9 算例 E:《拉长推断路径——去噪扩散概率模型 (DDPM)》

在前面的章节中，我们看到了 VAE 通过一个单步的推断网络 ϕ 将观测 X 压缩到隐空间 Z。然而，单步的压缩与重构往往导致信息丢失，生成的图像偏向模糊。如果我们把 VAE 的这一步推断“拉长”，拆分成成百上千个极其微小的步骤，并且放弃让神经网络去学习这个推断过程，而是用确定的物理热力学定律（扩散）来代替它呢？

本章将探讨生成模型的当前王者——去噪扩散概率模型（DDPM）。我们将证明，扩散模型在数学本质上就是一个极深层级的、推断路径被物理学锁死的特殊 VAE。

9.1 扩散模型的算例定义

为了透彻理解扩散模型，我们将其核心组件与标准 VAE 进行完全对齐。在扩散模型中，隐变量不再是一个孤立的 Z，而是一条马尔可夫链 $X_1, X_2, \ldots, X_T$。

9.1.1 变量定义与逻辑对齐 (对比 VAE)

符号	描述	在 VAE 中的逻辑对应
X_0	真实观测变量（干净数据）	对应 VAE 的输入 X
$X_{1:T}$	逐渐加噪的隐变量序列	对应 VAE 的隐变量 Z
β_t	前向加噪的方差表（超参数，固定值）	**取代了** VAE 的 Encoder 变分参数 ϕ
θ	去噪神经网络参数 (U-Net)	对应 VAE 的 Decoder 参数 θ

9.1.2 概率模型设定：固定的推断与参数化的生成

在扩散模型的马尔可夫框架下，概率结构定义如下：

1. **前向推断过程 (Forward Inference Process, q)**：这对应于 VAE 的 $q_\phi(Z|X)$。但极其关键的是，这里没有可学习的参数 ϕ。推断过程被硬编码为一个逐步添加高斯噪声的物理过程：

$$q(X_t|X_{t-1}) = \mathcal{N}(X_t; \sqrt{1-\beta_t}X_{t-1}, \beta_t\mathbf{I}) \tag{9-1}$$

   整个前向链的联合分布为 $q(X_{1:T}|X_0) = \prod_{t=1}^{T} q(X_t|X_{t-1})$。当 $T \to \infty$ 时，X_T 趋近于标准正态分布。

2. **隐变量先验 (Prior)**：与 VAE 一致，终点隐空间的先验是标准正态分布：

$$P(X_T) = \mathcal{N}(X_T|0, \mathbf{I})$$

3. **反向生成过程 (Reverse Generative Process, p_θ)**：这对应于 VAE 的解码器 $P_\theta(X|Z)$。由于逆转一个微小的高斯加噪步骤，其真实后验在数学上依然是高斯分布，因此我们用神经网络 θ 来拟合这个逐步去噪的过程：

$$p_\theta(X_{t-1}|X_t) = \mathcal{N}(X_{t-1}; \mu_\theta(X_t, t), \Sigma_\theta(X_t, t))$$

   整个反向链的联合分布为 $p_\theta(X_{0:T}) = P(X_T)\prod_{t=1}^{T} p_\theta(X_{t-1}|X_t)$。

9.1.3 概率图模型 (PGM)

扩散模型的概率图是一条展开的马尔可夫链。它清晰地展示了“确定性推断”与“参数化生成”的对立统一。

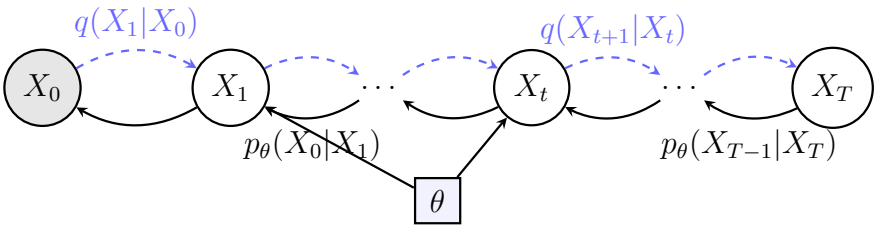

图 9: 扩散模型的 PGM 表示。蓝色虚线表示无需学习、硬编码的前向扩散推断链 q；黑色实线表示由参数 θ 控制的马尔可夫反向生成链 p_θ。神经网络 θ 共享于所有时间步 t。

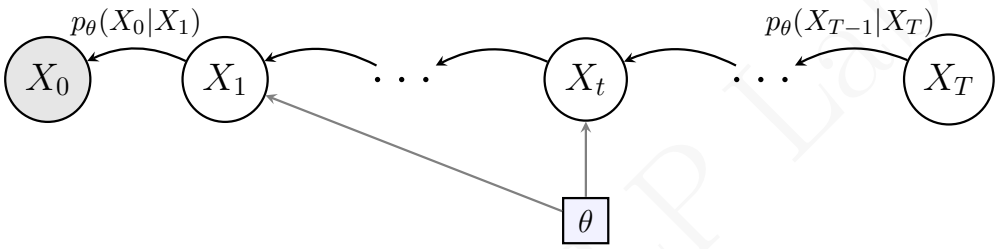

图 10: 反向生成过程 $p_\theta(X_{0:T})$: 采样并去噪 (Noise → Data)

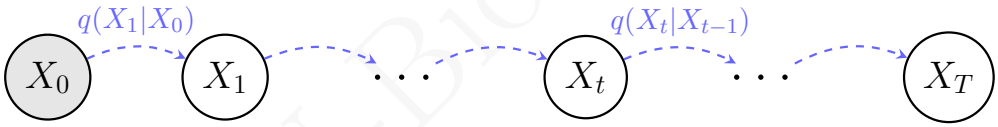

图 11: 前向扩散过程 $q(X_{1:T}|X_0)$: 注入高斯噪声 (Data → Noise)

9.2 扩散模型的目标函数：分层 ELBO

既然扩散模型也是隐变量模型，我们同样要最大化数据的对数边缘似然 $\log p(X_0)$。我们将 VAE 的 ELBO 公式直接平移过来，并将其在时间步上展开。

9.2.1 ELBO 的基本展开

代入扩散模型的所有序列变量 $X_{1:T}$，根据 Jensen 不等式，变分下界定义为：

$$\log p(X_0) \geq \mathbb{E}_{q(X_{1:T}|X_0)}\left[\log \frac{p_\theta(X_{0:T})}{q(X_{1:T}|X_0)}\right] \triangleq \mathcal{L}_{ELBO}(\theta)$$

利用马尔可夫链的性质以及贝叶斯定理，可以将这极其复杂的一长串分式，完美地分解解耦为三部分可计算的项（过程类似于算例 B 中的分解）：

$$\mathcal{L}_{ELBO}(\theta) = \mathbb{E}_q \left[\underbrace{\log p_\theta(X_0|X_1)}_{\mathcal{L}_0} - \underbrace{\mathcal{D}_{KL}(q(X_T|X_0) \parallel p(X_T))}_{\mathcal{L}_T} - \sum_{t=2}^{T} \underbrace{\mathcal{D}_{KL}(q(X_{t-1}|X_t, X_0) \parallel p_\theta(X_{t-1}|X_t))}_{\mathcal{L}_{t-1}} \right] \tag{9-2}$$

证明：

对于 $\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q(X_{1:T}|X_0)} \left[\log \frac{p_\theta(X_{0:T})}{q(X_{1:T}|X_0)} \right]$，利用马尔可夫链的性质，分别展开分母（前向加噪）和分子（反向生成），有

$$\begin{cases} q(X_{1:T}|X_0) = \prod_{t=1}^{T} q(X_t|X_{t-1}) \\ p_\theta(X_{0:T}) = p(X_T) \prod_{t=1}^{T} p_\theta(X_{t-1}|X_t) \end{cases}$$

代入原式得：

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_q \left[\log \frac{p(X_T) \prod_{t=1}^{T} p_\theta(X_{t-1}|X_t)}{\prod_{t=1}^{T} q(X_t|X_{t-1})} \right]$$

利用恒等式 $q(X_t|X_{t-1}) = \frac{q(X_{t-1}|X_t, X_0) q(X_t|X_0)}{q(X_{t-1}|X_0)}$ 替换分母项。当连乘时，由于 $q(X_t|X_0)$ 项产生对数抵消：

$$\prod_{t=1}^{T} q(X_t|X_{t-1}) = q(X_1|X_0) \cdot \frac{q(X_1|X_2, X_0) q(X_2|X_0)}{q(X_1|X_0)} \cdots \frac{q(X_{T-1}|X_T, X_0) q(X_T|X_0)}{q(X_{T-1}|X_0)}$$

化简后分母仅剩：

$$\prod_{t=1}^{T} q(X_t|X_{t-1}) = q(X_T|X_0) \prod_{t=2}^{T} q(X_{t-1}|X_t, X_0)$$

将化简后的分母代回对数式，并利用对数性质拆分为三项：

$$\mathcal{L}_{\text{ELBO}} = \underbrace{\mathbb{E}_q[\log p_\theta(X_0|X_1)]}_{\text{重建项}\mathcal{L}_0} - \underbrace{\mathcal{D}_{\text{KL}}(q(X_T|X_0) \parallel p(X_T))}_{\text{先验匹配}\mathcal{L}_T} - \sum_{t=2}^{T} \underbrace{\mathbb{E}_q \left[\mathcal{D}_{\text{KL}}(q(X_{t-1}|X_t, X_0) \parallel p_\theta(X_{t-1}|X_t)) \right]}_{\text{去噪匹配}\mathcal{L}_{t-1}}$$

9.2.2 各项的物理意义对齐

这三项在 VAE 体系中都能找到对应：

1. $\mathcal{L}_T$ (终点先验匹配) 要求前向加噪的终点 $q(X_T|X_0)$ 匹配标准正态先验 $p(X_T)$。这完全对应 VAE 的隐空间约束项。由于在扩散模型中，β_t 是人为设计好的，当 T 足够大时，X_T 必然坍缩为纯噪声。因此这一项是个常数（常为 0），在训练中可以直接丢弃，没有参数需要优化。

2. $\mathcal{L}_0$ (底层观测重构) 给定仅剩一丝噪声的 X_1，生成干净数据 X_0 的对数似然。这完全对应 VAE 的重建损失 (MSE/BCE)。

3. $\mathcal{L}_{t-1}$ (反向去噪匹配——核心优化项) 这是扩散模型独有的分层结构。在已知终点 X_t 和原始起点 X_0 的上帝视角下，真实的后验“撤销一步噪声”的分布 $q(X_{t-1}|X_t, X_0)$ 是可以通过贝叶斯公式推导出解析解的（且它是一个高斯分布）。这一项要求我们的神经网络 $p_\theta(X_{t-1}|X_t)$ 去尽力模仿这个真实的高斯后验分布。

9.3 扩散模型实现 ELBO 的两个终极 Trick

VAE 用重参数化解决了推断的求导问题。扩散模型用两个大 Trick 将复杂的 ELBO 简化成了一行代码。

9.3.1 第一个 Trick：任意时间步的直接采样 (Reparameterization 再升级)

如果每次训练都要跑完 $T = 1000$ 步的马尔可夫链，计算代价不可接受。利用高斯分布独立可加的性质（重参数化的连续应用），我们可以直接从 X_0 跳跃到任意时间步 X_t。定义 $\alpha_t = 1 - \beta_t$，以及 $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$，我们有：

$$q(X_t|X_0) = \mathcal{N}(X_t; \sqrt{\bar{\alpha}_t}X_0, (1 - \bar{\alpha}_t)\mathbf{I}) \tag{9-3}$$

$$X_t = \sqrt{\bar{\alpha}_t}X_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{9-4}$$

这意味着在训练时，我们可以随机抽取一个时间步 t，直接一步生成加噪数据 X_t，而不需要一步步递推。这使得训练可以完全并行化。

9.3.2 第二个 Trick：从匹配“均值”到预测“噪声”

在计算核心项 $\mathcal{L}_{t-1}$ 时，我们需要计算两个高斯分布之间的 KL 散度：真实的后验 q 与网络预测的 p_θ。正如我们在算例 C 中证明的，两个等方差高斯分布的 KL 散度等价于它们均值之间的均方误差 (MSE)。经过代数展开发现，真实的后验均值 $\tilde{\mu}_t$ 是由 X_t 和注入的噪声 ϵ 决定的。如果我们让神经网络 θ 去预测这个 ϵ，整个复杂的 KL 散度公式奇迹般地化简为了一个极其朴素的 L2 损失：

与其让网络预测高斯均值：
$$\mathcal{L}_{t-1} \propto \mathbb{E}\left[\|\tilde{\mu}_t(X_t, X_0) - \mu_\theta(X_t, t)\|^2\right] \tag{9-5}$$

不如让网络直接预测注入的噪声：
$$\mathcal{L}_{simple}(\theta) = \mathbb{E}_{t,X_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(X_t, t)\|^2\right] \tag{9-6}$$

在这里，网络 ϵ_θ 的输入是加噪后的图像 X_t 和时间步 t，它的任务是输出“这一步到底加了什么噪声 ϵ”。

9.3.3 DDPM 核心 Trick 的 PyTorch 实现

DDPM 核心训练逻辑的 PyTorch 实现

```
import torch
import torch.nn.functional as F

def train_diffusion_step(x0, model, alphas_cumprod):
    """
    x0: 真实的干净数据 Batch
    model: 预测噪声的 U-Net 模型 epsilon_theta
    alphas_cumprod: 预计算好的 \bar{\alpha}_t 列表
    """
    batch_size = x0.size(0)

    # --- Trick 1: 随机均匀采样时间步 t ---
    t = torch.randint(0, len(alphas_cumprod), (batch_size,), device=x0.device)

    # 提取对应的 \bar{\alpha}_t
    a_bar_t = alphas_cumprod[t].view(-1, 1, 1, 1)

    # --- Trick 1.5: 任意时间步的一步加噪采样 ---
    # 采样真实的纯高斯噪声 epsilon
    noise = torch.randn_like(x0)
    # X_t = sqrt(\bar{\alpha}_t) * X_0 + sqrt(1 - \bar{\alpha}_t) * epsilon
    xt = torch.sqrt(a_bar_t) * x0 + torch.sqrt(1 - a_bar_t) * noise

    # --- Trick 2: 噪声预测与极其简单的 L2 损失 (简化版 ELBO) ---
    # 让 U-Net 根据此时的加噪图像 xt 和时间步 t 去预测噪声
    predicted_noise = model(xt, t)

    # L_simple: 真实噪声与预测噪声的 MSE 损失
    loss = F.mse_loss(predicted_noise, noise)

    return loss
```

9.4 总结：扩散是终极的 VAE?

通过对比 VAE，扩散模型有如下特点：

1. **舍弃推断的优化：**VAE 使用复杂的神经网络 ϕ 去努力学习数据的隐空间结构 $q_\phi(Z|X)$。扩散模型认为这太吃力不讨好，直接用暴力的高斯物理扩散摧毁数据结构。虽然它因此丢失了像 VAE 那样清晰的可解释低维流形（Latent Space），但由于去噪网络只需要完成极小的高斯步撤销，大大降低了单步学习难度，从而获得了更好的生成质量。

2. **相痛的贝叶斯框架：**无论是 VAE 的重参数化（$Z = \mu + \sigma\epsilon$），还是扩散模型的一步加噪（$X_t = \sqrt{\bar{\alpha}_t}X_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$）；无论是 VAE 的 ELBO，还是扩散模型的 $\mathcal{L}_{ELBO}(\theta)$。它们在底层由同一套变分贝叶斯推断的思想。

Take home message:

在未来自主设计的模型中，如果我们的目标是提取用于聚类、分型或寻找数据关联的“特征（Representation）”，我们坚守 VAE / 混合 CAVI 模型；如果我们的核心需求是毫无瑕疵地“生成”逼真的数据以进行数据增强，我们考虑 GAN 或者扩散模型。