

# Mind-Diffusion

方桂安, 刘梦莎, 张焯霖, 周敏, 唐迅

## 摘要

从文本生成图像 (text to image) 是近几年的 AI generated content (AIGC) 中的一大研究方向, 其主要任务是从一句描述性文本生成一张与文本内容相对应的图片。主流方法有 VAE(Variational Auto-Encoder), GAN (Generative Adversarial Networks) 以及 Diffusion Model 等。作为图像生成领域最近出现的“颠覆性”方法, Diffusion Model 将文本图像生成效果和稳定性拔高到了一个新的高度。Diffusion 最早是 2015 年的一篇文章提出的, 但当时所能达到的效果十分有限, 直到 2020 年 Denoising Diffusion Probabilistic Models (DDPM) 的诞生才让文本图像生成有了较不错的成果。从 21 年底到 22 年间, OpenAI 的 GLIDE、DALLE2 和 Google 的 Imagen 先后对该技术方向进行了探索, 直至最近 Stable Diffusion 开源的大火出圈。本次作业中我们深入讨论了 Diffusion model 的工作原理与基本思想, 使用 mindspore 复现了三个 Diffusion model 经典模型: DDPM, GLIDE 和 Latent Diffusion, 并在此基础上我们提出自己的模型改进与创新思路。最后我们从定量和定性地角度展示了模型的性能。

## 关键词

Generative Models, Diffusion Models, Mindspore, DDPM, GLIDE, Latent Diffusion Model

## I. 概述

DIFFUSION model 是一种新兴的生成模型, 它受到非平衡热力学的启发, 定义了扩散步骤的马尔可夫链, 以缓慢地将随机噪声添加到数据中, 然后学习反向扩散过程, 以从噪声中构建期望的数据样本。它不需要像 VAE 那样对齐后验分布, 像 EBM 那样处理难以处理的分割函数, 像 GAN 那样训练额外的鉴别器, 或者像标准化流那样施加网络约束。在不断发展中, Diffusion model 已经成为了深度生成模型领域的新 SOTA, 在图像生成领域已经超越了 GAN, 并且在诸多应用领域都有出色的表现, 如计算机视觉, 自然语言处理、波形信号处理、多模态建模、分子图建模、时间序列建模等。因此, 在本次任务中, 我们选择了 Diffusion model 来完成此次的图像生成任务。我们将介绍 Diffusion model 的基本原理, 分析该模型不同的思想基础, 描述该模型在前沿领域做出的改进, 最后介绍我们使用 mindspore 复现的 DDPM、Taichu、Wukong 模型。

## II. DIFFUSION 模型的基本原理

Diffusion 模型的灵感来自于非平衡热力学。它定义了一个扩散步骤的马尔可夫链 (当前状态只与上一时刻的状态有关), 缓慢地向真实数据中添加随机噪声 (正向扩散过程), 然后学习反向扩散过程, 从噪声中构建所需的数据样本。图 1 为 Diffusion 模型对应的图像扩散过程。

### A. Diffusion 模型的正向扩散过程

给定一个从真实数据分布中采样的数据点  $\mathbf{x}_0 \sim q(\mathbf{x})$ , 定义一个正向扩散过程, 在这个过程中, 分  $T$  步向样本中添加少量的高斯噪声, 产生一个噪声样本  $\mathbf{x}_1, \mathbf{x}_T$ , 步长由方差表  $\{\beta_t \in (0, 1)\}_{t=1}^T$  控制。

$$\begin{aligned} q(\mathbf{x}_t | \mathbf{x}_{t-1}) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}\right) \\ q(\mathbf{x}_{1:T} | \mathbf{x}_0) &= \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}) \end{aligned}$$

方桂安, 20354027, (e-mail: fanggan@mail2.sysu.edu.cn), 负责 SOTA 的复现。

刘梦莎, 20354091, (e-mail: liumsh6@mail2.sysu.edu.cn), 负责收集资料, 撰写报告。

张焯霖, 20354156, (e-mail: zhangzhlin8@mail2.sysu.edu.cn), 负责收集资料, 撰写报告。

唐迅, 20354121, (e-mail: tangx66@mail2.sysu.edu.cn), 负责 DDPM 的复现。

周敏, 20354187, (e-mail: zhoum87@mail2.sysu.edu.cn), 负责收集资料, 撰写报告。

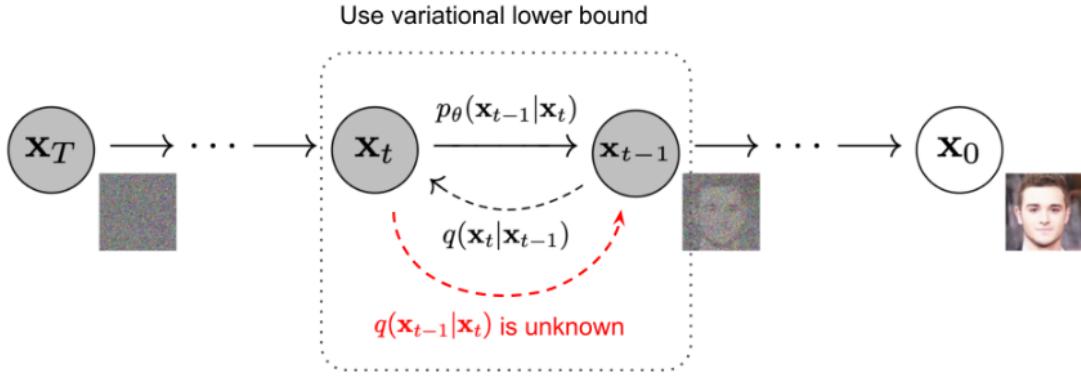


Fig. 1. 缓慢加入（去除）噪声生成样本的正向（反向）扩散过程的马尔可夫链

随着步长  $t$  变大，数据样本  $\mathbf{x}_0$  逐渐失去其可区分的特征。最终当  $T \rightarrow \infty, \mathbf{x}_T$  等价于各向同性高斯分布。

在上述过程中，可以使用重新参数化技巧以封闭形式对  $x_t$  在任意时间步长  $t$  进行采样。令  $\alpha_t = 1 - \beta_t$  和  $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ ：

$$\begin{aligned}\mathbf{x}_t &= \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \boldsymbol{\varepsilon}_{t-1} \\ &= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} \boldsymbol{\varepsilon}_{t-2} \\ &= \dots \\ &= \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon} \\ q(\mathbf{x}_t | \mathbf{x}_0) &= \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})\end{aligned}$$

当合并两个具有不同方差的高斯分布  $\mathcal{N}(\mathbf{0}, \sigma_1^2 \mathbf{I})$  和  $\mathcal{N}(\mathbf{0}, \sigma_2^2 \mathbf{I})$  时，可以得到新的分布  $\mathcal{N}(\mathbf{0}, (\sigma_1^2 + \sigma_2^2) \mathbf{I})$ 。合并的标准差为  $\sqrt{(1 - \alpha_t) + \alpha_t(1 - \alpha_{t-1})} = \sqrt{1 - \alpha_t \alpha_{t-1}}$ 。

当样本噪声变大时，就可以承担更大的更新步长，所以  $\beta_1 < \beta_2 < \dots < \beta_T$ ， $\bar{\alpha}_1 > \dots > \bar{\alpha}_T$ 。

## B. Diffusion 模型的反向扩散过程

如果可以反转上述过程并从  $q(\mathbf{x}_{t-1} | \mathbf{x}_t)$  中采样，就能够从高斯噪声输入  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  中重新创建真实样本。只要  $\beta_t$  足够小， $q(\mathbf{x}_{t-1} | \mathbf{x}_t)$  也将是高斯分布的。然而，估计  $q(\mathbf{x}_{t-1} | \mathbf{x}_t)$  并不容易，因为它需要使用整个数据集，因此需要学习一个模型  $p_\theta$  近似这些条件概率，以便运行反向扩散过程。

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

当条件为  $\mathbf{x}_0$  时，反向条件概率是可处理的：

$$q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$$

基于贝叶斯规则，有：

$$\begin{aligned}q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) &= q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0) \frac{q(\mathbf{x}_{t-1} | \mathbf{x}_0)}{q(\mathbf{x}_t | \mathbf{x}_0)} \\ &\propto \exp\left(-\frac{1}{2} \left( \frac{(\mathbf{x}_t - \sqrt{\alpha_t} \mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0)^2}{1 - \bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \mathbf{x}_0)^2}{1 - \bar{\alpha}_t} \right)\right) \\ &= \exp\left(-\frac{1}{2} \left( \frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t} \mathbf{x}_t \mathbf{x}_{t-1} + \alpha_t \mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 \mathbf{x}_{t-1} + \bar{\alpha}_{t-1} \mathbf{x}_0^2}{1 - \bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \mathbf{x}_0)^2}{1 - \bar{\alpha}_t} \right)\right) \\ &= \exp\left(-\frac{1}{2} \left( \left( \frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}} \right) \mathbf{x}_{t-1}^2 - \left( \frac{2\sqrt{\alpha_t}}{\beta_t} \mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}} \mathbf{x}_0 \right) \mathbf{x}_{t-1} + C(\mathbf{x}_t, \mathbf{x}_0) \right)\right)\end{aligned}$$

按照标准高斯密度函数，均值和方差可以参数化如下：

$$\begin{aligned}\tilde{\beta}_t &= 1 / \left( \frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}} \right) = 1 / \left( \frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t (1 - \bar{\alpha}_{t-1})} \right) = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \\ \tilde{\mu}_t (\mathbf{x}_t, \mathbf{x}_0) &= \left( \frac{\sqrt{\alpha_t}}{\beta_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}} \mathbf{x}_0 \right) / \left( \frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}} \right) \\ &= \left( \frac{\sqrt{\alpha_t}}{\beta_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}} \mathbf{x}_0 \right) \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \\ &= \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} \mathbf{x}_0\end{aligned}$$

将  $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon}_t)$  代入上述等式，得到：

$$\begin{aligned}\tilde{\mu}_t &= \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon}_t) \\ &= \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\varepsilon}_t \right)\end{aligned}$$

可以使用变分下界来优化负对数似然。

$$\begin{aligned}-\log p_{\theta}(\mathbf{x}_0) &\leq -\log p_{\theta}(\mathbf{x}_0) + D_{\text{KL}}(q(\mathbf{x}_{1:T} | \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{1:T} | \mathbf{x}_0)) \\ &= -\log p_{\theta}(\mathbf{x}_0) + \mathbb{E}_{\mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T} | \mathbf{x}_0)} \left[ \log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T}) / p_{\theta}(\mathbf{x}_0)} \right] \\ &= -\log p_{\theta}(\mathbf{x}_0) + \mathbb{E}_q \left[ \log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T})} + \log p_{\theta}(\mathbf{x}_0) \right] \\ &= \mathbb{E}_q \left[ \log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T})} \right] \\ \text{Let } L_{\text{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})} \left[ \log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T})} \right] \geq -\mathbb{E}_{q(\mathbf{x}_0)} \log p_{\theta}(\mathbf{x}_0)\end{aligned}$$

为了将方程中的每一项转换为可分析计算的，目标可以进一步重写为几个  $KL$  散度和熵项的组合。

$$\begin{aligned}L_{\text{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})} \left[ \log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T})} \right] \\ &= \mathbb{E}_q \left[ \log \frac{\prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1})}{p_{\theta}(\mathbf{x}_T) \prod_{t=1}^T p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} \right] \\ &= \mathbb{E}_q \left[ -\log p_{\theta}(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t | \mathbf{x}_{t-1})}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} \right] \\ &= \mathbb{E}_q \left[ -\log p_{\theta}(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t | \mathbf{x}_{t-1})}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} + \log \frac{q(\mathbf{x}_1 | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1)} \right] \\ &= \mathbb{E}_q \left[ -\log p_{\theta}(\mathbf{x}_T) + \sum_{t=2}^T \log \left( \frac{q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t | \mathbf{x}_0)}{q(\mathbf{x}_{t-1} | \mathbf{x}_0)} \right) + \log \frac{q(\mathbf{x}_1 | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1)} \right] \\ &= \mathbb{E}_q \left[ -\log p_{\theta}(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t | \mathbf{x}_0)}{q(\mathbf{x}_{t-1} | \mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1 | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1)} \right] \\ &= \mathbb{E}_q \left[ -\log p_{\theta}(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T | \mathbf{x}_0)}{q(\mathbf{x}_1 | \mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1 | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1)} \right] \\ &= \mathbb{E}_q \left[ \log \frac{q(\mathbf{x}_T | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)} - \log p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1) \right] \\ &= \mathbb{E}_q \underbrace{[D_{\text{KL}}(q(\mathbf{x}_T | \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_T))]_{L_T}}_{L_T} + \underbrace{\sum_{t=2}^T D_{\text{KL}}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t))}_{L_{t-1}} - \underbrace{\log p_{\theta}(\mathbf{x}_0 | \mathbf{x}_1)}_{L_0}\end{aligned}$$

分别标记变分下界损失中的每个组件:

$$L_{\text{VLB}} = L_T + L_{T-1} + \cdots + L_0$$

where  $L_T = D_{\text{KL}}(q(\mathbf{x}_T | \mathbf{x}_0) \| p_\theta(\mathbf{x}_T))$

$$L_t = D_{\text{KL}}(q(\mathbf{x}_t | \mathbf{x}_{t+1}, \mathbf{x}_0) \| p_\theta(\mathbf{x}_t | \mathbf{x}_{t+1})) \text{ for } 1 \leq t \leq T-1$$

$$L_0 = -\log p_\theta(\mathbf{x}_0 | \mathbf{x}_1)$$

### C. 训练损失 $\mathbf{L}_t$ 的参数化

为达到目标，需要学习一个神经网络来近似反向扩散过程中的条件概率分布:

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

期望通过训练  $\mu_\theta$  来预测  $\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_t \right)$ 。因为  $\mathbf{x}_t$  在训练时可用作输入，可以重新参数化高斯噪声项，使其在时间步长为  $t$  时，通过输入  $\mathbf{x}_t$  来预测  $\boldsymbol{\varepsilon}_t$ :

$$\begin{aligned} \mu_\theta(\mathbf{x}_t, t) &= \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) \right) \\ \mathbf{x}_{t-1} &= \mathcal{N} \left( \mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) \right), \Sigma_\theta(\mathbf{x}_t, t) \right) \end{aligned}$$

损失项  $\mathbf{L}_t$  被参数化，以最小化与  $\tilde{\mu}$  的差异:

$$\begin{aligned} L_t &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}} \left[ \frac{1}{2 \|\Sigma_\theta(\mathbf{x}_t, t)\|_2^2} \|\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)\|^2 \right] \\ &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}} \left[ \frac{1}{2 \|\Sigma_\theta\|_2^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_t \right) - \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) \right) \right\|^2 \right] \\ &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}} \left[ \frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t) \|\Sigma_\theta\|_2^2} \|\boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)\|^2 \right] \\ &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}} \left[ \frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t) \|\Sigma_\theta\|_2^2} \|\boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t} \boldsymbol{\varepsilon}_t, t)\|^2 \right] \end{aligned}$$

而正是为了简化损失项  $\mathbf{L}_t$ ，以获得更好的训练效果，Diffusion model 的开山之作 DDPM 被提出来了。

## III. DIFFUSION 模型的主要思想

Diffusion 模型是一系列概率生成模型，通过加入噪声逐步破坏数据，然后学习反转这一过程以生成样本。图 2 中展示了直观 Diffusion 模型的过程。目前对 Diffusion 模型的研究主要基于三种主要思想：去噪 Diffusion 概率模型 (DDPM)、基于分数的生成模型 (SGM) 和随机微分方程 (score SDE)。在本节中，我们对这三种思想进行了介绍，同时讨论了它们之间的联系。

### A. 去噪 Diffusion 概率模型 (DDPM)

去噪 Diffusion 概率模型 (DDPM) 使用了两条马尔可夫链：一条将数据通过扰动变成噪声的正向链，一条将噪声转换为数据的反向链。前者通常是手动设计的，目的是将任何数据分布转换为简单的先验分布（例如，标准高斯分布），而后者马尔可夫链通过学习深度神经网络参数化的 Transition Kernel 来逆转前者。随后，通过首先从先验分布中采样随机向量，然后通过反向马尔可夫链进行原始采样来生成新的数据点。

给定数据分布  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ ，前向马尔可夫过程可以通过 Transition Kernel  $q(x_t | x_{t-1})$  生成随机变量序列  $x_1, x_2, \dots, x_T$ 。使用概率链式法则和马尔可夫性质，可以分解成以  $x_0$  为条件  $q(x_t | x_{t-1})$  的联合分布，即  $q(x_1, x_2, \dots, x_T | x_0)$ ，转换为：

$$q(\mathbf{x}_1, \dots, \mathbf{x}_T | \mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1})$$

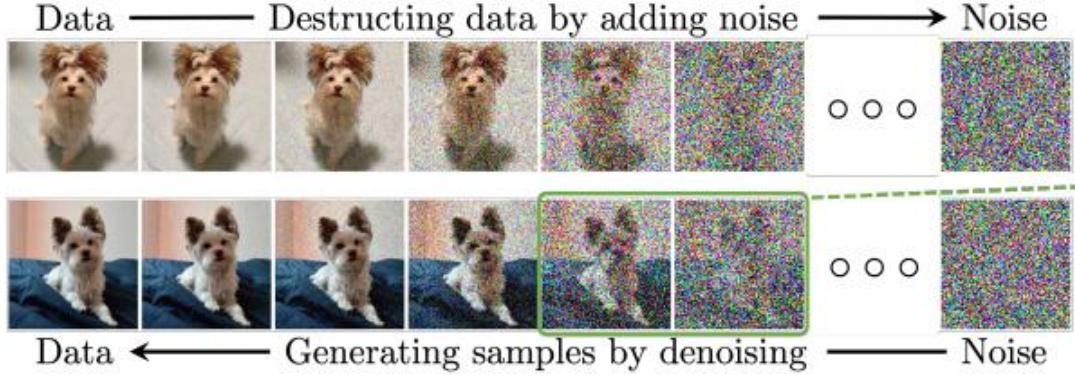


Fig. 2. Diffusion Model 的正向和逆向过程

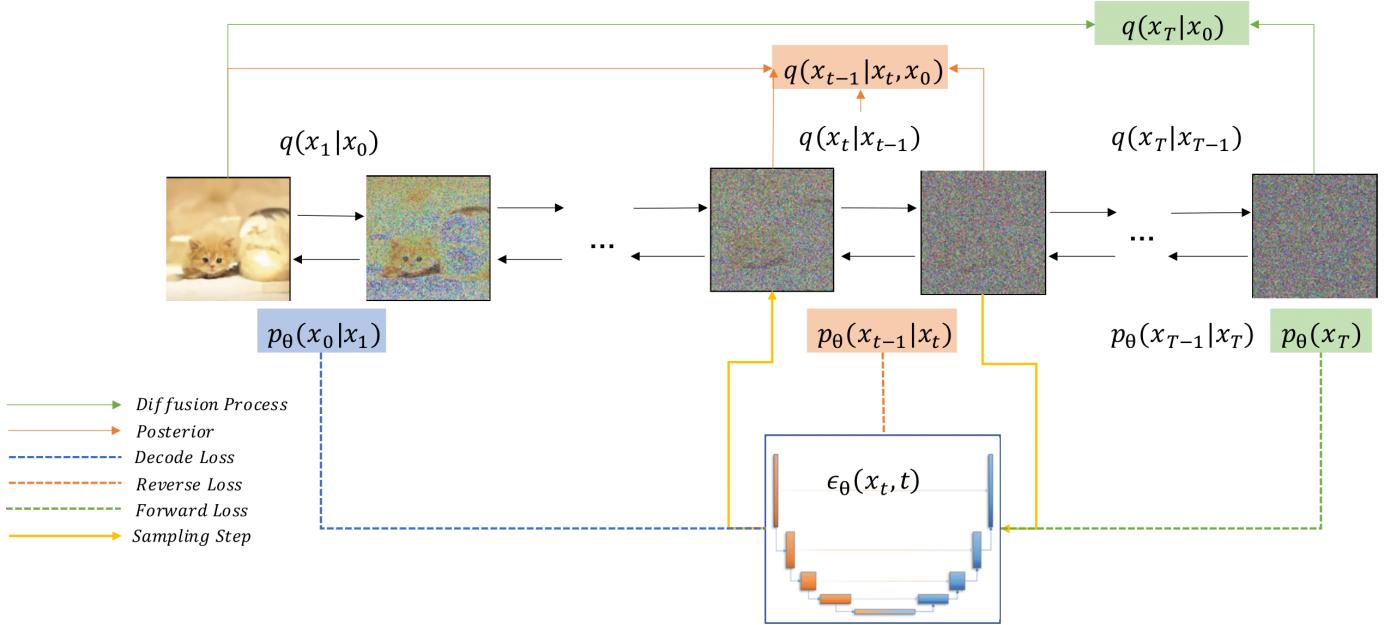


Fig. 3. 去噪 Diffusion 概率模型的流程图。从左到右的箭头表示扩散过程，相反方向的箭头表示反向过程。

在 DDPM 中，人工定义 Transition Kernel  $q(x_t|x_{t-1})$  将数据分布转换为可控制的先验分布  $q(x_0)$ 。Transition Kernel 的一个典型设计是高斯扰动，而过渡核最常见的选择是：

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N} \left( \mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I} \right)$$

其中  $\beta_t \in (0, 1)$  是在模型训练之前选择的超参数。在这里使用这个内核来简化讨论。这个高斯 Transition Kernel 能够将方程中的联合分布边缘化，以获得  $q(x_t|x_0)$  对于所有的  $t \in 0, 1, \dots, T$  的解析式。当  $\alpha_t := 1 - \beta_t$  且  $\bar{\alpha}_t := \prod_{s=0}^t \alpha_s$  时，有：

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N} \left( \mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I} \right)$$

给定  $x_0$ ，可以很容易地通过对高斯向量  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$  进行采样得到  $x_t$  的样本并且应用到变换之中：

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

当  $\bar{\alpha}_t \approx 0, x_t$  就几乎是高斯分布了，所以可以得到  $q(\mathbf{x}_T) = \int q(\mathbf{x}_T | \mathbf{x}_0) q(\mathbf{x}_0) d\mathbf{x}_0 \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ 。

直观地说，这个正向过程缓慢地向数据加入噪声，直到所有结构都丢失。为了生成新的数据样本，DDPM 首先从先验分布生成非结构化噪声向量（通常很难获得），然后通过在反向时间方向上运行可学习的马尔可夫链来逐

渐去除其中的噪声。更具体来讲，反向马尔可夫链由一个先验分布  $p(x_t) = \mathcal{N}(\mathbf{x}_t; 0, I)$  和一个可学习的 transition kernel  $p_\theta(x_{t-1}|x_t)$  参数化。因为正向过程被构造为： $p(x_t) \approx \mathcal{N}(\mathbf{x}_t; 0, I)$ ，选择先验分布  $p(x_t) = \mathcal{N}(\mathbf{x}_t; 0, I)$ 。可学习的 transition kernel:  $p_\theta(x_{t-1}|x_t)$  采取以下形式：

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

其中， $\theta$  代表着模型的参数，并且均值  $\mu_\theta(x_t, t)$  和方差  $\Sigma_\theta(x_t, t)$  都是由深度神经网络参数化的。有了这个反向马尔可夫链，可以首先通过采样噪声向量  $X_T \sim p(x_T)$ ，然后从可学习的 transition kernel:  $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$  迭代采样直至  $t = 1$ 。

该采样过程成功的关键是训练反向马尔可夫链以匹配正向马尔可夫链的实际过程反转。也就是说，必须调整参数  $\theta$  使得逆向马尔可夫链的联合分布  $p_\theta(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T) := p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$  非常近似前向步骤中的  $q(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T) := q(\mathbf{x}_0) \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1})$ ，这可以通过最小化两者的 KL 散度来完成：

$$\begin{aligned} & \text{KL}(q(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T) \| p_\theta(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T)) \\ & \stackrel{(i)}{=} -\mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T)} [\log p_\theta(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T)] + \text{const} \\ & \stackrel{(ii)}{=} \underbrace{\mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T)} \left[ -\log p(\mathbf{x}_T) - \sum_{t=1}^T \log \frac{p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)}{q(\mathbf{x}_t | \mathbf{x}_{t-1})} \right]}_{:= -L_{\text{VLB}}(\mathbf{x}_0)} + \text{const} \\ & \stackrel{(iii)}{\geq} \mathbb{E}[-\log p_\theta(\mathbf{x}_0)] + \text{const}, \end{aligned}$$

其中 (i) 式来自 KL 散度的定义，(ii) 式是因为  $q(x_0, x_1, \dots, x_t)$  和  $p_\theta(x_0, x_1, \dots, x_t)$  都是分布。(iii) 式来自 Jensen 不等式。等式 (ii) 中的第一项是数据  $x_0$  的对数似然的变分下界 (VLB)，这是训练概率生成模型的共同目标。不妨使用“const”来表示不依赖于模型参数的常量  $\theta$ ，因此不影响优化。DDPM 训练的目标是最大化 VLB(等效为最小化 VLB 的负数，即  $-VLB$ )，它是独立项的总和，因此可以通过蒙特卡洛采样进行有效估计，并通过随机优化进行有效优化。

为了得到更好的样本质量，Jonathan Ho 等人提出对  $L_{\text{VLB}}$  内的各项形式重新赋权，并且提出所得损失函数与噪声条件分数网络 (NCSN，一种基于分数的生成模型) 的训练目标之间的重要等价性。损失函数的形式为：

$$\mathbb{E}_{t \sim \mathcal{U}[1, T], \mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\varepsilon} \sim \mathcal{N}(0, I)} \left[ \lambda(t) \|\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)\|^2 \right]$$

其中  $\lambda(t)$  是正加权函数， $x_t$  是根据  $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon}$  计算的， $\mathcal{U}[1, T]$  是集合  $\{1, 2, \dots, T\}$ ， $\boldsymbol{\varepsilon}_\theta$  是一个具有参数  $\theta$  的能够根据给定的  $x_t$  和  $t$  来预测噪声矢量  $\boldsymbol{\varepsilon}$  的深度神经网络。对于选取特定的加权函数  $\lambda(t)$ ，该目标可以简化为式 (ii)，并且具有与基于训练分数的生成模型的多个噪声尺度上的去噪分数匹配损失相同的形式，Diffusion model 的另一个公式将在下一节讨论。

## B. 基于评分的生成模型 (SGM)

基于分数的生成模型的核心是 (Stein) 分数 (也称为分数或分数函数) 的概念。给定概率密度函数  $p(x)$ ，其得分函数定义为对数概率密度的梯度  $\nabla_x \log p_x$ 。与在统计学中常用的 Fisher 评分  $\nabla_\theta \log p_\theta$  不同，这里考虑的 Stein 分数是数据  $x$  的函数，而不是模型参数  $\theta$ ，它是一个指向概率密度函数具有最大增长率方向的向量场。

基于分数的生成模型 (SGM) 的关键思想是通过训练基于噪声水平的深度神经网络模型 (称为噪声条件分数网络，NCSN)，用一系列增强的高斯噪声干扰数据，并联合估计所有噪声数据分布的分数函数。通过使用基于分数的采样方法 (包括 Langevin Monte Carlo、随机微分方程、常微分方程及其各种组合) 在降低噪声水平下链接分数函数来生成样本。在基于分数的生成模型的制定中，训练和采样是完全解耦的，因此可以在估计分数函数之后使用多种采样技术。

使用上一小节中的类似符号，令  $q(x_0)$  是数据分布，并且  $0 < \sigma_1 < \sigma_2 < \dots < \sigma_t < \dots < \sigma_T$  是一系列噪声水平。SGM 的一个典型涉及扰动数据点  $x_0$  通过高斯噪声分布  $q(x_t | x_0) = \mathcal{N}(x_t; x_0, \sigma_t^2 I)$  迭代到  $x_t$ 。这会产生一系列噪声数据密度  $q(x_1), q(x_2), \dots, q(x_T)$ ，其中  $q(\mathbf{x}_t) := \int q(\mathbf{x}_t) q(\mathbf{x}_0) d\mathbf{x}_0$ 。噪声条件得分网络是一个经过训练后能够估计得分函数  $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$  的深度神经网络  $s_\theta(x, t)$ 。从数据中学习分数函数（也称为分数估计）已经建立了分数匹配、去噪分数匹配和切片分数匹配等技术，因此可以直接使用其中的一种来从扰动数据点训练噪声条件分数网络。例如，训练目标由下式给出：

$$\begin{aligned} & \mathbb{E}_{t \sim \mathcal{U}[1, T], \mathbf{x}_0 \sim q(\mathbf{x}_0), \mathbf{x}_t \sim q(\mathbf{x}_t | \mathbf{x}_0)} \left[ \lambda(t) \sigma_t^2 \| \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) - \mathbf{s}_\theta(\mathbf{x}_t, t) \|^2 \right] \\ & \stackrel{(i)}{=} \mathbb{E}_{t \sim \mathcal{U}[1, T], \mathbf{x}_0 \sim q(\mathbf{x}_0), \mathbf{x}_t \sim q(\mathbf{x}_t | \mathbf{x}_0)} \left[ \lambda(t) \sigma_t^2 \| \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) - \mathbf{s}_\theta(\mathbf{x}_t, t) \|^2 \right] + \text{const} \\ & \stackrel{(ii)}{=} \mathbb{E}_{t \sim \mathcal{U}[1, T], \mathbf{x}_0 \sim q(\mathbf{x}_0), \mathbf{x}_t \sim q(\mathbf{x}_t | \mathbf{x}_0)} \left[ \lambda(t) \left\| -\frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t} - \sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t) \right\|^2 \right] + \text{const} \\ & \stackrel{(iii)}{=} \mathbb{E}_{t \sim \mathcal{U}[1, T], \mathbf{x}_0 \sim q(\mathbf{x}_0), \varepsilon \sim \mathcal{N}(0, I)} \left[ \lambda(t) \|\varepsilon + \sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)\|^2 \right] + \text{const}, \end{aligned}$$

其中，(ii) 来自于假设  $q(x_t | x_0) = N(x_t; x_0, \sigma_t^2 I)$ ，(iii) 来自于定理  $x_t = x_0 + \sigma_t \varepsilon$ 。同样地，可以用  $\lambda(t)$  表示一个正加权函数，“const” 是一个不依赖于可训练参数  $\theta$  的常数。很明显只要设定  $\varepsilon_\theta(x, t) = -\sigma_t s_\theta(x, t)$ ，DDPM 和 SGM 的训练目标就是等效的。

对于样本生成，SGM 利用迭代方法依次从  $\mathbf{s}_\theta(\mathbf{x}, T), \mathbf{s}_\theta(\mathbf{x}, T-1), \dots, \mathbf{s}_\theta(\mathbf{x}, 0)$ 。由于 SGM 中训练和推理的解耦，存在许多采样方法，其中一些将在下一节中讨论。这里介绍 SGM 的第一种采样方法，称为 annealed Langevin dynamics(ALD)。令  $N$  为每个时间步长的迭代次数， $s_t > 0$  是步长。首先用  $x_T^{(N)} \sim \mathbf{N}(0, I)$  初始化 ALD，然后应用将 Langevin Monte Carlo 依次应用于  $t = T, T-1, \dots, 1$ 。在每一次更新  $0 \leq t < T$  时，设置  $x_t^{(0)} = x_{t+1}^{(N)}$ ，然后对于  $i = 0, 1, \dots, N-1$  根据以下更新规则进行迭代：

$$\begin{aligned} \varepsilon^{(i)} &\leftarrow N(0, I) \\ x_t^{(i+1)} &\leftarrow x_t^{(i)} + \frac{1}{2} s_t s_\theta(x_t^{(i)}, t) + \sqrt{s_t} \varepsilon^{(i)} \end{aligned}$$

根据 Langevin Monte Carlo 理论： $s_t \rightarrow 0$  和  $N \rightarrow \infty$ ， $x_0^{(N)}$  能够成为数据分布  $q(x_0)$  中的有效样本。

### C. 随机微分方程 (Score SDE)

DDPM 和 SGM 可以进一步推广到无限时间步长或噪声级的情况，其中扰动和去噪过程是随机微分方程 (SDE) 的解。可以称此公式为 Score SDE，因为它利用 SDE 进行噪声扰动和样本生成，并且去噪过程需要估计噪声数据分布的分数函数。

Score SDEs 通过以下随机微分方程 (SDE) 控制的扩散过程将数据扰动为噪声：

$$dx = f(x, t) dt + g(t) dw$$

其中  $f(x, t)$  和  $g(t)$  是 SDE 的扩散和漂移函数， $w$  是标准的维纳过程（也称为布朗运动）。DDPM 和 SGM 中的正向过程都是该 SDE 的离散化。对于 DDPM，对应的 SDE 为：

$$ds = -\frac{1}{2} \beta(t) x dt + \sqrt{\beta(t)} dw$$

其中  $\beta(\frac{t}{T}) = T\beta(t)$  随着  $T$  趋于无穷。这里可以用  $q_t(x)$  表示  $x_t$  在前进过程中的分布。

关键在于，对于  $dx = f(x, t) dt + g(t) dw$  的任何扩散过程，可以通过求解以下 reverse-time SDE 来反转：

$$dx = [f(x, t) - g(t)^2 \nabla_x \log q_t(x)] dt + g(t) d\bar{w}$$

其中  $\bar{\omega}$  是时间反向回转的标准维纳过程,  $dt$  表示无穷小的负时间步长。该反向 SDE 的解轨迹与正向 SDE 的边界密度相同, 只是它们沿相反的时间方向发展。即 reverse-time SDE 的解法是逐渐将噪声转换为数据的扩散过程。还存在一个常微分方程 (ODE), 即概率流 ODE, 其轨迹具有与反向时间 SDE 相同的边界。概率流 ODE 由下式给出:

$$dx = \left[ f(x, t) - \frac{1}{2} g(t)^2 \nabla_x \log q_t(x) \right] dt$$

reverse-time SDE 和概率流 ODE 都允许从相同的数据分布进行采样, 因为它们的轨迹具有相同的边界。

只要在每个时间步骤  $t$  处的  $\nabla_x \log q_t(x)$  是已知的, 即可解出 reverse-time SDE 和概率流 ODE, 随后可以通过使用各种数值技术 (如 annealed Langevin dynamics 数值 SDE 解算器、数值 ODE 解算器和预测修正器方法 (MCMC 和数值 ODE/SDE 解算器的组合)) 来解算它们从而生成样本。参数化一个与时间相关的分数模型  $s_\theta(x_t, t)$  通过将  $\mathbb{E}_{t \sim \mathcal{U}[0, T], x_0 \sim q(x_0), [x_t \sim q(x_t | x_0)]} [\lambda(t) \| s_\theta(x_t, t) - \nabla_{x_t} \log q_{\theta}(x_t | x_0) \|^2]$  中的分数匹配目标推广到连续时间来估计分数函数, 从而实现以下目标:

$$\mathbb{E}_{t \sim \mathcal{U}[0, T], x_0 \sim q(x_0), x_t \sim q(x_t | x_0)} [\lambda(t) \| s_\theta(x_t, t) - \nabla_{x_t} \log q_{\theta}(x_t | x_0) \|^2]$$

其中,  $\mathcal{U}[0, T]$  表示  $[0, T]$  的均匀分布。

随后对扩散模型的研究侧重于从三个方向改进这些经典方法 (DDPM、SGM 和 Score SDE): 更快和更有效的采样、更准确的似然和密度估计, 以及处理具有特殊结构 (如排列不变性、流形结构和离散数据) 的数据。接下来我们介绍如何更快和更有效的采样。

#### IV. 加速采样的 DIFFUSION 模型

从 Diffusion 模型生成样本通常涉及大量评估步骤的迭代方法。最近的很多研究工作都聚焦于如何加快采样过程, 同时提高所得样本的质量。将这些有效的采样方法分为两大类: 不涉及学习的方法 (无学习采样) 和在 Diffusion 模型训练后需要额外学习过程的方法 (基于学习的采样)。

##### A. 无学习采样

在求解 Diffusion SDE 时, 减小离散化步长可以加快采样过程。然而, 这种方法会导致离散化出错, 并对模型性能带来负面影响。因此, 研究者们已经开发了许多方法来优化离散化方案, 同时通过求解随机微分方程 (SDE) 或常微分方程 (ODE) 来减少采样 steps, 以保持良好的样本质量。

1) **SDE 求解器**: 已经有许多方法来解决连续时间设置中的反向扩散 SDE。SGM 旨在以与前向 SDE 相同的方式离散化反向时间 SDE。将前向 SDE 和反向 SDE 离散化为下列方程。并假设以下迭代规则是前向 SDE 的离散化:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{f}_i(\mathbf{x}_i) + \mathbf{g}_i \mathbf{z}_i, i = 0, 1, \dots, N-1$$

其中  $z_i \sim \mathcal{N}(0, \mathbf{I})$  和  $\mathbf{g}_i$  由 SDE 和离散化方案确定。可以用经过训练的得分函数  $s_{\theta^*}(x_i, i)$  离散化逆时间的 SDE:

$$\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{f}_{i+1}(\mathbf{x}_{i+1}) + \mathbf{g}_{i+1} \mathbf{g}_{i+1}^T s_{\theta^*}(\mathbf{x}_{i+1}, i+1) + \mathbf{g}_{i+1} \mathbf{z}$$

该过程可以应用于任何 Diffusion 模型, 经验结果表明, 该采样器的性能略好于 DDPM。此外, SGM 提出在 SDE 求解器 (solver) 中添加一个“校正器 (corrector)”, 以生成具有正确边际分布的样本。具体来说, 在数值 SDE 求解器在每个时间 step 都对样本进行估计后, “校正器”将使用马尔可夫链蒙特卡罗方法校正估计样本的边际分布。实验结果表明, 添加校正器比使用更多的 steps 更有效。

另一方面, 一种自适应步长 SDE 求解器将快速高阶 SDE 求解器与由 Jolicoeur-Martineau 等人开发的精确低阶 SDE 求解器相结合。在每个时间 steps, 高阶和低阶求解器从先前的样本  $\mathbf{x}'_{\text{rev}}$  生成新的样本  $\mathbf{x}'_{\text{high}}$  和  $\mathbf{x}'_{\text{low}}$

。如果  $\mathbf{x}'_{\text{high}}$  和  $\mathbf{x}'_{\text{low}}$  相似，算法会接受  $\mathbf{x}'$  作为在当前步对  $\mathbf{x}'_{\text{high}}$  和  $\mathbf{x}'_{\text{low}}$  进行插值得到的新样本，并增加步长。 $\mathbf{x}'_{\text{high}}$  和  $\mathbf{x}'_{\text{low}}$  之间的不一致性 (dissimilarity) 定义为：

$$E_q = \left\| \frac{\mathbf{x}'_{\text{low}} - \mathbf{x}'_{\text{high}}}{\delta(\mathbf{x}, \mathbf{x}'_{\text{prev}})} \right\|^2$$

其中  $\delta(\mathbf{x}'_{\text{low}}, \mathbf{x}'_{\text{high}}) = \max(\epsilon_{\text{abs}}, \epsilon_{\text{rel}} \max(|\mathbf{x}'|, |\mathbf{x}'_{\text{prev}}|))$  是公差 (tolerance)，且  $\epsilon_{\text{abs}}$  和  $\epsilon_{\text{rel}}$  是超参数。如果  $E_q \leq 1$ ，那么  $\mathbf{x}'_{\text{high}}$  与  $\mathbf{x}'_{\text{low}}$  相近，结果是比较准确的。因此，可以将外推样本 (extrapolation sample) 用作该时间 step 的新样本，因为它可以被视为以大步长绘制的准确样本。

受 SDE 收缩理论的启发，CCDF 表明，从具有更好初始化的单个前向扩散出发，可以显著减少采样 steps 的数量。在每个生成 steps 中，每个样本都通过非扩展线性映射进行转换：

$$\begin{aligned} \tilde{\mathbf{x}}_{i-1} &= f(\mathbf{x}_i, i) + g(\mathbf{x}_i, i) \mathbf{z}_i \\ \mathbf{x}_{i-1} &= \mathbf{A}\tilde{\mathbf{x}}_{i-1} + \mathbf{b} \end{aligned}$$

其中  $\mathbf{z}_i \sim \mathcal{N}(0, \mathbf{I})$ ，且  $\mathbf{A}$  是非扩展的：

$$\|\mathbf{Ax} - \mathbf{Ay}\| \leq \mathbf{x} - \mathbf{y}, \quad \forall \mathbf{x}, \mathbf{y}$$

使用随机收缩理论，可以证明：存在一条等待  $\mathbf{A}$  的较短采样路径，并且该较短采样路径的估计误差受原始估计误差的限制。因此，我们可以在更早的 step 开始生成，以获得更好的结果。最近，DSB 通过将逆过程学习视为薛定谔桥 (Schrödinger Bridge) 问题来扩展 SGM。它提出了使用迭代比例拟合过程的近似来解决具有一定离散化的 DSB 问题。此外，DSB 提供了生成建模和最优运输问题 (optimal transport problem) 之间的联系。

**2) ODE 求解器:** ODE 求解器不需要高斯采样作为 SDE 求解器的过程，因此可以提高 Diffusion 模型的采样效率。如 SGM 所示，每个 Diffusion 模型都有一个对应的 ODE，其边缘分布与扩散 SDE 相同：

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}} \log q_t(\mathbf{x}) dt$$

因此我们可以设计更好的 ODE 求解器，并使用派生的离散化 ODE 生成数据。DPM 求解器演示了扩散 ODE 解的过程。该公式计算了该解的线性部分，通过神经网络的指数加权积分来近似非线性部分。它将以下包含 DDPM 的扩散 ODE 视为特例：

$$\begin{aligned} \frac{d\mathbf{x}_t}{dt} &= f(t)\mathbf{x}_t - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}} \log q_t(\mathbf{x}_t) \\ \mathbf{x}_t &= e^{\int_s^t f(\tau)d\tau} \mathbf{x}_s + \int_s^t \left( e^{\int_t^\tau f(r)dr} \frac{g^2(\tau)}{2\sigma_\tau} \epsilon_\theta(\mathbf{x}_\tau, \tau) \right) d\tau \end{aligned}$$

其中学习的得分函数  $-\frac{\epsilon_\theta(\mathbf{x}_\tau, \tau)}{\sigma_\tau}$  用于替换 oracle 得分函数。该公式可以减少线性部分的离散化误差，从而可以在保持样本质量的同时减少采样步数。此外，它可以通过将得分函数扩展到高阶来减少拟合误差，这种方法进一步提出了结合不同阶数的求解器来动态调整步长。从而提高效率。

DEIS 还利用了学习扩散过程的半线性结构，提出使用二次时间步长离散化扩散 ODE，如 DDIM。它还提出使用指数积分器来近似计算 ODE 在时间步中的解，并使用多项式外推法来进行更好的分数估计。在估计时间离散化  $\{t_i\}_{i=0}^N$  时的得分函数  $\epsilon_\theta(x_{t_i}, t_i)$  后，它拟合了关于插值点  $(t_i, \epsilon_\theta(\mathbf{x}_{t_i}, t_i))$  的  $r$  次多项式  $\mathbf{P}_r(t)$ ：

$$\mathbf{P}_r(t) = \sum_{j=0}^r \left[ \prod_{k \neq j} \frac{t - t_{i+k}}{t_{i+j} - t_{i+k}} \right] \epsilon_\theta(\mathbf{x}_{t_{i+j}}, t_{i+j})$$

然后它使用  $\mathbf{P}_r(t)$  在区间  $[t_{i-1}, t_i]$  上逼近  $\epsilon_\theta(x_\tau, \tau)$ 。减少离散化和拟合误差，从而减少了估计 steps。PNDM 证明我们可以通过求解流形上的 DDPM 微分方程来设计更快的高阶 ODE。

马尔可夫过程只依赖于最后一步的样本进行预测，限制了对先前丰富的样本信息进行利用。相比之下，非马尔可夫过程的转换核可以依赖更多的样本，并且可以利用来自这些样本的更多信息。因此，它可以以较大的步长

进行准确的预测，从而相应地加快了采样过程。DDIM 将原始的 DDPM 扩展到非马尔可夫情况。形式上，DDIM 提出前向扩散链为：

$$\begin{aligned} q(\mathbf{x}_1, \dots, \mathbf{x}_T | \mathbf{x}_0) &= \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0) \\ q_\sigma(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) &= \mathcal{N}(\mathbf{x}_{t-1} | \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \mathbf{I}) \\ \tilde{\mu}_t(\mathbf{x}_t | \mathbf{x}_0) &= \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 + \sqrt{\bar{\beta}_{t-1} - \sigma_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \mathbf{x}_0}{\sqrt{\bar{\beta}_t}} \end{aligned}$$

该公式将 DDPM 和 DDIM 封装为特殊情况。此外，DDIM 的生成方案可以看作是 ODE 的特殊离散化。因此，可以确定性地对数据进行采样。

## B. 基于学习的采样

基于学习的抽样是 Diffusion 模型的另一种有效方法。除了非马尔可夫过程外，也可以仅使用反向过程中的部分 steps 来生成样本，以样本质量换取采样速度。使用动态规划算法可以选择一个最优子轨迹，使训练的 VLB 最大化。那么这个子轨迹只能用一代。这里关键的一点是，我们可以根据子轨迹分解 VLB。给定推理时间 steps  $0 \leq t'_0 \leq t'_1 \leq \dots \leq t'_{K-1} \leq t'_K = T$  的路径，可以得出相应的 ELBO：

$$\begin{aligned} -L_{VLB} &= \mathbf{E}_q \mathbf{D}_{KL}[q(x_1 | x_0) \| p_\theta(x_1)] + \sum_{i=1}^K L(t'_i, t'_{i-1}) \\ L(t, s) &= \begin{cases} -\mathbf{E}_q \log p_\theta(\mathbf{x}_t | \mathbf{x}_0) s = 0 \\ \mathbf{E}_q \mathbf{D}_{KL} q(\mathbf{x}_s | \mathbf{x}_t, \mathbf{x}_0) p_\theta(\mathbf{x}_s | \mathbf{x}_t) s > 0 \end{cases} \end{aligned}$$

因此，每个  $L(i,j)$  都会被记住，并且可以应用动态规划，为所有预训练的 DDPM 选择具有连续时间 steps 的最优轨迹。结合 Analytic-DPM 中的结果，可以选择最优的子轨迹。

**1) 优化离散化**: Watson 等人提出了一个新的策略，如果给定一个预训练的 Diffusion 模型，可以通过选择最佳时间步来最大化训练目标，来找到最佳离散化方案 DDPM。这种方法的关键是观察到 DDPM 目标可以分解为离散的和，使其非常适合动态规划。随后通过 Kernel Inception Distance (KID) 直接优化样本质量来解决此问题。

**2) 截断扩散**: 可以通过截断正向和反向扩散过程来提高采样速度。核心思想是在几步之后尽早停止向前扩散过程，并开始具有非高斯分布的反向去噪。通过从预训练的生成模型（如变分自编码器或生成对抗网络）中扩散样本，可以有效地从该分布中获得样本。

**3) 知识蒸馏**: 使用知识蒸馏的方法可以显著提高 Diffusion 模型的采样速度。具体来说，在逐级蒸馏中，作者提出将整个采样过程蒸馏到一个只需一半步骤的更快的采样器中。通过将新的采样器参数化为深度神经网络，作者能够训练采样器来匹配 DDIM 采样过程的输入和输出。重复此过程可以进一步减少采样步骤，但这会导致样本质量降低。为了解决这个问题，作者提出了新的参数化 Diffusion 模型和新的加权方案。

## V. 与其他生成模型的联系

生成模型是一个有很多种应用的热门研究领域。例如，它们可用于生成高质量图像、合成逼真的语音和音乐、推进半监督学习、识别对抗性示例、进行模仿学习，并在强化学习中进行优化。在本节中，我们首先介绍其他五类重要的生成模型，并分析它们的优点和局限性。然后我们介绍 Diffusion 模型是如何与它们联系起来的，并说明了这些生成模型是如何通过结合 Diffusion 模型来提升的。Diffusion 模型与其他生成模型集成的算法总结在表 I 中，我们还在图 4 中提供了示意图。

Model	Article	Year
VAE	Luo et al.	2022
	Hunag et al.	2021
	Vadhat et al.	2021
GAN	Wang et al.	2022
	Xiao et al.	2021
Normalizing Flow	Zhang et al.	2021
	Gong et al.	2021
Autoregressive Model	Hoogeboom et al.	2021
	Rasul et al.	2021
Energy-based Model	Gao et al.	2021
	Yu et al.	2022

TABLE I

算法统计

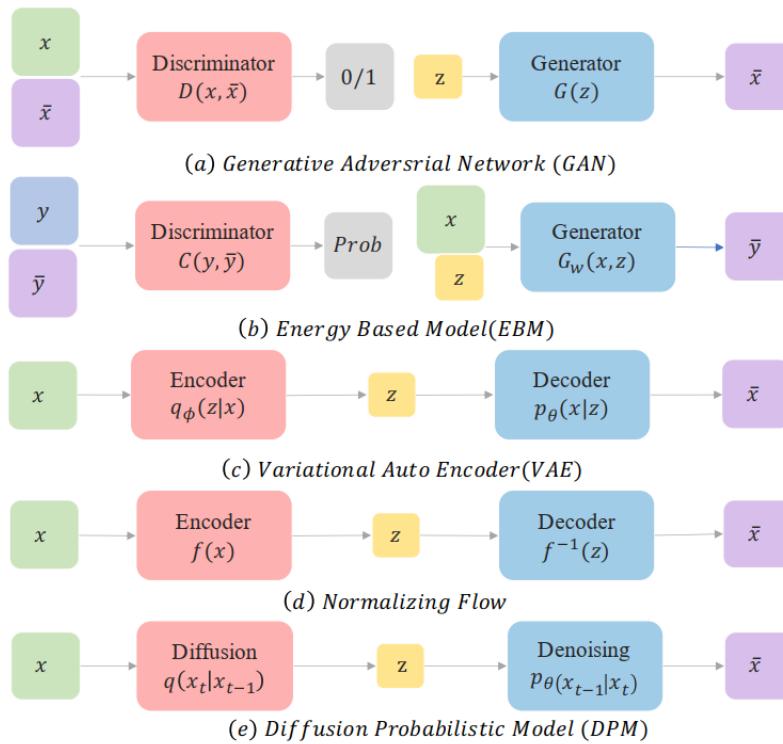


Fig. 4. 将 Diffusion 模型与其他生成模型相结合的作品

### A. 变分自编码器及其与 Diffusion 模型的联系

变分自动编码器期望同时学习编码器和解码器，以将输入数据映射到连续隐空间中的值。它们提供了一个公式，其中嵌入（embedding）可以解释为概率生成模型中的隐变量，概率解码器可以通过参数化似然函数来公式化。该模型假设数据  $\mathbf{x}$  是由一些未观察到的隐变量  $\mathbf{z}$  使用条件分布  $p_\theta(\mathbf{x} | \mathbf{z})$  生成的，并且它使用  $q_\phi(\mathbf{z} | \mathbf{x})$  来近似推断  $\mathbf{z}$ 。通过最大化证据下限（ELBO）实现了一种变分贝叶斯方法和这个过程：

$$\mathcal{L}(\phi, \theta; \mathbf{x}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} | \mathbf{x})]$$

其中  $\mathcal{L}(\phi, \theta; \mathbf{x}) \leq \log p_\theta(\mathbf{x})$ 。假设参数化似然函数  $p_\theta(\mathbf{x} | \mathbf{z})$  和参数化后验近似值  $q_\phi(\mathbf{z} | \mathbf{x})$  可以逐点计算，并且可以通过它们的参数进行微分，则可以通过梯度下降来最大化 ELBO。它允许灵活选择编码器和解码器模型。通常它们由指数族分布表示，其参数由多层神经网络生成。

DDPM 可以被视为分层马尔可夫 VAE 并带有前缀编码器。前向过程表示编码器，反向过程表示解码器。此外，DDPM 跨多个层共享解码器，并且所有隐变量与样本数据的大小相同。在连续时间设置中，分数匹配目标可以通过深度分层 VAE 的 ELBO 进一步近似。通过这种方式，优化 Diffusion 模型可以被视为训练一个无限的、深度的、分层的 VAE。

### B. 生成对抗网络及其与 Diffusion 模型的联系

生成对抗网络 (GAN) 因其有趣的对抗思想和效果优越的架构而受到了广泛关注。GAN 主要由两个模型组成：生成器和鉴别器。这两个模型通常由神经网络构建，但可以以任何形式的可微系统实现，该系统将输入数据从一个空间映射到另一个空间。生成器会对真实示例的分布进行建模，并生成新的示例。鉴别器通常是一个二元分类器，从真实示例中识别生成的示例。GAN 的优化可以看作是一个 minimax 优化问题，其价值函数  $V(G,D)$  如下：

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(\mathbf{x})] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]$$

优化在鞍点 (saddle point) 处结束，该鞍点产生关于生成器的最小值和关于判别器的最大值。这意味着 GAN 优化的目标是实现纳什平衡 (Nash equilibrium)。此时，可以认为生成器已经捕获了真实示例的准确分布。

GAN 的主要实际问题之一是训练过程中的不稳定性，这主要是由于输入数据的分布与生成数据的分布不重叠造成的。一个实用的解决方案是将噪声作为鉴别器的输入。利用灵活的 Diffusion 模型，有学者提出使用由 Diffusion 模型确定的自适应噪声调度向鉴别器注入噪声。相反，GAN 可以帮助提高 Diffusion 模型的采样速度。但也有人认为慢采样是由去噪 step 中的高斯假设引起的，这仅适用于小步长。因此，建议使用条件 GAN 对每个去噪 step 进行建模，从而允许更大的步长。

### C. 归一化流及其与 Diffusion 模型的联系

因其精确的密度评估和对高维数据建模的能力，归一化流 (Normalizing flows) 是一类很有前途的模型。归一化流可以将简单的概率分布转化为极其复杂的概率分布，可用于生成模型、强化学习、变分推理等领域。构建它所需的工具是行列式、雅可比矩阵和随机变量的变量替换定理 (Change of Variable Theorem)。归一化流的轨迹由微分方程表示。在离散时间设置中，归一化流中从数据  $\mathbf{x}$  到隐式  $\mathbf{z}$  的映射是一系列双射函数的组合，如  $F = F_N \circ F_{N-1} \circ \dots \circ F_1$ 。

对所有  $i \leq N$  轨迹  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_i\}$  都满足：

$$\mathbf{x}_i = F_i(\mathbf{x}_{i-1}, \theta), \quad \mathbf{x}_{i-1} = F_i^{-1}(\mathbf{x}_i, \theta)$$

与连续条件类似，在基于随机变量的变量替换定理的规范化流中可以访问精确的对数似然。然而，对双射的要求导致了对复杂数据的建模的局限性。一些工作试图放宽双射的要求。Diffflow 提出了一种新的生成建模算法，结合了基于流的模型和 Diffusion 模型的优点。因此，它不仅可以获得比归一化流更清晰的边界，还可以通过比 Diffusion 概率模型更少的离散化步骤来学习更一般的分布。

### D. 自回归模型及其与 Diffusion 模型的联系

自回归模型 (ARM) 是使用概率链规则，将数据的联合分布分解为条件分布的乘积：

$$\log p(\mathbf{x}_{1:T}) = \sum_{t=1}^T \log p(\mathbf{x}_t | \mathbf{x}_{<t})$$

其中  $\mathbf{x}_{<t}$  是  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{t-1}$  的简写。

深度学习的最新技术在各种模式上都取得了巨大的进步，例如图像、音频和文本，它们被称为语言文本建模中的模型。可以通过调用单个神经网络来检索 ARM 的可能性。尽管自回归模型是一个非常强大的密度估计器，

但采样本质上也是一个连续过程，并且在高维数据上可能会非常慢。自回归 Diffusion 模型 (ARDM) 提出了一种不同的自回归模型，它学习生成任意顺序的数据。ARDM 扩展了与顺序无关的自回归模型和离散 Diffusion 模型。与 ARM 不同，ARDM 不需要对表示进行因果屏蔽，因此可以以有效的目标进行训练，这类似于 Diffusion 概率模型。在测试过程中，ARDM 能够并行地生成，因此它可以应用于任意预算生成任务。

### E. 基于能量的模型及其与 Diffusion 模型的联系

基于能量的模型 (EBM) 最近受到了很多关注。EBM 可以看作是鉴别器的一种生成版本，同时可以从未标记的输入数据中学习。令  $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$  表示训练样本， $p_{\theta}(\mathbf{x})$  表示一个旨在逼近  $p_{\text{data}}(\mathbf{x})$  模型的概率密度函数。基于能量的模型 (EBM) 定义如下：

$$p_{\theta}(\mathbf{x}) = \frac{1}{Z_{\theta}} \exp(f_{\theta}(\mathbf{x}))$$

其中  $Z_{\theta} = \int \exp(f_{\theta}(\mathbf{x})) d\mathbf{x}$  是划分函数。对于高维  $\mathbf{x}$ ，在分析上是难以处理的。对于图像， $f_{\theta}(\mathbf{x})$  由具有标量输出 (scalar output) 的卷积神经网络参数化。

尽管 EBM 具有许多理想的特性，但在高维数据上训练 EBM 仍然存在两个挑战。首先，通过最大化似然性来学习 EBM 需要马尔可夫链蒙特卡罗 (MCMC) 方法从模型中生成样本，这可能非常耗费精力。其次，使用非收敛 MCMC 学习的能量势 (energy potentials) 不稳定，因为来自长期马尔可夫链的样本可能与观察到的样本有显著差异，因此很难评估学习的能量势。

最近的一项研究提出了一种扩散恢复似然方法，可以在 Diffusion 模型的逆过程中从一系列 EBM 中轻松学习样本。每个 EBM 都经过恢复似然训练，旨在使数据在特定噪声水平下的条件概率最大化，因为它们的噪声版本处于较高的噪声水平。EBM 最大化恢复似然比边际似然更容易处理，因为从条件分布中抽样比从边际分布中抽样容易得多。该模型可以生成高质量的样本，且来自条件分布的长期 MCMC 样本类似于真实图像。

## VI. 文本到图像的生成

文本到图像生成任务最近在模型方面取得了很大进展，这些模型可以在文本输入下生成越来越逼真的图像，生成图像的内容和风格具有前所未有的灵活性。目前提出了许多引领发展的模型，如 DALL·E 模型、Imagen 模型和 Stable Diffusion 模型。



Fig. 5. DALL·E 的生成效果

### A. DALL·E

DALL·E 模型于 2021 年 1 月发布，DALL·E 利用了 Transformer 以及变分自动编码器 (VAE)，这种经过训练的模型可以将图像编码为低维概率分布，然后将其解码，可用于中间分布采样和通过解码器来生成新图像。

Transformer 在一系列文本标记和一系列图像标记的串联上进行自回归训练，使其尝试根据过去的值预测未来的值，其中第二个是事先通过 VAE 训练获得的。在采样过程中，转换器会收到全文提示，并按顺序生成图像中的每个令牌，稍后由 VAE 解码以获得完整的输出图像。

在发布时，DALL·E 在生成卡通/艺术外观的图像时取得了很好的效果。然而，它在尝试生成逼真的现实图像时令人大失所望。

## B. DALL · E 2

针对 DALL · E 的缺点，OpenAI 做出了改进，于 2022 年 4 月发布了 DALL · E 2。DALL · E 2 不同于 GLIDE 中提出的原始文本嵌入，它建立在 GLIDE 建立的基础上，并更进一步用 CLIP 图像嵌入来调节扩散过程。

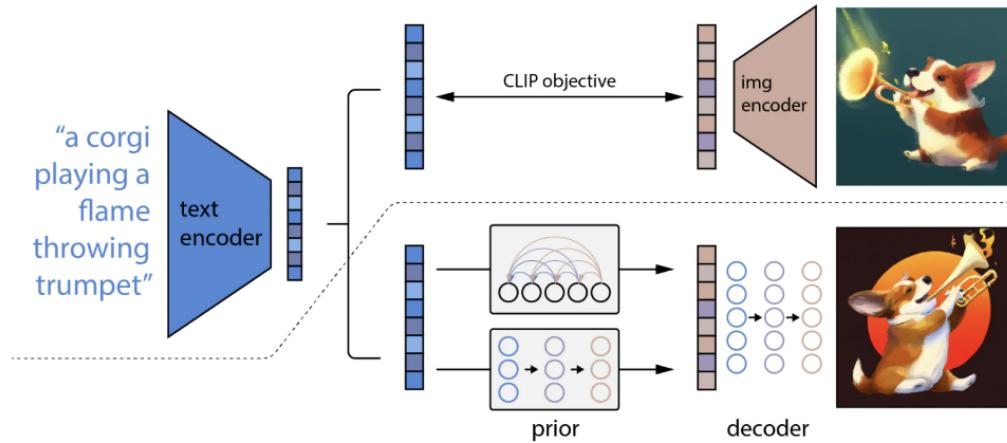


Fig. 6. DALL · E 2 的架构

为了获得这种图像嵌入，首先训练了 CLIP 和 DALL · E 的组合数据集和先验辅助模型，以产生编码标题及其 CLIP 嵌入为条件的图像嵌入。然后，此图像嵌入（以及可选的编码标题）调节用于生成最终图像的扩散模型，称为解码器。与 GLIDE 相比，使用生成的 CLIP 图像嵌入作为条件可以提高样本多样性。

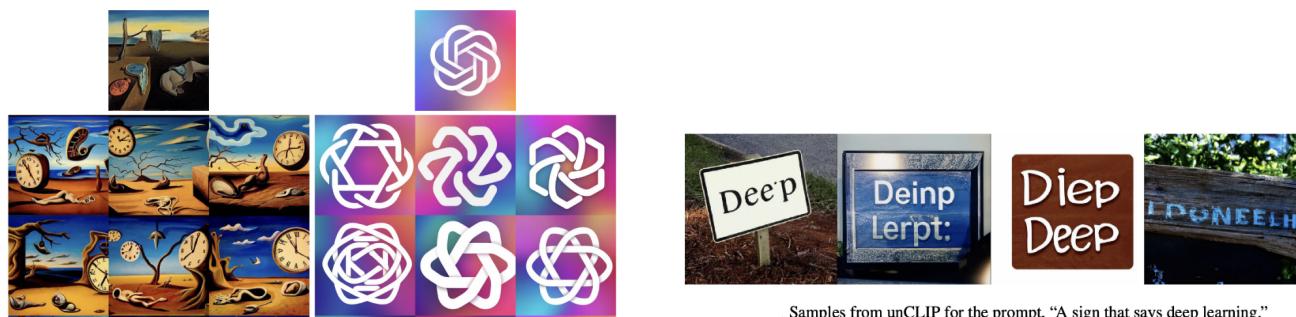


Fig. 7. DALL · E 2 生成效果

## C. Imagen

谷歌于 2022 年 5 月发布了 Imagen 模型，他们声称模型拥有“前所未有的照片级真实感和深层次的语言理解”，它将其预先训练的文本编码器替换为在纯文本语料库上预先训练的通用大型语言模型，以此来改进 GLIDE，而不是像 GLIDE 一样在 DALL · E 上训练的 1.2B 参数 transformer 作为训练制度的一部分。

Imagen 使用 T5-XXL 模型，这个 transformer 在更大的数据集上训练了 4.6B 参数。提出者发现缩放文本编码器的大小比缩放图像 Diffusion 模型的大小更能提高样本质量。除了核心文本编码器和文本到图像模型外，Imagen 还使用了另外两个 Diffusion 模型将输出图像从 64x64 放大到 1024x1024 像素。

Imagen 还介绍了：

- (1) 动态阈值：它允许在采样期间执行更强的监督，以提高照片真实感和文本相匹配，而不会生成过度饱和的图像。

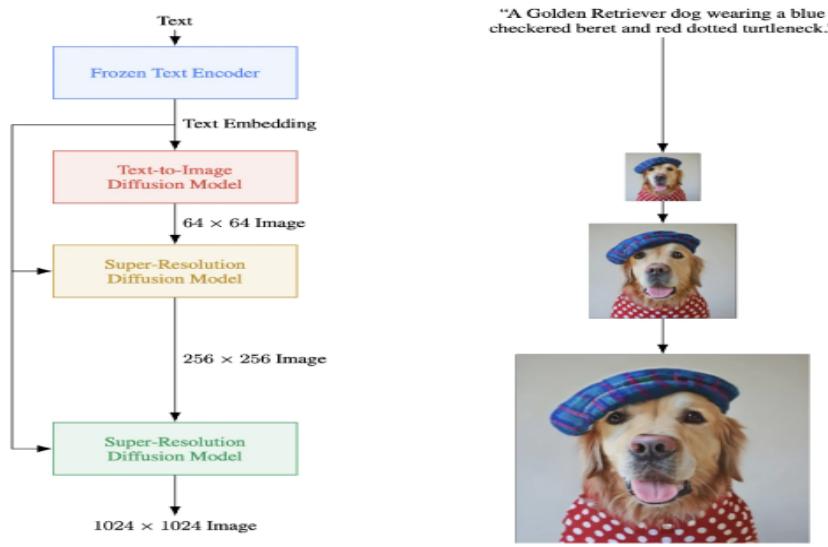


Fig. 8. Imagen 架构

- (2) 高效的 U-Net: 一种新的 U-Net，更简单，更节省内存，收敛速度更快。
- (3) DrawBenc: 用于正则化文本到图像模型的评估。



Fig. 9. Imagen 模型的生成示例

#### D. Stable Diffusion

虽然 DALL·E 2、Imagen 取得了惊人的成绩，但前者目前还处于测试阶段，只有部分用户可以免费使用，而后者根本没有向公众发布。直到 Stability AI 提出了一个新的模型 Stable Diffusion，并向全世界开放了源代码。

Stable Diffusion 的模型很小，即使是消费级 GPU 也足以运行。一些研究人员发现，直接在图像的像素空间上运行 Diffusion 速度慢、计算成本高。他们建议在一个强大的预训练自编码器的低维潜在空间上应用 Diffusion 模型，然后使用自编码器本身解码图像。这让 Diffusion 模型通过在感知上等同于图像空间的空间上生成样本，大大降低了计算复杂度，并让经过测试的自动编码器完成繁重的工作，以获得全尺寸的最终图像。从而实现了一个性能可与 DALL·E 2 相媲美但空间压缩得非常小的模型。



Fig. 10. 输入 “Triceratops programming on a MacBook in a startup office with other dinosaurs on the back, oil canvas”, Stable Diffusion 模型的生成结果

## VII. 实验

在本次实验中我们计划使用 mindspore 复现了 Diffusion models 中的三个经典工作：DDPM、GLIDE 和 Latent Diffusion。在这个部分中，我们将会介绍使用到的数据集，代码及超参数讲解，实验结果展示和关于模型的创新与改进的思考。

### A. Wukong 数据集

除了 DDPM 的简易实现是使用著名数据集 cifar10，其他两个模型都是用华为诺亚方舟实验室发布的一个名为「悟空」的大型中文跨模态数据集，其中包含来自网络的 1 亿个图文对。悟空数据集具有多样性和泛化性，是迄今为止最大的中文视觉语言跨模态数据集，涵盖了广泛的视觉和文本概念。虽然目前开源的英文大规模图文数据集较为丰富，如 CC12M、YFCC100M、LAION-400M，但是中文却始终缺少这样的大规模数据集可供自由下载研究。悟空作为首个中文领域开源的大规模图文数据集，填补了这样的空缺，能够加快中文跨模态预训练大模型的研究工作。

悟空数据集为了涵盖足够多样的视觉概念，原始数据的采集围绕 20 万个基础关键词进行，针对每一个基础关键词，通过输入搜索引擎，获取图片 URL 列表和相应的标题信息，用返回的图片及对应的文本来构建数据集。此外，为了平衡关键词对应样本的数量，每一个基础关键词所采集的样本最多保留 1000 个。

在此之后，数据集又经过一系列的过滤策略来得到最终的版本，最终共收集了 1.66 亿个图文对。具体的过滤策略分为基于图片的过滤和基于文本的过滤。下图 11 显示了悟空数据集中的一些样本。

**1) 基于图像的过滤:** 数据首先根据图像的大小和长宽比对进行过滤。只保留长或宽超过 200 像素且长宽比不超过 3 的图像。这种方式过滤掉了太小、太高或太宽的图像，以此来保证图片能够较为清晰地呈现视觉概念。因为这些图像在预训练期间经过上采样和方形裁剪等图像增强手段后，可能变成低分辨率。

**2) 基于文本的过滤:** 其次，为了使选择的样本具有对应图像的高质量中文描述，进一步根据图像所附文本的语言、长度和频率对数据进行过滤。研究者首先检查了语言和长度，保留了包含至少一个但少于 32 个汉字的句子。同时还会丢弃无意义的图像描述，例如「000.jpg」。之后，与太多图片配对的文字通常与图片内容无关，例如「查看源网页」(View source page)、「展开全文」(Expand text)、「摄影部落」(Photography community)。实际上，研究者将此阈值设置为 10，即丢弃掉在收集的整个语料库中出现超过 10 次的图文对。为了保护文本中出现的个人隐私，人名被替换为特殊标记「< 人名 >」，此外还构建了一个中文敏感词列表，包含敏感词的图文对也被丢弃。



Fig. 11. 悟空数据集的样本

应用上述过滤策略后，最终得到一个约 1 亿对的数据集。包括用于预训练的数据集“悟空”和用于模型测试的数据集“悟空测试”。下表 II 显示了数据集的统计量：数据集文本中有 20,442 个唯一 token，每个描述中的平均 token 数为 22。

	mage-text Pairs	Unique Tokens	Tokens per Caption		
			mean	std	median
Wukong	101,483,885	20,442	22	7	24
Wukong-Test	33,365	5,155	22	7	24

TABLE II

数据集统计

在下图 12 中可视化了数据集中单词（由一个或多个 token 组成）的分布，研究人员使用中文文本分词工具 Jieba 来截取单词并构建数据集的词云。



Fig. 12. 数据集的词云

## B. 代码及超参数讲解

开始着手进行这项任务之后，我们小组的想法是首先使用 mindspore 复现 Diffusion model 中最基础的 DDPM，然后再尝试实现目前领域中的前沿 sota。

前者利用了 github 中一个 pytorch 版本的实现, 观察代码可以得知, 模型部分涉及框架使用的主要是在 unet.py 和 diffusion.py, 我们根据 mindspore 官网的 api 映射表一一修改、调试, 最终成功实现了 mind-DDPM。

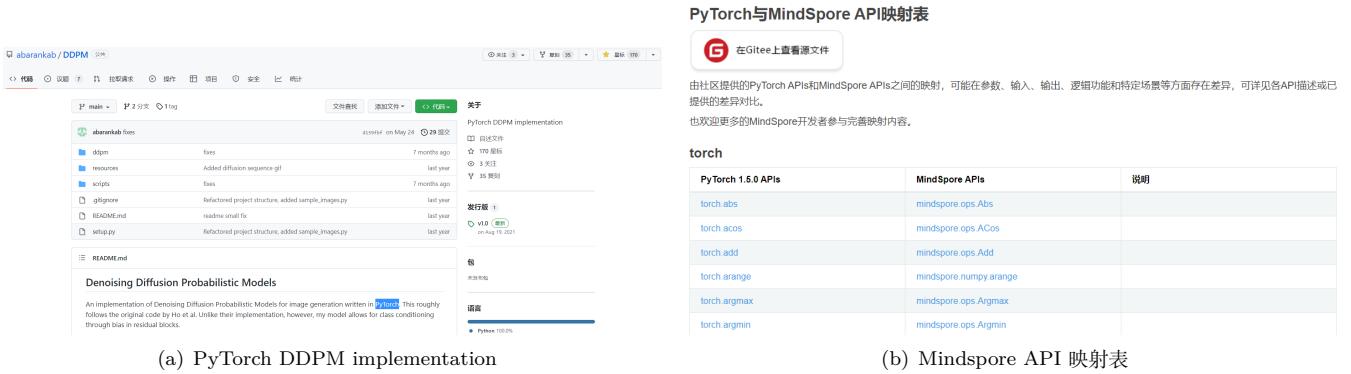


Fig. 13. DDPM 复现思路

后者在实现的过程中遇到了很多阻碍, 幸运地是我们通过 github 联系到了华为香港研究中心的工作人员, 对方非常热情地指导了我们, 并大方地开源了一系列基于 MindSpore 的经典 SOTA Diffusion 模型。

1) *DDPM*: 修改自用 PyTorch 编写的用于图像生成的去噪扩散概率模型的实现。内容大致遵循 DDPM 最早论文的原始代码。然而, 与它们的实现不同, 我们的模型允许通过残差块中的偏差进行类调节。

#### 默认超参数:

- `learning_rate`: 默认为  $2e-4$ , 这是优化器在训练期间使用的学习率。它确定优化器更新模型参数的步骤大小。
- `batch_size`: 默认为 128, 这是每次训练迭代中使用的示例数。增加批次大小可以加快训练速度, 但可能需要更多内存。
- `iterations`: 默认为 800000, 这是在训练期间运行的迭代数。每次迭代都涉及通过模型传递一批示例并更新模型的参数。
- `log_to_wandb`: 默认为 True, 这是一个布尔值, 表示是否将训练度量值记录到 Weights & Biases 平台。
- `log_rate`: 默认为 1000, 这是将训练度量值记录到控制台或 Weights & Biases 的频率。
- `checkpoint_rate`: 默认为 1000, 这是在训练过程中保存模型和优化器状态的频率。
- `log_dir`: 这是保存日志和检查点的目录。
- `project_name`: 这是要记录的 Weights & Biases 项目的名称。
- `run_name`: 这是 Weights & Biases 运行的名称。

2) *Taichu-GLIDE*: Taichu-GLIDE 是华为昇腾计算携手武汉人工智能研究院、中科院自动化所基于昇腾昇思全栈开发的中文文生图大模型 (紫东·太初系列模型之一), 该模型也采用了扩散模型 (Diffusion Model) 技术, 代码和预训练模型权重均对外进行开源。

#### 环境要求:

- 1) CANN (5.1.RC2 版本) 及其配套版本的驱动 (driver) 和固件 (firmware)
- 2) MindSpore 1.8.1 版本
- 3) ipython、regex、sentencepiece、blobfile、toolz、tqdm、pathlib2

训练分为两个阶段, 生成阶段和超分阶段。

生成阶段使用的 `batch_size` 是 10, 共训练 10 个 epoch, 初始学习率是  $1e-4$ , 最终学习率是  $1e-9$ 。这个阶段的目标是训练模型根据文本信息生成符合该文本信息描述的图片的能力, 可以使用多卡分布式训练。

超分阶段使用的 `batch_size` 是 2, 共训练 2 个 epoch, 学习率设置与生成阶段保持一致。这个阶段的目标是使模型具有超分辨率的功能, 在生成图片之后, 利用算法继续放大图片, 获得更多的细节。初始图片尺寸是 64\*64, 超分会先扩展到 256\*256, 最终扩展到 1024\*1024。

3) *Wukong-huahua*: Wukong-Huahua 是基于 Latent Diffusion 的中文文生图大模型，由华为诺亚团队携手中软分布式并行实验室，昇腾计算产品部联合开发。模型基于 Wukong dataset 训练，并使用昇思框架 (MindSpore)+ 昇腾 (Ascend) 软硬件解决方案实现。

环境要求：

- 1) CANN (6.0.RC1 版本) 及其配套版本的驱动 (driver) 和固件 (firmware)
- 2) MindSpore 1.9.0 版本
- 3) opencv-python、regex、omegaconf、ftfy

目前只开源了测试代码，要进行文图生成，可以运行 txt2img.py 或者直接使用默认参数运行 infer.sh。更高的分辨率需要更大的显存，对于 Ascend 910 芯片，我们可以同时生成 2 张 1024x768 的图片或者 16 张 512x512 的图片。

### C. 定性展示图片结果



Fig. 14. DDPM: cifat10 类图片生成



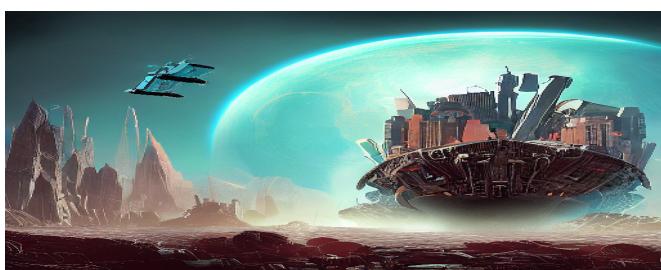
Fig. 15. Taichu: 一幅画着柯基的油画



(a) 城市夜景 赛博朋克 格雷格·鲁特科夫斯基



(b) 莫奈 撑阳伞的女人 月亮 梦幻



(c) 诺亚方舟在世界末日起航 科幻插画



(d) 来自深渊 风景 绘画 写实风格

Fig. 16. Wukong-Huahua: 效果展示

#### D. 定量计算指标 FID

Frechet Inception 距离得分 (Frechet Inception Distance score, FID) 是计算真实图像和生成图像的特征向量之间距离的一种度量。

FID 从原始图像的计算机视觉特征的统计方面的相似度来衡量两组图像的相似度，这种视觉特征是使用 Inception v3 图像分类模型计算的得到的。分数越低代表两组图像越相似，或者说二者的统计量越相似，FID 在最佳情况下的得分为 0.0，表示两组图像相同。

众所周知，在 Diffusion 模型大火之前，图像生成领域的 SOTA 一直是 GAN 的各项变种，恰好我们在人工智能原理实验的期末任务中复现了各类 GAN，使用 cifar10 中的 horse 图片训练并生成马的图片。为了能够横向对比我们复现的三个模型性能，我们统一生成“horse/马”来计算 FID，结果如下：



(a) DDPM: FID=219



(b) Wukong: FID=18



(c) Taichu: FID=10

实验编号	模型架构	FID
1	GAN	198
2	AAE	299
3	VAE	327
4	DCGAN	111
5	WGAN	241
6	LSGAN	172
7	DRAGAN	184
8	BEGAN	205
9	styleGAN 3	100
10	styleGANxl	22

(d) 代码: [github.com/Enderfga/Enderfga/tree/main/horse\\_generation](https://github.com/Enderfga/Enderfga/tree/main/horse_generation)**Fig. 17.** horse/马的生成示例及其 FID 分数

我们可以看到，虽然原始的 DDPM 生成的分数与 GAN 差距不大，但 SOTA 的 Diffusion 模型所生成的图片无论是质量还是大小，都远远超过了 GAN，即使当前我们使用的模型生成的图片仍然有“缺胳膊少腿”的现象，但大体效果已经非常让人满意了。

## E. 关于模型的创新与改进的思考

正所谓知其然，更要知其所以然，我们除了通过视频、文章、论文等资料来了解扩散模型，在动手复现 mindspore 上的扩散模型之后，我们继续查阅资料，思考如何创新与改进模型。

1) **DDPM**: 近几年，在 Transformer 的推动下，机器学习正在经历复兴。过去五年中，用于自然语言处理、计算机视觉以及其他领域的神经架构在很大程度上已被 transformer 所占据。不过还有许多图像级生成模型仍然不受这一趋势的影响，例如过去一年扩散模型在图像生成方面取得了惊人的成果，几乎所有这些模型都使用卷积 U-Net 作为主干。

我们调研了最新的论文之后，认为 U-Net 归纳偏置对扩散模型的性能不是至关重要的，并且可以很容易地用标准设计（如 transformer）取代。

这一发现表明，扩散模型可以从架构统一趋势中受益，例如，扩散模型可以继承其他领域的最佳实践和训练方法，保留这些模型的可扩展性、鲁棒性和效率等有利特性。标准化架构也将为跨领域研究开辟新的可能性。

2) **GLIDE**: 查阅资料得知，该模型具有以下的不足之处：

- 1) 容易将物体和属性混淆，例如不容易将红色和蓝色分辨出来。这可能来源于 CLIP 的 embedding 过程没有将属性绑定到物体上；并且 decoder 的重建过程也经常混淆属性和物体。
- 2) 对于将文本放入图像中的能力不足，模型常常将单词/词组拼写得很离谱。这个问题可能来源于 CLIP embedding 不能精确地从输入地文本提取出“拼写”信息。
- 3) 在生成复杂场景图片时，对细节处理有缺陷。这个可能来源于 decoder 的分层（hierarchy）结构，先生成  $64 \times 64$  的图像，再逐步上采样得到最终结果的。如果将 decoder 先生成的图像分辨率提高，比如从  $64 \times 64$  提升到  $128 \times 128$ ，那么这个问题可能可以缓解，但要付出更大计算量和训练成本的代价。

上述问题主要集中在 CLIP 上，并且 OpenAI CLIP 的预训练主要使用英文世界的图文数据，不能天然支持中文。即便是社区有研究者通过翻译的文本，蒸馏出多语言版本的 Multilingual-CLIP (mCLIP)，同样无法很好满足中文世界的需求，对于中文领域的文本理解不很到位。

故可以通过使用达摩院的 Chinese-CLIP 来优化模型，该项目针对中文领域数据以及在中文数据上实现更好的效果做了优化，多项实验数据表明，Chinese-CLIP 可以在中文跨模态检索取得最优表现。

3) **LDM**: Stable Diffusion v2.1 (基于 Latent Diffusion) 已经在 12 月 7 号正式更新，新版本使用了全新文本编码器 (OpenCLIP) 训练的强大的文本到图像模型，该模型由 LAION 在 Stability AI 的支持下开发，与早期的 V1 版本相比，它大大提高了生成图像的质量。此版本中的文本到图像模型可以生成默认分辨率为 512x512 像素和 768x768 像素的图像。

此外还包括了：

- Upscaler Diffusion 模型，可以将图像分辨率提高 4 倍。
- Depth2img 深度图像扩散模型，能够推断输入图像的深度信息，然后使用文本和深度信息生成新图像。
- text-guided 修复模型，能够很容易快速实现替换一个图像的一部分。

当然这些更多是应用层面的创新，在技术层面也有许多改进值得我们学习例如 Diffusion Models as Plug-and-Play Priors (NeurIPS): 一个不需要训练 time-dependent classifier 的 guidance 方法，Fast Solver for Guided Sampling of Diffusion Probabilistic Models(ICLR): 效果极佳的快速采样算法 .....

## VIII. 总结

Diffusion model 是 2022 年最热门的 AI 关键词之一，在 Stable Diffusion 等文图生成项目开源发布之后，其强大的生成效果和优美的数学形式引起了诸多研究者的关注，引领了技术社区中各种 Diffusion model + X 的风潮。决定以这个选题为我们小组的方向之后短短两个月，新的论文新的作品也如雨后春笋般涌现，让我们深刻地感受到计算机视觉这一领域的魅力与深度，希望在不久的将来我们也能够紧跟学术前沿，产出自己的作品。