

变分自编码器

xbZhong

2025-06-20

[本页PDF](#)

VAE (Variational Autoencoder 变分自编码器)

- 与 AE 结构相似，但是用了特殊技巧使得 VAE 的 decoder 对于一个 随机的向量 可以产生结果较好的图

自编码器(AE)与VAE的对比

自编码器(AE)的局限性

- 自编码器将输入图像压缩为低维潜在向量，然后通过解码器重建图像
- 关键局限：形成的是严格的一对一映射关系
 - 特定输入 → 特定潜在向量 → 特定重建图像
- 潜在空间不连续，缺乏良好的插值特性
- 无法生成新的、有意义的样本

VAE的突破

- VAE 不是学习确定的潜在向量，而是学习概率分布
- 每个输入被映射到潜在空间中的分布(通常是高斯分布)，由均值向量(μ)和方差向量(σ^2)确定
- 核心思想：**通过学习分布而非单点表示，实现了潜在空间的连续性和平滑性
- 通过学习分布而非固定点，VAE 能够在潜在空间中的相邻点产生相似但不完全相同的图像，使得整个潜在空间连续且有意义，从而可以从中随机采样生成新的图像。

核心概念

- $P(x)$ 为真实数据的分布，但我们并不知道
- z 是 x 的特征，作为解码器的输入，解码器会将 z 映射到 $P(x)$ 上，输出 x 的条件分布
 - 在这里假设 z 服从标准正态分布，但 z 是无法穷举的，因为特征是一个抽象的东西
- VAE 的真正训练目标是最大化数据的边缘似然 $P(x)$
 - $P(x) = \int P(z)P(x|z)$
 - z 是一个正态分布(通常是标准正态分布 $N(0, I)$)
 - 解码器将 z 空间中的每个点映射到 x 空间中的概率分布
 - 最终的 $P(x)$ 正是这些单个正态分布的积分(叠加)

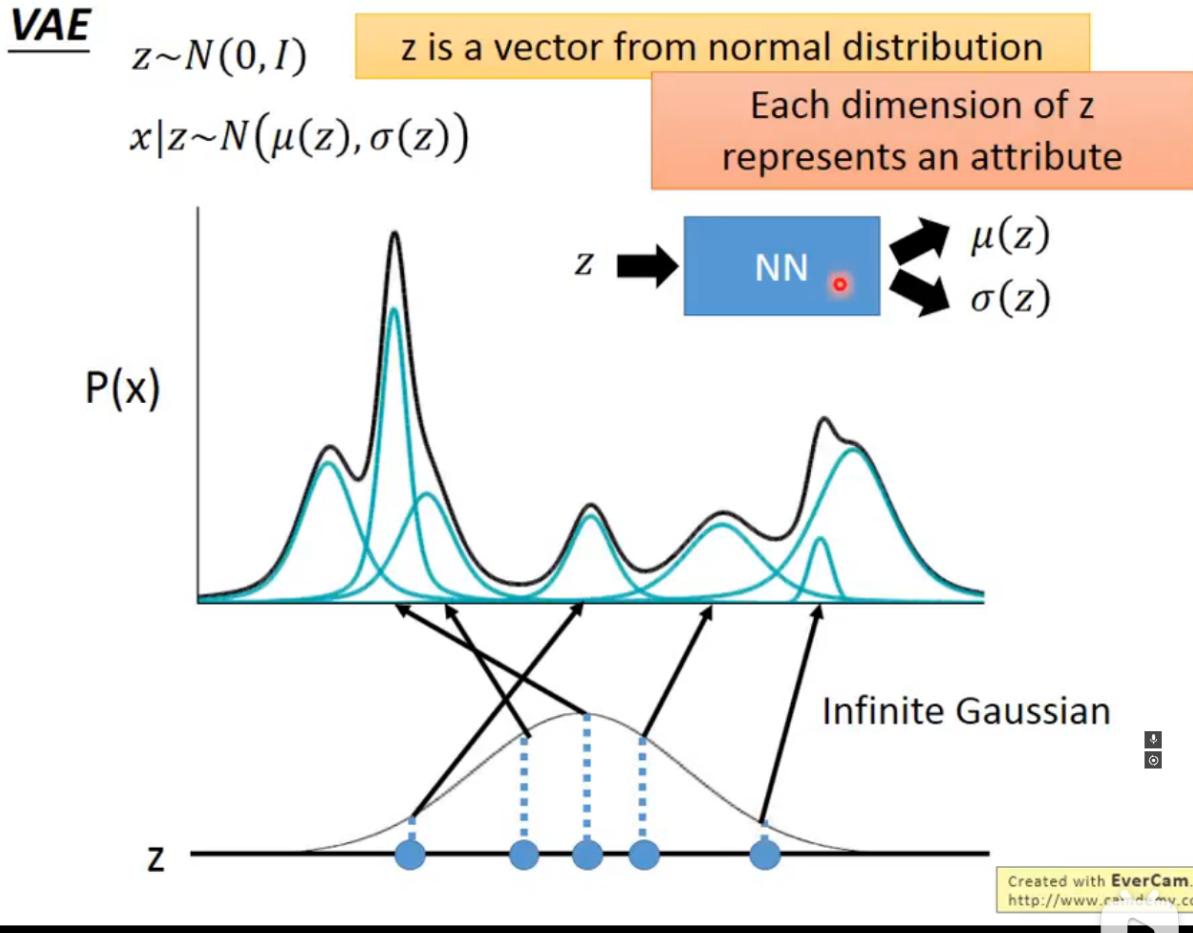


image-20250430233601234

- **Decoder:** 将潜在变量 z 作为输入，通过神经网络 (NN) 输出条件分布 $P(x|z)$ 的参数，即均值 $\mu(z)$ 和方差 $\sigma(z)$ 。这决定了给定 z 时 x 的概率分布 $P(x|z)$
- **Encoder:** 将观测数据 x 作为输入，通过另一个神经网络 (NN') 输出近似后验分布 $q(z|x)$ 的参数，即均值 $\mu'(x)$ 和方差 $\sigma'(x)$ 。这用于近似真实但难以计算的后验分布 $P(z|x)$

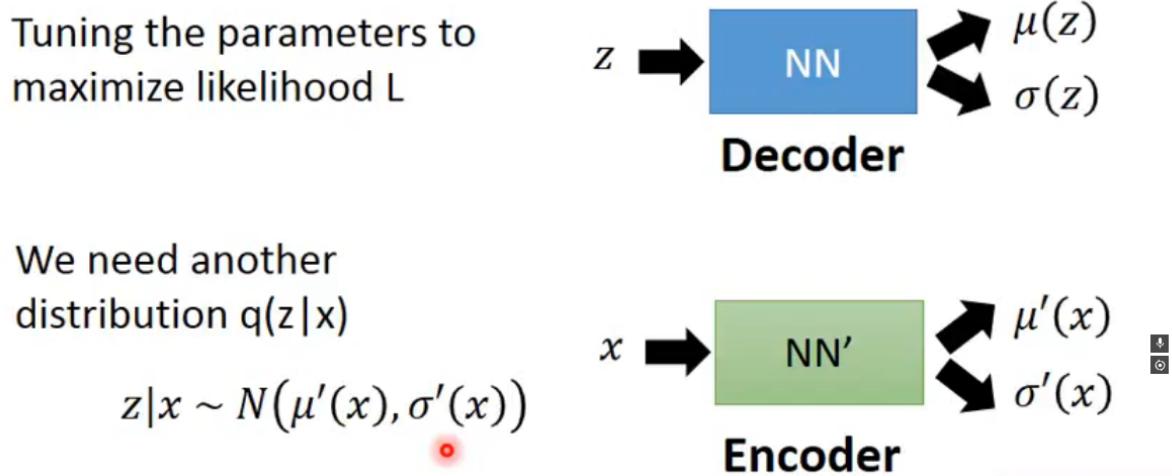


image-20250501000147212

数学推导

有疑问可以参考[1. 变分自编码器（Variational Autoencoder） — 张振虎的博客 张振虎 文档](#)

X为可观测变量（Observed variable），Z为不可观测变量（Unobserved variable）

X为图片样本，Z表示潜在空间中的数据

- 从X到Z相当于给定X求Z的条件概率 $P(Z|X)$
- 从Z到X相当于给定Z求X的条件概率 $P(X|Z)$

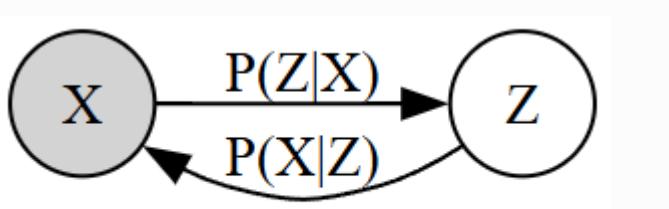


图 1.2 X 和 Z 的图表示

image-20250502115738249

完整的模型表示是二者的联合概率分布 $P(X, Z)$

我们用 \mathcal{D} 来表示观测样本的集合，如果X和Z都可以观测到，那么 $\mathcal{D} = \{(z^{(1)}, x^{(1)}), \dots, (z^{(N)}, x^{(N)})\}$ ，则目标损失函数为：

$$\mathcal{L}(\theta; \mathcal{D}) = \sum_{i=1}^N \ln p_\theta(z^{(i)}, x^{(i)}) \quad (1.1)$$

但是我们并没有Z的观测样本，此时观测样本集合为 $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ ，目标损失函数变成

$$\mathcal{L}(\theta; \mathcal{D}) = \sum_{i=1}^N \ln p_\theta(x^{(i)}) = \sum_{i=1}^N \ln \int_z p_\theta(x^{(i)}, z) dz \quad (1.2)$$

积分难以计算，无法直接求解

证据下界

补充 Jensen 不等式

对于凹函数 $f(x)$ ，有如下关系 $f(\mathbb{E}[X]) \geq \mathbb{E}[f(X)]$ 而对于凸函数，不等号方向相反

定义一个变量Z的概率密度函数 $q_\phi(z|x)$

推导出公式(1.2)的下界函数

$$\begin{aligned}
\mathcal{L}(\theta; x) &= \ln p_\theta(x) \\
&= \ln \int_z p_\theta(x, z) \\
&= \ln \int_z q_\phi(z|x) \frac{p_\theta(x, z)}{q_\phi(z|x)} \quad \text{同时乘除 } q_\phi(z|x), \text{ 等于没变化} \\
&= \ln \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(x, z)}{q_\phi(z|x)} \right] \\
&\geq \mathbb{E}_{q_\phi(z|x)} \ln \left[\frac{p_\theta(x, z)}{q_\phi(z|x)} \right] \quad \text{根据Jensen不等式} \\
&= \int_z q_\phi(z|x) \ln \left[\frac{p_\theta(x, z)}{q_\phi(z|x)} \right] \\
&= \left[\int_z q_\phi(z|x) \ln p_\theta(x, z) - \int_z q_\phi(z|x) \ln q_\phi(z|x) \right] \\
&\triangleq \mathcal{L}(q, \theta)
\end{aligned}$$

找到下界函数 (ELBO) $\mathcal{L}(q, \theta)$

$$\begin{aligned}
\mathcal{L}(q, \theta) &= \left[\int_z q_\phi(z|x) \ln p_\theta(x, z) - \int_z q_\phi(z|x) \ln q_\phi(z|x) \right] \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x, z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z|x)]
\end{aligned}$$

而 $p(x, z) = p(z)p(x|z) = p(x)p(z|x)$, 说明有两种分解方法变换下界函数ELBO

第一种形式

使用 z 的后验 $p(z|x)$ 进行分解

$$\begin{aligned}
\mathcal{L}(q, \theta) &= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x, z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z|x)] \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x) + \ln p_\theta(z|x)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z|x)] \\
&= \underbrace{\mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x)]}_{\text{与 } z \text{ 无关, 期望符号可以直接去掉}} + \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(z|x)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z|x)] \\
&= \underbrace{\ln p_\theta(x)}_{\text{观察数据对数似然/证据}} + \underbrace{\mathbb{E}_{z \sim q_\phi} [\ln p_\theta(z|x)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z|x)]}_{\text{KL散度}} \\
&= \underbrace{\ell(\theta; x)}_{\text{观察数据对数似然/证据}} - \underbrace{KL(q_\phi(z|x) || p_\theta(z|x))}_{\text{KL散度}}
\end{aligned}$$

整理后可得到

$$\underbrace{\ell(\theta; x)}_{\text{观察数据对数似然/证据}} = \underbrace{\mathcal{L}(q, \theta)}_{\text{下界函数ELBO}} + \underbrace{KL(q_\phi(z|x) || p_\theta(z|x))}_{\text{KL散度}}$$

第二种形式 (主要优化方向)

$$\begin{aligned}
\mathcal{L}(q, \theta) &= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x, z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z)] \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p(z) + \ln p_\theta(x|z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z)] \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p(z)] + \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x|z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z)] \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x|z)] + \underbrace{\mathbb{E}_{z \sim q_\phi} [\ln p(z)] - \mathbb{E}_{z \sim q_\phi} [\ln q_\phi(z)]}_{\text{KL散度}} \\
&= \mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x|z)] - \underbrace{KL(q_\phi(z|x) || p(z))}_{q_\phi(z) \text{ 和 先验 } p(z) \text{ 的 KL 散度}}
\end{aligned}$$

根据前文的结论，当 $q_\phi(z)$ 等于 z 的后验 $p_\theta(z|x)$ 时，下界函数 $\mathcal{L}(q, \theta)$ 和观测数据的对数似然函数 $\ell(\theta; x)$ 是相等的。因此，我们令 $q_\phi(z) = p_\theta(z|x)$ ，为了符号区分这里记作 $q_\phi(z) = q_\phi(z|x)$ ，可得

$$\begin{aligned}
\mathcal{L}(q, \theta) &= \underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} [\ln p_\theta(x|z)]}_{\text{重建项(reconstruction term)}} - \underbrace{KL(q_\phi(z|x) || p(z))}_{\text{先验匹配项(prior matching term)}} \\
&= \ell(\theta; x)
\end{aligned}$$

疑难杂点

分布	含义	是否学习	网络对应
$p(z)$	潜变量的先验分布	否，通常固定为 $N(0, I)$	无
$q_\phi(z x)$	近似后验分布	是，学习参数 ϕ	编码器
$p_\theta(x z)$	条件生成分布	是，学习参数 θ	解码器
$p_\theta(z x)$	真实后验分布	理论上是，但难以直接计算	无
$p_\theta(x)$	数据的边缘分布	理论上是，但难以直接优化	整个VAE

- $p(z)$ 是未看见任何数据之前对 z 的**假设分布**，通常是正态分布
- $p(z|x)$ 是看到数据之后 z 的分布，为**真实后验**
- 实际情况中真实后验**难以计算**（因为要对 z 积分，但是我们不知道 z 的特征），不采用第一种方式进行训练，而使用第二种我们对 z 假设的分布进行训练
- 我们希望编码器的输出可以接近我们假设的 z 的分布，可以使得 **ELBO** 最大，从而使对数似然最大

后验分布-编码器

后验分布 $q_\phi(z|x)$ 是一个**高斯分布**，但我们不知道其均值和方差。这里我们分别用 μ_z 和 Σ_z 表示后验分布的均值参数和方差参数，此时有 $q_\phi(z|x) = \mathcal{N}(\mu_z, \Sigma_z)$ ，由于模型假设 Z 的各维度之间是相互独立的，因此其协方差 Σ_z 是一个对角矩阵，非对角线位置元素值为 0，对角线元素是**未知参数**。

在VAE中， Z 是一个**随机变量**，不能从 X 直接映射到 Z ，但是输入变量 X 是通过影响 Z 的均值参数和方差参数间接影响到 Z ，就是说均值参数 μ_z 和方差参数 Σ_z 是和 x 相关的，即：

$$\begin{aligned}
\mu_z &= \mu_\phi(x) = \text{encoder}_\phi(x) \\
\Sigma_z &= \Sigma_\phi(x) = \text{encoder}_\phi(x)
\end{aligned}$$

KL散度-正则项

$$KL(q_\phi(z)||p(z)) = KL(\mathcal{N}(\mu_z, \Sigma_z)||\mathcal{N}(0, I)) \\ = \frac{1}{2}(\text{tr}(\Sigma_z) + \mu_z^T \mu_z - k - \log \det(\Sigma_z))$$

上述是模型训练过程目标函数的一项， k 是向量的维度，他的作用相当于一个正则项，使得后验 $q_\phi(z|x)$ 尽量接近 z 先验。

生成分布-解码器

$\underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} [\ln p_\theta(x|z)]}$ 对应着解码器的部分，从变量 Z 重建回 X 。 z 到 x 的映射是通过为 z 和 X 的均值参数 μ_x 建立重建项(reconstruction term)

映射函数实现的。这里并没有建立从 z 到 X 方差参数 Σ_x 之间的映射，这是因为模型为了简单，假设 X 的方差为常量，即单位方差 I

$$\mu_x = \mu_\theta(z) = \text{decoder}_\theta(z)$$

因为编码器的输出是 Z 的分布，所以没办法用解析计算的方式计算它的积分。这时可以借助**马尔科夫链蒙特卡洛法**，即**采样法近似实现**。其实就是从后验概率分布 $q_\phi(z|x)$ 随机采样很多个 z 值，代入进去算平均值。

$$\mathbb{E}_{z \sim q_\phi(z|x)} [\ln p_\theta(x|z)] \approx \frac{1}{L} \sum_{l=1}^L [\ln p_\theta(x|z^{(l)})]$$

采样次数 L 可以作为模型的超参数，可以人为指定，根据作者的经验 $L = 1$ 其实也可以。然而这有产生了新的问题，从编码器网络到解码器网络中间有个**随机采样**，即 z 是通过随机采样参数的，而随机采样过程是**不可导的**，这导致梯度不能从解码器传递到编码器。VAE 的作者，在这里采用**重参数化** (reparameterization trick) 的技巧来解决这个问题。

参数重整

重参数化的思想其实很简单，就是稍微调整了一下采样的方法

我们要从后验分布 $q_\phi(x|z)$ 中随机采样 z 的值，这个后验分布是一个高斯分布 $\mathcal{N}(\mu_z, \Sigma_z)$ ，直接从这个分布中采样会导致模型不可导，梯度无法传递。这里可以利用高斯分布的一个特点来改变采样过程，**任意均值和方差的高斯分布都可以从一个标准正态分布 $\mathcal{N}(0, I)$ 变换得到**，我们用符号 ϵ 表示一个多维标准正态分布，即 $\epsilon \sim \mathcal{N}(0, I)$ ，任意另一个高斯分布 $\mathcal{N}(\mu_z, \Sigma_z)$ 的值可以通过下式直接计算得到

$$z = \mu_z + \sqrt{\Sigma_z} \odot \epsilon \\ = \mu_\theta(x) + \sqrt{\Sigma_\phi(x)} \odot \epsilon, \epsilon \sim \mathcal{N}(0, I)$$

也就是说，可以先从标准正态分布 $\mathcal{N}(0, I)$ 随机采样一个值，然后通过上述公式计算得到 z 的值，其中 \odot 表示元素乘法。这相当于在 encoder 的输出 $\mu_\phi(x)$ 的基础上加上高斯噪声，再乘上 encoder 的另一个输出 $\sqrt{\Sigma_\phi(x)}$ ，随机采样的是高斯噪声 ϵ ，而它不影响模型的梯度传递

损失函数推导

回顾一下**多维正态分布的表达式**

$$p(x) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right\}$$

在前面我们假设 $p_\theta(x|z)$ 是一个单位方差的高斯分布，根据高斯分布的概率密度函数， $p_\theta(x|z)$ 的形式为

$$\begin{aligned} p(x) &= \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right\} \\ &\propto \exp\left\{-\frac{1}{2}(x - \mu_x)^T (x - \mu_x)\right\} \\ &= \exp\left\{-\frac{1}{2}(x - \mu_\theta(z))^T (x - \mu_\theta(z))\right\} \\ &= \exp\left\{-\frac{1}{2}(x - \text{decoder}(z))^T (x - \text{decoder}(z))\right\} \end{aligned}$$

最后下界函数 $ELBO$ 的形式为

$$\begin{aligned} \mathcal{L}(q, \theta) &= \underbrace{\mathbb{E}_{z \sim q_\phi} [\ln p_\theta(x|z)]}_{\text{对应解码过程}} - \underbrace{KL(q_\phi(z|x) || p(z))}_{\text{对应编码过程}} \\ &= \frac{1}{L} \sum_{l=1}^L [\ln p_\theta(x|z^{(l)})] - KL(\mathcal{N}(\mu_z, \Sigma_z) || \mathcal{N}(0, I)) \\ &\propto \frac{1}{L} \sum_{l=1}^L \left[-\frac{1}{2}(x - \mu_x)^T (x - \mu_x) \right] - \left[\frac{1}{2}(\text{tr}(\Sigma_z) + \mu_z^T \mu_z - k - \log \det(\Sigma_z)) \right] \\ &\propto -\frac{1}{L} \sum_{l=1}^L \left[(x - \mu_x)^T (x - \mu_x) \right] - \left[(\text{tr}(\Sigma_z) + \mu_z^T \mu_z - k - \log \det(\Sigma_z)) \right] \end{aligned}$$

其中：

$$\begin{aligned} \mu_x &= \mu_\theta(z^{(l)}) = \text{decoder}(z^{(l)}) \\ z^{(l)} &= \mu_z + \sqrt{\Sigma_z} \odot \epsilon, \epsilon \sim \mathcal{N}(0, I) \\ \mu_z &= \mu_\phi(x) = \text{encoder}(x)_{\mu_z} \\ \Sigma_z &= \Sigma_\phi(x) = \text{encoder}(x)_{\Sigma_z} \end{aligned}$$

由于我们是通过极大化 $\mathcal{L}(q, \theta)$ 进行参数求解，其中有一些参数项可以去掉，最后可以等价于同时极小化下面两项

$$\frac{1}{L} \sum_{l=1}^L \underbrace{[(x - \mu_x)^T (x - \mu_x)]}_{\text{均方误差}}$$

和

$$(\text{tr}(\Sigma_z) + \mu_z^T \mu_z - k - \log \det(\Sigma_z))$$

可以见得，第一项是均方误差，第二项是正则项。

个人理解

- 重建项是为了能够通过 z 还原出 x
- 先验匹配项是希望编码器生成的 z 的分布可以接近我们给定的分布
 - 使用的时候在给定的分布里抽样即可，这样解码器就能正确还原
- 编码器接收 x ，将 x 映射到不同均值，不同方差的正态分布，解码器根据不同均值和不同方差采样出来的点去还原图像
 - 由于每张图像具有不同的特征，所以编码器要为每张图像输出属于他自己的高斯分布

$$\mathcal{L}(q, \theta) = \underbrace{\mathbb{E}_{z \sim q_\phi(z|x)} [\ln p_\theta(x|z)]}_{\text{重建项(reconstruction term)}} - \underbrace{KL(q_\phi(z|x) || p(z))}_{\text{先验匹配项(prior matching term)}}$$

训练流程

1. 输入图像 x
2. 编码器输出该图像对应的分布 μ, σ
3. 从分布中采样出 z
4. 解码器用 z 生成图像 \hat{x}
5. 对比 x 和 \hat{x} , 计算重建误差
6. 同时计算这个分布与 $\mathcal{N}(0, 1)$ 的 KL 距离
7. 两部分误差加在一起做反向传播更新参数

AE (Auto-encoder 自编码器)

基本概念：

自编码器是一种特殊的神经网络，主要目的是**学习如何压缩和重建数据**。想象一下，你要把一张照片通过一个狭窄的管道传输，然后在另一端重新组装成原来的样子。自编码器就是学习如何**进行这种压缩和重建的过程**

核心特点：

试图将输入数据编码成一个低维表示（称为潜在空间或编码），然后再从这个低维表示重建原始输入数据

主要组成部分

自编码器主要由三个部分组成：

1. 编码器 (**Encoder**)：
 - 将输入数据转换为较低维度的表示（称为“潜在表示”或“编码”）
 - 相当于压缩过程
 - 通常由几层神经网络组成，逐渐减少神经元数量
2. 潜在空间 (**Latent Space**)：
 - 编码后的数据存在的压缩表示空间
 - 通常维度比原始数据小得多
 - 包含了数据的主要特征信息
3. 解码器 (**Decoder**)：
 - 将潜在表示转换回原始数据维度
 - 相当于解压缩过程
 - 结构通常是编码器的镜像，神经元数量逐渐增加

工作流程

1. 输入数据（如图像）进入编码器
2. 编码器将数据压缩到潜在空间（例如，从784维压缩到32维）
3. 解码器尝试从潜在空间重建原始输入
4. 比较重建结果与原始输入的差异（称为“重建误差”）
5. 通过反向传播调整编码器和解码器的参数，使重建误差最小化

自编码器的用途包括：

- 数据降维
- 特征学习

- 异常检测
- 图像去噪
- 生成模型的基础