生成式 AI 学习笔记

Author: 张哲源 **Email:** **Date:** 2026-02-09

1 变分自编码器 (Variational Autoencoder, VAE)

符号说明 (Notation)

符号	含义
$\mathcal{D}$	数据集，$\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$
$\mathbf{x}^{(i)}$	第 i 个数据样本，维度为 d
$\mathbf{z}$	瓶颈层学习到的压缩隐变量 (Latent Code)
$g_\phi(\cdot)$	编码器函数，参数为 ϕ
$f_\theta(\cdot)$	解码器函数，参数为 θ
$q_\phi(\mathbf{z}\|\mathbf{x})$	推断网络（近似后验），概率编码器
$p_\theta(\mathbf{x}\|\mathbf{z})$	生成网络（似然），概率解码器

1.1 前置知识：自编码器及其变体 (Autoencoders)

Definition 1.1 (自动编码器)**.** 自动编码器（Autoencoder, AE）是一种无监督学习模型，其核心目标是学习一个恒等函数（Identity Function），即输出 $\mathbf{x}' \approx \mathbf{x}$。它通过一个具有窄瓶颈层（Bottleneck）的神经网络将高维输入压缩为低维编码（降维），再从该编码重构出原始输入。

1.1.1 基本自动编码器 (Autoencoder)

自动编码器通常包含两个部分：

- **编码器 (Encoder)** $g_\phi(\cdot)$：将高维输入 $\mathbf{x}$ 映射为低维隐变量 $\mathbf{z} = g_\phi(\mathbf{x})$。
- **解码器 (Decoder)** $f_\theta(\cdot)$：将隐变量 $\mathbf{z}$ 映射回重构数据 $\mathbf{x}' = f_\theta(\mathbf{z})$。

训练目标是最小化重构误差，例如均方误差 (MSE)：

$$L_{\text{AE}}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}^{(i)} - f_\theta(g_\phi(\mathbf{x}^{(i)})))^2 \tag{1}$$

1.1.2 去噪自动编码器 (Denoising Autoencoder, DAE)

为了避免过拟合（即单纯地记忆输入数据），去噪自动编码器引入了“破坏”机制。

Note 1.1 (核心思想). DAE 在输入 $\mathbf{x}$ 中主动加入噪声得到 $\tilde{\mathbf{x}}$（例如将部分像素置零，类似 Dropout），然后训练模型从损坏的 $\tilde{\mathbf{x}}$ 中恢复出**原始的**无噪数据 $\mathbf{x}$。这迫使模型学习数据维度间的依赖关系，从而提取更鲁棒的特征。

损失函数调整为：

$$L_{\text{DAE}}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}^{(i)} - f_\theta(g_\phi(\tilde{\mathbf{x}}^{(i)})))^2 \tag{2}$$

1.1.3 稀疏自动编码器 (Sparse Autoencoder, SAE)

稀疏自动编码器通过在损失函数中增加稀疏约束，强制隐层神经元在大部分时间处于“未激活”状态。假设隐藏层神经元 j 的平均激活度为 $\hat{\rho}_j$，我们希望它接近一个很小的稀疏参数 ρ（如 0.05）。通常使用 KL 散度作为惩罚项：

$$L_{\text{SAE}}(\theta) = L(\theta) + \beta \sum_{j=1}^{s_l} D_{\text{KL}}(\rho \| \hat{\rho}_j) \tag{3}$$

此外，还有 **k-稀疏自动编码器**（k-Sparse Autoencoder）。其前向传播步骤如下：

1. 运行编码器得到潜在代码 $\mathbf{z} = g(\mathbf{x})$。
2. 对 $\mathbf{z}$ 进行排序，仅保留前 k 个最大的激活值，将其余值置零，得到 $\mathbf{z}' = \text{Sparsify}(\mathbf{z})$。
3. 计算重构损失 $L = \|\mathbf{x} - f(\mathbf{z}')\|_2^2$。

注意：反向传播时，梯度仅通过这前 k 个激活的神经元传递。

1.1.4 收缩自动编码器 (Contracting Autoencoder, CAE)

收缩自动编码器旨在使学习到的表示对输入的微小扰动具有鲁棒性。它在损失函数中加入雅可比矩阵（Jacobian Matrix）的 Frobenius 范数作为惩罚项，抑制编码器激活值对输入的敏感度：

$$\|J_f(\mathbf{x})\|_F^2 = \sum_{ij} \left(\frac{\partial h_j(\mathbf{x})}{\partial x_i} \right)^2 \tag{4}$$

这促使模型学习到的特征分布在低维流形上，而对流形正交方向的变化保持不变。

1.2 VAE 原理详解 (The Principle of VAE)

> **Definition 1.2.** 变分自编码器（Variational Autoencoder, VAE）是一种强大的深度生成模型，它巧妙地将深度学习的表示能力与贝叶斯推断的概率框架相结合。与传统的自编码器（Autoencoder）不同，VAE 并非学习一个从输入到编码的确定性映射，而是学习一个从输入到隐变量后验概率分布的映射。通过从这个学习到的分布中采样，VAE 能够生成与训练数据相似但全新的数据样本。

关键词：生成模型，变分自编码器，贝叶斯推断，证据下界，重参数化技巧

1.2.1 引言 (Introduction)

生成模型的根本目标是学习观测数据 X 的真实概率分布 $P_{data}(X)$。一旦这个分布被成功建模，我们便可以从中采样，生成新的数据。早期的生成模型如受限玻尔兹曼机（RBM）面临训练困难的问题，而传统的自编码器（AE）虽能学习数据的有效压缩表示，但其隐空间（latent space）缺乏良好的结构性，无法直接用于生成新样本。 挑战

Note 1.2 (VAE 的核心思想). 为了解决这一问题，Kingma 和 Welling 在 2013 年提出了变分自编码器。VAE 的核心思想是，不再将输入数据 X 编码为一个确定的隐向量 z，而是将其编码为一个概率分布（通常是高斯分布）。具体来说，模型学习这个分布的参数（例如均值 μ 和方差 σ^2）。生成过程则变为：

1. 首先从这个分布中采样一个隐向量 z。
2. 然后解码器（Decoder）基于这个 z 来重构出数据 $\hat{X}$。

通过这种方式，VAE 不仅学习了如何重构数据，更重要的是，它学习到了一个结构化、连续的隐空间，使得我们可以在这个空间中进行采样和插值，从而生成多样化的新数据。

1.2.2 模型架构与概率框架 (Model Architecture & Probabilistic Framework)

从概率的视角来看，VAE 假设所有的数据样本 X 是由一个我们无法直接观测到的隐变量 z 生成的。这个生成过程包含以下步骤：

1. 从一个简单的先验分布 $p(z)$ 中采样一个隐变量 z。通常，我们选择标准正态分布，即 $z \sim \mathcal{N}(0, I)$。
2. 根据隐变量 z，通过一个以神经网络（解码器）参数化的条件概率分布 $p_\theta(X|z)$ 来生成数据 X。

我们的最终目标是最大化观测数据的边际似然 $p(X)$，即所有可能隐变量 z 生成 X 的概率的积分：

$$p(X) = \int p_\theta(X|z)p(z)dz \tag{5}$$

然而，这个积分通常是难以计算的（intractable），因为它需要在整个高维的隐空间上进行。这就引出了变分推断（Variational Inference）的必要性。

VAE 模型架构 VAE 由两个核心部分组成，通常由深度神经网络实现：

- **编码器 (Encoder)，或称推断网络 (Inference Network):**
  - **输入：** 数据样本 X。
  - **输出：** 隐变量 z 的后验分布 $p(z|X)$ 的近似分布 $q_\phi(z|X)$ 的参数。这里，ϕ 是编码器网络的参数。具体地，编码器输出近似后验分布的参数，例如高斯分布的均值 μ_X 和对数方差 $\log(\sigma_X^2)$。即 $q_\phi(z|X) = \mathcal{N}(z; \mu_X, \sigma_X^2 I)$。

- **解码器 (Decoder)，或称生成网络 (Generative Network):**
  - **输入：**从分布 $q_\phi(z|X)$ 中采样的隐向量 z。
  - **输出：**重构的数据 $\hat{X}$，或者更准确地说，是条件概率分布 $p_\theta(X|z)$ 的参数。

1.2.3 核心推导：证据下界 (Derivation of the Evidence Lower Bound, ELBO)

由于直接最大化 $p(X)$ 不可行，我们转而最大化它的一个下界，即**证据下界（ELBO）**。我们从对数边际似然 $\log p(X)$ 开始，引入近似后验 $q_\phi(z|X)$，并根据詹森不等式（Jensen's Inequality），得到：

$$\begin{aligned}\log p(X) &\geq \int q_\phi(z|X) \log \frac{p_\theta(X|z)p(z)}{q_\phi(z|X)} dz \\ &= \mathbb{E}_{z\sim q_\phi(z|X)}\left[\log \frac{p_\theta(X|z)p(z)}{q_\phi(z|X)}\right]\end{aligned} \tag{6}$$

不等式的右侧就是证据下界 (ELBO)，记为 $\mathcal{L}(\phi, \theta; X)$。

Note 1.3 (ELBO 的分解).

$$\begin{aligned}\mathcal{L}(\phi,\theta;X) &= \mathbb{E}_{z\sim q_\phi(z|X)}\left[\log p_\theta(X|z) + \log p(z) - \log q_\phi(z|X)\right] \\ &= \underbrace{\mathbb{E}_{z\sim q_\phi(z|X)}[\log p_\theta(X|z)]}_{\text{重构项}} - \underbrace{D_{KL}(q_\phi(z|X)||p(z))}_{\text{KL 散度正则化项}}\end{aligned} \tag{7}$$

Note 1.4 (为什么使用反向 KL 散度？). 我们通过最小化 $D_{KL}(q_\phi(z|X)||p_\theta(z|X))$ 来逼近真实后验，这被称为**反向 KL 散度**（Reverse KL）。

- **前向 KL ($D_{KL}(P||Q)$):** 要求近似分布 Q 覆盖真实分布 P 的所有非零区域。
- **反向 KL ($D_{KL}(Q||P)$):** 倾向于将近似分布 Q “挤压”在真实分布 P 的峰值下方。这使得 q_ϕ 能够捕捉到 p_θ 的主要模式（Mode-seeking behavior），对于生成任务更为稳健。

Note 1.5 (目标函数解读). VAE 的训练目标是最大化 ELBO，它由两部分构成：

- **重构项 (Reconstruction Term):** 该项最大化在给定隐变量 z 的情况下，重构出原始数据 X 的对数似然。这驱使解码器学习如何精确地从 z 恢复 X。在实践中，常使用均方误差（MSE）或二元交叉熵（BCE）损失的负值。
- **KL 散度正则化项 (KL Divergence Regularization Term):** 该项衡量近似后验分布 $q_\phi(z|X)$ 与先验分布 $p(z)$（通常是 $\mathcal{N}(0, I)$）之间的差异。最小化 KL 散度，相当于对编码器施加约束，使其产生的隐变量分布接近标准正态分布，从而使隐空间变得平滑和连续。

因此，训练目标可以概括为：**最大化重构质量**的同时，**保持隐空间分布的规整性**。

1.2.4 关键技术：重参数化技巧 (The Reparameterization Trick)

Definition 1.3 (重参数化技巧)**.** 在目标函数中，从 $q_\phi(z|X)$ 中采样 z 的过程是随机的，不可微分，这会阻碍梯度的反向传播。**重参数化技巧**通过将随机性与模型参数分离来解决此问题。对于高斯分布 $z \sim \mathcal{N}(z; \mu_X, \sigma_X^2 I)$，其采样过程可重写为：

1. 从一个固定的标准正态分布中采样噪声 $\epsilon \sim \mathcal{N}(0, I)$。
2. 通过确定性变换生成 z：$z = \mu_X + \sigma_X \odot \epsilon$。

这样，随机采样被移出计算图，梯度可以顺利地从损失函数回传到编码器的参数 ϕ（即 μ_X 和 σ_X）。

1.2.5 结论与讨论 (Conclusion and Discussion)

变分自编码器（VAE）通过最大化证据下界（ELBO）成功地将深度神经网络与概率图模型结合起来。其损失函数由重构损失和 KL 散度正则化项构成，分别保证了模型的重构能力和隐空间的良好结构。关键的重参数化技巧使得模型能够通过标准的梯度下降算法进行端到端的训练。

Note 1.6 (VAE vs. GAN)*.* 与 GAN 相比，VAE 的训练过程更稳定，能够显式地学习数据的概率密度。然而，VAE 生成的样本通常比 GAN 生成的样本**模糊**，这是因为其损失函数倾向于覆盖数据的所有模式，导致在细节上有所妥协。

尽管如此，VAE 作为一种 foundational 的深度生成模型，其思想，特别是对隐空间的概率建模，对后续的生成模型（包括扩散模型）产生了深远的影响。理解 VAE 是掌握现代生成式 AI 版图不可或缺的一步。

1.3 进阶变分模型 (Advanced VAE Models)

1.3.1 Beta-VAE (β-VAE)

Definition 1.4 (解耦表示 (Disentangled Representation))**.** 如果隐变量 $\mathbf{z}$ 中的每个维度仅对数据的一个生成因子敏感（例如一个维度控制“发色”，另一个控制“肤色”），且对其他因子保持不变，则称这种表示为解耦的。解耦表示具有极佳的可解释性。

β-VAE 旨在通过调整 ELBO 中的 KL 散度项权重来发现解耦的潜在因子。无论是从直觉上增加 KL 惩罚，还是从约束优化的角度来看，都有坚实的理论基础。我们希望最大化生成概率，同时约束后验分布与先验分布的距离小于 δ：

$$\max_{\phi,\theta} \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})] \quad \text{subject to } D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) < \delta \tag{8}$$

使用拉格朗日乘数法，这等价于最大化：

$$\mathcal{F}(\theta, \phi, \beta) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - \beta(D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})) - \delta) \tag{9}$$

去掉常数项，得到 β-VAE 的损失函数：

$$L_{\beta\text{-VAE}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] + \beta D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})) \tag{10}$$

Note 1.7 (β 的作用).

- 当 $\beta = 1$ 时，模型退化为标准的 VAE。
- 当 $\beta > 1$ 时，模型对隐变量施加了更强的独立性约束，鼓励学习更有效的、解耦的潜在编码。但这通常会带来重构质量与解耦程度之间的权衡（Trade-off）。

1.3.2 VQ-VAE (Vector Quantized VAE)

传统的 VAE 使用连续的隐变量分布（如高斯分布），这可能不适合某些模态（如语言、推理）。VQ-VAE 引入了**离散潜在变量**。

Note 1.8 (向量量化 (Vector Quantization)). VQ-VAE 维护一个“代码簿”（Codebook），包含 K 个嵌入向量 $\mathbf{e}_i$。编码器的输出 $\mathbf{z}_e(\mathbf{x})$ 不再直接传给解码器，而是通过最近邻搜索映射到代码簿中最接近的嵌入向量 $\mathbf{e}_k$：

$$\mathbf{z}_q(\mathbf{x}) = \mathbf{e}_k, \quad \text{where } k = \arg\min_i \|\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_i\|_2 \tag{11}$$

由于 arg min 操作不可导，VQ-VAE 使用直通估计器（Straight-Through Estimator）将解码器的梯度直接复制回编码器（即 $\nabla_z L \approx \nabla_{z_q} L$）。其总损失函数如下（其中 sg[·] 表示这个梯度的截断算子 stop gradient）：

$$L = \underbrace{\|\mathbf{x} - D(\mathbf{e}_k)\|_2^2}_{\text{Reconstruction Loss}} + \underbrace{\|\mathrm{sg}[E(\mathbf{x})] - \mathbf{e}_k\|_2^2}_{\text{VQ Loss}} + \underbrace{\beta\|E(\mathbf{x}) - \mathrm{sg}[\mathbf{e}_k]\|_2^2}_{\text{Commitment Loss}} \tag{12}$$

代码簿更新 (EMA): 为了更稳定地更新嵌入向量，可以使用指数移动平均 (Exponential Moving Average)：

$$N_i^{(t)} = \gamma N_i^{(t-1)} + (1-\gamma) n_i^{(t)}, \quad \mathbf{m}_i^{(t)} = \gamma \mathbf{m}_i^{(t-1)} + (1-\gamma)\sum_j \mathbf{z}_{i,j}^{(t)}, \quad \mathbf{e}_i^{(t)} = \frac{\mathbf{m}_i^{(t)}}{N_i^{(t)}} \tag{13}$$

VQ-VAE-2 进一步引入了层级结构（Hierarchical VQ-VAE），分离局部纹理和全局形状信息，并结合自回归模型（如 PixelCNN）来学习离散编码的先验分布，生成了极高质量的图像。

1.3.3 TD-VAE (Temporal Difference VAE)

TD-VAE 专为序列数据设计，旨在解决传统序列模型推断效率低的问题。它基于三个核心思想：

1. **状态空间模型 (State Space Model):** 假设观测序列由一系列不可见的隐状态控制。
2. **信念状态 (Belief State):** 智能体应学会将过去的所有信息编码为一个信念状态 b_t，用于推理未来。
3. **跳跃预测 (Jumpy Prediction):** 模型应具备根据当前信息直接预测遥远未来的能力，而无需一步步模拟中间过程。

TD-VAE 需要学习四种分布：

- **解码器分布** $p_D(x_t|z_t)$: 从隐状态生成观测。
- **转移分布** $p_T(z_t|z_{t-1})$: 捕捉隐变量的序列依赖。
- **信念分布** $p_B(z_t|b_t)$: 从信念状态推断隐变量。
- **平滑分布** $p_S(z_{t-1}|z_t, b_{t-1}, b_t)$: 回溯推断。

TD-VAE 的最终目标函数是在两个时间戳 $t_1 < t_2$ 之间最大化以下期望（Jump Prediction）：

$$J_{t_1,t_2} = \mathbb{E}[\log p_D(x_{t_2}|z_{t_2}) + \log p_B(z_{t_1}|b_{t_1}) + \log p_T(z_{t_2}|z_{t_1}) - \log p_B(z_{t_2}|b_{t_2}) - \log p_S(z_{t_1}|z_{t_2}, b_{t_1}, b_{t_2})] \tag{14}$$

参考文献

[1] WENG L. From autoencoder to beta-vae[J/OL]. lilianweng.github.io, 2018. https://lilianweng.github.io/posts/2018-08-12-vae/.