

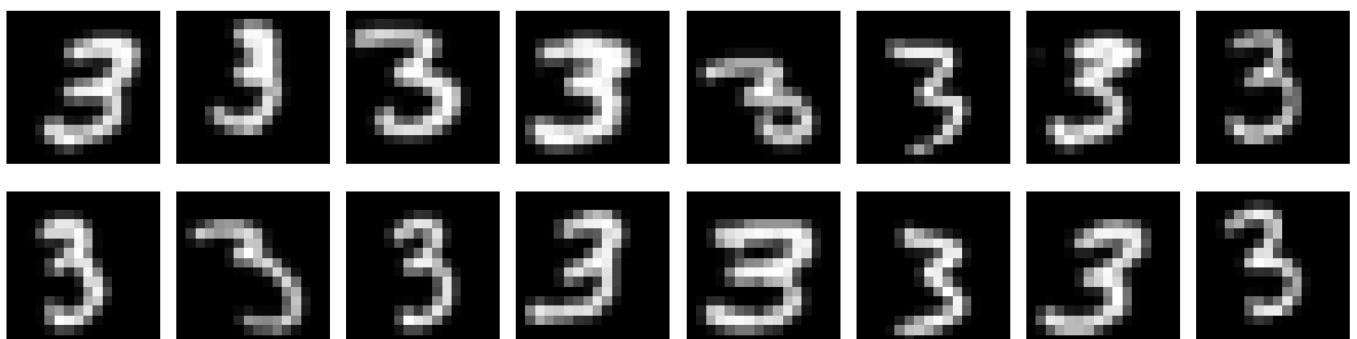
# Generate Image Base On Flow Matching

- [An Introduction to Flow Matching and Diffusion Models](#)
- 苏剑林. (Feb. 23, 2023). 《生成扩散模型漫谈（十七）：构建ODE的一般步骤（下）》 [Blog post]. Retrieved from [生成扩散模型漫谈（十七）：构建ODE的一般步骤（下） - 科学空间|Scientific Spaces](#)
- [通俗易懂的Flow Matching原理解读（附核心公式推导和源代码）](#)

## 从概率分布的角度理解图像

### 下采样后的手写数字数据集MNIST

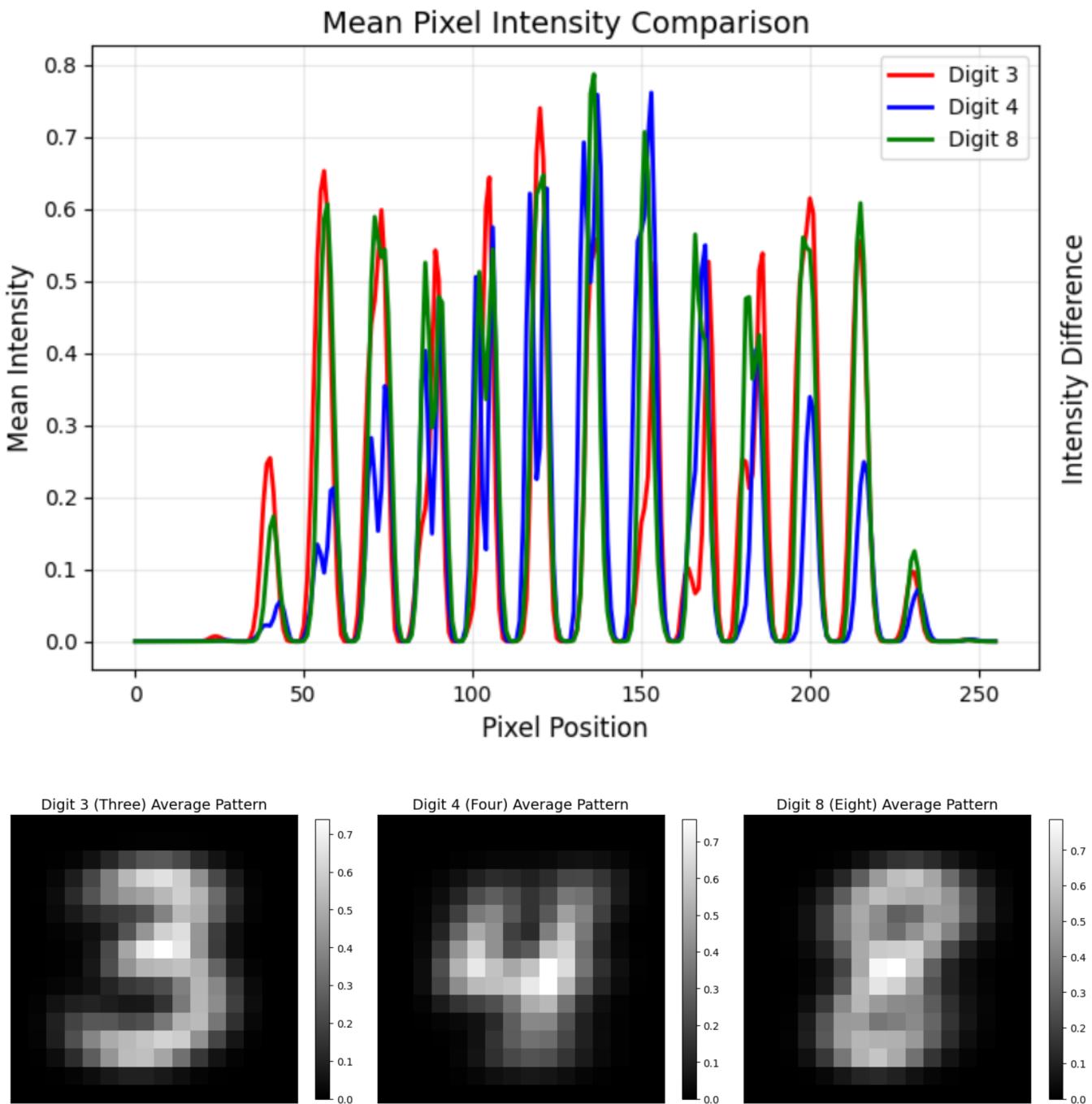
16 labeled images of digit 3 after downsampling to 16x16



- 这种图片总共有 $16 \times 16$ 个像素，怎么理解这些图像的分布是什么？

最粗暴直接的理解是，一张 $16 \times 16$ 的图片，可以被拉直成为一个256维的向量 $V$ 。维度之间并不独立，因为相邻像素之间存在空间相关性，例如数字“3”中尖角的形状，往往由特定位置的像素组合共同决定。

- 这些数字3的图片，都属于256维空间里的一个分布。
- 其中任意一张数字3的图片，都属于这些分布下的一个采样。
- 不同的手写数字图片，分布长不一样（下图对比了数字3,4,8）



## 怎么定义“图像生成”

"Creating noise from data is easy; creating data from noise is generative modeling."

- 用数值表示图像：

- 考虑  $H \times W$  像素  $H$  描述高度和  $W$  图像宽度，每个宽度包含三个颜色通道（RGB）。对于每个像素和每个颜色通道，我们都给出了 的强度值  $\mathbb{R}$  .因此，图像可以用一个元素表示  $z \in \mathbb{R}^{H \times W \times 3}$  .

- 生成图像，即生成采样：
  - 当我们说“生成一张狗的图”时，并不存在唯一标准的答案。哈士奇是狗，柯基也是狗；侧面的狗是狗，正面的狗也是狗。人类对好坏的判断是主观的且多样的。
- 为了让计算机理解这种多样性，我们引入了概率分布（Probability Distribution）的概念，记作  $p_{data}$ 
  - **高概率区域：**看起来非常像狗的图片（比如一张清晰的金毛照片），在这个分布中对应的数值（可能性）就很高。
  - **低概率区域：**看起来不太像狗，或者完全是乱码的图片，在这个分布中对应的数值就很低。
- 对计算机来说，图片生成可以描述成，从真实数据分布  $p_{data}$  中采样
- 用数据集模拟  $p_{data}$ 
  - 在现实世界中，我们**根本不知道**  $p_{data}$  这个完美的数学公式长什么样。
    - 所以我们要去收集一大堆现实中的狗的图片，**它是真实分布的一个有限的近似采样**。数据集越丰富，它能更好地代表  $p_{data}$  分布。
  - 从噪声到数据

- 到目前为止，我们讨论了生成建模的“**是什么 (what)**”：即从  $p_{data}$  中生成样本。在这里，我们将简要讨论“**怎么做 (how)**”。
- 为此，我们假设我们可以访问某种**初始分布 (initial distribution)**， $p_{init}$ ，且我们可以很容易地从这个分布中进行采样。比如高斯分布  $p_{init} = N(0, I_d)$ 。
- 那么，生成建模的目标就是：将从初始分布  $x_0 \sim p_{init}$  中采样的样本，**转换为**符合数据分布  $x_1 \sim p_{data}$  的样本。

## 怎么用数学描述 $x_0$ 到 $x_1$ 的变化过程？

常微分方程(ODEs)描述的是**状态随一个自变量 (通常是时间) 变化的规律**，我们需要构建一个常微分方程来描述这个变换。

一个 ODE 的解被定义为一条**轨迹 (trajectory)**，即形式为如下的函数

$$X(t) \in R^d, t \in [0, 1]$$

该函数将时间  $t$  映射到空间  $R^d$  中的某个位置  $X_t$ 。

每一个 ODE 都是由一个**向量场 (vector field)**  $u$  定义的，即形式为如下的函数：

$$u(x, t) \in R^d, x \in R^d, t \in [0, 1]$$

这个公式的意思是，对于每一个时间  $t$  和 位置  $x$ ，我们都能得到一个向量  $u(x, t)$ ，它指定了空间的一个**速度**。 $u(x, t)$  可以用  $u_t(x)$  表示

我们希望轨迹  $X$ ，能够顺着向量场  $u_t(x)$  方向延伸。我们可以将这样一条轨迹形式化为以下的的解，这个方程就是一个 常微分方程 (ODEs)：

$$\begin{cases} \frac{d}{dt} X_t = u_t(X_t), \\ X_0 = x_0 \end{cases}$$

这个方程组要求  $X_t$  的导数 (即变化率) 必须由  $u_t$  给出的方向来指定。并且要求我们在时间  $t = 0$  时从  $x_0$  出发。

再进一步，如果我们想要知道  $t = 0$  时采样  $x_0$  开始沿着向量场变化，那么在时间  $t$  时我们在哪里 (即  $X_t$  是多少)，我们要怎么描述这个事情？

这个问题的答案由一个被称为 **flow** ( $\psi$ ) 的函数给出。

$$X_t = \psi(x_0, t) \in R^d$$

下面记作  $\psi_t(x_0)$

所以现在有

$$\begin{cases} \frac{d}{dt}\psi_t(x_0) = u_t(\psi_t(x_0)), \\ \psi_0(x_0) = x_0 \end{cases}$$

根据这个方程组，给定一个初始条件  $X_0 = x_0$ ，ODE 的轨迹可以通过  $X_t = \psi(X_0)$  被恢复出来。到目前为止， $x_0$  到  $x_1$  的过程就被建模出来了。

## 这套ODE一定有解吗？

这套ODE一定有解吗？换句话说，给定了采样  $x_0$  和  $x_1$ ，一定存在一条轨迹（flow），让他们变换吗？

这里给出的结论是：如果  $u_t$  连续可微且导数有界，这套 ODE 方程存在解且存在唯一解（路径唯一）

## 神经网络Flow match

到目前为止，我们可以确定，这个转换流程可以用数学建模了。

接下来我们可以用神经网络  $u_t^\theta(Xt)$ ，来拟合向量场  $u_t$ 。模型在前向推理的过程，就是不断构建流的过程。

这种用神经网络拟合变换向量场的思想，也就是所谓的 flow match

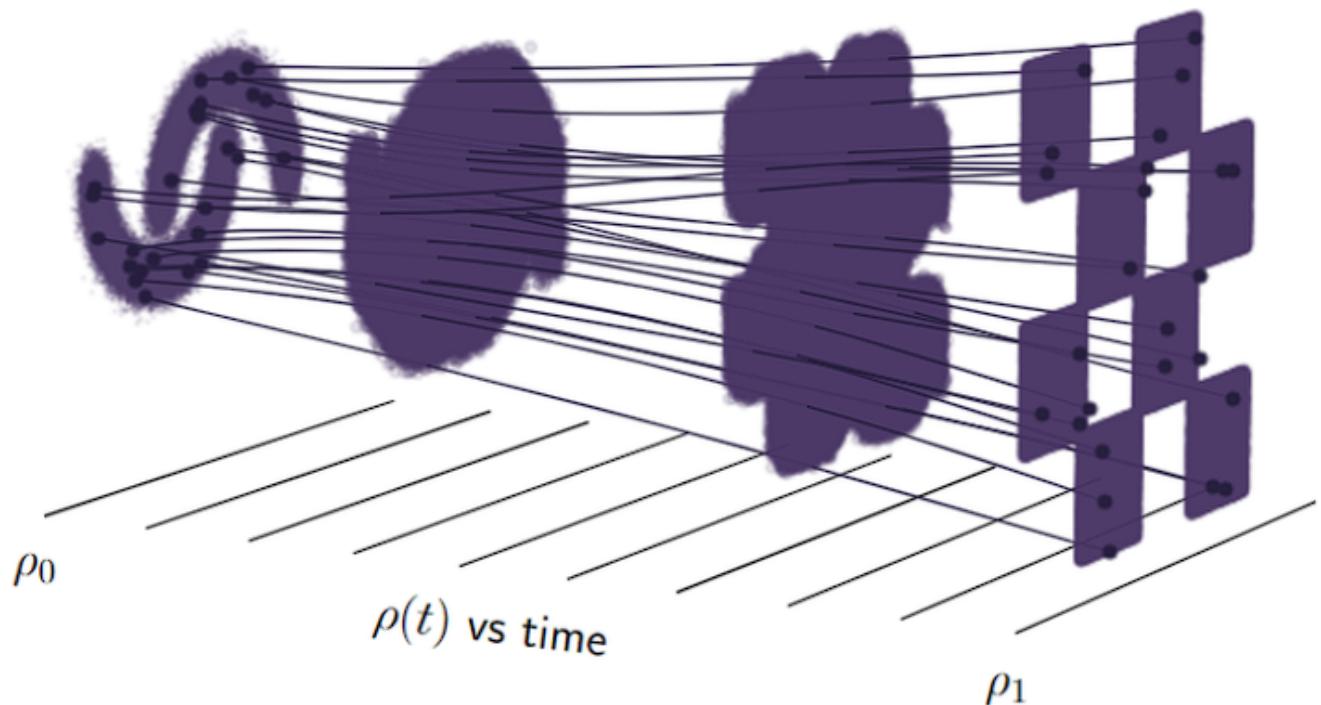
神经网络参数化（学习）的是向量场，而不是流本身

为了计算出流（即得到最终结果），我们需要模拟求解这个 ODE

## 神经网络的训练目标

回顾一下，现在起始分布是  $p_{init}$ ，目标分布是  $p_{data}$ ，转换过程之的分布是  $p_t$

# ODE



为了训练出一个模拟向量场的神经网络，我们写出这样的目标函数（论文就是用这个二范数来拟合）

$$\mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t) - u_t(x_t)\|^2]$$

这个函数的意思是，不断在分布转移过程中的采样  $X_t$ ，把它输入神经网络  $v_t^\theta$  得到一个值，然后放进向量场  $u_t$ ，得到一个值，来计算损失值。

很明显，我们没法根据这个目标函数训练我们的网络，因为我们不知道  $P_t$ ，也不知道  $u_t$

---

## 改写训练函数

---

为了克服这些困难，并进行训练。我们需要实现两个关键目标：

- 找到构建  $P_t(X_t)$  的方法：必须找到一种方式来构建或定义中间分布  $P_t(X_t)$ 。这个分布定义了在时间  $t$  时的采样到样本  $X_t$  的概率。
- 找到获取  $u_t(X_t)$  向量的方法：必须找到一种方法来获取（即计算或近似）由理想向量场  $u_t(X_t)$  定义的向量，因为这些向量是神经网络  $v_t^\theta$  的训练目标

在开始之前，先改写一下这个目标函数 展开这个平方项，我们要计算梯度的部分主要包含交叉项：

$$\begin{aligned} & \mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t) - u_t(x_t)\|^2] \\ &= \mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t)\|^2 - 2 \cdot v_t^\theta(x_t) \cdot u_t(x_t) + \|u_t(x_t)\|^2] \\ &= \mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t)\|^2 + \|u_t(x_t)\|^2] - \mathbb{E}_{x_t \sim p_t(x_t)} [2 \cdot v_t^\theta(x_t) \cdot u_t(x_t)] \end{aligned}$$

下面要用到一个叫**边缘化**的技巧，这个技巧用来展开  $u_t$

$$\begin{aligned} & \mathbb{E}_{x_t \sim p_t(x)} [2 \cdot v_t^\theta(x_t) \cdot u_t(x_t)] \\ &= 2 \cdot \int v_t^\theta(x_t) \cdot u_t(x_t) \cdot p_t(x_t) dx_t \\ &= 2 \cdot \int v_t^\theta(x_t) \cdot (\int u_t(x_t|x_1) \frac{p_t(x_t|x_1)p_{data}(x_1)}{p_t(x_t)}, dx_1) \cdot p_t(x_t) dx_t \end{aligned}$$

下面我们来解释一下这个公式展开的含义：

$$u_t(x_t) = \int u_t(x_t|x_1) \frac{p_t(x_t|x_1)p_{data}(x_1)}{p_t(x_t)} dx_1$$

我们无法直接得到边缘向量场  $u_t$ ，因为这需要遍历所有数据；但我们可以构造一个“条件向量场”，只要神经网络拟合了这个条件向量场的期望，它也就拟合了边缘向量场。这利用了期望的性质

$$\nabla_\theta E[\|v_\theta - u\|^2] = \nabla_\theta E[\|v_\theta - u_{cond}\|^2]$$

具体来说，我们会遍历我们的整个数据集，对于数据集中的每一个数据点  $x_1$ ，我们都会计算每个向量（即从  $x_t$  指向该  $x_1$  的向量）。然后，我们会用一个加权因子乘这个向量。这个权重（加权因子）是  $P_t(x_t|x_1) = \frac{p_t(x_t|x_1)p_{data}(x_1)}{p_t(x_t)}$ 。这个概率密度函数决定每个数据点/向量有多少影响力，即它衡量了数据点  $x_1$  有多大可能性在时间  $t$  时出现在位置  $x_t$ 。

接下来继续回到上面的公式，我们现在要将这个积分化回期望的形式

$$\begin{aligned} &= 2 \cdot \int v_t^\theta(x_t) \cdot (\int u_t(x_t|x_1) \frac{p_t(x_t|x_1)p_{data}(x_1)}{p_t(x_t)}, dx_1) \cdot p_t(x_t) dx_t \\ &= 2 \cdot \iint v_t^\theta(x_t) \cdot u_t(x_t|x_1) p_t(x_t|x_1) p_{data}(x_1) dx_1 dx_t \\ &= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1)] \end{aligned}$$

现在把这个精简后的项放回一开始的目标函数

$$\begin{aligned} & \mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t) - u_t(x_t)\|^2] \\ &= \dots \end{aligned}$$

$$= \mathbb{E}_{x_t \sim p_t(x_t)} [\|v_t^\theta(x_t)\|^2 + \|u_t(x_t)\|^2] - \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1)]$$

这里同样要改变前面那一项的期望的采样

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t)\|^2 + \|u_t(x_t)\|^2] - \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1)]$$

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t)\|^2 + \|u_t(x_t)\|^2 - 2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1)]$$

根据论文讲解，这里有一个trick，就是加一项  $\|u_t(x_t|x_1)\|^2$ ，再减一项，我们用这个项可以凑一个完全平方公式

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t)\|^2 + \|u_t(x_t)\|^2 - 2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1) + \|u_t(x_t|x_1)\|^2 - \|u_t(x_t|x_1)\|^2]$$

把第二项移动到后面

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t)\|^2 - 2 \cdot v_t^\theta(x_t) \cdot u_t(x_t|x_1) + \|u_t(x_t|x_1)\|^2 + \|u_t(x_t)\|^2 - \|u_t(x_t|x_1)\|^2]$$

前三项凑成一个完全平方公式

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t) - u_t(x_t|x_1)\|^2 + \|u_t(x_t)\|^2 - \|u_t(x_t|x_1)\|^2]$$

$$= \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t) - u_t(x_t|x_1)\|^2] + \mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|u_t(x_t)\|^2 - \|u_t(x_t|x_1)\|^2]$$

从上面的式子可以看到，后一项期望和我们要训练的参数无关，直接把它当成常数就好（只要训练的初始分布  $p_{init}$  和目标分布  $p_{data}$  确定了，后一项期望就确定了）。自此，我们可以将目标函数简化为，这个又叫做 Conditional Flow Matching

$$\mathbb{E}_{x_t \sim p_t(x_t|x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t) - u_t(x_t|x_1)\|^2]$$

## 实例化路径-重参数化

从现在这个目标函数看，数据集已经用上了。尽管目标函数的结构已经确定，但构成这个目标函数的两个关键组件——条件向量场  $u_t(x_t | x_1)$  和 条件概率密度函数  $P_t(x_t | x_1)$ ——在数学上仍然是泛化公式，需要给出具体、可计算的定义。

### 定义 $x_0$

Flow Matching 方法和其他扩散模型一样，都是从一个已知的分布  $p_{init}(x_0)$  开始。作者们首先规定了流的输入  $x_0$  来自一个正态分布 (normal distribution)。

## 定义流的变换：

作者用高斯分布的规范变换 (canonical transformation for gaussian distributions) 作为流的变化形式。

$$\psi_t(x_0) = \sigma_t(x_1)x_0 + \mu_t(x_1)$$

Flow Matching 框架允许我们自由定义从噪声  $x_0$  到数据  $x_1$  的变换路径。为了计算简便，我们通常选择**简单的线性插值路径**，即让

$$\sigma_t(x_1) = 1 - t$$

$$\mu_t(x_1) = tx_1$$

代入那个高斯分布的规范变换之后就变成

$$\psi_t(x_0) = (1 - t)x_0 + tx_1$$

## 重参数化目标函数

现在有了flow函数的定义，我们要进一步实例化我们的目标函数。

在训练时，虽然我们不知道真实的边缘向量场， $u_t(x_t)$ ，但在每个训练步里， $u_t(x_t)$  等价于  $u_t(x_t | x_1)$ ，我们只需要拟合**条件向量场**即可

进一步有

$$x_t = \psi_t(x_0) = (1 - t)x_0 + tx_1$$

$$u_t(x_t | x_1) = \frac{d}{dt}x_t = \frac{d}{dt}((1 - t)x_0 + tx_1) = x_1 - x_0$$

这意味着：条件向量场是一个恒定的速度向量，方向直指目标样本  $x_1$ ，大小为起点到终点的距离。

所以训练函数可以写成

$$\begin{aligned} \mathbb{E}_{x_t \sim p_t(x_t | x_1), x_1 \sim p_{data}} [\|v_t^\theta(x_t) - u_t(x_t | x_1)\|^2] &= \\ \mathbb{E}_{x_t \sim p_t(x_t | x_1), x_1 \sim p_{data}} [\|v_t^\theta(\psi_t(x_0)) - x_1 - x_0\|^2] \end{aligned}$$

这就是 Conditional Flow Matching (CFM) 的全部奥秘：将复杂的生成问题转化为了一个简单的**监督回归问题**。