE 原因



现在我有两张图片，分别是圆月和半月的图片，我把这两张图片进行训练，可以得到相应的 z 编码，假如圆月代表的 z 编码为 1，半月代表的为 10。现在我有一个问题，比如我输入 z 编码为 5，希望能够生成相应的图像。对我们人来说，由于 5 在 1 和 10 之间，故希望能够生成介于圆月和半月之间的图像（比如 3/4 个月亮），但是我们在输入数据里面并没有这样的图像，故在实际输出时可能达不到我们的想法。所以引入了 VAE（变分自编码器）。

4.VAE 流程：



输入数据 x，不是直接生成码 z，而是生成了一个码(m1,m2,m3)和方差($\sigma_1, \sigma_2, \sigma_3$)。这代表了一个分布，其中 m 可以代表生成分布的均值，σ代表标准差(这里是三维的)。这样的话我在生成图像时便是从这个分布中随机取一个值(由均值、标准差、偏移量三个条件确定这个值)来得到我们的重构的图像。

Z 的公式为：　（重参数化技巧）

$$\boldsymbol{z} = \boldsymbol{m} + \exp(\boldsymbol{\sigma}) * \boldsymbol{e}$$

标粗是因为代表一个向量，按照上图的话便是三维的。这样的话我们发现 VAE 在得到 z 时是运用了预测出来的分布并在其中随机采了一个值。该分布是从原始数据中学习的。这样的话就由此方法得到的 z 可以解决上述 AE 中的问题。

那么问题又来了，假如我的损失函数还是$\|x - \hat{x}\|^2$可以吗?

答案是否，如果还是上述损失函数，可能会导致我们的模型倒退为自编码（AE）。

5.大概思路：

在 VAE 中，我们使用概率分布来表示潜在空间中的数据点。编码器网络将输入数据映射到潜在空间中的概率分布（连续的），而解码器网络则将潜在空间中的点映射回原始数据空间。训练 VAE 的目标是最大化数据的似然性，即最大化观测数据 x 的条件概率 p(x)同时最小化编码器网络生成的潜在空间分布与预先定义的先验分布之间的差异，通常是一个高斯分布。

VAE 的一个重要原因是它能够学习数据的连续潜在表示，这使得它能够在潜在空间中进行插值和采样，从而生成具有变化的新样本。这种连续潜在表示有助于生成更加真实和多样化的数据，同时也提供了一种有效的方式来进行数据压缩和降维。因此，VAE 在生成模型和数据表示学习方面具有广泛的应用。

二、公式推导

对于 vae 模型，我们有$P(x) = \int P(z) * P(x|z)\, dz$ [1]

其中$P(z)$为隐变量 z 先验分布的概率密度函数，$P(x|z)$为已知 z 时观测变量 x 的条件概率密度函数。但是 P(x)实际很难求出，因为 P(z)是连续的，（我们假设其为标准正态分布）要求出 P(x)需要对 z 求积分。

我们考虑对 P(x)进行极大似然估计：

$$L = \sum_{x} \log P(x)$$

$$logP(x) = \int q(z|x) logP(x) dz$$

其中 q(z|x)是引入的变分密度函数，它可以代表任何分布。

类似 EM 算法，我们对上述公式进行处理：

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

$$= ELBO + KL(q(z|x)||P(z|x))$$

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

$$= -KL(q(z|x)\big|\big|P(z)) + E_{q(z|x)}[logP(x|z)]$$

我们可以通过 EM 算法对其进行求解，步骤为：

（1） E 步：寻找一个密度函数$q(z|x)$使其接近或等于后验密度函数$P(z|x)$。

（2） M 步：最大化 ELBO。

在 EM 算法每次迭代中。理论上最优的$q(z|x)$为隐变量后验概率密度函数$P(z|x)$。但$P(z|x)$也是很难计算的，通常需要变分推断进行估计。具体地说，在 VAE 中，我们使用编码器 $q(z|x)$ 来近似表示后验分布 $P(z|x)$，编码器输出的均值和方差（或其他参数化方式）被用于参数化 $q(z|x)$ 。然后，我们使用参数化的 $q(z|x)$来代替真实的 $P(z|x)$，从而简化了推断问题，并且使得模型的训练变得可行。

VAE 的思想就是利用神经网络分别建模两个复杂的条件概率密度函数。

（1） 用神经网络估计变分分布$q(z|x)$，这一步骤称为推断网络。推断网络输入为 x，输出变分分布为$q(z|x)$。

（2） 用神经网络估计概率分布$P(x|z)$，称为生成网络，生成网络输入为 z，输出概率分布$P(x|z)$。

变分自编码器的名称来自于其整个网络结构和自编码器比较类似。推断网络看作是“编码器”，将可观测变量映射为隐变量。生成网络可以看作是“解码器”，将隐变量映射为可观测变量。但变分自编码器背后的原理和自编码器完全不同。变分自编码器中的编码器和解码器的输出为分布（或分布的参数），而不是确定的编码。

1.推断网络的目标：

推断网络的目标是使得$q(z|x)$ 尽可能接近后验分布 $P(z|x)$，假设φ为$q(z|x)$的参数，我们要找到φ^*去最小化两个分布的 KL 散度。

即：

$$\varphi^* = arg\min_{\varphi} KL(q(z|x)||P(z|x))$$

我们可以把上式转换一下：即推断网络的目标函数是

$$\varphi^* = arg\max_{\varphi} ELBO$$

2.生成网络的目标：

假设θ为$P(x|z)$的参数，生成网络的目标是找到一组网络参数θ^*去最大化 ELBO，即：

$$\theta^* = arg\max_{\theta} ELBO$$

3.模型的汇总：

推断和生成网络的目标为最大化 ELBO，因此，vae 总目标函数为：

$$\max_{\theta,\ \varphi} ELBO = max\left(-KL(q(z|x;\varphi)||P(z)) + E_{q(z|x)}[logP(x|z;\theta)]\right)$$

其中 P(z)为先验分布，θ、φ 为生成网络和推断网络对应的参数。

其中 KL 散度项可以计算，为：

$$KL = -0.5\sum_{i=1}^{K}\left(1 + \log\left(\sigma_i^2\right) - \sigma_i^2 - \mu_i^2\right)$$

其中$q(z|x)$是多元正态分布，P(z)是标准正态分布，μ、σ为$q(z|x)$对应的均值和标准差，也即推断网络的输出。

4.损失函数：

VAE 损失函数一般包括两部分：重建损失和 KL 散度。

如果我们的重建损失是 MSE，则

$$recon\_loss = \frac{1}{N}\sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2$$

$$KL = -0.5\sum_{i=1}^{K} (1 + \log(\sigma_i^2) - \sigma_i^2 - \mu_i^2)$$

$$VAE_{Loss} = recon\_loss + KL$$

其中 N 代表样本个数。

x_i代表第 i 个样本。

$\hat{x}_i$代表重建后的第 i 个样本。

参考文献：

[1]神经网络和深度学习-邱锡鹏