扩散生成模型的理论演进与前沿探索

戴煊朗

Talk Roadmap

1. 生成图像背后到底在做什么
2. 生成领域还在被探索的领域

会讲什么：
1. 生成模型基础和可做方向的总览
2. 在一些topic上的一些小想法

不会讲什么：
1. 具体方法和建模的细节数学推导（补充的推导放在这个blog里了 Diffusion Model 简单介绍 | Xuanlang Dai）
2. 涉及paper的大量实验分析

Text to Image

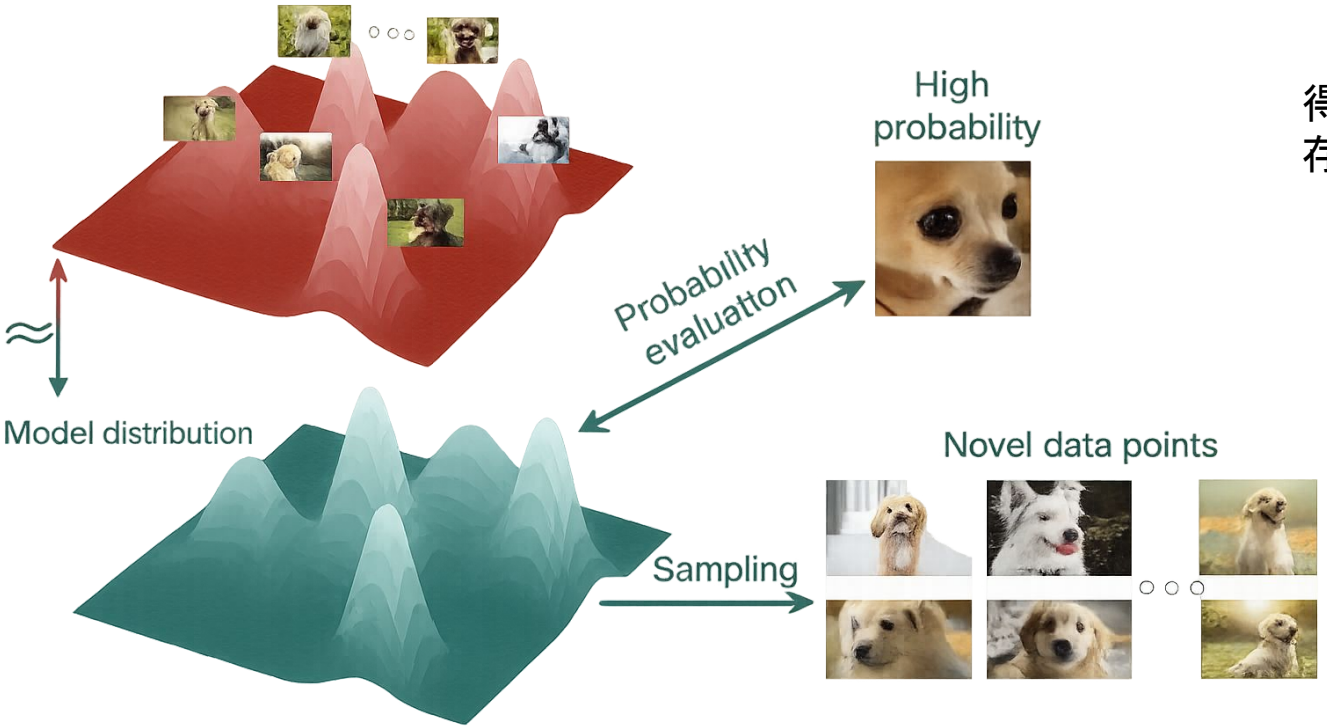
Image Editing

Prompt: Retro 80s Monster Horror Comedy Movie Scene: Color film, children's bedroom bathed in soft, warm light. Plush monsters of various sizes and colors are having a chaotic party, jumping on the bed, dancing to upbeat music, and throwing confetti. The walls are adorned with posters of classic 80s movies, and the room is filled with the playful laughter of children.

Prompt: A sepia-toned vintage photograph depicting a whimsical bicycle race featuring several dogs wearing goggles and tiny cycling outfits. The canine racers, with determined expressions and blurred motion, pedal miniature bicycles on a dusty road. Spectators in period clothing line the sides, adding to the nostalgic atmosphere. Slightly grainy and blurred, mimicking old photos, with soft side lighting enhancing the warm tones and rustic charm of the scene. 'Bicycle Race' captures this unique moment in a medium shot, focusing on both the racers and the lively crowd.

Text to Vedio

生成模型需要做什么



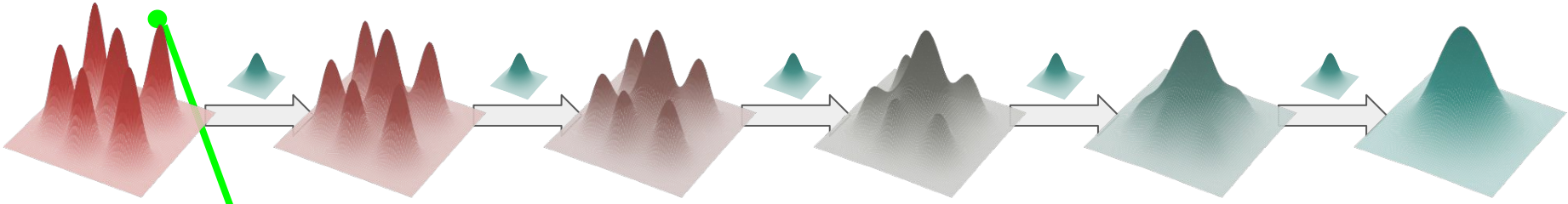
得到一个尽可能接近于世界上存在的所有真实图像的分布

怎么做？

1. 预测过程
2. 预测梯度
3. 预测变化

Fig 来源 Diffusion and Score-Based Generative Models

预测过程——DDPM & DDIM



真实图像

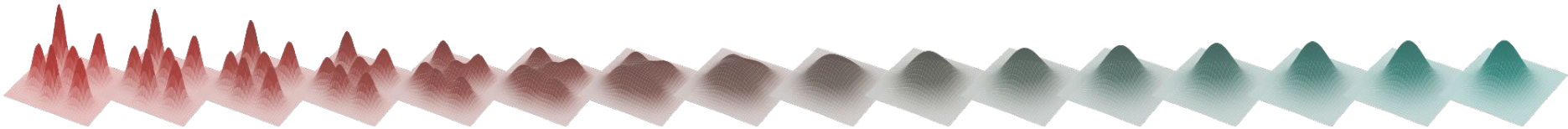
纯噪声

出发点：如果直接预测真实图像分布太困难，那不如预测”差值”，也就是怎么让输入图像稍微好一点

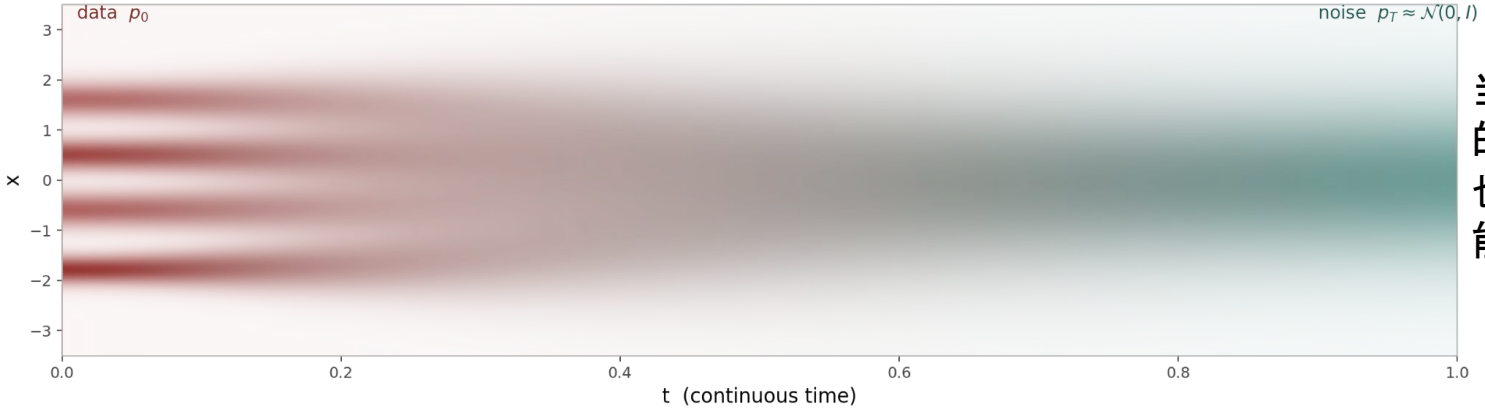
预测过程 -> 预测噪声

$$Loss = \|\mathbf{x}_{t-1} - \boldsymbol{\mu}(\mathbf{x}_t)\|^2 = \frac{\beta_t^2}{\alpha_t^2}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2$$

预测梯度——Score-Based Modeling



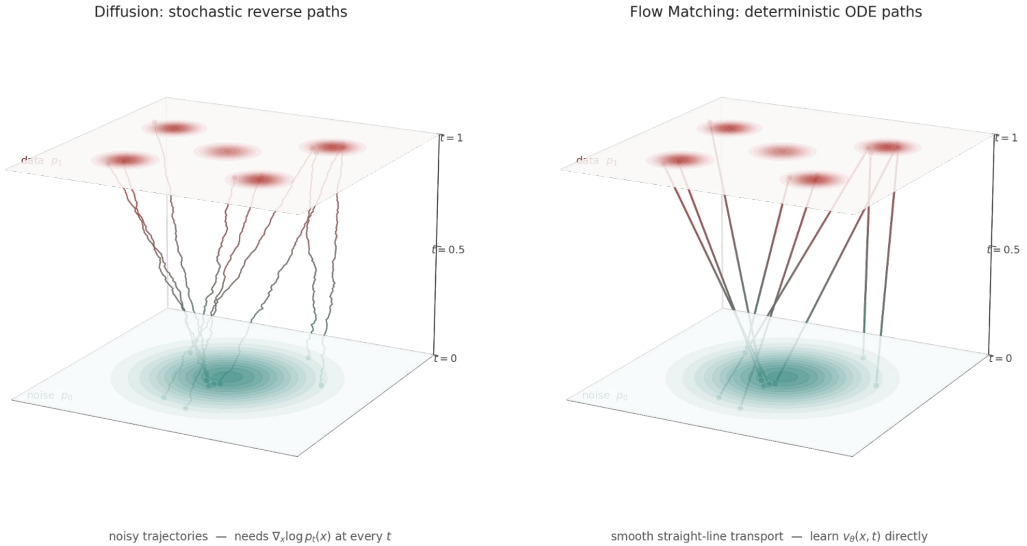
出发点：越连续的信号，描述的越准确



当这个切分越来越细的时候，预测的噪声也趋向于0，所以只能转而预测“梯度”

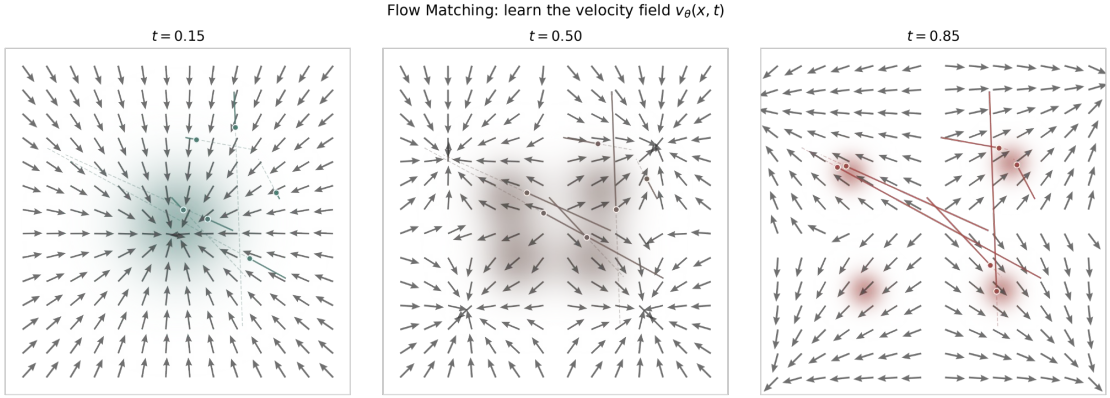
$$Loss = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t \sim p(\mathbf{x}_t|\mathbf{x}_0)\tilde{p}(\mathbf{x}_0)} \left[\left\| \mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) \right\|^2 \right]$$

预测变化——Flow Model



出发点：模拟去噪过程是否是最优的？

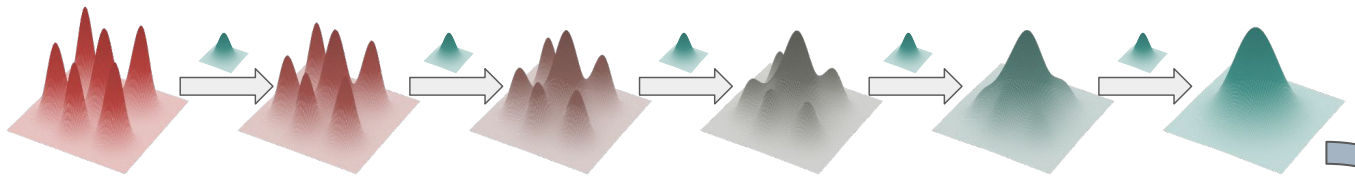
数据在纯噪声的位置
和
数据在最终分布的位置
连个线，告诉它应该这样走

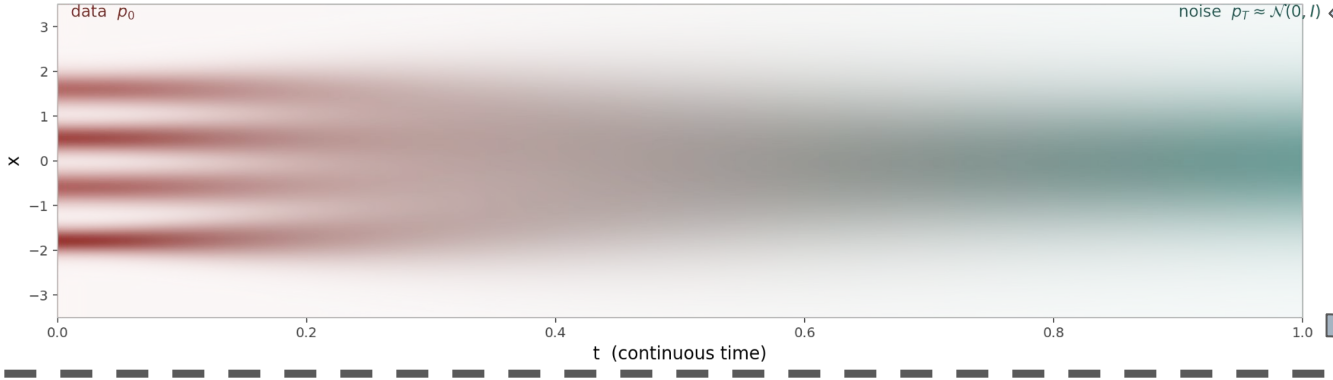


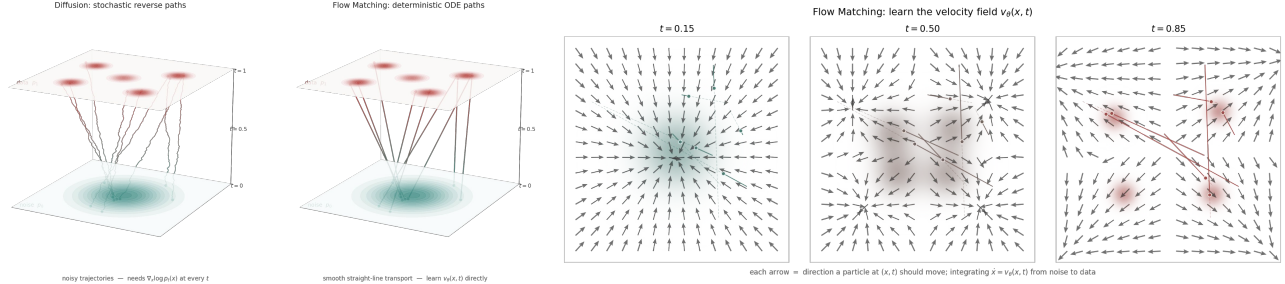
模型预测的目标：
“去噪”到目前这个时候，这个数据点应该往哪里运动（预测速度）

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1} \| v_\theta(x_t, t) - (x_1 - x_0) \|^2$$

怎么做生成模型







怎么做？

1. 预测过程->预测噪声
2. 预测梯度
3. 预测变化->预测速度

Prerequisite: Flow Matching

目标：从噪声到数据

逐步加噪的过程是否真的必须？✗ 我们关心的只是它能做的事：把 p_0（噪声）传输成 p_1（数据）。

不妨假设存在这样一个变量是时间t，图片latent x的流 ϕ_t 它能把随机噪声点送到目标数据点

可以理解成前两种方法，所有加上的噪声的总和

它满足：

$$\frac{d\phi_t(x)}{dt} = v_\theta(\phi_t(x), t), \quad \phi_0(x) = x$$

在我们实际使用中，基本都是从这个大流形中采样一个点，所以完全没必要预测出整个 ϕ_t

它的梯度的几何意义是：*在时间t的这个位置的数据点该往哪里走*

训练两者完全等价，所以就不妨让模型预测一个v，去拟合真实的u

Prerequisite: Flow Matching

目标：从噪声到数据

可以写出损失函数：

$$\mathcal{L} = \mathbb{E}_{t,p_t(x)} \left| v_\theta(x,t) - u_t(x) \right|^2$$

不过我们还缺少每个时刻的u_t(x)，要精确计算的话依赖整个数据分布 q，这个cost是无法接受的。

但是有个数值上的小技巧：

我向一个正确的方向 优化一步，等效于我向正确方向的所有分量 优化一步

Prerequisite: Flow Matching

目标：从噪声到数据

具体来说，不去想整体的概率路径，而是只考虑"从噪声走向某个特定数据点 x"的条件路径：

$$p_t(x|x_1) = \mathcal{N}(x; \mu_t(x_1), \sigma_t(x_1)^2 I)$$

这里的 $\mu_0(x_1) = 0; \mu_1(x_1) = x_1$ $\sigma_0(x_1) = 1; \sigma_1(x_1) = 0$

数据集有多少张图，就有多少条条件路径，叠加起来就是等效的正确优化方向。

可以得到粒子的位置随时间的演化：

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

这里出现的新变量： $x_0 \sim \mathcal{N}(0, I)$

Prerequisite: Flow Matching

目标：从噪声到数据

那么速度就是求个导：

$$u_t(x|x_1) = \frac{\sigma_t'(x_1)}{\sigma_t(x_1)}(x - \mu_t(x_1)) + \mu_t'(x_1)$$

这下真实的u就知道了，我们最终的CFM loss也可以写出来了：

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],x_1\sim q(x_1),x_0\sim p(x_0)}\left\|v_\theta(x_t,t) - u_t(x_t|x_1)\right\|^2$$

Prerequisite: Flow Matching

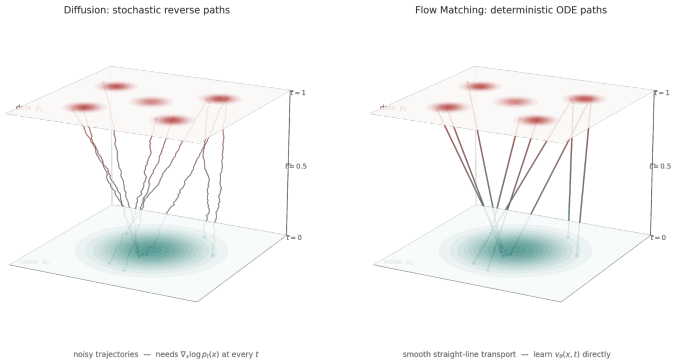
目标：从噪声到数据 $u_t(x|x_1) = \frac{\sigma_t'(x_1)}{\sigma_t(x_1)}(x - \mu_t(x_1)) + \mu_t'(x_1)$

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],x_1\sim q(x_1),x_0\sim p(x_0)} \left\| v_\theta(x_t, t) - u_t(x_t|x_1) \right\|^2$$

那之前说的直线是咋来的呢？先带入俩特殊值

$$\mu_t(x_1) = \bar{\alpha}_{1-t}x_1 \qquad \sigma_t = \bar{\beta}_{1-t}$$

推导出来的结果和 Score-Based 概率流 ODE 完全一致
（这里没推score-based model，就不具体放式子了）



$$\mu_t(x_1) = t \cdot x_1, \quad \sigma_t = 1 - t \quad \Longrightarrow \quad \mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1} \left\| v_\theta(x_t, t) - (x_1 - x_0) \right\|^2$$

几何上也很好理解，这里都是线性变化，流 ϕ_t 自然就变成了仿射插值

生成模型的具体建模

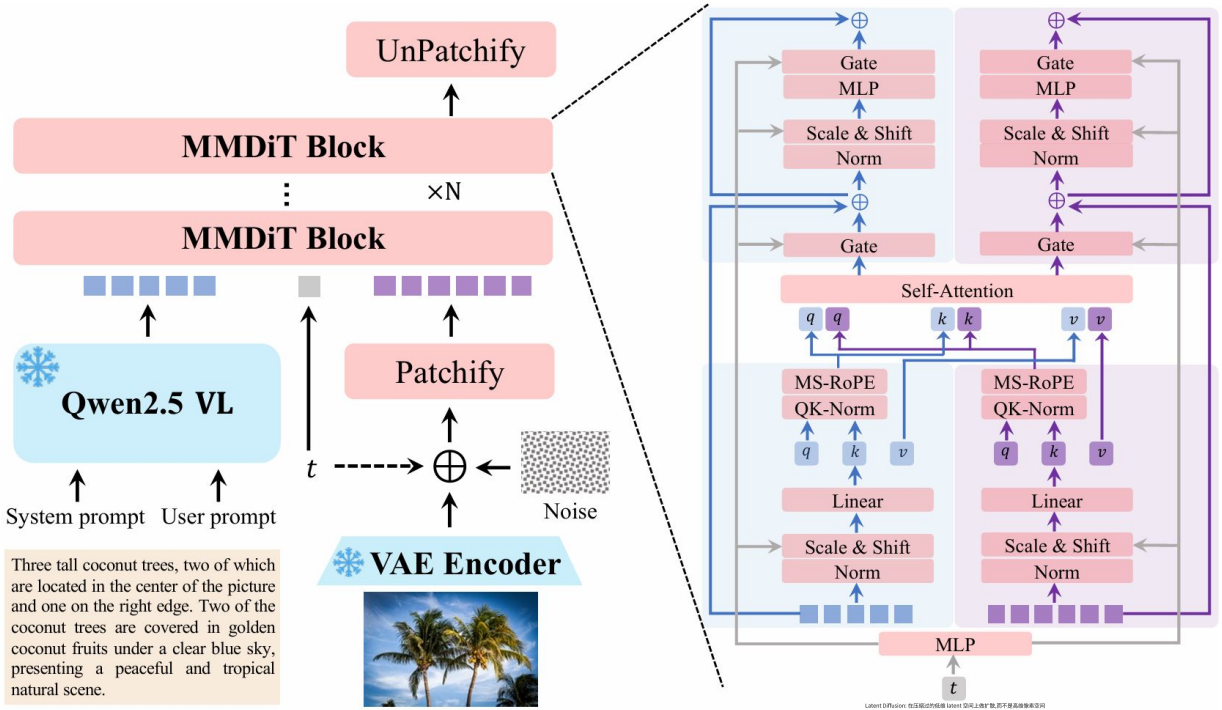
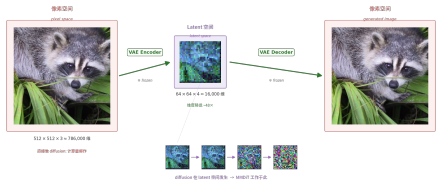
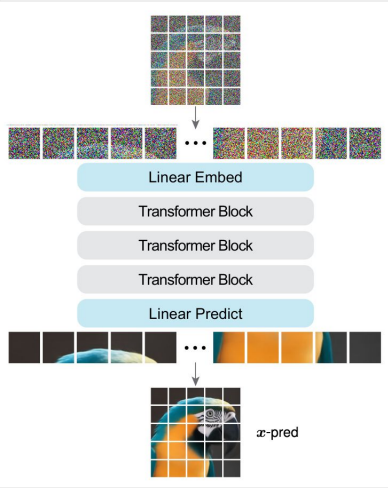


Fig 来源 Fig 来源 [2508.02324] Qwen-Image Technical Report



用什么来预测前面提到的噪声/梯度/速度？

1. Text encoder
2. VAE Encoder
3. MMDiT

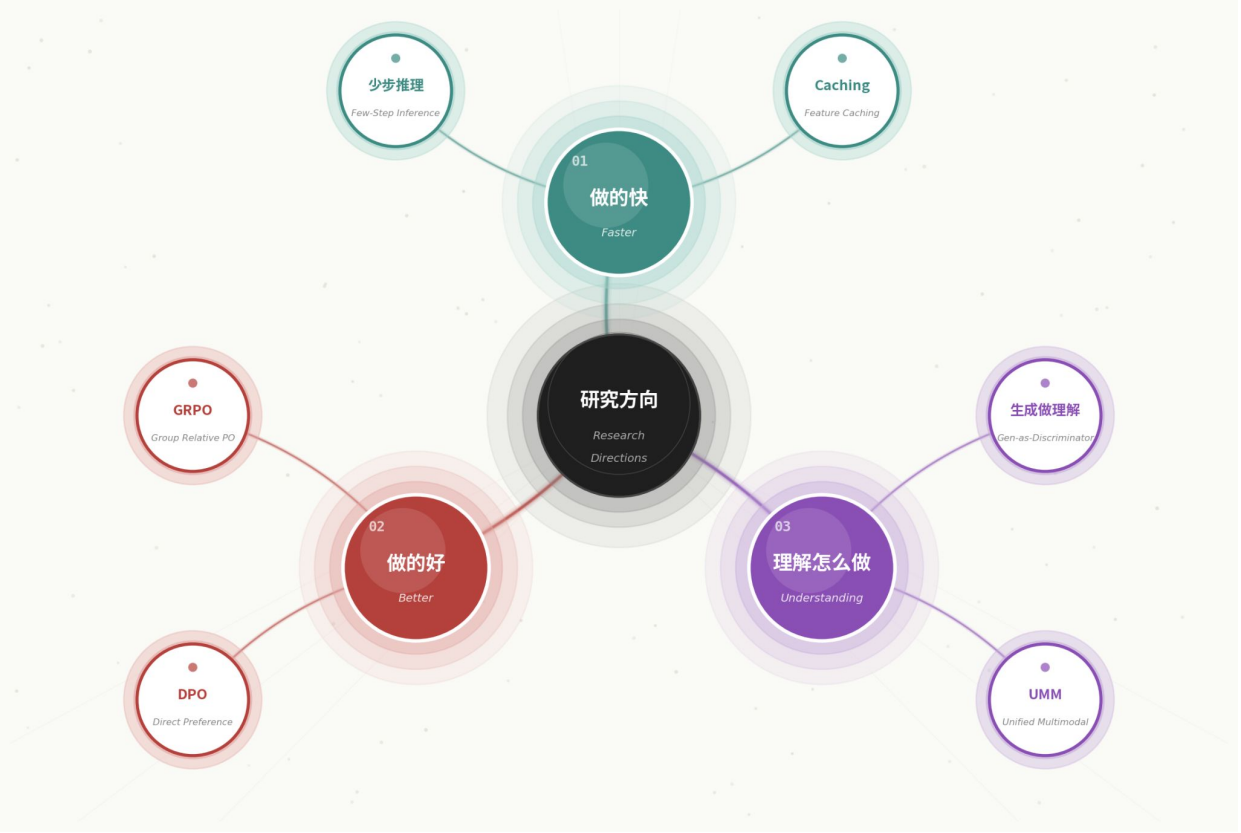


当然也有纯在pixel space做的生成工作。

[2511.13720] Back to Basics: Let Denoising Generative Models Denoise

生成模型关注什么方向

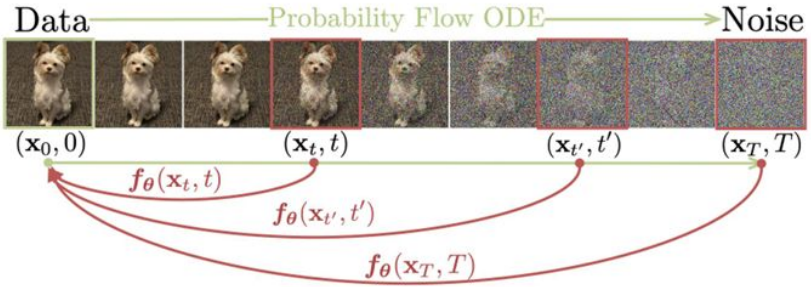
1. 提升推理效率
   - 少步推理
   - caching
2. 优化生成质量
   - DPO
   - GRPO（强化学习）
   - OPD
3. 探索统一范式
   - 用生成模型做理解任务
   - UMM
4. 语言模型的连续建模



少步推理

假设你用扩散模型生成一张图，要等 30 秒，这对产品来说并不是个好消息。
最直接的想法是：能不能**一步**或者**少步**就出图？
所以一整条研究方向就在解决 **怎么让它快起来**

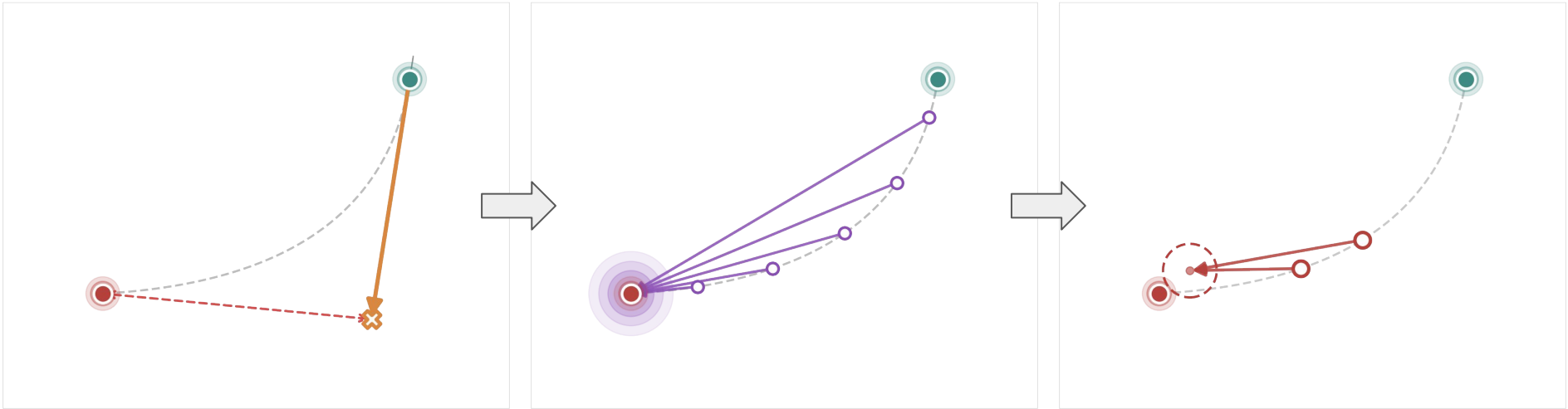
1. Consistency Models (算法上的改进)
2. MeanFlow (训练目标的更换)



少步推理

Consistency Models

从绿点出发前往红点



$$x_0 = x_T - T \cdot v_\theta(x_T, T)$$

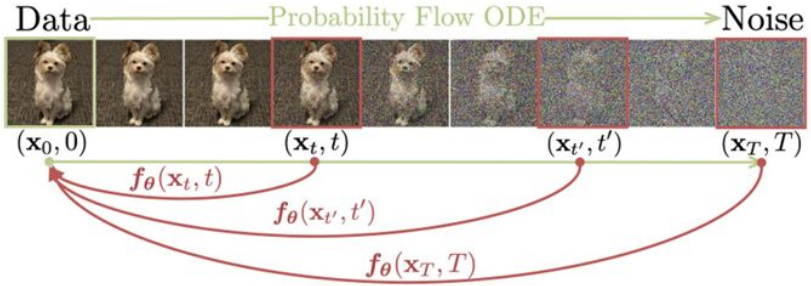
$$\mathcal{L}_{CD} = \mathbb{E}_{t,x_0,x_1} \| f_\theta(x_t, t) - x_0 \|^2$$

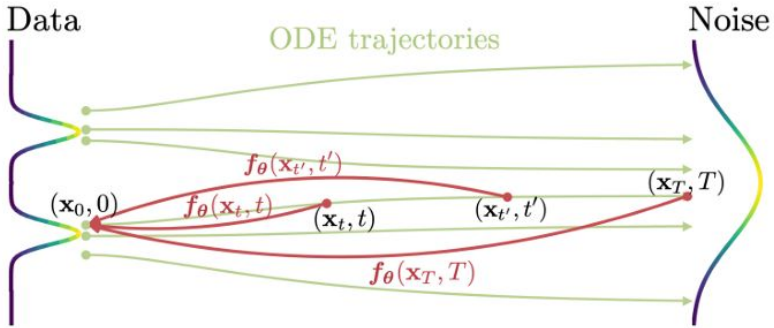
$$\mathcal{L}_{CD} = \mathbb{E} \left\| \underbrace{x_{t_{k+1}} - t_{k+1} \cdot v_\theta(x_{t_{k+1}}, t_{k+1})}_{\text{从 } t_{k+1} \text{ 预测的 } x_0} - \underbrace{(\hat{x}_{t_k} - t_k \cdot v_{\theta^-}(x_{t_k}, t_k))}_{\text{走一步后，从 } t_k \text{ 预测的 } x_0} \right\|$$

出发点：尽可能让模型有全局的视野，让任意时间都知道怎么往终点走

少步推理

Consistency Models



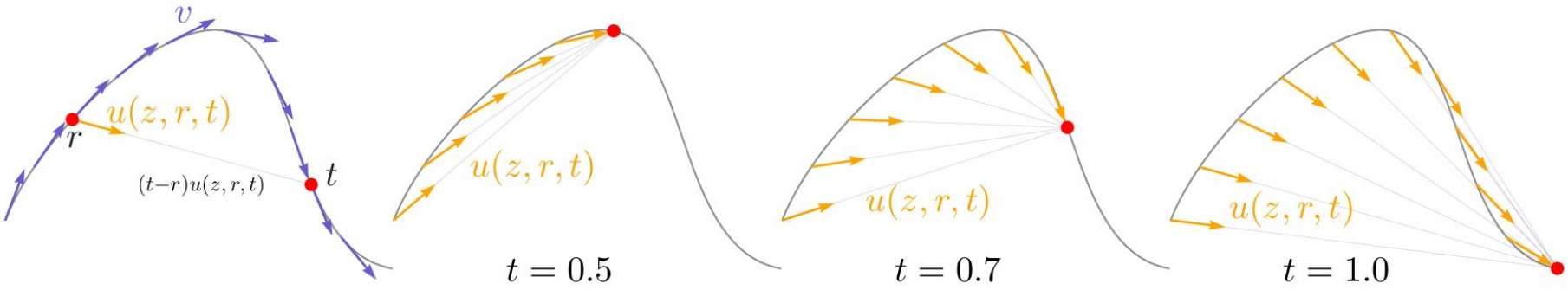


出发点：尽可能让模型有全局的视野，让任意时间都知道怎么往终点走

少步推理

MeanFlow

生成步数少 ⇨ $(t_{k-1} - t_k)$ 的值大　　瞬时速度 ⇨ 平均速度



出发点：与其学瞬时速度再多步积分，不如让网络直接预测任意区间 [r,t]上的平均速度

$$\boldsymbol{u}_\theta(\boldsymbol{x}_t, t) \quad \Rightarrow \quad \boldsymbol{u}_\theta(\boldsymbol{x}_t, r, t)$$

少步推理

MeanFlow

我们原本是需要解一个如下ODE:

$$\frac{dx_t}{dt} = v_\theta(x_t, t)$$

把瞬时速度变成平均速度，其实只需要在两边都做下积分：

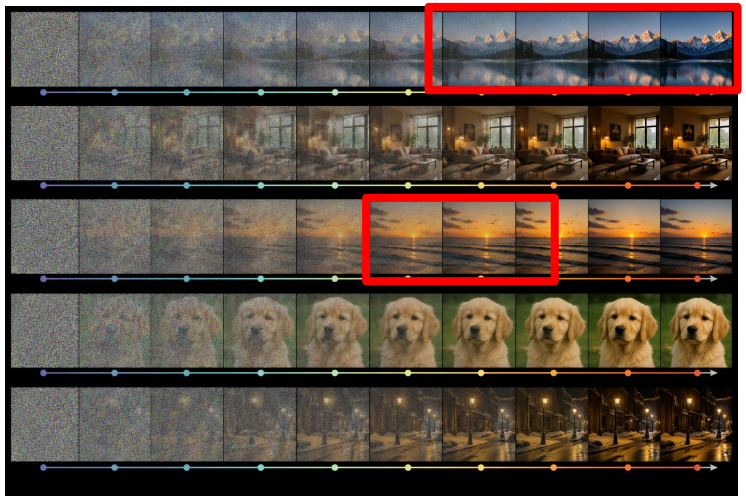
$$x_t - x_r = \int_r^t v_\theta(x_\tau, \tau)d\tau = (t-r) \times \frac{1}{t-r}\int_r^t v_\theta(x_\tau, \tau)d\tau$$

再化简下，左边就变成t到r的平均速度了：

$$u_\theta(x_t, r, t) = \frac{1}{t-r}\int_r^t v_\theta(x_\tau, \tau)d\tau$$

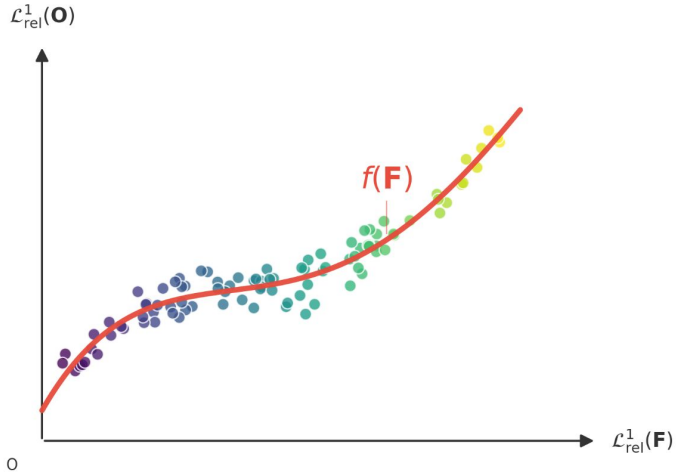
Caching

Teacache



有些timesteps的实际变动很小，那就跳过这些不太影响最后结果的步骤

怎么判断哪些去噪步数是 值得被跳过的呢

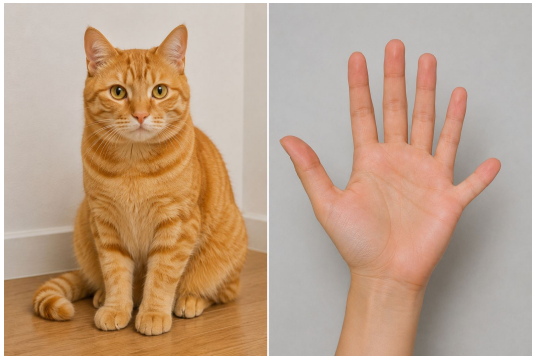


出发点：复用旧特征

$$\mathcal{L}^1_{\text{rel}}(O,t) = \frac{\|O_t - O_{t+1}\|}{\|O_{t+1}\|} \Rightarrow \sum_{t=t_a}^{t_b-1} \text{L1}_{\text{rel}}(\mathbf{O},t) \le \delta < \sum_{t=t_a}^{t_b} \text{L1}_{\text{rel}}(\mathbf{O},t) \Rightarrow \sum_{t=t_a}^{t_b-1} f(\text{L1}_{\text{rel}}(\mathbf{F},t)) \le \delta < \sum_{t=t_a}^{t_b} f(\text{L1}_{\text{rel}}(\mathbf{F},t))$$

优化生成质量

如何通过后训练让一个本来就能生成图片的模型生成的更好



怎么告诉模型'什么是好、什么是不好'?

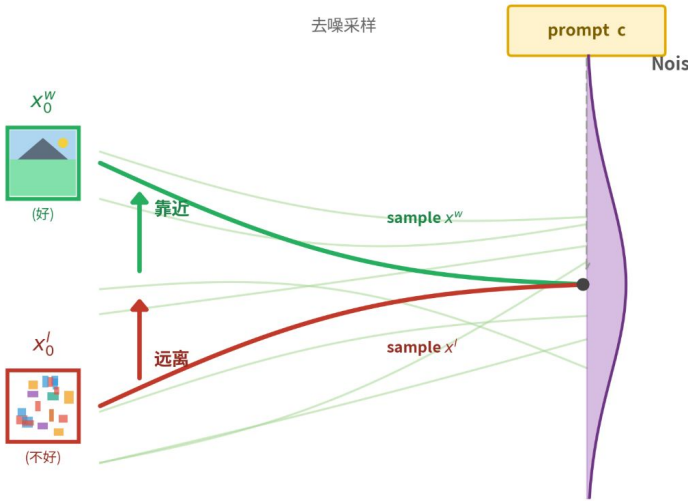
1. **人类标注**:让人来评判'A 图比 B 图好(DPO)
2. **打分模型**:训练一个 AI 当裁判,给每张图打分(GRPO)
3. **更强的老师模型**:让一个更厉害的模型来教这个学生(OPD)

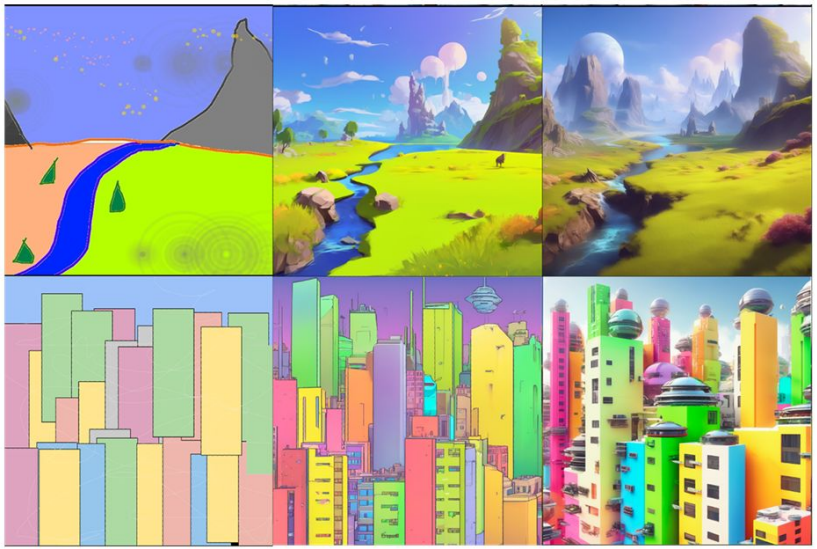
DPO

如何通过后训练让一个本来就能生成图片的模型生成的更好

DiffusionDPO

出发点:让生成的内容靠近好的数据,远离不好的数据



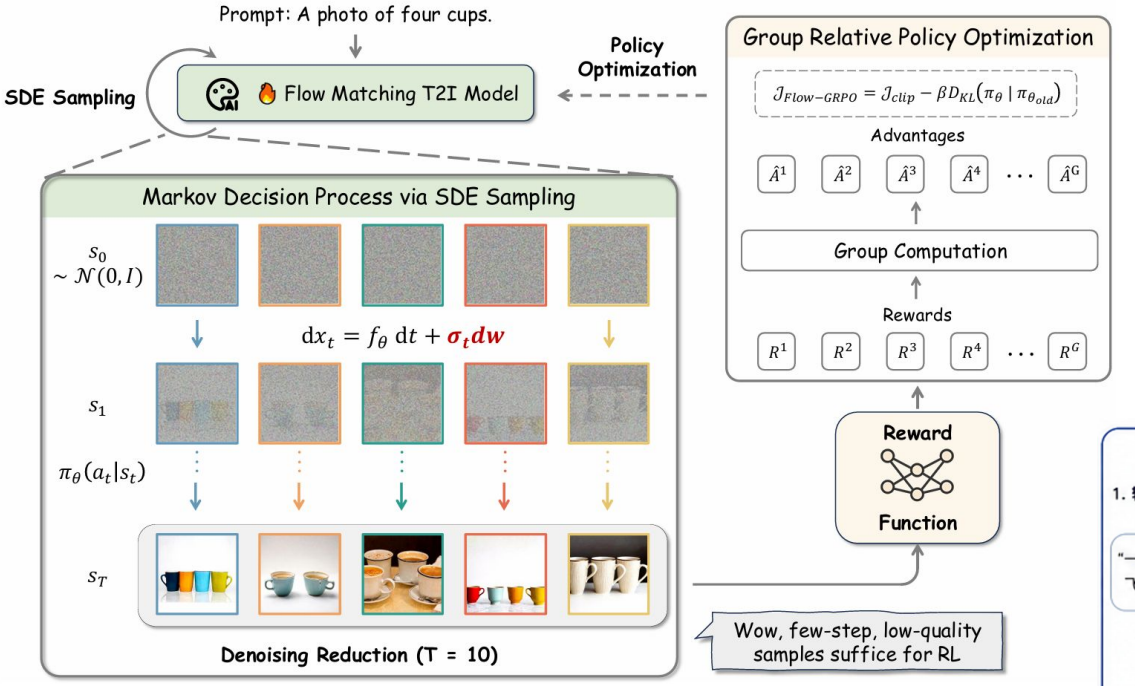


$$\mathcal{L}(\theta) = -\mathbb{E}_{x_0^w, x_0^l, t, \epsilon^w, \epsilon^l}\left[\ln \sigma\left(-\beta\omega_t\left(\|\epsilon^w - \epsilon_\theta(x_t^w, t)\|^2 - \|\epsilon^w - \epsilon_{\text{ref}}(x_t^w, t)\|^2 - \|\epsilon^l - \epsilon_\theta(x_t^l, t)\|^2 + \|\epsilon^l - \epsilon_{\text{ref}}(x_t^l, t)\|^2\right)\right)\right]$$

GRPO（强化学习）

FlowGRPO

这类方法虽然在训练的cost上大于DPO，但是他的好处是对训练数据没有那么严格的要求，只需要一句能生成图片的prompt就行了。



问题：如何根据同一个prompt，生成一组各不相同的图片 ？

$$dx = v_\theta(x,t)dt \Rightarrow dx = f(x_t,t)dt + \sigma_t dw$$

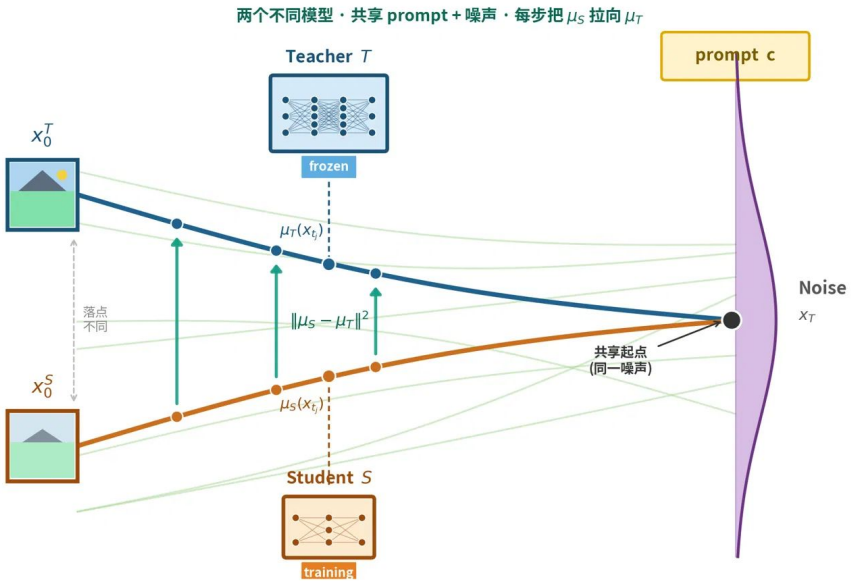
s.t.：ode和sde任意时刻粒子密度都一致

生成->评分->更新

OPD

DiffusionDPO

出发点：如何把一个更好模型的知识内化给更弱的模型



$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{x_{0:N}\sim p_S}\left[\sum_{j=0}^{N-1} \mathrm{KL}\big(p_S(\cdot|x_{t_j}) \,\|\, p_T(\cdot|x_{t_j})\big)\right]$$

$$\mathcal{L}_{\mathrm{OPD}}^{\text{diffusion-ODE}}(\theta) = \mathbb{E}_{x_{0:N}\sim p_{S,\theta}}\left[\sum_{j=0}^{N-1} \frac{1}{2}\|\mu_S(x_{t_j};\theta) - \mu_T(x_{t_j})\|_2^2\right]$$

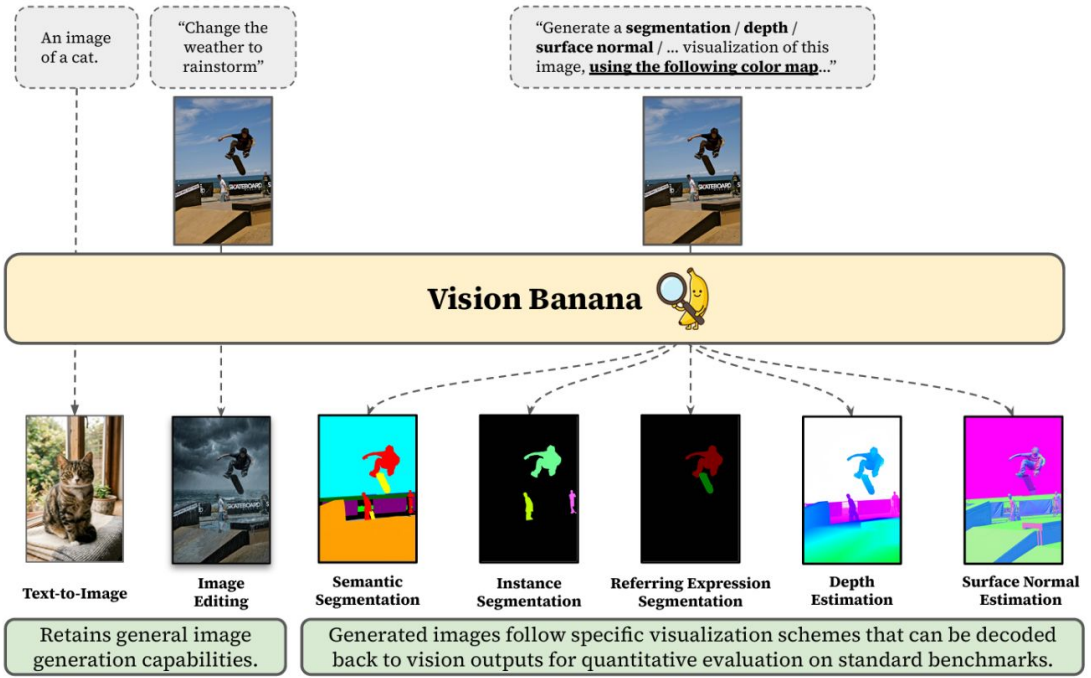
TIPS: 如果和DPO对比的话，可以这样理解：

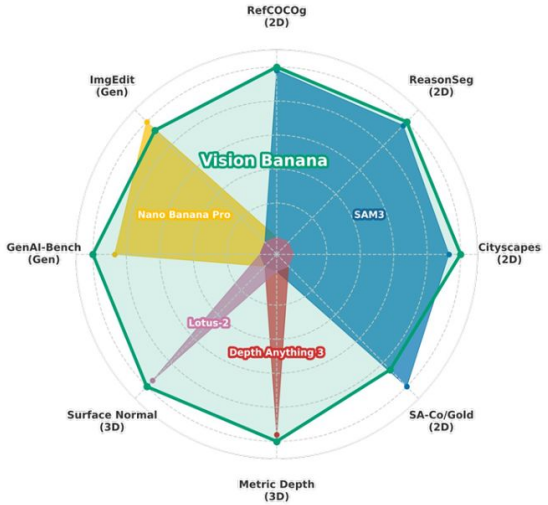
1. DPO 是"一好一坏从两个点出发往反方向拉"
2. OPD 是"两条路径共享起点，但落点不同"。

用生成模型做理解任务

VisionBanana

出发点：通过数据层面的增强，能不能让生成模型做更多的理解任务





Vision Banana is a generalist vision model in both visual generation and understanding, surpassing or rivaling specialist models.

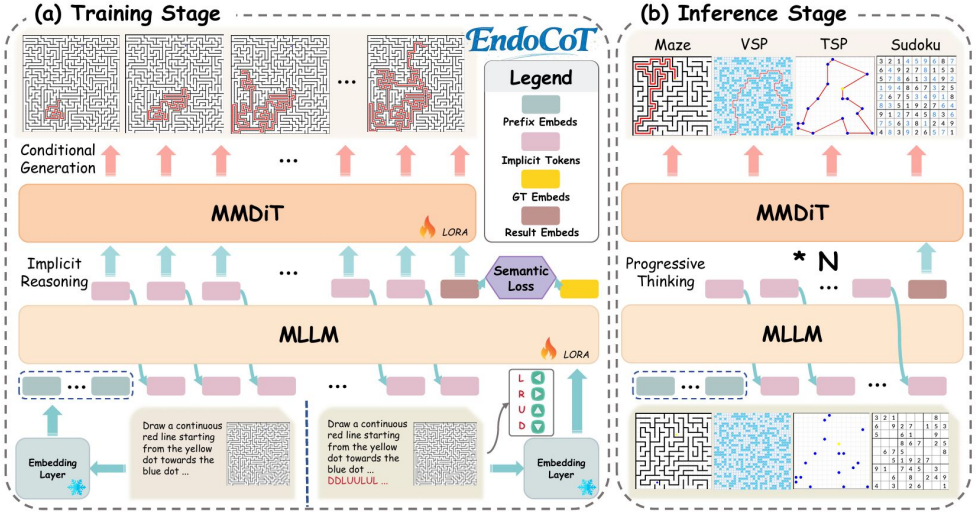
用生成模型做理解任务

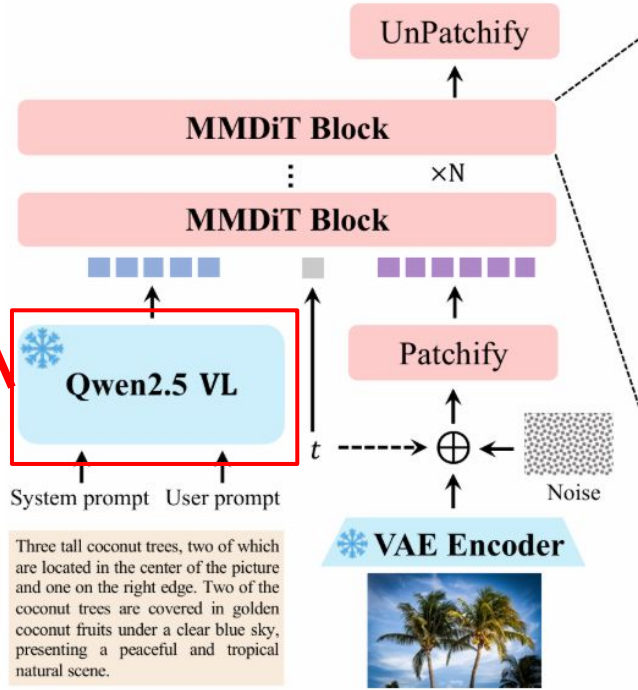
EndoCoT

出发点：如何更好的利用用语言模型的reasoning能力

参考目前生成模型的结构图，普遍使用了vl model作为text encoder

很朴素的想法是，既然语言模型能做cot，能不能用ar的条件来让生成模型也逐步生成图片。

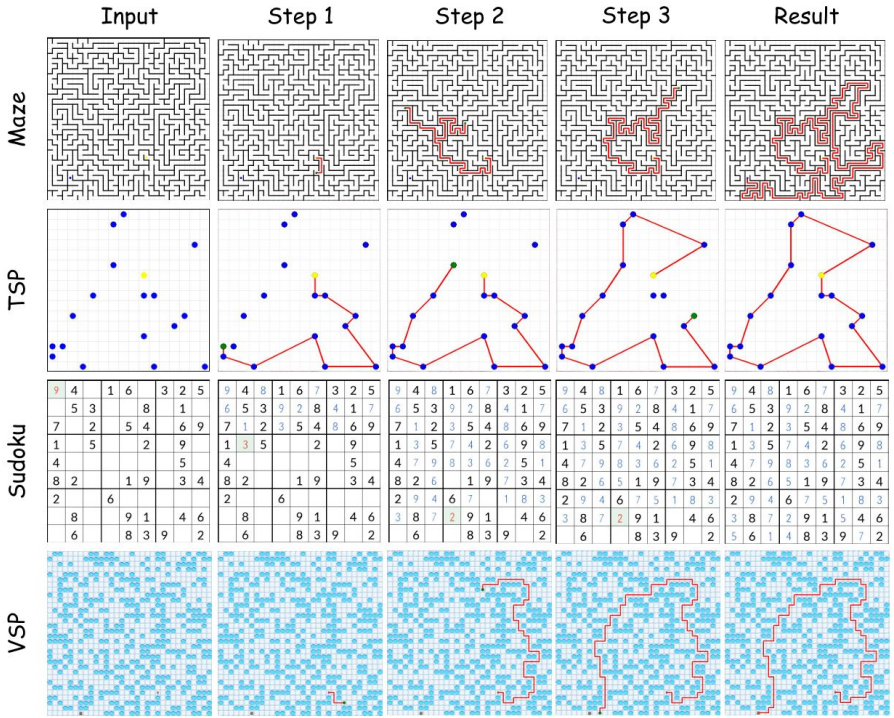




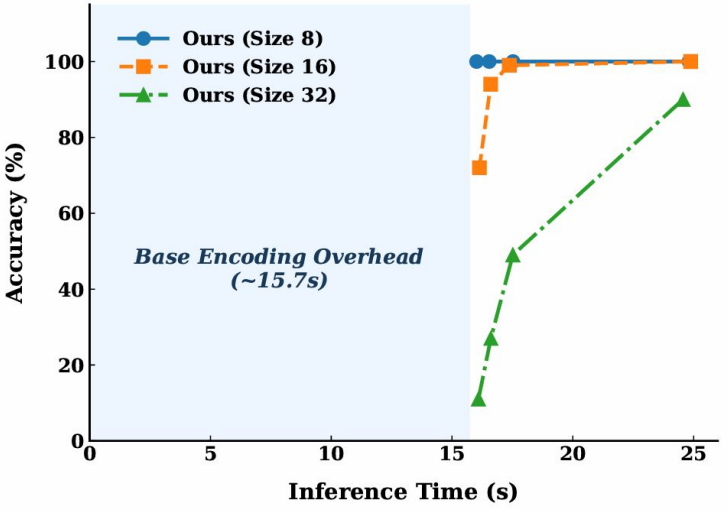
用生成模型做理解任务

EndoCoT

出发点：如何更好的利用用语言模型的reasoning能力

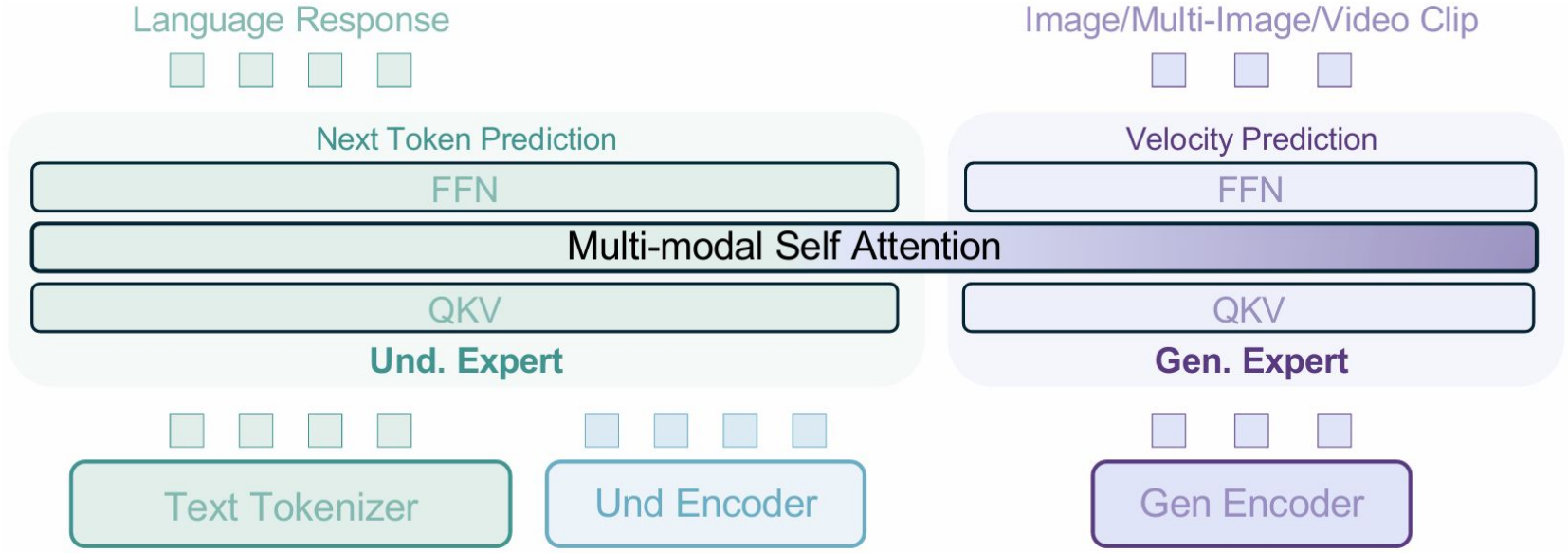


得到的是能逐步推理解决理解任务的生成模型



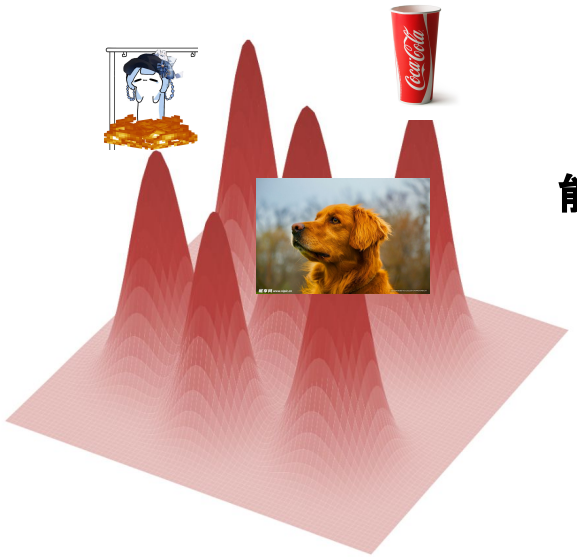
UMM

Bagel



出发点:让生成和理解都共用同一组参数，让一个模型只需要更改Encoder就能做到统一的建模

语言模型的连续建模

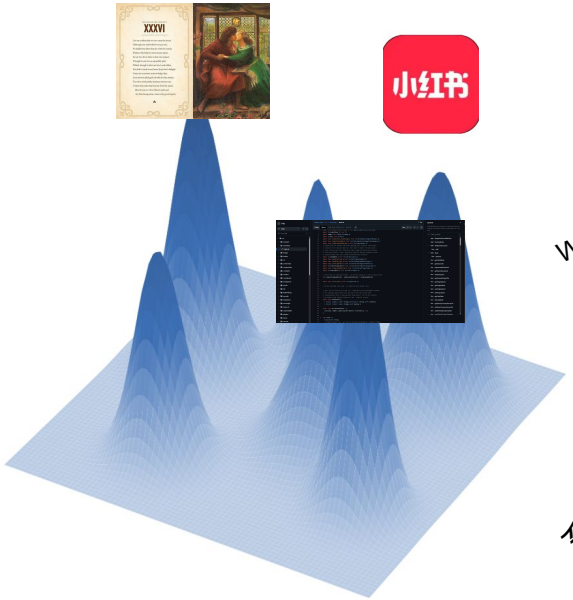


图像

能否用相似的方法建模？

文本

语言模型的连续建模

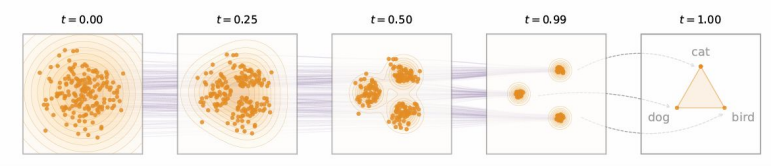


文本

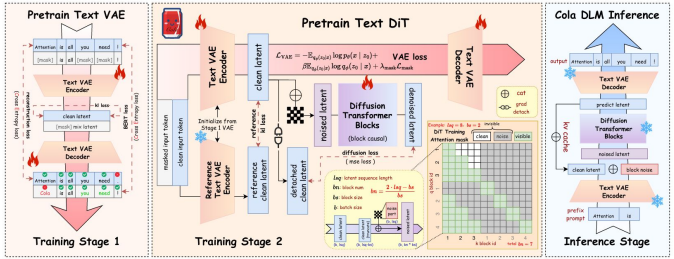
Word Embeds是天然的latent

(Word2Vec)

从头开始训一个latent space



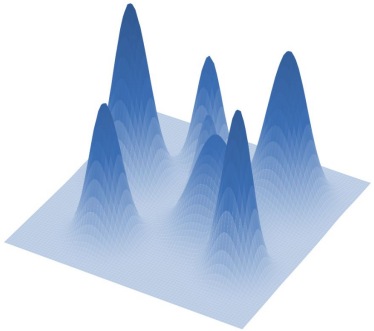
ELF: Embedded Language Flows



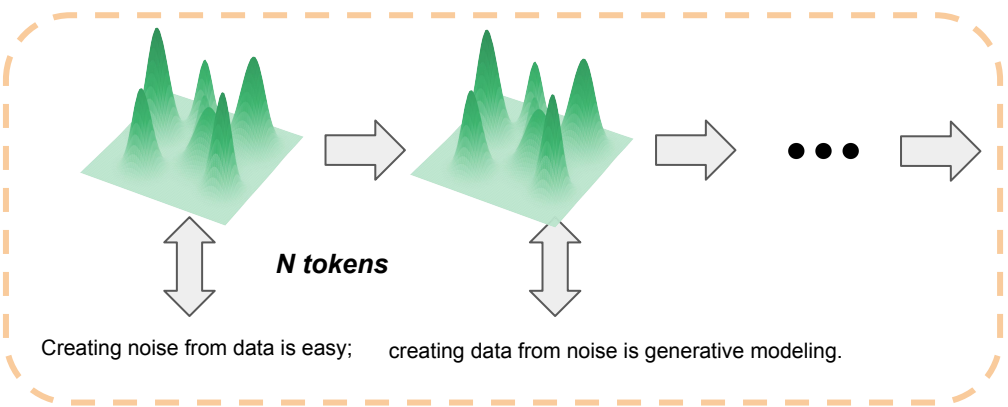
Continuous Latent Diffusion Language Model

语言模型的连续建模

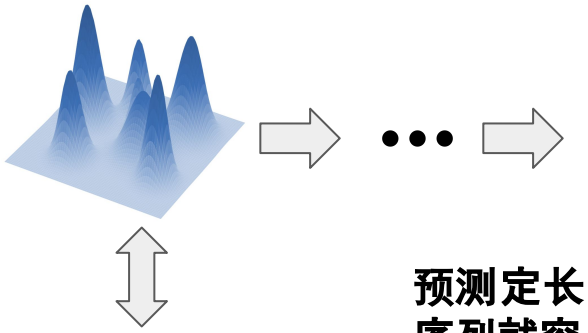
问题简化：AR (Block-by-Block解码）



Creating noise from data is easy; creating data from noise is generative modeling. We present a stochastic differential equation (SDE) that smoothly transforms a complex data distribution to a known prior distribution by slowly injecting noise, and a corresponding reverse-time SDE that transforms the prior distribution back into the data distribution by slowly removing the noise. Crucially, the reverse-time SDE depends only on the time-dependent gradient field (\aka, score) of the perturbed data distribution. By leveraging advances in score-based generative modeling .....



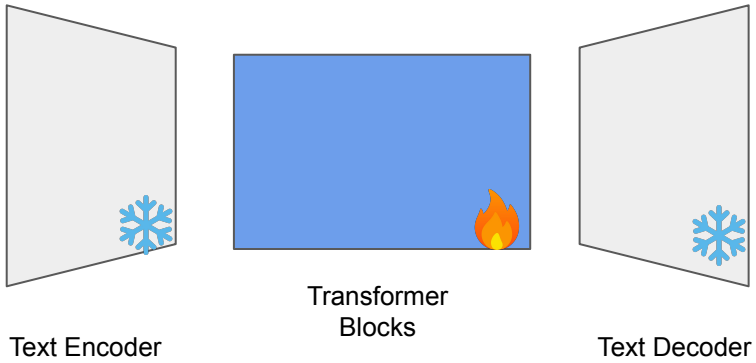
已知的Prefix



生成的未知新内容（1）

预测定长的token序列就容易多了

语言模型的连续建模

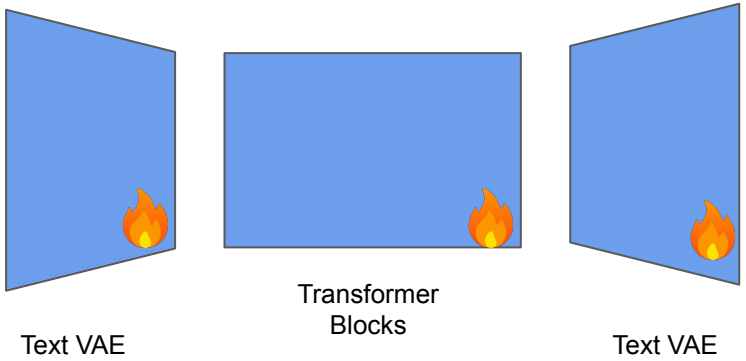


Pros:

1. Latent space天生可解释
2. 建模简洁

Cons:

1. MSE & CE联合监督
2. 测试的任务比较toy
3. 还是在语义空间



Pros:

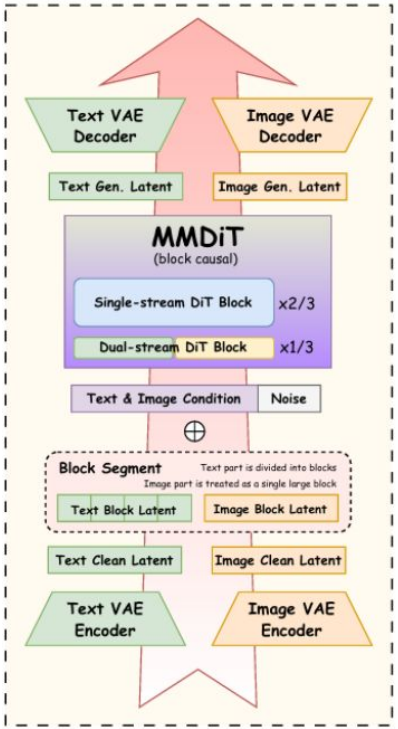
1. 和生成空间天然对齐，是真正的UMM
2. 从实验上看Scaling存在优势（算力而非参数）

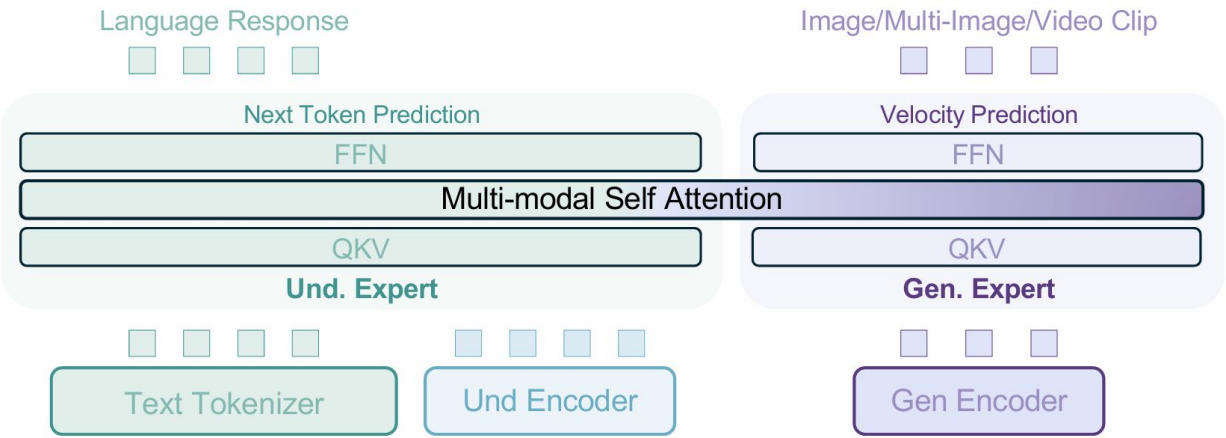
Cons:

1. 目前的实现方式是在不能算是优雅

语言模型的连续建模

CoLa





用这种连续建模去做umm，理论上会比bagel这一套更本质

总结

1. 生成图像背后到底在做什么
   a. DDPM / DDIM（预测噪声）
   b. Score based Model（预测梯度）
   c. Flow Model（预测速度）

2. 生成模型的架构
   Mllm + VAE + mmDiT

3. 生成领域还在被探索的领域
   a. 提升推理效率
   少步推理
   Caching
   b. 优化生成质量
   DPO
   GRPO
   OPD
   c. 探索统一范式
   用生成模型做理解任务
   UMM
   d. 语言模型的连续建模

Text to Image

Image Editing

Text to Vedio