《高级机器学习》小组项目报告

题目： Denoising Diffusion Probabilistic Model

组号： 24 组

小组成员姓名： 郜寿枫　　陈泽磊

小组成员学号： 231027163　231027162

评分：

1、 语言与排版（25 分）：

2、 问题难易程度（25 分）：

3、 实践掌握程度（50 分）：

总分：

目录

一、 背景问题及分析

给定一个训练数据$x:=\{x_i: i=1\ldots N\}.i.i.d\sim p^*(x)$，生成模型旨在寻找数据的潜在概率分布$p^*(x)$，一般可以通过参数（分析）或无参（GAN[1]）或score-matching[2]（基于内核）方法来用一个概率分布$p(x)$来近似$p^*(x)$，之后从中采样来生成新的数据样例，即$x\sim p(x)$。

因此生成式模型要解决的问题可简单划分为两块，分别是概率估计和采样。对于参数估计方法来说就分别对应分布参数的估计和从由该分布参数构成的分布中采样。

基于最大似然估计原则的参数估计法是常用的解决方法。

$$\underset{\theta}{\operatorname{argmax}}\, \mathrm{E}_{\mathrm{p}^*(\mathrm{x})}[\log \mathrm{p}(\mathrm{x})] \approx \underset{\theta}{\operatorname{argmax}} \sum_{\mathrm{i}=1}^{\mathrm{N}} \log \mathrm{p}(\mathrm{x}_\mathrm{i}) \text{ or } \underset{\theta}{\operatorname{argmin}}\, \mathrm{KL}(\mathrm{p}\|\mathrm{p}^*) \quad (1)$$

二、 解决思路及方法

（一）确定形式的概率估计

指定$p(x;\theta)$的具体形式，例如高斯分布$p(x;\theta):=\mathcal{N}(x;\mu,\sigma^2),\theta=\{\mu,\sigma^2\}$

参数估计（对θ的推理）：

$$\mu=\frac{1}{N}\sum_{i=1}^{N}x_i$$

$$\sigma^2=\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2$$

采样：

$$x=\mu+\sigma^2\epsilon,\epsilon\sim\mathcal{N}(0,1)$$

缺点&概括

简单分布形式在表示$p^*(x)$上缺少灵活性和丰富性，我们可能需要更复杂的分布形式的假设。

（二）更加复杂形式的概率估计

以下内容参考自[3]

采用具有更复杂形式的$p(x;\theta)$，例如混合高斯分布

$$\mathrm{p(x)}:=\sum_{\mathrm{z}}\mathrm{p(z)p(x|z)}=\sum_{\mathrm{k=1}}^{\mathrm{K}}\pi_{\mathrm{k}}\mathrm{N(x|}\,\mu_{\mathrm{k}},\,\sigma_{\mathrm{k}}^{2}),\ \theta=\{\pi_{\mathrm{k}},\,\mu_{\mathrm{k}},\,\sigma_{\mathrm{k}}^{2}|\mathrm{k}=1\ldots\mathrm{K}\}$$

这里有两种方法可以得到优化下界

（1）基于重要性采样

$$log\,p(x)=log\sum_{z}p(x,z)=\log\sum_{z}p(z|x)\frac{p(x,z)}{p(z|x)}=\log E_{p(z|x)}\left[\frac{p(x,z)}{p(z|x)}\right]$$

由 Jensen 不等式，可得

$$log\,p(x)\geq\sum_{z}p(z|x;\theta^{old})\,log\frac{p(x,z;\theta)}{p(z|x;\theta^{old})}$$

$$=\sum_{z}p(z|x;\theta^{old})\,log\,p(x,z;\theta)+\sum_{z}-p(z|x;\theta^{old})\,log\,p(z|x;\theta^{old})$$

$$=\sum_{z}p(z|x;\theta^{old})\,log\,p(x,z;\theta)+const,const\,w.r.t\,\theta,const\geq0$$

$$\geq\sum_{z}p\big(z\big|x;\theta^{old}\big)\,log\,p(x,z;\theta)=:\mathcal{L}\,(\theta^{old},\theta)$$

（2）基于贝叶斯规则

$$\log p(x,z|\theta)=\log p(x|\theta)+\log p(z|x,\theta^{old})$$

$$\Rightarrow log\,p(x|\theta)=log\,p(x,z|\theta)-log\,p(z|x,\theta^{old})$$

$$\Rightarrow log\,p(x|\theta)=\sum_{z}p(z|x,\theta^{old})\,log\,p(x|\theta)$$

$$=\sum_{z}p(z|x,\theta^{old})\,log\,p(x,z;\theta)+\sum_{z}-p(z|x;\theta^{old})\,log\,p(z|x;\theta^{old})$$

$$=\sum_{z}p(z|x;\theta^{old})\,log\,p(x,z;\theta)+const,const\,w.r.t\,\theta,const\geq0$$

$$\geq\sum_{z}p(z|x;\theta^{old})\,log\,p(x,z;\theta)=:\mathcal{L}\,(\theta^{old},\theta)$$

参数估计：基于 EM 算法训练（θ的推理）：

以迭代的方式执行以下步骤

$$E\,Step:\ evaluate\,p(z|x;\theta^{old})$$

$$M\,Step:\ \theta^{old}\leftarrow\underset{\theta}{\mathrm{argmax}}\,\mathcal{L}\,(\theta^{old},\theta)$$

采样：

$$z\sim p(z),\text{且}z_k=1,z_{\neq k}=0$$

$$x=\mu_k+\sigma_k^2\epsilon,\epsilon\sim\mathcal{N}(0,1)$$

缺点：

该分布形式对于$p^*(x)$仍然在表示能力上有所受限，主要来源于$p(z|x;\theta^{old})$的形式受限

推广：

这里我们在生成模型中引入一个潜在变量 Z，这样我们就可以将其扩展到更一般的情况。本质上是用有向概率图模型将潜在的概率分布展开，然后模拟在该图中数据的生成过程；

可得到推广的 EM 算法：

引入在潜在变量上定义的分布 q(z)来替换 p(z|x)

$$\log p(x|\theta) = \sum_z q(z)\log\frac{p(x,z|\theta)}{q(z)} + \sum_z -q(z)\log\frac{p(z|x,\theta)}{q(z)} = L(q,\theta) + KL(q||p) \quad (2)$$

参数估计（θ推理）：

以迭代的方式执行以下步骤

$$E\ Step:\ \underset{q(z)}{\operatorname{argmax}}\mathcal{L}(q,\theta)\ \mathcal{L}(\theta^{old},\theta) \Rightarrow q(z) = p(z|x,\theta^{old})$$

$$M\ Step:\ \theta^{old} \leftarrow \underset{\theta}{\operatorname{argmax}}\mathcal{L}(q,\theta)$$

采样：

$$z \sim p(z)$$

$$x \sim p(x|z)$$

缺点&概括：

分布中θ仍然需要谨慎设置，这意味着我们可能仍然受到分布形式的限制。

$p(z|x,\theta^{old})$在这种情况下很难处理，无法得到可解析的解。

例如，当我们使用神经网络来模拟$p(z|x)$时，网络的非线性和复杂性使得它在解析上变得困难，无法达到闭解析式。

（三）变分推理自编码器 VAE[4]

利用确定性的方式估计后验分布：$q(z) \to p(z|x)$

将θ也看作潜在变量，对于贝叶斯模型来说，q(z)也是可追踪的。

$$\log p(x) = \int q(z|x,\phi)\log\frac{p(x,z|\theta)}{q(z|x,\phi)}dz + \int -q(z|x,\phi)\log\frac{p(z|x,\theta)}{q(z|x,\phi)}dz$$

$$= \mathcal{L}(q,\theta) + KL(q||p) \quad (3)$$

where

$$\mathcal{L}(q,\theta) = \int q(z|x,\varphi)\,log\frac{p(x|z,\theta)}{q(z|x,\varphi)}dz + \int q(z|x,\varphi)\,log\frac{p(z|\theta)}{q(z|x,\varphi)}dz$$

$$= E_q[\,log\frac{p(x|z,\theta)}{q(z|x,\varphi)}] - KL(q(z|x,\varphi)||p(z|\theta))$$

这引入了变分自编码器的基本结构

编码器

$$x^{(i)}\sim p(x).i.i.d \xrightarrow{recognition} z^{(i)}\sim q(z|x=x^{(i)},\varphi)$$

$$\rightarrow \underset{\varphi}{\operatorname{argmin}}\, KL\big(q(z|x,\varphi)||p(z|\theta)\big)$$

解码器

$$z^{(i)} \xrightarrow{generation} \bar{x}^{(i)}\sim p(x|z=z^{(i)},\theta) \rightarrow \underset{\varphi,\theta}{argmin}\, d\big(x^{(i)},\bar{x}^{(i)}\big)$$

$$= \underset{\varphi,\theta}{argmin} - E_q\left[\log\frac{p(x|z,\theta)}{q(z|x,\varphi)}\right]$$

（四）扩散概率模型[5]

采用序列化的潜变量

$$z = (x_{1:T}), x_0 = x$$

这意味着我们可以使用马尔科夫链来将变分自编码器扩展成如下两个过程

前向过程：扩散过程，缓慢破坏数据分布

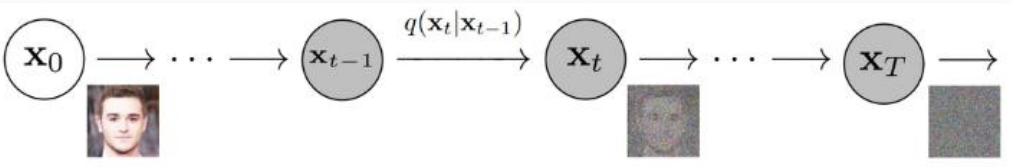

图 1

$$q(x_{0:T}|\phi) = q(x_0)q(x_{1:T}|x_0,\phi) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1},\phi) \quad (4)$$

$$q\big(x_t|x_{t-1},\phi\big) = \mathcal{N}(x_t;\sqrt{\alpha_t}\boldsymbol{x_{t-1}},(1-\alpha_t)\boldsymbol{I}),\ \phi = \{\beta_1,\ldots,\beta_T\},\beta_t = 1-\alpha_t$$

反向过程：生成过程，缓慢恢复数据分布

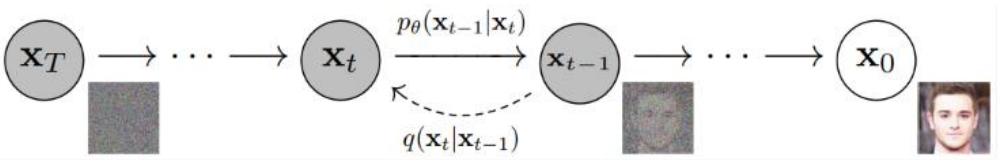

图 2

$$p(x_{0:T}|\theta) = p(x_{1:T}|\theta)p(x_0|x_{1:T},\theta) = p(x_T)\prod_{t=1}^{T} p(x_{t-1}|x_t,\theta), p(x_T)\sim\mathcal{N}(0,I) \quad (5)$$

参数估计：

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1,\dots,T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$$
6: **until** converged

图 3

$$\text{MML objective: } \underset{\phi,\theta}{\operatorname{argmax}}\, E_{q(x_0)}[\log p(x_0)],\ x_0\sim q(x_0).\,i.\,i.\,d \quad (6)$$

利用在“（二）更加复杂形式的概率估计”中相同的技巧，可以得到优化变分下界

$$\log p(x_0) \geq E_{q\left(x_{1:T}\middle|x_0,\phi\right)}\left[\log\frac{p\left(x_0,x_{1:T}\middle|\theta\right)}{q\left(x_{1:T}\middle|x_0,\phi\right)}\right] \quad (7)$$

$$\Rightarrow MML\ objective \propto \underset{\varphi,\theta}{argmax}\, E_{q(x_{0:T})}\left[log\frac{p(x_0,x_{1:T}|\theta)}{q(x_{1:T}|x_0,\varphi)}\right]$$

由于参数φ既为可学习的也可以是固定的，至于为什么它是可固定的将在后面给出解释，为方便目标优化函数的推到，现在假定它是固定的，目标函数可以进一步缩减为

$$\underset{\theta}{argmax}\, E_{q(x_{0:T})}\left[log\frac{p(x_0,x_{1:T}|\theta)}{q(x_{1:T}|x_0,\varphi)}\right]$$

$$=\underset{\theta}{argmax}\, E_{q(x_{0:T})}\left[log\, p(x_T)+\sum\nolimits_{t\geq 1} log\frac{p(x_{t-1}|x_t,\theta)}{q(x_t|x_{t-1})}\right]$$

基于马尔可夫假设，该目标可以分解为各时间点优化目标的和

$$\Rightarrow E_{q(x_{0:T})}\left[log\, p(x_T)+\sum\nolimits_{t>1} log\frac{p(x_{t-1}|x_t,\theta)}{q(x_t|x_{t-1})}+log\frac{p(x_0|x_1,\theta)}{q(x_1|x_0)}\right]$$

再利用贝叶斯公式，可得

$$q(x_t|x_{t-1})=q(x_t|x_{t-1},x_0)=\frac{q(x_{t-1}|x_t,x_0)q(x_t|x_0)}{q(x_{t-1}|x_0)}$$

$$\Rightarrow q(x_{t-1}|x_t,x_0)=\frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}$$

代入可得

$$\Rightarrow E_{q(x_{0:T})}\left[log\, p(x_T)+\sum\nolimits_{t>1} log\frac{p(x_{t-1}|x_t,\theta)}{q(x_{t-1}|x_t,x_0)}+log\frac{p(x_0|x_1,\theta)}{q(x_1|x_0)}\right]$$

$$\sum\nolimits_{t>1} log\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}=log\, q(x_1|x_0)+\sum\nolimits_{t=3}^{T-1} log\frac{q(x_t|x_0)}{q(x_t|x_0)}-log\, q(x_T|x_0)$$

$$\Rightarrow E_{q(x_{0:T})}\left[log\frac{p(x_T)}{q(x_T|x_0)}+\sum\nolimits_{t>1} log\frac{p(x_{t-1}|x_t,\theta)}{q(x_{t-1}|x_t,x_0)}+log\, p(x_0|x_1,\theta)\right]$$

最终得到 KL 散度形式的目标函数

$$=\mathrm{E}_{q(x_0)}[\mathrm{KL}(q(x_T|x_0)||p(x_T))]+\sum\nolimits_{t>1}\mathrm{E}_{q(x_0,x_t)}[\mathrm{KL}(q(x_{t-1}|x_t,x_0)||p(x_{t-1}|x_t,\theta))]+\mathrm{E}_{q(x_0,x_1)}[\log p(x_0|x_1,\theta)]\quad(8)$$

由于所有这些分布都以x_0为条件，使得其对应的概率分布具有解析式，而不需要使用蒙特卡罗方法进行估计。

注意从$q(x_0,x_1)$中采样可以分解为

$$x_0\sim q(x_0),x_t\sim q(x_t|x_0)$$

由于$p(x_T)$固定为正态分布，因此优化时可以忽略第一项，得到进一步缩减的目标函数

$$\Rightarrow\sum\nolimits_{t>1}\mathrm{E}_{q(x_0,x_t)}[\mathrm{KL}(q(x_{t-1}|x_t,x_0)||p(x_{t-1}|x_t,\theta))]+\mathrm{E}_{q(x_0,x_1)}[\log p(x_0|x_1,\theta)]\quad(9)$$

最后一项需要转换为离散的形式来匹配图片离散的表示形式，即[−1, 1]

→ {0, 1, 2, ⋯, 255}

这相当于累积以 x 为中心、宽度为 2/255 的小区间的概率。

$$p(x_0|x_1, \theta) = \prod_{i=1}^{D} \int_{\delta_-(x_0^i)}^{\delta_+(x_0^i)} N(x; \mu_\theta^i(x_1, I), \sigma_1^2) dx \quad (10)$$

$$\text{其中} \delta_+(x_0^i) = \begin{cases} \infty, & x = 1 \\ x + 1/255, & x < 1 \end{cases}, \delta_-(x_0^i) = \begin{cases} -\infty, & x = -1 \\ x - 1/255, & x > -1 \end{cases}$$

$$D = H * W$$

与 VAE 的最后阶段相同，我们在生成的样本和数据样本之间估计重构误差。

从这个角度来看，我们可以将每一步的优化视为一个特殊的 VAE，从而产生总共 T 个“迷你 VAE”；因此，每个时间步都对应一个重构的任务，这个重构任务负责还原该时刻被噪声破坏的样本。不同在于，这些时间步的估计目标并不是重构的中间时刻样本，而是在该时刻的逆向转移概率分布。

通过进一步假设反向转移概率与前向转移概率有相同的方差，可得到反向转移概率参数化的解析式，此时分布之间的度量等价于均值之间的度量。

$$p(x_{t-1}|x_t, \theta) = \begin{cases} \mathcal{N}\left(f_\theta(x_1), \sigma_1^2 I\right), & t = 1 \\ q\left(x_{t-1}|x_t, f_\theta(x_t)\right), & t > 1 \end{cases} \quad (11)$$

其中$f_\theta(x_t)$是对x_0的预测，借助 $q(x_t|x_0)$的参数化可以得到

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\overline{\alpha}_t} x_0, \left(1 - \overline{\alpha}_t\right) I\right), \overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s \quad (12)$$

$$x_0 = \frac{x_t - \sqrt{1 - \overline{\alpha}_t} \cdot \epsilon_t}{\sqrt{\overline{\alpha}_t}}, \epsilon_t \sim \mathcal{N}(0, I)$$

$$f_\theta(x_t) = \frac{x_t - \sqrt{1 - \overline{\alpha}_t} \cdot \epsilon_\theta(x_t, t)}{\sqrt{\overline{\alpha}_t}}$$

进一步，借助$q(x_{t-1}|x_t, x_0)$与$q(\boldsymbol{x_t}|\boldsymbol{x_0})$的贝叶斯等式

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1}) q(x_{t-1}|x_0)}{q(x_t|x_0)} = \mathcal{N}\left(x_{t-1}; \widetilde{\mu}_t(x_t, x_0), \widetilde{\beta}_t I\right) \quad (13)$$

$$where \tilde{\boldsymbol{\mu}}_t(x_t, x_0) := \frac{\sqrt{\overline{\alpha}_{t-1}} \beta_t}{1 - \overline{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1 - \overline{\alpha}_{t-1})}{1 - \overline{\alpha}_t} x_t \quad and \quad \tilde{\beta}_t := \frac{1 - \overline{\alpha}_{t-1}}{1 - \overline{\alpha}_t} \beta_t$$

将x_0替代为$f_\theta(x_t)$，便得到了$p(x_{t-1}|x_t, \theta)$的参数化解析式

$$p(x_{t-1}|x_t, \theta) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t), \widetilde{\beta}_t I\right) \quad (14)$$

$$\boldsymbol{\mu}_\theta(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$$

因此均值之间的误差估计进一步转换为噪声之间的误差估计

$$\underset{\theta}{\operatorname{argmin}}\, E_{x_\theta, \epsilon_t}[KL(q(x_{t-1}|x_t, x_0)||p(x_{t-1}|x_t, \theta))] \quad (15)$$

$$\Rightarrow \underset{\theta}{\operatorname{argmin}}\, E_{x_0, \epsilon_t}\left[\|\epsilon_\theta(x_t, t) - \epsilon_t\|_2^2\right]$$

事实上，我们最终发现，整个目标函数是建立在$q(x_t|x_0)$上的，因此只要$q(x_t|x_0)$不变，那么目标函数就不变。

采样:

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

图 4

$$x_T \sim p(x_T)$$

$$x_t \sim p(x_t|x_{t+1}): x_t = g(z, x_{t+1}, \theta), \begin{cases} z \sim p(z), t > 1 \\ z = 0, t = 1 \end{cases}$$

三、 类比分析

（一）与马尔科夫蒙特卡洛（MCMC[6]）算法的联系

上述介绍是从优化的角度进行的。但还有另一种观点可以追溯到 MCMC 的核心思想。

（1）扩散过程——具有平稳分布的马尔可夫链

在对转移概率和平稳分布施加弱限制（ergodic）的情况下，在马尔科夫链上不断扩散的目标分布可以收敛到简单的平稳分布，例如正态高斯分布

$$\pi_j = \sum_k \pi_k p_{kj} \tag{16}$$

（2）反向过程——从平稳分布中恢复到原分布

从平稳分布中恢复的关键是马尔可夫链是可逆的，当马尔科夫链满足细致均衡（detailed balance）时，马尔科夫链是可逆的

$$\pi_j p_{ji} = \pi_i p_{ij} \tag{17}$$

另外，平稳分布的充分条件是细致均衡，这也保证了扩散过程能够收敛到一个平稳分布。

$$\sum_j \pi_j p_{ji} = \sum_j \pi_i p_{ij} = \pi_i \tag{18}$$

并且，根据 Metropolis 算法可知当转移概率分布是对称时，即两者有相同的分布，细致均衡是成立的。

这里我们或许可以对论文中最终的回归损失函数形式展开一定的解释。

也就是说，diffusion 的损失函数可能也构建了一个可逆的马尔可夫链，它可能具有每个时间步都保持正向转移概率分布和反向转移概率分布之间的对称性，但也可能寻找到了其他的方式来符合一个可逆的马尔科夫链，正如利用模型能够发现更高效的矩阵乘法那样。但是鉴于不以x_0的正反向转移概率分布的解析式是未知的，这种对称性没有办法用符号来证明。

（3）和 MCMC 的不同点

在 Metropolis-Hasting 算法中，采样是通过可分析地计算移动概率来实现的

$$\alpha(x,y) = min\left\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\right\} \Rightarrow u \sim U(0,1) \begin{cases} u \le \alpha(x,y), reject\ y \\ u > \alpha(x,y), reject\ y,\ reuse\ x \end{cases} \tag{19}$$

因为移动概率是保持细致平衡的关键

$$\pi(x)q(x,y)\alpha(x,y) = \pi(y)q(y,x)\alpha(y,x) \quad (20)$$

而在 Diffusion 中，MC 在训练后已经满足了细致平衡，因此可以直接沿学习得到的马尔科夫链迭代采样来从目标分布中得到生成的样本。

（二）与 GAN 的联系

扩散模型（DDPM）和生成式对抗模型（GAN）分别是描述概率模型和隐式概率模型的经典实例。为了估计数据的潜在分布，DDPM 在最大似然假设下，通过模拟层级的数据生成过程来用一个参数化的近似分布来逼近它；而 GAN 以对比方式学习，直接从中采样，而没有显示地重构潜在分布。 所以 DDPM 的生成结果天然就被赋予了多样性，但生成样本的质量不足，而 GAN 则恰恰相反。因此它们两者构成了一种生成多样性和生成质量的平衡，这也就意味着存在一些方法能够将两者结合来达到比较好的综合效果。

其实对于两者我们都可以发现，采样都是从随机噪声开始的，而 DDPM 在每一步采样都引入了这种随机性，导致了生成的多样性较高，但也带来了采样效率低的问题。

因此，通过减少这种随机性，同时保持优化目标不变，我们可以将其采样过程修改为类似于 GAN 的采样过程。

四、 存在问题及改进

DDPM 的问题：推理缓慢且效率低下

（一）推理时间步过于漫长

为了实现$p(x_T)\sim\mathcal{N}(0,1)$，我们需要一个较大的采样时间步 T：

$$q(x_T|x_0)=N(x_T;\sqrt{\overline{\alpha}_T}x_0,(1-\overline{\alpha}_T)I)$$

$$\Rightarrow x_T=\sqrt{\overline{\alpha}_T}x_0+(1-\overline{\alpha}_T)\epsilon,\epsilon\sim N(0,1)$$

$$x_T\to\epsilon\Rightarrow\overline{\alpha}_T\to 0\Rightarrow T\to+\infty$$

通过采用固定的前向概率分布参数φ便可以满足这个条件，这也是为什么可以把φ设置为固定值的原因。

由于 DDPM 的学习目标——逆向转移分布是建立在基于马尔可夫假设的前向过程上的

$$\mathrm{q}(\mathrm{x}_{t-1}|\mathrm{x}_t,\mathrm{x}_0)=\frac{\mathrm{q}(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{x}_0)\mathrm{q}(\mathrm{x}_{t-1}|\mathrm{x}_0)}{\mathrm{q}(\mathrm{x}_t|\mathrm{x}_0)}=\frac{\mathrm{q}(\mathrm{x}_t|\mathrm{x}_{t-1})\mathrm{q}(\mathrm{x}_{t-1}|\mathrm{x}_0)}{\mathrm{q}(\mathrm{x}_t|\mathrm{x}_0)}\qquad(21)$$

因此，作为它的估计项的反向分布 $p(x_{t-1}|x_t)$，同样需要满足马尔可夫假设，这也导致了逆向采样的过程同样需要 T 个时间步

$$\mathrm{p}(\mathrm{x}_{t-1}|\mathrm{x}_t,\theta)=\begin{cases}\mathrm{N}(\mathrm{f}_\theta(\mathrm{x}_1),\sigma_1^2\mathrm{I}),\mathrm{t}=1\\ \mathrm{q}(\mathrm{x}_{t-1}|\mathrm{x}_t,\mathrm{f}_\theta(\mathrm{x}_t)),\mathrm{t}>1\end{cases}\qquad(22)$$

$$f_\theta(x_t)=\frac{x_t-\sqrt{1-\overline{\alpha}_t}\cdot\epsilon_t(x_t)}{\sqrt{\overline{\alpha}_t}}$$

因此，为了从前向过程中解锁采样过程，需要将前向过程改造成一个非马尔可夫过程

扩散隐模型（DDIM[7]）正是基于这个想法实现了对 DDPM 的采样的加速

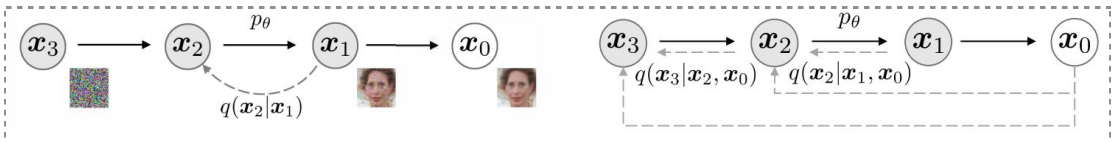

图 5

$$\mathrm{q}_\sigma(\mathrm{x}_{t-1}|\mathrm{x}_t,\mathrm{x}_0)=\frac{\mathrm{q}_\sigma(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{x}_0)\mathrm{q}_\sigma(\mathrm{x}_{t-1}|\mathrm{x}_0)}{\mathrm{q}_\sigma(\mathrm{x}_t|\mathrm{x}_0)},\mathrm{q}_\sigma(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{x}_0)\neq\mathrm{q}_\sigma(\mathrm{x}_t|\mathrm{x}_{t-1})\qquad(23)$$

论文发现 DDPM 的目标函数和$q(x_t|x_0)$是高度绑定的，因此只要$q(x_t|x_0)$不变，就能够复用 DDPM 的目标函数和训练过程。最终只需要改变 DDPM 的采样算法即可实现加速。

在假设前向过程不满足马尔可夫假设的前提下，作者引入了一个参数化的逆向分布$q_\sigma(\mathrm{x_{t-1}}|\mathrm{x_t},\mathrm{x_0})$，在保持$q(x_t|x_0)$不变的前提下，通过构建等式方程组来求得其解析式。

$$q_\sigma(\mathrm{x_{t-1}}|\mathrm{x_t},\mathrm{x_0}) = N(sx_t + mx_0,\ \sigma_t^2 I)$$

$$x_{t-1} = sx_t + mx_0 + \sigma_t\epsilon_1$$

利用条件$q(x_t|x_0) = N(x_t;\sqrt{\overline{\alpha}_t}x_0,(1-\overline{\alpha}_t)I)$

$$\mathrm{x_t} = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon_2$$

代入可得

$$\Rightarrow x_{t-1} = (s\sqrt{\alpha_t} + m)x_0 + \sigma_t\epsilon_1 + s\sqrt{1-\alpha_t}\epsilon_2$$
$$= (s\sqrt{\alpha_t} + m)x_0 + \sqrt{\sigma_t^2 + s^2(1-\alpha_t)}\epsilon$$

再利用条件$q(x_{t-1}|x_0) = N(x_t;\sqrt{\overline{\alpha}_{t-1}}x_0,(1-\overline{\alpha}_{t-1})I)$

$$\Rightarrow x_{t-1} = \sqrt{\alpha_{t-1}}x_0 + \sqrt{1-\alpha_{t-1}}\epsilon$$

可得等式方程组

$$\Rightarrow \begin{cases} s\sqrt{\alpha_t} + m = \sqrt{\alpha_{t-1}} \\ \sqrt{\sigma_t^2 + s^2(1-\alpha_t)} = \sqrt{1-\alpha_{t-1}} \end{cases} \Rightarrow \begin{cases} \mathrm{s} = \sqrt{\dfrac{1-\alpha_{t-1}-\sigma_t^2}{1-\alpha_t}} \\ \mathrm{m} = \sqrt{\alpha_{t-1}} - \sqrt{\dfrac{1-\alpha_{t-1}-\sigma_t^2}{1-\alpha_t}}\sqrt{\alpha_t} \end{cases}$$

最终得到参数化的目标逆向分布

$$\Rightarrow \mathrm{q}_\sigma(\mathrm{x_{t-1}}|\mathrm{x_t},\mathrm{x_0}) = \mathrm{N}\left(\sqrt{\frac{1-\alpha_{t-1}-\sigma_t^2}{1-\alpha_t}}(\mathrm{x_t} - \sqrt{\alpha_t}\mathrm{x_0}) + \sqrt{\alpha_{t-1}}\mathrm{x_0},\ \sigma_t^2\mathrm{I}\right) \quad (24)$$

可以使用相同的结构导出用于估计的反向分布

$$\mathrm{p}_\sigma(\mathrm{x_{t-1}}|\mathrm{x_t},\ \theta) = \mathrm{q}_\sigma(\mathrm{x_{t-1}}|\mathrm{x_t},\mathrm{f}_\theta(\mathrm{x_t})) \quad (25)$$

$$= \mathcal{N}\left(\sqrt{1-\overline{\alpha}_{t-1}-\sigma_t^2}\epsilon_\theta(x_t)+\sqrt{\alpha}_{t-1}\frac{x_t-\sqrt{1-\overline{\alpha}_t}\epsilon_\theta(x_t)}{\sqrt{\alpha}_t},\ \sigma_t^2 I\right)$$

便得到了 DDPM 的推广形式，并且在以下情况成立时退化为 DDPM：

$$\sigma_t=\sqrt{\frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}}\sqrt{1-\frac{\overline{\alpha}_t}{\overline{\alpha}_{t-1}}},\ p_\sigma(x_{t-1}|x_t,\theta)=p(x_{t-1}|x_t,\theta)$$

此时前向转移分布又变回了一个马尔科夫链

而当σ_t=0 时，每个时间步的随机性变消失来了，从而产生了类似于 GAN 那样的采样过程，并同时具备采样一致性，即相似的初始噪声能够得到相似的生成样本。

由于采样过程的确定性大大增加，因此可以引入跨步式的采样方式来加速采样，通过将$q_\sigma(x_{t-1}|x_t,x_0,f_\theta(x_t))$推广为$q_\sigma(x_s|x_k,f_\theta(x_k))),s\le k-1$，便能够在采样时仅需要采样时间上的一个子序列，从而实现采样的加速。

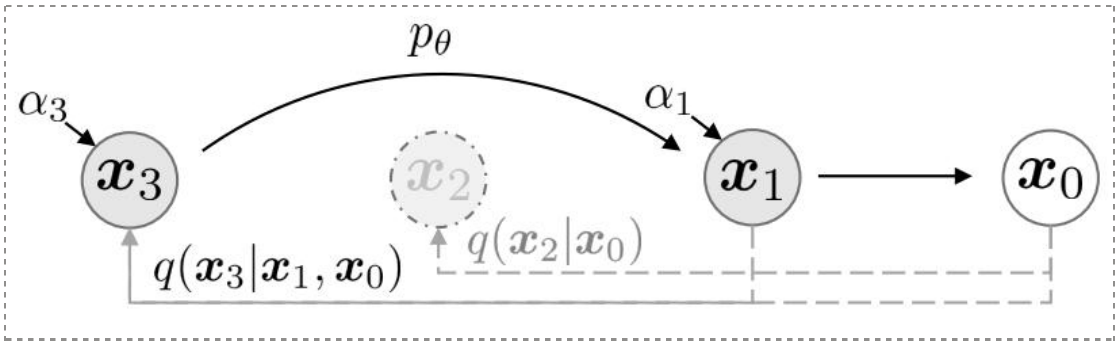

图 6

（二）像素级别的样本生成消耗巨大

DDPM 的训练过程和采样过程都是直接在像素级别上进行的，这导致了以下几个问题：

（1） 内存消耗巨大且执行效率底下，最终的结果是训练和采样的消耗都很大

（2） 像素级别的学习会使得模型过于关注那些微小的不可感知的像素数值的变化，而忽略了语义的正确性，最终导致采样的质量不够高

潜在扩散模型（LDM[8]）：通过引入变分自编码器，将原本像素级别的前向

和后向过程迁移到维度更低，且更贴近语义表示的潜在空间内。并且通过自编码器使得能够将采样结果投射到更高维度上，从而适用于产生具有更高分辨率的样例。

五、 论文中的实验效果

（一）CIFAR10 逐时间步采样的结果

图 7 从左到右依次为时间 T 到时间 0，时间 T 为随机噪声，时间 0 为最终生成数据样本

（二）不同起始时间步采样的差异

利用$p(x_0|x_t,\theta)$的分布特性，实际上每个时间步都能够直接采样得到对应的最终样本

$$x_0 \sim p(x_0|x_t,\theta)$$

图 8

当起始时间步越小时，采样得到的样本之间的相似程度越高，且采样一致性的程度也越高

当起始时间步越大时，只有偏全局的特征被保留下来，最终生成的样本的具有很强的多样性。

（三）插值生成

类似于 VAE，Diffusion 本质上也构建了一个平滑的编码空间，因此可通过对不同数据样本对应某个时间步的编码作凸组合得到新的编码，经过反向解码过程，可生成融合两个样本特征的新样本

$$x_0, x_0^{'} \sim q(x_0) \xrightarrow{Diffusion} x_t, x_t^{'} \sim q(x_t|x_0)$$

$$\overline{x}_t = (1-\lambda)x_t + \lambda x_t^{'} \xrightarrow{Reversal} \overline{x}_0 \sim p(x_0|\overline{x}_t)$$

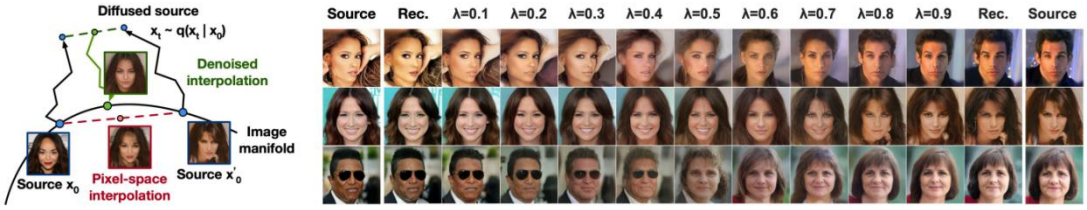

图 9

其中 Rec 表示对单源样本的重构结果

不同时间长度下的插值生成

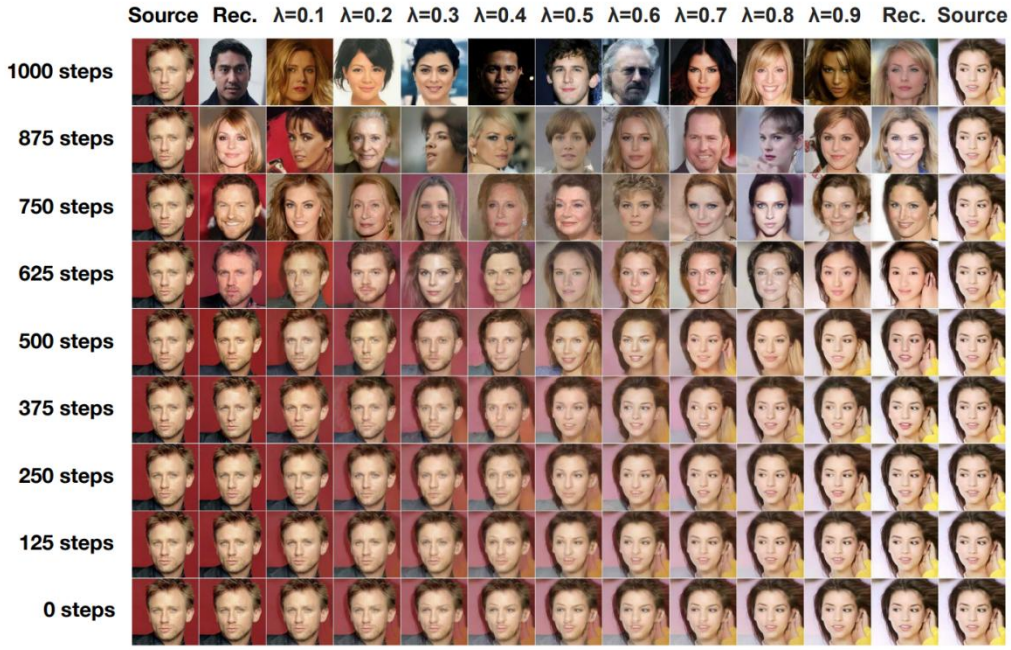

图 10

可以发现，时间步越大，生成样本的多样性越大，甚至于在某种程度上已经完全偏离了原始的语义，这是由于所有时间步之间的随机性相互叠加导致的结果。

六、 实验结果及分析

（一）实验设置

（1） 实验构想

由于目前的 diffusion 模型往往训练消耗大，且采样时间步长。出于训练资源紧缺和快速实验的考究需要采用一个简化版本的 diffusion 模型，且其在推断时的采样能够加速，且仅在一些小批量的简易数据集上作实验。

因此本次实验整体参考自 Keras 官方的简易版本的 DDIM 模型实验。

（2） 实验环境

显卡：Nvidia 2060s

系统：Ubuntu 22.04.2

机器学习框架：Pytorch 1.12.1

（3） 数据设置

本实验使用 Oxford Flowers 102[9]数据集，该数据集包含了 8000 张从自然界中采集的花朵图片。鉴于原数据集中大多数样本位于测试样本集中，因此需要重新划分测试和训练集。在新的划分方案中，训练集占八成，测试集占二成，并对所有样本做中心裁切处理，从而得到长宽一致的图片。另外再将样本馈入模型前，会使用归一化的处理方式将其转换为均值为 0，方差为 1 的浮点数形式。

（4） 模型设置

由于原 DDPM 的模型训练和推断所需要的资源消耗巨大，因此在本次实验中采用了简化版本的扩散模型，并且引入了 DDIM 的采样算法，除此之外还采用了以下配置：

a. 使用方差作为位置信息嵌入而不是时间

扩散模型往往将位置信息嵌入到扩散过程中，而基于时间匹配（score-matching）的模型往往嵌入的是不同级别的噪声方差。后者的方式能够使得在训练时对不同的噪声更为敏感，且在采样时能够采用不同的噪声方差，从而控制每个时间步采样的随机程度。而前者要达到这一点则需要重新训练模型。

b. 仅在模型第一层馈入位置信息

扩散模型往往会将嵌入信息输入到每一层。考虑到扩散模型使用了具有跳跃链接的 Unet[10]结构，这些嵌入信息理论上也会随着跳跃链接馈入更深的网络层，因此出于简化的目的可以仅在第一层馈入位置信息。

c. 不使用注意力机制[11]

出于缩减模型容易的考虑，将原 DDPM 的注意力模块删去。

d. 不使用可学习的 Batch Normalization[12]

因为每个时间步学习的都是一个正态分布形式的表示，利用 Batch Normalization 得到正态分布的隐含表示可以加速学习过程。

e. 将最后一层的卷积的权重初始化为 0

使得网络在初始化后对任意输入都输出 0。由于输入的图片都会预处理为具有均值为 0，方差为 1 的表示形式，因此此举相当于为网络的学习设置了一个靠近目标值的初始点，从而加快学习进程。

（5） 训练设置

训练过程除了训练模型本身外，还会使用 EMA（Exponential Moving Average）的方式更新模型的一个副本，这个模型副本将用于模型的评估。另外，为了提高生成样本的质量，最终的损失度量采用了绝对平均误差，相对于均方误差损失，后者虽然能够增加生成样本的多样性，但同时也会导致生成的样本较为平滑，即生成质量有所损失。模型将在 Oxford Flowers 102 的训练集上训练 50 个 epoch，学习率固定为 0.001，权重衰减率固定为 0.0001。并且采用边训练边评估的方式，在整个训练过程中仅保存评估值最好的权重。

（6） 采样设置

本实验使用了 DDIM 的采样算法，并且将σ_t设置为 0，因此最终的采样是一个确定性的过程，仅在初始时间步从一个高斯噪声中采样，类似于 GAN 的采样过程。

所有的采样都是在模型的 EMA 副本上进行的。

（二）实验过程及结果

（1） 官方代码实验结果

代码来源：

https://github.com/keras-team/keras-io/blob/master/examples/generative/ddim.py

以下是理想的官方采样的结果

图 11

但是我们实际训练得到的结果却是下面这样的（使用 DDIM 采样 20 步）

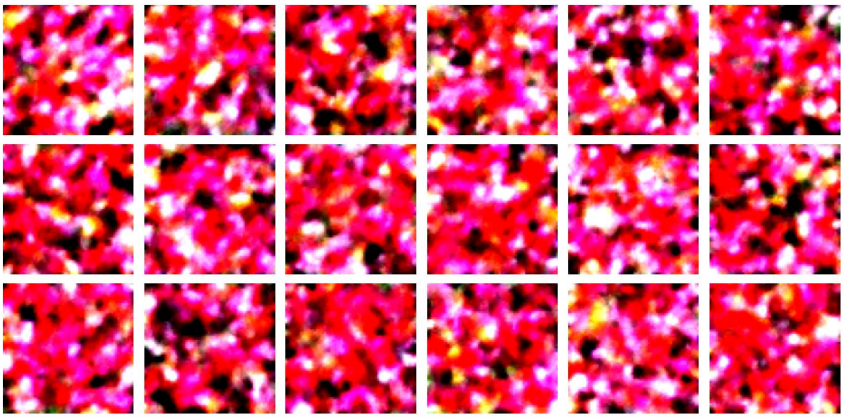

图 12

以下是 DDPM 采样 1000 步的生成结果

图 13

采样输出更像是花瓣和叶子杂糅的产物

而且训练过程中记录的损失变换也证实了这次的训练有问题，如下训练图所示

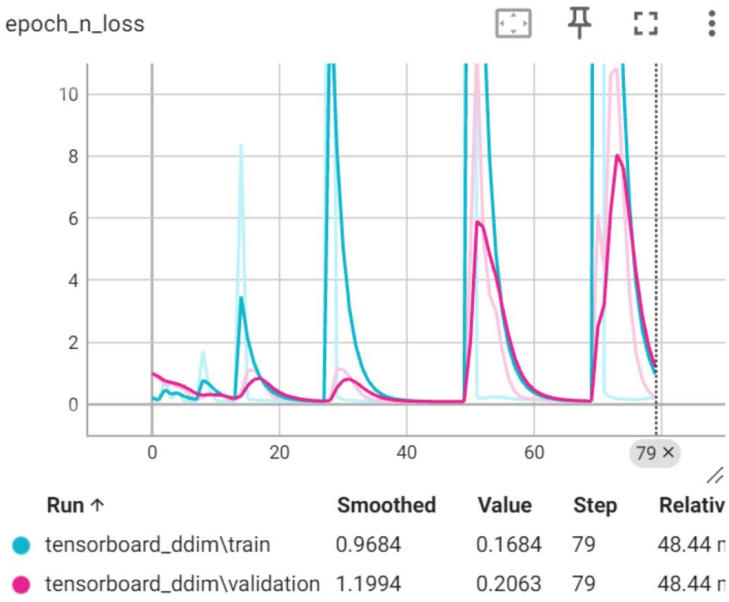

图 14

但是我们在训练时使用的是完完全全没有更改过的官方代码。

为此我们在官方的 Github 上发布了一个 ISSUE：

https://github.com/beresandras/clear-diffusion-keras/issues/9

（2） 改造官方的 DDIM 代码作对比实验

我们进一步改造官方的 DDIM 代码，使其具有相同的实验设置。

为了确保改造后的代码的正确性，我们从以下几个方面展开了调试：

a. 确定数据管道的正确性

通过保证没有将数据输入模型的情况下，数据预处理和后处理得到的是一个对数据的恒等变换。

b. 确定推断过程的正确性

这主要包含模型和采样算法的正确性，通过加载 DDPM 的预训练模型，和官方的采样结果进行对比，排除模型和采样算法中存在的错误。

对于采样算法：

其中就发现了我们在实现 DDIM 的采样算法时忽略了一个论文中提到的细节导致采样的样本的数值出现了 NAN。

具体的是在计算采样的方差时，即 DDIM 中的公式（15）中的σ_{t_i}，如下所示

$$\sigma_{\tau_i}(\eta) = \eta\sqrt{(1-\alpha_{\tau_{i-1}})/(1-\alpha_{\tau_i})}\sqrt{1-\alpha_{\tau_i}/\alpha_{\tau_{i-1}}},$$

在$\tau_i = 1$的采样时，我们的代码实现中$\alpha_0 = \alpha_T$，这导致α_1远远大于α_0，进而导致$1 - \frac{\alpha_0}{\alpha_{-1}}$是一个负数，而对负数开根便得到了 NAN。而 DDIM 中即将α_0设置为 1，从而避免 NAN 的数值计算问题。

对于模型：

其中就发现了 DDIM 中的上采样采用的是最邻近插值，而 Kears 版本中采用的是双线性插值，当我们使用 Kears 版本中的配置时，加载预训练模型采样得到的样本表现很糟糕。如下图所示：

图 15 DDIM_lsun_bedroom

但是我们很难确定仅仅是一个采样方法的不同导致 Kears 版本的采样结果出现问题。

c. 确定训练流程的正确性

在我们的实验设定中，对数据进行归一化处理便会使得输入的数值以 0 为中心，且方差为 1，再进一步对最后一层的网络的权重做 0 初始化，理论上模型训练时的初始损失将限定在 1 以内。但是实际训练时损失初始为一千多。后面我们发现是在损失函数的计算中没有对每一个像素点取平均。当然，数值的量级并不影响具体的训练，但是这样能够和 Keras 版本的损失作对比。

接下来，我们通过配置不同的模型参数，进行对照实验，希望能够找到能够正确生成样本的那个模型，并反向推演为什么 Keras 版本的生成结果和官方的结果差距如此之大。我们主要进行了四个版本模型架构的训练实验：

1）baseline_keras，记为 BK

该版本完全使用 Keras 版本中的模型架构，即本实验最开始的配置

2）baseline_bn_norm_out，记为 BBNO

相较于 1)，该版本使用了 DDIM 中的位置编码策略和更大的卷积核(kernel size = 3)用于图像编码嵌入，同时在最后一层卷积的前一层加入了一个归一化层，这个是 DDIM 中的架构设置。

3）baseline_bn_affine_norm_out，记为 BBANO

相较于 2)，该版本使用了可学习的 BN，主要用于探究为什么 Keras 版本中使用不可学习的 BN。

4）baseline_gn_affine_norm_out，记为 BGANO

相较于 3)，该版本使用 Group Normalization，是 DDIM 中的架构设置。

四种架构训练过程的损失变换如下图所示

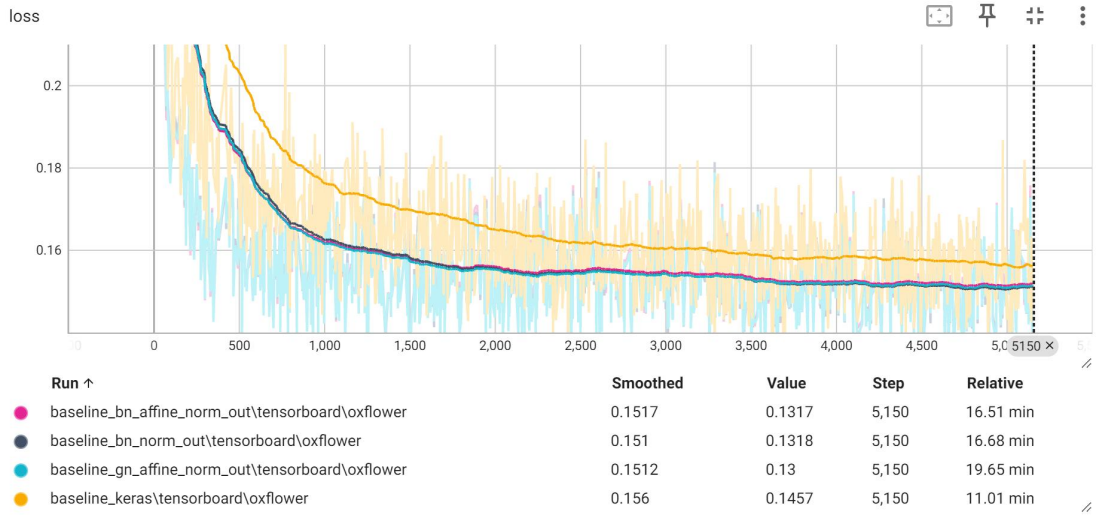

图 16

这是将 smooth 调整为 0.99 后的显示效果，方便对比查看。四种架构的损失变化基本一致，且当损失降到 0.14 后就在此处一直处于振荡的状态。

另外，四种架构采样生成的样本也并没有如官方采样结果那样理想，但相较于 Keras 版本的采样结果有所改善，但总体还是呈现出一种花瓣和叶子杂糅的状态。

如下图所示是同样的噪声由三种模型使用 DDIM 采样 20 步的每一步采样结果

图 17 从左到右对应时间步从 20 到 1

如下图所示是三种模型使用 DDIM 采样 50 步的结果，仅取最后一步的结果

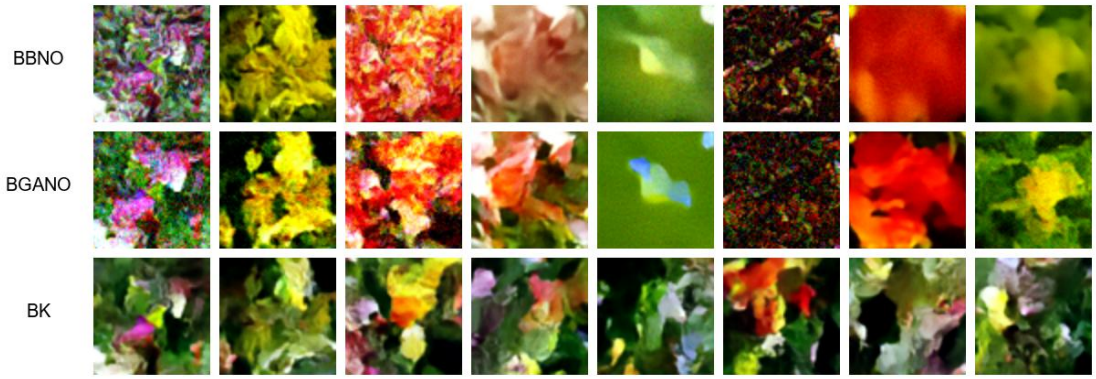

图 18

如下图所示是三种模型使用 DDIM 采样 100 步的结果，仅取最后一步的结果

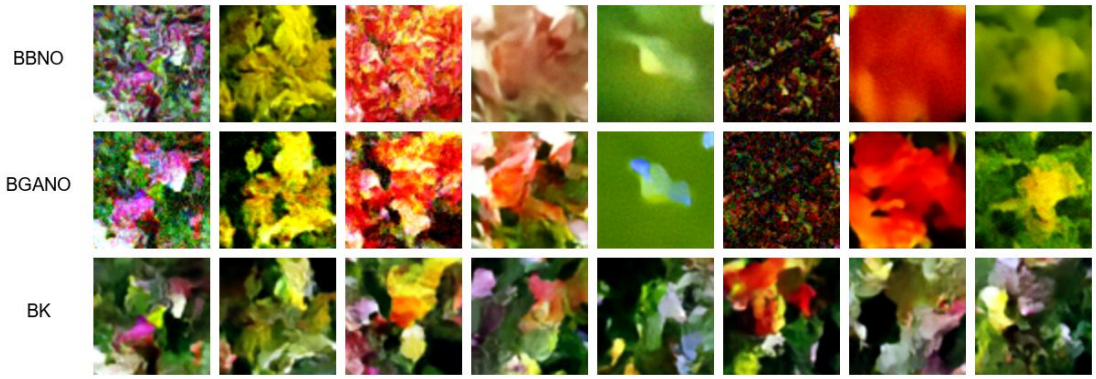

图 19

如下图所示是三种模型使用 DDPM 采样 1000 步的结果，仅取最后一步的结果

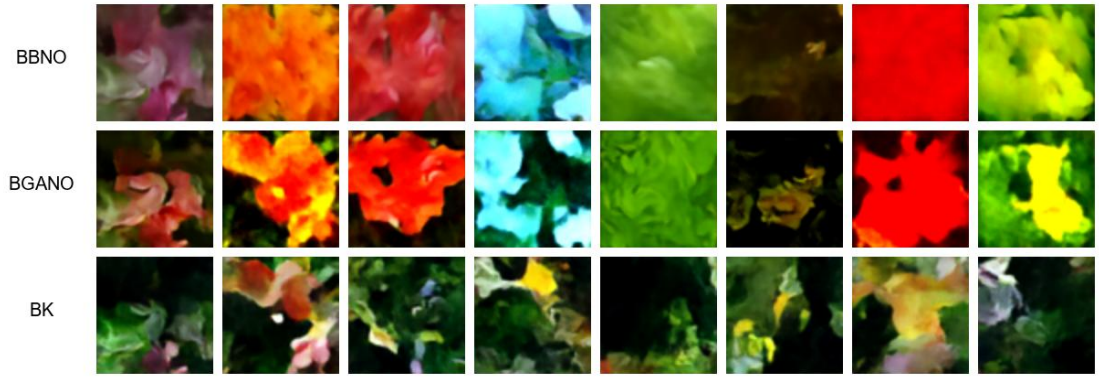

图 20

为了探究 Keras 版本使用不可学习的 BN 的原因， 我们专门对比了使用可学习的 BN 和不可学习的 BN 之间的采样结果，同时还对比了可学习 BN 本身分别使用 EMA 模型副本和使用训练所得模型的采样结果。

如下是 baseline_bn_affine_norm_out 内部分别使用和不使用 EMA 模型副本采样结果的对比

采用 DDIM 采样 20 步每步的结果

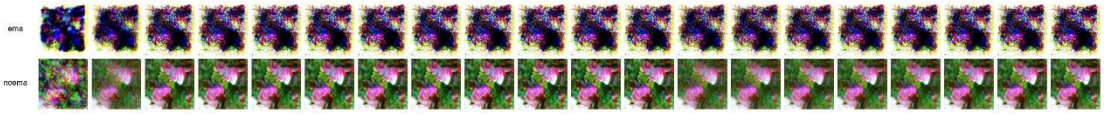

图 21

采用 DDIM 采样 50 步的结果，仅取最后一步

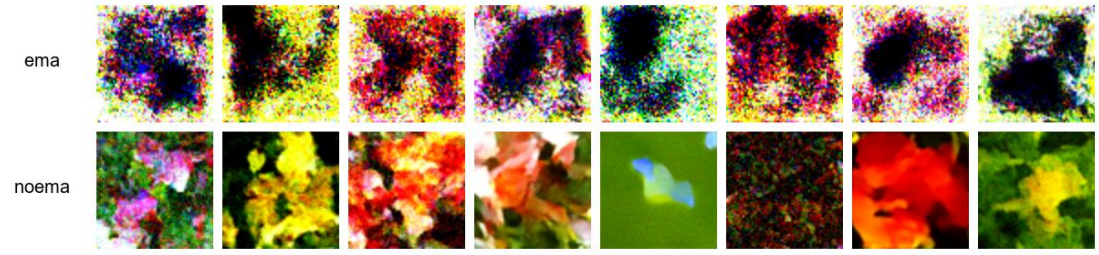

图 22

采用 DDIM 采样 100 步的结果，仅取最后一步

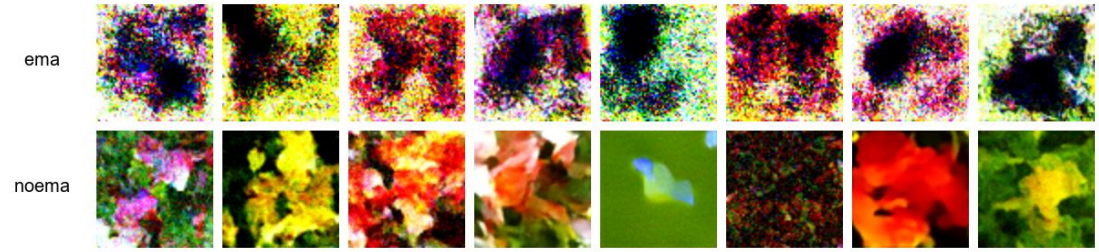

图 23

采用 DDPM 采样 1000 步的结果，仅取最后一步

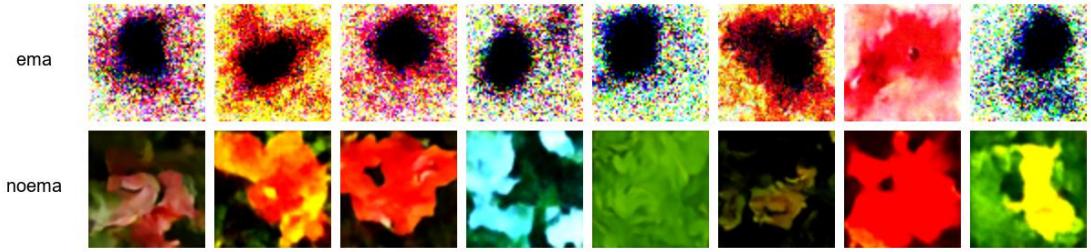

图 24

最终我们有如下结论：

通过对比 GN 和 BN，可以发现，BN 在归一化时的统计量需要依赖于多个样本，这个统计量同样需要使用 EMA 的方式来计算，因此在 EMA 模型副本上更新这些统计量是一个问题。

由于统计量本身是和数据分布绑定的，所以按理来说它不应该再使用 EMA 的方式更新到 EMA 模型副本上，而应该直接拷贝过去。另外由于 BN 的可学习参数也是和统计量直接绑定的，因此如果将它们以 EMA 的方式更新到 EMA 模型副本上时，则最终采样时会导致 EMA 模型副本的 BN 仿射参数于统计量之间不兼容的问题。

而由于 GN 的统计量仅依赖于样本本身，因此它的可学习参数也依赖于样本本身，而 EMA 模型副本得到的迁移的输出和同样迁移的仿射参数之间是相互兼容的，因此它的 EMA 模型副本的采样结果和训练所得模型的采样结果是一致的。

通过同样对比了可学习 BN 本身分别使用 EMA 模型副本和使用训练所得模型的采样结果。

如下是 baseline_gn_affine_norm_out 内部分别使用和不使用 EMA 模型副本采样结果的对比

采用 DDIM 采样 20 步每步的结果

图 25

采用 DDIM 采样 50 步的结果，仅取最后一步

图 26

采用 DDIM 采样 100 步的结果，仅取最后一步

图 27

采用 DDPM 采样 1000 步的结果，仅取最后一步

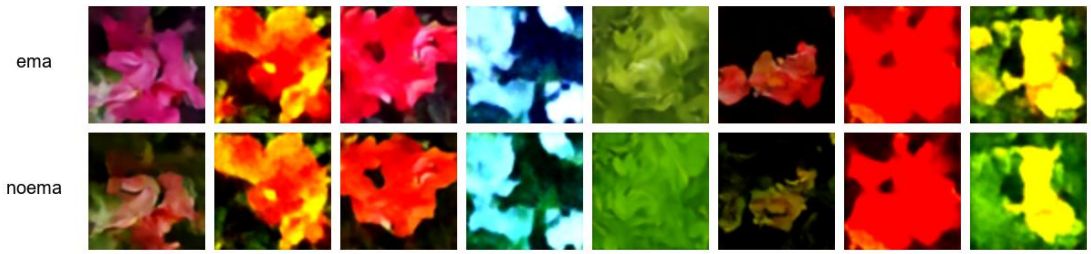

图 28

七、 总结及后续工作

我们最初的构想是训练一个足够简化的 diffusion 模型，并在这个基础上进一步简化，其中能够进行简化的对照配置包含

a. U-Net 结构
b. 位置编码
c. 方差序列生成方式
d. 数据的归一化
e. EMA 的训练方式

可以进一步探究对以上几个部分进行剔除或者是用更简单的方式替换，从而探究哪些配置对 diffusion 来说是必不可少的。然后能够得到一个足够简单的 diffusion 模型。

正如在 SegNet[13]中提到的，要验证一个想法的可行性最好是在一个足够简单的模型上展开实验，从而剔除其他的增益，独立地探究该想法的可行性。这也是“奥卡姆剃刀”的思维的一个延生。

在得到这个简化模型后，我们打算再引入 Latent Diffusion 中的自编码器结果，将生成过程投射到维度更小的潜在编码空间，在减少运算消耗的同时，间接加速训练和推断的过程，并能够应用于生成分辨率更高的图片并同时保持较高的语义正确性。在得到这一整套模型后，便能够将它快速地用于各种下游的实验任务。

不过可惜的是本次实验并没有得到理想的结果，不过我们仍会进一步探究其中存在的问题，使它达到我们想要的效果，并按计划的那样一步一步往下走。

参考文献

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair , Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. arXiv preprint arXiv:1406.2661, 2014.

[2] Song, Yang, and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. arXiv preprint arXiv:1907.05600, 2019.

[3] Christopher M Bishop. Pattern recognition and machine learning. springer, 2006. 423-455

[4] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114, 2013.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. arXiv preprint arXiv:2006.11239, 2020.

[6] Siddhartha CHIBand Edward GREENBE. Understanding the Metropolis-Hastings Algorithm. 1995.

[7] Jiaming Song, Chenlin Meng & Stefano Ermon. DENOISING DIFFUSION IMPLICIT MODELS. arXiv preprint arXiv: 2010.02502, 2020.

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Bjorn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv preprint arXiv:2112.10752, 2022.

[9] Nilsback, M-E. and Zisserman, A. Automated Flower Classification over a Large Number of Classes. Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing. 2008.

[10]Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv preprint arXiv:1505.04597, 2015.

[11]Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin. Attention Is All You Need. arXiv preprint arXiv: 1706.03762, 2017.

[12]Sergey Ioffe Google Inc, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift. arXiv preprint arXiv: 1502.03167, 2015.

[13]Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, Senior Member. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. arXiv preprint arXiv:1511.00561, 2015.

附录 1

1. 论文“Denoising Diffusion Probabilistic Model”公式(4)的推导

“前向过程的一个显着特性是它允许在任意时间步长 t 以封闭形式采样 x”

$$\alpha_t = 1 - \beta_t, \overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s \, , p(x_t|x_{t-1}) := N(x_t;\ \sqrt{\alpha_t}x_{t-1}, \beta_t I)$$

$$\mathrm{q}(x_t|x_0) = \mathrm{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1 - \overline{\alpha}_t)I)$$

基本证明思路

$$x_t \sim N(x_{t-1}, 1) \Leftrightarrow x_t = N(0,1) + x_{t-1}$$

基本证明

$$\mathrm{p}(x_t) = \int \mathrm{p}(x_t|x_{t-1})\mathrm{p}(x_{t-1})\mathrm{d}\,x_{t-1}$$

$$= \int \mathrm{N}(x_t - x_{t-1}; 0,1)\mathrm{p}(x_{t-1})\mathrm{d}\,x_{t-1}$$

$$= (\mathrm{N}(0,1) * p_{x_{t-1}})(x_t), \text{卷积运算}$$

$$\Longrightarrow x_t = N(0,1) + x_{t-1}$$

令 $\mathrm{t} = 2$

$$\mathrm{q}(\mathrm{x}_2|\mathrm{x}_0) = \int \mathrm{q}(\mathrm{x}_2|\mathrm{x}_1)\mathrm{q}(\mathrm{x}_1|\mathrm{x}_0)\mathrm{dx}_1 \text{ should be } \mathrm{N}(\mathrm{x}_2; \sqrt{\alpha_2\alpha_1}\mathrm{x}_0, (1 - \alpha_2\alpha_1)I)$$

$$\begin{cases} q(\mathrm{x}_2|\mathrm{x}_1) := N(\mathrm{x}_2; \sqrt{\alpha_2}\mathrm{x}_1, \beta_2 I) = N(\mathrm{x}_2 - \sqrt{\alpha_2}\mathrm{x}_1; 0, \beta_2 I) \\ q(\mathrm{x}_1|\mathrm{x}_0) := N(\mathrm{x}_1; \sqrt{\alpha_1}\mathrm{x}_0, \beta_1 I) = N(\sqrt{\alpha_2}\mathrm{x}_1; \sqrt{\alpha_2\alpha_1}\mathrm{x}_0, \alpha_2\beta_2 I) \end{cases}$$

$$\Rightarrow \mathrm{q}(\mathrm{x}_2|\mathrm{x}_0) = \int \mathrm{N}(\mathrm{x}_2 - \sqrt{\alpha_2}\mathrm{x}_1; 0, \beta_2 I) * N(\sqrt{\alpha_2}\mathrm{x}_1; \sqrt{\alpha_2\alpha_1}\mathrm{x}_0, \alpha_2\beta_2 I)\mathrm{dx}_1$$

$$= (\mathrm{N}(0, \beta_2 I) * \mathrm{N}(\sqrt{\alpha_2\alpha_1}\mathrm{x}_1, \alpha_2\beta_2 I))(\mathrm{x}_2), \text{卷积运算}$$

$$\text{marked as } \mathrm{Z} = \mathrm{X} + \mathrm{Y}, \begin{cases} \mathrm{X} = \mathrm{x}_2 - \sqrt{\alpha_2}\mathrm{x}_1 \\ \mathrm{Y} = \sqrt{\alpha_2}\mathrm{x}_1 \end{cases}$$

$$\mathrm{E}[\mathrm{Z}] = \mathrm{E}[\mathrm{X} + \mathrm{Y}] = 0 + \sqrt{\alpha_2\alpha_1}\mathrm{x}_0 = \sqrt{\alpha_2\alpha_1}\mathrm{x}_0$$

$$\mathrm{Var}(\mathrm{Z}) = \mathrm{E}[(\mathrm{X} + \mathrm{Y} - \mathrm{E}[\mathrm{Z}])^2] = \mathrm{E}[\mathrm{X}^2] + 2\mathrm{E}[\mathrm{XY}] + \mathrm{E}[\mathrm{Y}^2] - \mathrm{E}^2[\mathrm{Z}]$$

$$= (1 - \alpha_2\alpha_1)I$$

遵循归纳法，即可得证！