Consistency Models基本原理解析

作者：Tong Tong

B站主页：Double童发发

版本

- v1.1-20240911：修改部分概率表达符号不规范错误
- v1.0-20240904：初版形成

前言

一致性模型（Consistency Models，CM）主要解决的是扩散生成模型迭代采样速度慢的问题，支持一步采样快速生成和多步采样高精度生成，同时支持zero-shot图像编辑（图像重绘、超分辨率重构等）。如何实现尽可能少的采样或者说一步到位的生成呢？按照老司机开车理论的逻辑，最直接的办法就是只要上路开车了，随时都能够“瞬移”到采样的终点就可以。这个瞬移的技能可以通过一个神经网络模型学习，也即采用一个神经网络f_θ将任意时刻的噪声图像$\mathbf{x}_t$变换回初始图像$\mathbf{x}_0$，很容易写成下面的这种表示形式：

$$f_\theta(\mathbf{x}_t, t) = f_\theta(\mathbf{x}_{t'}, t'),\ \forall t' \in [\varepsilon, T]$$

其中，边界条件为$f_\theta(\mathbf{x}_\varepsilon, \varepsilon) = \mathbf{x}_\varepsilon$。CM的“瞬移”技能支持两种训练方式，第一种从预训练好的扩散模型中偷学（也即蒸馏），第二种直接从零开始训练。实验结果也充分说明了CM的有效性，在当时也属于一种SOTA的算法。本讲稿涉及的论文为宋博士发表的《Consistency Models》，这篇论文逻辑性很严密，关键数学推导过程非常详细（虽然一半我也看不懂），非常推荐大家去读一读！

问题1：一致性模型的理论依据是什么？

一致性模型的故事还要从随机微分方程（Stochastic Differential Equation，SDE）大一统扩散模型开始讲起，先来回顾一下SDE统一扩散模型视角下的加噪和去噪过程。

前向加噪过程

基于SDE的前向加噪过程可描述为：

$$\mathrm{d}\mathbf{x}_t = f(\mathbf{x}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t \tag{1}$$

其中，$t \in [0, T]$且$T > 0$，f和g分别是漂移（drift）和扩散（diffusion）因子，$\mathbf{w}_t$是维纳（Wiener）过程，也是布朗运动过程，$p_T(\mathbf{x}_T)$表示先验分布，$p_0(\mathbf{x}_0)$表示数据分布。

逆向去噪过程

基于SDE的逆向去噪过程可描述为：

$$\mathrm{d}\mathbf{x}_t = \left[f(\mathbf{x}_t, t) - \frac{1}{2}g^2(t)\nabla \log p_t(\mathbf{x}_t)\right]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t \tag{2}$$

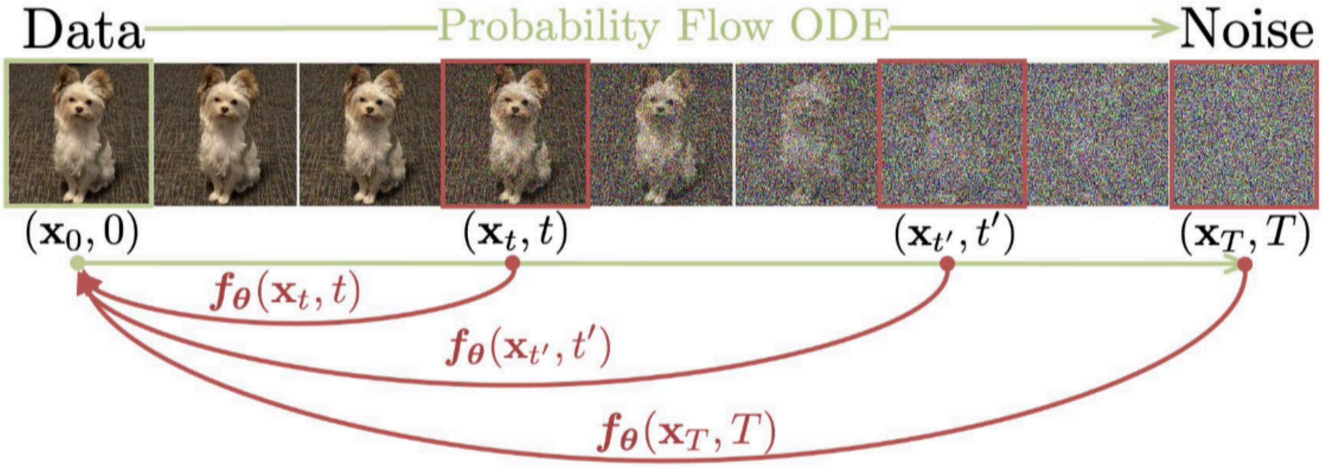
其中，$\nabla \log p_t(\mathbf{x}_t)$是分数score，$\bar{\mathbf{w}}_t$是逆向维纳过程。**SDE通常存在一个对应的ODE形式，也就是把维纳过程去掉，变为一种流（Flow），这个流叫做概率流，也是一个常微分方程，因此叫做概率流常微分方程（Probability Flow Ordinary Differential Equation，PF-ODE）**，形式如下：

$$\mathrm{d}\mathbf{x}_t = \left[f(\mathbf{x}_t, t) - \frac{1}{2}g^2(t)\nabla \log p_t(\mathbf{x}_t)\right]\mathrm{d}t \tag{3}$$

在这里，f和g的取法就多种多样了，作者在这里采用了Karras论文中的一种，令$f(\mathbf{x}_t, t) = 0$，$g(t) = \sqrt{2t}$。根据score matching对应SDE形式，可得$p_t(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}, t^2\mathbf{I})$，时间和噪声等价了！再根据score matching算法，用一个模型s_θ去进行近似，也即$s_\theta = \nabla \log p_t(\mathbf{x}_t)$。将上述已知信息代入PF-ODE方程中，可得：

$$\mathrm{d}\mathbf{x}_t = -ts_\theta(\mathbf{x}_t, t)\mathrm{d}t \tag{4}$$

这个公式也被成为经验（empirical）PF-ODE。和flow matching类似，这里从先验分布中采样一个样本，然后采用欧拉法等数值求解方法，逐步迭代，可获得起点$\mathbf{x}_0$的图像。需要注意的是，通常这个$\mathbf{x}_0$都替换为$\mathbf{x}_\varepsilon$，ε是一个非常小的正数（例如0.002），因为直接求到$\mathbf{x}_0$可能会导致数值不稳定现象。



有了PF-ODE后，就存在了一条固定而不是随机的逆向去噪路线，也就可以训练一个神经网络，把这条固定路线上的每个点的神经网络输出都等于x_ε，这也就是一致性模型构建的理论基础！

问题2：如何利用神经网络训练出一致性模型？

一致性函数

给定一个PF-ODE路径$\{\mathbf{x}_t\}_{t\in[\varepsilon,T]}$，它的一致性函数$f$的形式为：

$$f(\mathbf{x}_t, t) = \begin{cases} \mathbf{x}_\varepsilon, & t = \varepsilon \\ f(\mathbf{x}_{t'}, t'), & t \in (\varepsilon, T], \forall t' \in [\varepsilon, T] \end{cases} \tag{5}$$

一致性函数需要满足每个点$(\mathbf{x}_t, t)$都在同一个PF-ODE的路径上

一致性模型

一致性模型就是采用一个神经网络，去模仿一致性函数的特性，实现方式多种多样。论文采用了输入和输出维数一致的**任意神经网络$F_\theta(\mathbf{x}_t, t)$**，并给定了如下一种模仿一致性函数实现方式：

$$f_\theta(\mathbf{x}_t, t) = C_{\text{skip}}(t)\mathbf{x}_t + C_{\text{out}}(t)F_\theta(\mathbf{x}_t, t) \tag{6}$$

其中C_{skip}和C_{out}是两个关于时间t的可微函数，保证$C_{\text{skip}}(\varepsilon) = 1$和$C_{\text{out}}(\varepsilon) = 0$，满足一致性函数的要求。实际上在代码实现的时候，作者玩了一点“小技巧”，并没有严格按照上面的式子来进行：

一致性模型实现代码

```python
def denoise(self, model, x_t, sigmas, **model_kwargs):
        import torch.distributed as dist

        if not self.distillation:
                c_skip, c_out, c_in = [
                        append_dims(x, x_t.ndim) for x in self.get_scalings(sigmas)
                ]
        else:
                c_skip, c_out, c_in = [
                        append_dims(x, x_t.ndim)
                        for x in self.get_scalings_for_boundary_condition(sigmas)
                ]
        rescaled_t = 1000 * 0.25 * th.log(sigmas + 1e-44)
        model_output = model(c_in * x_t, rescaled_t, **model_kwargs)
        denoised = c_out * model_output + c_skip * x_t
        return model_output, denoised
```

可以发现，在实现的时候多了一个C_{in}，来限制输入图像，有点像LSTM的输入门。C_{skip}、C_{out}为**人为设定**，和时间t相关，当t增加时，图像噪声水平增加，离采样终点越来越远，C_{skip}的值会下降，C_{out}的值会上升，更少的输入信号$\mathbf{x}_t$被保留，更多依靠模型去进行预测。相反，当t减少时，图像噪声水平下降，离采样终点越来越近，C_{skip}的值会上升，C_{out}的值会下降，更多的输入信号$\mathbf{x}_t$被保留，更少依靠模型去进行预测。至于C_{in}，先不讨论，因为是Karras提出的EDM算法中涉及的。

综上所述，一致性模型的特点如下：

1. 支持一步生成
2. 支持多步采样，也即拥有用“时间换质量”的能力
3. 能进行zero-shot data editing任务

损失函数

一致性模型采用了让PF-ODE**相邻两个时间点模型输出值差距最小化的方式**实现利用神经网络逼近一致性函数的目标，损失函数可以写为：

$$\mathcal{L}^N(\theta) = \mathbb{E}[\|f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n)\|_2^2] \tag{7}$$

其中，N表示时间点设置的数目，$\hat{\mathbf{x}}_{t_n}$是通过一种ODE求解器获得的上一个时刻t_n的图像，**这里用t_n表示第n个时间点对应的时间**，$\mathbf{x} \sim p_{\text{data}}$，$\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}, t_{n+1}^2 \mathbf{I})$，$n \sim \mathcal{U}[1, N-1]$且为整数。$N$是一个超参数，理论这个值越大，在PF-ODE上的路径点数目就会越多，两点之间越靠近，$\hat{\mathbf{x}}_{t_n}$的求解会更准，模型精度越好。实际上当N足够大以后，这个数值对模型性能影响已经不敏感。作者在这里并没有直接使用上面的损失函数形式，而是将上式后一项的f的权重θ换成了模型的指数滑动平均值（Exponential Moving Average，EMA）θ^-。根据EMA的性质，给定衰减系数$0 \le \mu < 1$，可得：

$$\theta^- \leftarrow \text{stopgrad}(\mu\theta^- + (1-\mu)\theta) \tag{8}$$

此时，损失函数可写为：

$$\mathcal{L}^N(\theta, \theta^-) = \mathbb{E}[\|f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f_{\theta^-}(\hat{\mathbf{x}}_{t_n}, t_n)\|_2^2] \tag{9}$$

该式便是论文中损失函数的一种简化形式。采用EMA相当于用一个多轮加权平均的结果**充当最终的模型**，每一步不同的模型权重θ对其影响较小，因此f_{θ^-}又被称作“目标模型”。采用EMA的原因是可以提升训练过程的稳定性，提升一致性模型的效果。有了损失函数的形式了，就可以考虑如何训练模型了。论文中给定了两种方法，一种是从已有模型切入，也即一致性蒸馏（Consistency Distillation，CD），一种是从零开始训练一个新的一致性模型，也即一致性训练（Consistency Training，CT）。

从已有模型切入 — 一致性蒸馏

一致性模型的训练可以从已有模型切入，比如采用score-based模型进行蒸馏。假设现在已经有一个模型$s_\theta(\mathbf{x}_t, t)$，很自然地，通过模型预测出一个score的值，再通过欧拉法采样一步，就可以获得在PF-ODE路径上相邻点的位置。根据PF-ODE形式，很显然有：

$$\hat{\mathbf{x}}_{t_n} = \mathbf{x}_{t_{n+1}} - (t_n - t_{n+1}) t_{n+1} s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) \tag{10}$$

代入损失函数中可得CD的损失函数形式为：

$$\mathcal{L}_{CD}^{N}(\theta, \theta^-) = \mathbb{E}[\| f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f_{\theta^-}(\mathbf{x}_{t_{n+1}} - (t_n - t_{n+1}) t_{n+1} s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), t_n) \|_2^2] \tag{11}$$

论文中给出了CD过程的算法步骤如下，结合损失函数来看不难理解。首先从数据中采样样本，均匀采样时间点n，根据时间加噪声。加完噪声后，采用ODE数值求解器获得上一个时间点的图像，紧接着计算损失函数并进行梯度反传。其中，Φ表示模型学习的“司机”，可以是score，也可以是速度场等ODE迭代必须元素。唯一需要注意的是，每一步需要同步更新EMA权重θ^-的值。

Algorithm 2 Consistency Distillation (CD)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, ODE solver $\Phi(\cdot, \cdot; \boldsymbol{\phi})$, $d(\cdot, \cdot)$, $\lambda(\cdot)$, and μ

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

  Sample $\mathbf{x} \sim \mathcal{D}$ and $n \sim \mathcal{U}[\![1, N-1]\!]$

  Sample $\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}; t_{n+1}^2 \boldsymbol{I})$

  $\hat{\mathbf{x}}_{t_n}^{\boldsymbol{\phi}} \leftarrow \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) \Phi(\mathbf{x}_{t_{n+1}}, t_{n+1}; \boldsymbol{\phi})$

  $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \boldsymbol{\phi}) \leftarrow$

    $\lambda(t_n) d(\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\mathbf{x}}_{t_n}^{\boldsymbol{\phi}}, t_n))$

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \boldsymbol{\phi})$

  $\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu \boldsymbol{\theta}^- + (1-\mu) \boldsymbol{\theta})$

until convergence

作者对CD的损失函数进行了深刻分析，证明了当$\mathcal{L}_{CD}^{N}(\theta, \theta^-) = 0$时，学习的神经网络模型$f_\theta$会逼近真实的一致性函数$f$。

从零开始训练 — 一致性训练

一致性模型难道一定要背靠一个现成的扩散模型吗？不！咱们也可以从零开始训练一个一致性模型。可以发现，没有现成扩散模型的最大问题是Φ没了，没办法进行ODE的数值求解了。作者就给定了一个score-based模型$s_\theta(\mathbf{x}_t, t)$预测结果$\nabla \log p_t(\mathbf{x}_t)$的平替，用这个平替让模型进行学习，进而替代$s_\theta(\mathbf{x}_t, t)$。这个平替需要至少满足以下两个条件：

> 1. 是$\nabla \log p_t(\mathbf{x}_t)$的无偏估计
> 2. 需要在训练过程中作为金标准，因此需要可以计算获得

基于上述两个条件，作者给出了平替的形式为：

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = -\mathbb{E}\left[\frac{\mathbf{x}_t - \mathbf{x}}{t^2} \middle| \mathbf{x}_t \right] \tag{12}$$

其中，$\mathbf{x}_t \sim \mathcal{N}(\mathbf{x}, t^2\mathbf{I})$，$\mathbf{x} \sim p_{\text{data}}$。上式获得的方法有两个，一个是作者提出的“功法正练”之法，按部就班进行推导，另外一种是我提出的“功法逆练”快速看破（但不严谨）。

功法正练：一板一眼推导

先来看“功法正练”对应的推导过程，论文中推导的前几步可能稍微难理解一点，在这里给大家全部补上。回忆一下概率论的基本概念，根据边缘概率密度与联合概率密度的关系，可得如下的等式：

$$p_t(\mathbf{x}_t) = \int p(\mathbf{x}_t, \mathbf{x}) \mathrm{d}\mathbf{x} = \int p(\mathbf{x}_t|\mathbf{x}) p_{\text{data}}(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{13}$$

接下来就是公式推导，公式推导的目的就是从score的定义出发，探索到底有没有一个关于score的无偏估计量。总体而言，宋博士在这里的推导过程写的相当清晰，几乎没有跳步，仅在开始两步稍微有点难度，在这里给大家一一分解。根据score的定义，$\log p_t(\mathbf{x}_t)$关于$\mathbf{x}_t$求导，实际上是一个**复合函数求导**！也即先对log求导，再对$p_t(\mathbf{x}_t)$求导！心中牢记复合函数求导，则推导过程并不难理解，如下所示：

$$\begin{aligned}
\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) &= \frac{1}{p_t(\mathbf{x}_t)} \cdot \nabla_{\mathbf{x}_t} p_t(\mathbf{x}_t) \quad \text{复合函数求导} \\
&= \frac{\nabla_{\mathbf{x}_t} \int p(\mathbf{x}_t|\mathbf{x}) p_{\text{data}}(\mathbf{x}) \mathrm{d}\mathbf{x}}{p_t(\mathbf{x}_t)} \\
&= \frac{\int \nabla_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{x}) p_{\text{data}}(\mathbf{x}) \mathrm{d}\mathbf{x}}{p_t(\mathbf{x}_t)} \quad \text{莱布尼兹法则} \\
&= \frac{\int \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}) p(\mathbf{x}_t|\mathbf{x}) p_{\text{data}}(\mathbf{x}) \mathrm{d}\mathbf{x}}{p_t(\mathbf{x}_t)} \quad \text{反用复合函数求导} \\
&= \int \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}) \frac{p(\mathbf{x}_t|\mathbf{x}) p_{\text{data}}(\mathbf{x})}{p_t(\mathbf{x}_t)} \mathrm{d}\mathbf{x} \\
&= \int \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}) p(\mathbf{x}|\mathbf{x}_t) \mathrm{d}\mathbf{x} \quad \text{贝叶斯公式} \\
&= \mathbb{E}_{\mathbf{x}_t \sim \mathcal{N}(\mathbf{x}, t^2\mathbf{I}), \mathbf{x} \sim p_{\text{data}}} [\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}) | \mathbf{x}_t] \quad \text{条件期望定义} \\
&= -\mathbb{E}_{\mathbf{x}_t \sim \mathcal{N}(\mathbf{x}, t^2\mathbf{I}), \mathbf{x} \sim p_{\text{data}}} \left[\frac{\mathbf{x}_t - \mathbf{x}}{t^2} \middle| \mathbf{x}_t \right] \quad \text{代入正态分布公式}
\end{aligned} \tag{14}$$

通过推导可以发现，这个平替确实是score的无偏估计。

功法逆练：一招快速看破

功法逆练就直接从咱们熟悉的score公式下手，还记得在score matching中咱们没办法直接获得score的金标准，采用的是原始图像加噪声获得不同噪声水平score的值，也即有$p(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}, t^2\mathbf{I})$。Score matching论文证明了采用条件概率$p(\mathbf{x}_t|\mathbf{x})$替代边缘概率$p(\mathbf{x}_t)$的等价性，二者不会对训练模型有任何影响（损失函数等价）。既然如此，条件概率所对应的score也一定是原始score的一个无偏估计，直接有：

$$\begin{aligned}
\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) &= \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{x}) \quad \text{看破之无偏估计} \\
&= \nabla_{\mathbf{x}_t} \log \left[\frac{1}{\sqrt{2\pi}\sigma} \exp \left(-\frac{(\mathbf{x}_t - \mu)^2}{2\sigma^2} \right) \right] \\
&= -\frac{\mathbf{x}_t - \mathbf{x}}{t^2} \quad \text{依据} p(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}, t^2\mathbf{I}), \mu = \mathbf{x}, \sigma = t^2
\end{aligned} \tag{15}$$

通过这种方法也能获得类似论文的平替形式。然而，虽然本方法看似简单，但功法逆练实属歪门邪道，不够严谨，切勿走火入魔！

CT损失函数形式

既然无偏估计出现了，对于原始损失函数，只需要把原来依靠模型预测值$s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1})$预测的$\hat{\mathbf{x}}_{t_n}$直接写为平替形式，也即

$$
\begin{aligned}
&\hat{\mathbf{x}}_{t_n} = \mathbf{x}_{t_{n+1}} - (t_n - t_{n+1}) t_{n+1} s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\
&\Longrightarrow \\
&\hat{\mathbf{x}}_{t_n} = \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) t_{n+1} \frac{\mathbf{x}_{t_{n+1}} - \mathbf{x}}{t_{n+1}^2} = \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) \frac{\mathbf{x}_{t_{n+1}} - \mathbf{x}}{t_{n+1}}
\end{aligned} \tag{16}
$$

所以，CT的损失函数形式为：

$$
\mathcal{L}_{CT}^{N}(\theta, \theta^-) = \mathbb{E}\left[\left\| f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f_{\theta^-}\left(\mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) \frac{\mathbf{x}_{t_{n+1}} - \mathbf{x}}{t_{n+1}}, t_n\right) \right\|_2^2\right] \tag{17}
$$

论文给出了基于CT损失函数的模型训练算法步骤，可以发现相比CD来说，主要不同点为相邻时刻（也即上一时刻）的图像直接通过原始图像加噪而非ODE迭代获得（与公式(16)等价）。

Algorithm 3 Consistency Training (CT)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, step schedule $N(\cdot)$, EMA decay rate schedule $\mu(\cdot)$, $d(\cdot,\cdot)$, and $\lambda(\cdot)$

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$ and $k \leftarrow 0$

repeat

  Sample $\mathbf{x} \sim \mathcal{D}$, and $n \sim \mathcal{U}[\![1, N(k) - 1]\!]$

  Sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

  $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-) \leftarrow$

    $\lambda(t_n) d(\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x} + t_{n+1}\mathbf{z}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\mathbf{x} + t_n\mathbf{z}, t_n))$

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$

  $\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu(k)\boldsymbol{\theta}^- + (1 - \mu(k))\boldsymbol{\theta})$

  $k \leftarrow k + 1$

until convergence

再来看一下代码的实现，代码实现过程与我写的损失函数完全一致！

CT训练代码

```
def consistency_losses(
        self,
        model,
        x_start,
        num_scales,
        model_kwargs=None,
        target_model=None,
        teacher_model=None,
        teacher_diffusion=None,
        noise=None,
    ):

 # 前面代码省略
 # 不分非关键代码省略

        @th.no_grad()
        def euler_solver(samples, t, next_t, x0):
                x = samples
                if teacher_model is None:
                        denoiser = x0
                else:
                        denoiser = teacher_denoise_fn(x, t)
                d = (x - denoiser) / append_dims(t, dims) # 平替出现!
                samples = x + d * append_dims(next_t - t, dims) # 这里加号的原因是因为t增大对应方差降低

                return samples

        # t与t2是已知量，PF-ODE上相邻两点

        x_t = x_start + noise * append_dims(t, dims) # x_t加噪
        distiller = denoise_fn(x_t, t) # f_{theta}

        x_t2 = euler_solver(x_t, t, t2, x_start).detach() # 通过欧拉法获得前一个时刻的值

        distiller_target = target_denoise_fn(x_t2, t2)  # f_{theta-}
        distiller_target = distiller_target.detach()

        # 以MSE损失为例
        diffs = (distiller - distiller_target) ** 2
        loss = mean_flat(diffs) * weights

# 后面代码省略

```

损失函数等价性证明

到此，我们看似解决了所有的问题，即使没有一个预先训练好的扩散模型，也能够通过计算上面的损失函数从零开始训练了。然而，平替毕竟是平替，在损失函数上，CD和CT是否同样具备一定的等价性，也即采用CT训练出的模型是否能达到和CD一样的效果？是否能够有足够的把握相信从零开始训练的CT模型呢？这就要牵扯到CT和CD损失函数的等价性分析问题。所以，**问题的出发点就是从$\mathcal{L}_{CD}^N$出发，推导获得有关$\mathcal{L}_{CT}^N$的表达式**，如果二者只差常数，或者差一个关于时间差分的无穷小量，则在某种情况下二者梯度能够处处相等，进而满足损失函数等价性。

宋博士在这里又展示了大佬级别的详细推导过程，全程只用了**泰勒展开**（正用和逆用）和**条件期望**的性质，过程上不存在跳步。在这里，仅对开头几步泰勒展开难点进行细化，只要前面几步理解了，后面的推导就很简单了。为了表达方便，并

且更靠近原始论文推导过程，这里使用距离度量d代替前面的均方误差损失。此外，假设CD中预训练的score-based模型已经完美匹配ground truth，此时有$s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), = \nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}})$。

$$\begin{aligned}\mathcal{L}_{CD}^N(\theta,\theta^-) &= \mathbb{E}[d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}} - (t_n - t_{n+1})t_{n+1}s_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), t_n))] \\ &= \mathbb{E}[d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}} + (t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}), t_n))]\end{aligned} \tag{18}$$

到这里，我们先暂停一下，再来回顾一下二元函数泰勒展开的形式，设二元函数$f(x,y)$，在(x_0,y_0)处展开，有：

$$f(x,y) = f(x_0,y_0) + \partial_1 f(x_0,y_0)(x-x_0) + \partial_2 f(x_0,y_0)(y-y_0) + o(\cdots) \tag{19}$$

其中，∂_1和∂_2分别表示对x和y求偏导数。对于泰勒展开，最好的方式就是写出x、y、x_0和y_0分别是什么，然后套上面公式就可以获得结果。对于上式的结果，作者首先对d当中的$f_{\theta^-}(\mathbf{x}_{t_{n+1}} + (t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}), t_n)$进行了泰勒展开，展开点为$(\mathbf{x}_{t_{n+1}}, t_{n+1})$，那么现在，对照上面的泰勒展开公式，可以轻松写出$x$、$y$、$x_0$和$y_0$的值如下，我也称之为二元泰勒展开的四个关键量：

$$\begin{cases} x = \mathbf{x}_{t_{n+1}} + (t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}) \\ y = t_n \\ x_0 = \mathbf{x}_{t_{n+1}} \\ y_0 = t_{n+1} \end{cases} \tag{20}$$

代入泰勒展开公式，可得:

$$\begin{aligned} f_{\theta^-}(\mathbf{x}_{t_{n+1}} + (t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}), t_n) &= f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\ &+\partial_1 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}) + \partial_2 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_n - t_{n+1}) \\ &+o(|t_{n+1} - t_n|) \end{aligned} \tag{21}$$

再带回到原式(18)中，可得：

$$\begin{aligned}\mathcal{L}_{CD}^N(\theta,\theta^-) &= \mathbb{E}[d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}} + (t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}), t_n))] \\ &= \mathbb{E}[d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\ &\quad +\partial_1 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}) \\ &\quad +\partial_2 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_n - t_{n+1}) + o(|t_{n+1} - t_n|))] \end{aligned} \tag{22}$$

到这一步后，作者又对距离度量函数d在$(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}))$处进行泰勒展开。这里需要明确，损失函数本身确实是一个函数，咱们还是按照老套路，分析二元函数泰勒展开的关键量x、y、x_0和y_0的值如下：

$$\begin{cases} x = f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\ y = f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}) + \partial_1 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}) \\ \qquad +\partial_2 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_n - t_{n+1}) + o(|t_{n+1} - t_n|) \\ x_0 = f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\ y_0 = f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}) \end{cases} \tag{23}$$

同理可得，期望中的d可以写为：

$$\begin{aligned} & d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})) \\ &\quad +\partial_1 d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}))\underbrace{(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}))}_{\text{这一项为0}} \\ &\quad +\partial_2 d(f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}), f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1}))[\partial_1 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_{n+1} - t_n)t_{n+1}\nabla \log p_{t_{n+1}}(\mathbf{x}_{t_{n+1}}) \\ &\quad +\partial_2 f_{\theta^-}(\mathbf{x}_{t_{n+1}}, t_{n+1})(t_n - t_{n+1}) + o(|t_{n+1} - t_n|)] \end{aligned} \tag{24}$$

到这里，就来到了论文定理2推导的第三个等号。后面的推导过程类似，需要注意有一个地方反用了泰勒展开形式，也即将展开式变为了原始形式。然而换汤不换药，只要大家写出二元函数泰勒展开四个关键量，问题便可迎刃而解。在这里，直接给出损失函数等价性推导证明的结论：

$$\mathcal{L}_{CD}^N(\theta,\theta^-) = \mathcal{L}_{CT}^N(\theta,\theta^-) + o(\Delta t) \tag{25}$$

其中，$\Delta t := \max_{n\in[1,N-1]}\{|t_{n+1}-t_n|\}$，也即任意两个相邻时间间隔的最大值。该结论可以反映CD和CT的损失函数在Δt接近0的时候可以趋近于相同，进而说明了二者损失函数存在等价性。

问题3：如何通过一致性模型采样获得图像?

根据一致性模型的特点，一致性模型支持概率流上任意一点的一步采样，也支持类似扩散模型的多步采样。

一步采样

一步采样形式简单，也即给定一个$\mathbf{x}_T\sim\mathcal{N}(\mathbf{0},T^2\mathbf{I})$，带入到CD或CT训练获得的$f_\theta(\mathbf{x}_t,t)$中，便可直接获得近似满足数据分布的图像$\hat{\mathbf{x}}_\varepsilon\sim\mathcal{N}(\mathbf{x},\varepsilon^2\mathbf{I})$，用公式描述就是：

$$\hat{\mathbf{x}}_\varepsilon = f_\theta(\mathbf{x}_T,T) \tag{26}$$

这里需要注意一个小问题，就是给定$\mathbf{x}_T$的分布并不是$\mathcal{N}(\mathbf{x},T^2\mathbf{I})$，而是$\mathcal{N}(\mathbf{0},T^2\mathbf{I})$，这是因为在采样过程中并不知道真实图像是什么。当$T$足够大时，方差能够大到淹没均值$\mathbf{x}$，可以近似认为$\mathcal{N}(\mathbf{x},T^2\mathbf{I})$就是一个容易获得的先验分布$\mathcal{N}(\mathbf{0},T^2\mathbf{I})$。

多步采样

一致性模型也可以多步采样，论文给出了多步采样的方法（如下图所示）：

Algorithm 1 Multistep Consistency Sampling

Input: Consistency model $\boldsymbol{f_\theta}(\cdot,\cdot)$, sequence of time points $\tau_1>\tau_2>\cdots>\tau_{N-1}$, initial noise $\hat{\mathbf{x}}_T$

$\mathbf{x}\leftarrow \boldsymbol{f_\theta}(\hat{\mathbf{x}}_T,T)$

for $n=1$ **to** $N-1$ **do**

  Sample $\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})$

  $\hat{\mathbf{x}}_{\tau_n}\leftarrow \mathbf{x}+\sqrt{\tau_n^2-\epsilon^2}\,\mathbf{z}$

  $\mathbf{x}\leftarrow \boldsymbol{f_\theta}(\hat{\mathbf{x}}_{\tau_n},\tau_n)$

end for

Output: $\mathbf{x}$

多步采样用一句话总结就是：**一个老司机在一条确定路径上反复瞬移**。具体而言，就是先对在概率流起点的先验分布样本$\mathbf{x}_T$使用一步一致性模型$f_\theta(\mathbf{x}_T,T)$获得一个$\hat{\mathbf{x}}_\varepsilon$，再把$\hat{\mathbf{x}}_\varepsilon$重新加一个更小的噪声变换回概率流$\mathbf{x}_T$前面某个时刻的位置$\mathbf{x}_{\tau_1}$，然后再使用一致性模型获得$\hat{\mathbf{x}}_\varepsilon$，如此反复数轮便可用更多的时间，获得质量更高的生成图像。
这里有几个问题大家可以一起思考：

1. 为什么噪声先大后小？能不能先小后大？或者每次都一样或者混乱排布？

> 个人理解：首先噪声先大是肯定的，因为先验分布$\mathcal{N}(\mathbf{0},T^2\mathbf{I})$对应的就是噪声最大的情形，排除先小后大的可能。此外，噪声从大到小也是扩散逆向过程重要特征，噪声都是从大到小逐步过渡。噪声如果每次一样，相当于又把图像从概率流终点变换到起点，多次本质上没有区别。如果噪声混乱排布没有规律，同样无法满足扩散模型噪声逐步下降的性质，容易走回头路，对采样没有好处。

2. 多步采样到底是如何提升模型生成效果的?

个人理解：注意f_θ的形式，当噪声越大时，f_θ的输出结果越依赖于神经网络F_θ的预测结果。通过不断加更小的噪声，相当于把生成图像不断退回PF-ODE路径上更靠近采样终点$\mathbf{x}_\varepsilon$的点，此时f_θ的输出结果更少依赖神经网络F_θ的预测值，更多的相信输入的噪声图像$\mathbf{x}_t$。明确上述条件，我这里给出一个定性分析：

第一步：输入为$\mathbf{x}_T$，C_{skip}最小，C_{out}最大

$$\hat{\mathbf{x}}_\varepsilon = f_\theta(\mathbf{x}_T, T) = C_{\text{skip}}(T)\mathbf{x}_T + C_{\text{out}}(T)F_\theta(\mathbf{x}_T, T)$$

第二步：把第一步获得的$\hat{\mathbf{x}}_\varepsilon$重新加噪

$$\mathbf{x}_{\tau_1} = \hat{\mathbf{x}}_\varepsilon + \sqrt{\tau_1^2 - \varepsilon^2}\mathbf{z} = f_\theta(\mathbf{x}_T, T) + \sqrt{\tau_1^2 - \varepsilon^2}\mathbf{z}$$

第三步：把第二步获得的$\mathbf{x}_{\tau_1}$输入到一致性模型f_θ中，C_{skip}的值增加，C_{out}的值下降

$$\hat{\mathbf{x}}_\varepsilon = f_\theta(\mathbf{x}_{\tau_1}, \tau_1) = C_{\text{skip}}(\tau_1)\mathbf{x}_{\tau_1} + C_{\text{out}}(\tau_1)F_\theta(\mathbf{x}_{\tau_1}, \tau_1)$$

$$= \underbrace{C_{\text{skip}}(\tau_1)}_{\text{信息保留占比增加}} (\underbrace{f_\theta(\mathbf{x}_T, T)}_{\text{信息保留}} + \sqrt{\tau_1^2 - \varepsilon^2}\mathbf{z}) + \underbrace{C_{\text{out}}(\tau_1)}_{\text{模型修正占比下降}} \underbrace{F_\theta(\mathbf{x}_{\tau_1}, \tau_1)}_{\text{模型修正}}$$

同理可得，当τ_n逐步下降，前面所有步骤信息保留占比增加，模型修正的比例下降，不稳定因素F_θ占比降低，输出结果确定性增加。不断反复采用一致性模型能够综合不同时刻f_θ的预测结果，相比单步采样，有类似将系统误差求均值的效果，有利于提升模型性能！