图像生成算法：从 DDPM 到 Diffusion Model

Illusionna

15:53, Tuesday 4th August, 2026

目录

1 DDPM 哲学

对于 $\boldsymbol{x}_0$ 真实图像数据，例如一张归一化的图片，定义 DDPM 两个方向：

前向： $\boldsymbol{x}_0 \to \boldsymbol{x}_1 \to \boldsymbol{x}_2 \to \cdots \boldsymbol{x}_T$ 不断加入高斯噪声
反向： $\boldsymbol{x}_T \to \boldsymbol{x}_{T-1} \to \cdots \to \boldsymbol{x}_1 \to \boldsymbol{x}_0$ 不断消除噪声

对于普通加噪过程，以 $\boldsymbol{x}_0$ 为均值，β 为标准差，进行正态分布采样：

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \beta\boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$$

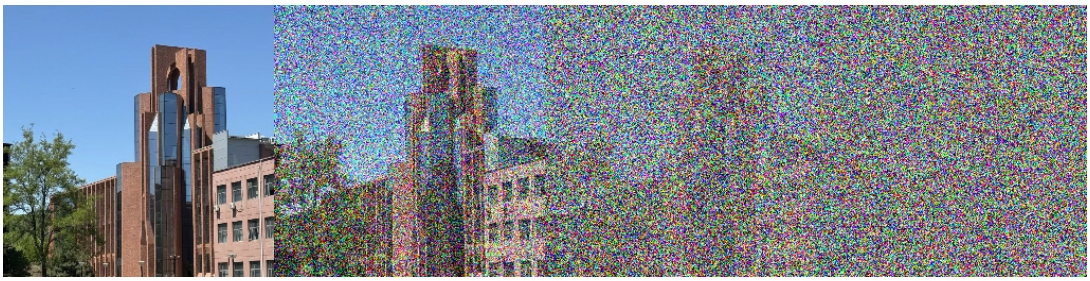
并且记该过程为 $q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)$ 前向转移的概率密度，我们的目标是希望随着前向链不断推进，最终的加噪图像呈现标准正态分布：

$$\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$$

显而易见，我们可以从 $\boldsymbol{x}_0$ 直接一步加噪得到 $\boldsymbol{x}_t$，即：

$$\boldsymbol{x}_t = \boldsymbol{x}_0 + t\beta\boldsymbol{\epsilon}$$

(a) 加噪 4 步过程片段



(b) 加噪 512 步过程片段

图 1: 随着图像 $\boldsymbol{x}_0$ 不断 DDPM 加噪声最终呈现标准高斯分布

但这存在两个问题，随着 t 增加，均值始终保持 $\boldsymbol{x}_0$，方差一直增大，最后 $\boldsymbol{x}_t$ 会变得越来越偏离标准正态分布，并且图像加噪后期要达到和前期相同的破坏效果，需要更大的噪声强度，而保持 β 始终常数是无法完成的.

因此 DDPM 进行了改进，对任意一步前向过程都是预先指定的马尔可夫链：

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\boldsymbol{E})$$

也就是 $\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}$，其中，有 $0 < \beta_1 < \beta_2 < \cdots < \beta_T < 1$，最终 DDPM 前向过程算法达到均值的缩放保证方差不会在每步无界累积的效果.

2 前向过程推演式

我们记 $\alpha_t = 1 - \beta_t$，观察现象：

$$\begin{aligned}
\boldsymbol{x}_1 &= \sqrt{\alpha_1}\boldsymbol{x}_0 + \sqrt{1-\alpha_1}\boldsymbol{\epsilon} \\
\boldsymbol{x}_2 &= \sqrt{\alpha_2}\boldsymbol{x}_1 + \sqrt{1-\alpha_2}\boldsymbol{\epsilon} \\
&= \sqrt{\alpha_2}\left(\sqrt{\alpha_1}\boldsymbol{x}_0 + \sqrt{1-\alpha_1}\boldsymbol{\epsilon}\right) + \sqrt{1-\alpha_2}\boldsymbol{\epsilon} \\
&= \sqrt{\alpha_2\alpha_1}\boldsymbol{x}_0 + \sqrt{1-\alpha_2\alpha_1}\boldsymbol{\epsilon} \\
\boldsymbol{x}_3 &= \sqrt{\alpha_3}\boldsymbol{x}_2 + \sqrt{1-\alpha_3}\boldsymbol{\epsilon} \\
&= \sqrt{\alpha_3}\left(\sqrt{\alpha_2\alpha_1}\boldsymbol{x}_0 + \sqrt{1-\alpha_2\alpha_1}\boldsymbol{\epsilon}\right) + \sqrt{1-\alpha_3}\boldsymbol{\epsilon} \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\boldsymbol{x}_0 + \sqrt{1-\alpha_3\alpha_2\alpha_1}\boldsymbol{\epsilon}
\end{aligned}$$

其中，这里使用统计学中正态分布性质，即多个独立同正态分布的随机变量值和仍然为正态分布，均值与方差等于多个正态分布均值与方差之和. 因此推导出前向过程 $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ 有：

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{x}_0, \sqrt{1 - \prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{E}\right)$$

其闭式采样为：

$$\boldsymbol{x}_t = \sqrt{\prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{x}_0 + \sqrt{1 - \prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{\epsilon}$$

这个现象是很明显的重参数化，训练只需随机抽一个 t 和一份噪声，而不用模拟 t 步. 此外，根据数学分析极限定理：

$$\lim_{t\to\infty} \prod_{n=1}^{t} \alpha_n = \lim_{t\to\infty} \prod_{n=1}^{t} (1 - \beta_n) = 0$$

因此 DDPM 前向过程解决了以上存在的两个问题，满足了当 t 增大时，最终加噪图像呈现 $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$ 标准正态分布.

例子

若 $\prod_{n=1}^{t} \alpha_n = 0.81$，则 $\boldsymbol{x}_t \approx 0.9\boldsymbol{x}_0 + 0.436\boldsymbol{\epsilon}$，其中信号功率为 0.81，噪声功率为 0.19.

3 反向过程去噪

理想的 DDPM 反向分布概率密度是 $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$，即根据上一个状态的 $\boldsymbol{x}_t$ 去噪生成当前状态 $\boldsymbol{x}_{t-1}$ 的图像，但问题在于它不可直接计算，因为我们计算一个实际的近似分布：

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$$

近似分布尽可能接近理想分布，即可达到去噪复原效果：

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \approx q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$$

在前向过程中，已知原图的后验，则有：

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}$$

上式含义是若同时知道 $\boldsymbol{x}_0$ 和 $\boldsymbol{x}_t$，那么最正确的单步去噪分布就是这个高斯分布. 由于给定 $\boldsymbol{x}_{t-1}$ 后，$\boldsymbol{x}_t$ 不再依赖 $\boldsymbol{x}_0$，则：

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}$$

由于一个 d 维度图像展平后，各向同性高斯密度对数为：

$$\log \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}, \sigma^2 \boldsymbol{E}\right) = -\frac{d}{2} \log\left(2\pi\sigma^2\right) - \frac{\|\boldsymbol{x} - \boldsymbol{\mu}\|^2}{2\sigma^2}$$

由于 $\boldsymbol{x}_{t-1}$ 是未知的随机变量，因此：

$$\begin{aligned}\log q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) &= \log q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) + \log q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0) - \log q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) \\ &= C - \frac{\|\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_{t-1}\|^2}{2\beta_t} - \frac{\left\|\boldsymbol{x}_{t-1} - \sqrt{\prod_{n=1}^{t-1}\alpha_n}\cdot\boldsymbol{x}_0\right\|^2}{2\left(1-\prod_{n=1}^{t-1}\alpha_n\right)}\end{aligned}$$

展开后得到：

$$\begin{cases}\|\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_{t-1}\|^2 = \alpha_t\|\boldsymbol{x}_{t-1}\|^2 - 2\sqrt{\alpha_t}\boldsymbol{x}_t^\top\boldsymbol{x}_{t-1} + C \\ \left\|\boldsymbol{x}_{t-1} - \sqrt{\prod_{n=1}^{t-1}\alpha_n}\cdot\boldsymbol{x}_0\right\|^2 = \|\boldsymbol{x}_{t-1}\|^2 - 2\sqrt{\prod_{n=1}^{t-1}\alpha_n}\cdot\boldsymbol{x}_0^\top\boldsymbol{y} + C\end{cases}$$

整合得到：

$$\log q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = C - \frac{1}{2}\left[\frac{1-\prod_{n=1}^{t}\alpha_n}{\beta_t\left(1-\prod_{n=1}^{t-1}\alpha_n\right)}\|\boldsymbol{x}_{t-1}\|^2 - 2\left(\frac{\sqrt{\alpha_t}}{\beta_t}\boldsymbol{x}_t + \frac{\sqrt{\prod_{n=1}^{t-1}\alpha_n}}{1-\prod_{n=1}^{t-1}\alpha_n}\boldsymbol{x}_0\right)^\top\boldsymbol{x}_{t-1}\right]$$

简记：

$$\boldsymbol{A} = \frac{1-\prod_{n=1}^{t}\alpha_n}{\beta_t\left(1-\prod_{n=1}^{t-1}\alpha_n\right)} \qquad \boldsymbol{v} = \frac{\sqrt{\alpha_t}}{\beta_t}\boldsymbol{x}_t + \frac{\sqrt{\prod_{n=1}^{t-1}\alpha_n}}{1-\prod_{n=1}^{t-1}\alpha_n}\boldsymbol{x}_0$$

则有：

$$\log q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = C - \frac{1}{2}\left(\boldsymbol{A}\|\boldsymbol{x}_{t-1}\|^2 - 2\boldsymbol{v}^\top\boldsymbol{x}_{t-1}\right)$$

又因为配方可得：

$$\boldsymbol{A}\|\boldsymbol{x}_{t-1}\|^2 - 2\boldsymbol{v}^\top\boldsymbol{x}_{t-1} = \boldsymbol{A}\left\|\boldsymbol{x}_{t-1} - \boldsymbol{v}\boldsymbol{A}^{-1}\right\|^2 + C$$

所以后验为：

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left[\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t), \tilde{\beta}_t\boldsymbol{E}\right]$$

其中：

$$\begin{cases}\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t) = \boldsymbol{v}\boldsymbol{A}^{-1} = \dfrac{\sqrt{\prod_{n=1}^{t-1}\alpha_n}\cdot\beta_t}{1-\prod_{n=1}^{t}\alpha_n}\boldsymbol{x}_0 + \dfrac{\sqrt{\alpha_t}\left(1-\prod_{n=1}^{t-1}\alpha_n\right)}{1-\prod_{n=1}^{t}\alpha_n}\boldsymbol{x}_t \\ \tilde{\beta}_t = \boldsymbol{A}^{-1} = \dfrac{1-\prod_{n=1}^{t-1}\alpha_n}{1-\prod_{n=1}^{t}\alpha_n}\beta_t\end{cases}$$

综上可知，当在确定原图 $\boldsymbol{x}_0$ 时，均值 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$ 是原图和当前噪声图的加权组合，方差 $\tilde{\beta}_t$ 是该步骤不可避免的不确定性.

4 AI 神经网络训练过程

由于真实的生成任务中没有 $\boldsymbol{x}_0$，只有 $\boldsymbol{x}_t$，所以不能直接使用 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$，因此我们定义一个神经网络来补上这个缺失的信息. 模型把每一步反向过程转移设置为一个高斯分布，把方差固定为已知度量，让 AI 神经网络学习决定均值：

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}\left[\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t), \sigma_t^2 \boldsymbol{E}\right]$$

其中与真实后验比较可知：

- 真实方差是 $\tilde{\beta}_t$，因此不需要网络学习方差，固定 $\sigma_t^2 = \tilde{\beta}_t$，工程设计中，可简化为 $\sigma_t^2 = \beta_t$.
- 需要 AI 神经网络学习的只有均值 $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$，使其尽可能接近真实均值 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$.

我们可以理解成 $\tilde{\beta}_t$ 用来设定反向采样的随机程度，而 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$ 是网络逼近正确的去噪方向.

5 从预测均值到拟合噪声

训练时，原始图像 $\boldsymbol{x}_0$ 经过前向加噪，有：

$$\boldsymbol{x}_t = \sqrt{\prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{x}_0 + \sqrt{1 - \prod_{n=1}^{t} \alpha_n} \cdot \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$$

其中的 $\boldsymbol{\epsilon}$ 是高斯分布，可以随即生成采样点，因而训练时它是已知答案，因此可以得到：

$$\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t - \sqrt{1 - \prod_{n=1}^{t} \alpha_n}}{\sqrt{\prod_{n=1}^{t} \alpha_n}}$$

因为真实均值：

$$\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t) = \frac{\sqrt{\prod_{n=1}^{t-1} \alpha_n} \cdot \beta_t}{1 - \prod_{n=1}^{t} \alpha_n} \boldsymbol{x}_0 + \frac{\sqrt{\alpha_t}\left(1 - \prod_{n=1}^{t-1} \alpha_n\right)}{1 - \prod_{n=1}^{t} \alpha_n} \boldsymbol{x}_t$$

再利用：

$$\prod_{n=1}^{t} \alpha_n = \alpha_t \prod_{n=1}^{t-1} \alpha_n$$

可得：

$$\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t) = \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1 - \prod_{n=1}^{t} \alpha_n}} \boldsymbol{\epsilon} \right)$$

改写后的 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$ 是神经网络训练的关键，真实去噪均值中的未知原图可以等价改写为当初加入的噪声 $\boldsymbol{\epsilon}$. 于是我们让噪声神经网络满足：

$$\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \approx \boldsymbol{\epsilon}$$

再以完全相同的形式定义神经网络模型均值：

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) = \frac{1}{\sqrt{\alpha_t}} \left[\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1 - \prod_{n=1}^{t} \alpha_n}} \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \right]$$

如果神经网络把噪声预测准确，则可达到我们想要的反向去噪效果：

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \approx \tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$$

6 更新神经网络参数 $\boldsymbol{\theta}$

训练过程中，真实噪声 $\boldsymbol{\epsilon}$ 已知，直接让网络输出与它接近即可：

$$\mathcal{L}(\boldsymbol{\theta}) = \|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) - \boldsymbol{\epsilon}\|_2^2$$

该损失函数是普通的 MSE，我们的目标是不断迭代使得损失尽可能小，而当前目标正是一个有现成标签的监督学习问题，因此一次训练迭代可概括为：

- 从图片集抽取一张干净的原始图像 $\boldsymbol{x}_0$.
- 对图片随机抽取时间步 t 生成高斯噪声图 $\boldsymbol{x}_t$.
- 神经网络输入 $\boldsymbol{x}_t$，得到预测 $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$.
- 计算 $\mathcal{L}(\boldsymbol{\theta})$ 损失函数 MSE.
- 反向传播更新梯度：
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \boldsymbol{\nabla} \mathcal{L}(\boldsymbol{\theta})$$

其中 η 是学习率，在 PyTorch 中通常用 AdamW 执行更新.

逻辑链

$$\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t) \implies \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \implies \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \boldsymbol{\nabla} \mathcal{L}(\boldsymbol{\theta})$$

7 神经网络推理生成图像

从 $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$ 标准正态分布噪声图像开始，对 $t = T, T-1, \ldots, 1$ 重复执行：

$$\boldsymbol{x}_{t-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) + \sigma_t \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$$

通常在最后一步 $t = 1$ 时，我们取 $\boldsymbol{\epsilon} = \boldsymbol{0}$，避免最后刚得到图像又额外引入一次随机扰动. 这便把我们前文求得的 $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t)$ 和 $\tilde{\beta}_t$ 与最终训练得到的参数 $\boldsymbol{\theta}$ 连成了完整的 DDPM 过程.

8 总结

DDPM 不是与 Diffusion Model 并列的另一个模型，而是 Diffusion Model 的一个具体实现. 更一般地说，扩散模型都遵循同一条生成思路：先定义一个逐渐破坏数据结构的正向过程，再学习其逆过程，最后从简单噪声分布逐步生成数据. 若把正向过程记为 q，模型学习的反向过程记为 $p_{\boldsymbol{\theta}}$，则这一共同结构可写为：

$$\boldsymbol{x}_0 \xrightarrow{q} \boldsymbol{x}_1 \xrightarrow{q} \cdots \xrightarrow{q} \boldsymbol{x}_T \approx \mathcal{N}(\mathbf{0}, \boldsymbol{E}) \qquad \Longrightarrow \qquad \boldsymbol{x}_T \xrightarrow{p_{\boldsymbol{\theta}}} \cdots \xrightarrow{p_{\boldsymbol{\theta}}} \boldsymbol{x}_1 \xrightarrow{p_{\boldsymbol{\theta}}} \boldsymbol{x}_0$$

DDPM 提供的是 Diffusion Model 的基本范式：**把复杂数据分布的生成问题转化为一系列较容易学习的局部去噪问题**.

DDPM 与 Diffusion Model 的关系

Diffusion Model 是“加噪 → 学习逆过程 → 从噪声生成”的总框架
DDPM 是该框架在离散高斯马尔可夫链上的一种具体设计实现