

图像处理与视觉—课程大作业实践报告

匿名

摘要

我们实现了简单的去噪扩散模型，并将其效果与生成对抗网络及变分自编码器进行对比。

目录

1 去噪扩散模型基础	1
1.1 简要介绍	1
1.2 公式推导	2
2 模型结构及实现	2
2.1 噪声方差	2
2.2 神经网络	3
2.2.1 U-Net	3
2.2.2 残差网络	3
2.2.3 注意力机制	4
2.2.4 位置编码	4
2.3 训练超参数	4
3 模型效果对比	4
3.1 生成效果对比	4
3.2 训练难度对比	6
4 总结	8
参考文献	8
图形目录	8

1 去噪扩散模型基础

1.1 简要介绍

去噪扩散概率模型 (*Denoising Diffusion Probabilistic Models*) 是基于分数匹配 (score matching) 理论的生成模型，最早在 [2] 中有了足够强的能力。简单来说，扩散模型试图通过变分推理，学习一个逐渐向图片添加噪声的过程的逆，从而从一个采样的噪声中生成图片。

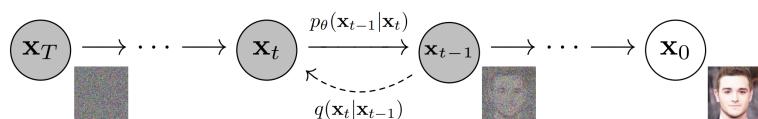


图 1. 扩散模型工作的过程. 图片来自 [2].

扩散模型仍可以看作隐变量模型。在这里，隐变量是和样本图片维度相同的高斯噪声，我们试图学习一个从隐变量到图像流形的过程。相较于 VAE，扩散模型的优势在于：

- 通过变分推理学习时，VAE 的后验概率 $q_\phi(z|x)$ 是通过自监督的编码器学习出来的，而扩散模型中这个函数是确定性的，即施加高斯噪声。
- VAE 的解码器是从一个学习出来的，没有明确特征的编码中解码，而扩散模型的解码器就是去噪声（或许除了一开始几轮），在图像上有明确的含义。

或许是因为这些优势，扩散模型的清晰度比 VAE 好很多。

1.2 公式推导

在加噪声时, 后一张图片 \mathbf{x}_t 满足

$$\mathbf{x}_t \sim \mathcal{N}(\sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}).$$

其中 \mathbf{x}_{t-1} 是上一张图片, \mathbf{I} 是对角矩阵, β_t 是方差. β 可以固定, 也可以学习. 这样采样的好处是, 我们可以快速采样任意时间 t 的一张图片而无需一步一步做, 因为

$$\begin{aligned} \mathbf{x}_t &= \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \boldsymbol{\epsilon}_{t-1} \\ &= \sqrt{1 - \beta_t} (\sqrt{1 - \beta_{t-1}} \mathbf{x}_{t-2} + \sqrt{\beta_{t-1}} \boldsymbol{\epsilon}_{t-2}) + \sqrt{\beta_t} \boldsymbol{\epsilon}_{t-1} \\ &= \sqrt{(1 - \beta_t)(1 - \beta_{t-1})} \mathbf{x}_{t-2} + \sqrt{(1 - \beta_t)\beta_{t-1}} \boldsymbol{\epsilon}_{t-2} + \sqrt{\beta_t} \boldsymbol{\epsilon}_{t-1} \\ &= \sqrt{(1 - \beta_t)(1 - \beta_{t-1})} \mathbf{x}_{t-2} + \sqrt{1 - (1 - \beta_t)(1 - \beta_{t-1})} \boldsymbol{\epsilon}' \\ &\vdots \\ &= \sqrt{\alpha_t} \mathbf{x}_0 + \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}. \end{aligned} \tag{1}$$

这里 $\boldsymbol{\epsilon}_i$ 是独立的单位高斯分布; $\alpha_t = \prod_{i=1}^t 1 - \beta_i$, 可以提前计算好. 注意这不代表逆过程也可以直接计算, 除非加上已知 \mathbf{x}_0 的条件.

我们试图优化一个最大似然的下界

$$\mathbb{E} \left[\log p(\mathbf{x}_t) + \sum_{i=1}^t \log \frac{p_\theta(\mathbf{x}_{i-1} | \mathbf{x}_i)}{q(\mathbf{x}_i | \mathbf{x}_{i-1})} \right] = \mathbb{E} \left[\log \frac{p_\theta(\mathbf{x}_0, \dots, \mathbf{x}_t)}{q(\mathbf{x}_1, \dots, \mathbf{x}_t | \mathbf{x}_0)} \right] \geq \mathbb{E} [\log p_\theta(\mathbf{x}_0)].$$

这里 p_θ 就是我们要训练的神经网络. 原始论文称, 当 β_t 足够小时, 逆过程也近似符合高斯分布, 故我们也以此为前提训练 $f_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) \sim \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$, 这里包含了时间 t 是因为每一时刻加噪的参数不同. 用高斯分布拟合逆过程带来的另一个好处是, 如果把上面的损失函数写作

$$\mathbb{E} \left[\text{KL}(q(\mathbf{x}_t | \mathbf{x}_0) \| p(\mathbf{x}_t)) + \sum_{i=2}^t \text{KL}(q(\mathbf{x}_{i-1} | \mathbf{x}_i, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{i-1} | \mathbf{x}_i)) - \log p_\theta(\mathbf{x}_0 | \mathbf{x}_1) \right].$$

其中, 第一项是常数; 最后一项在原论文中用单独的解码器预测, 而为了方便我们忽略了这一项; 剩下的所有项中, KL 距离的两个分布都是高斯分布, 有闭式解.

实际训练的过程还有进一步的参数化. 我们先固定 $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)$ 为 $\sqrt{\beta_t} \mathbf{I}$ 或 $\sqrt{\frac{1 - \alpha_{t-1}}{1 - \alpha_t}} \beta_t \mathbf{I}$: 原始论文称这两者训练效果相近, 且分别对应于某个熵的上下界, 在这里不做探讨. 也就是说, 我们只预测均值 (不过, [3] 中称同时学习方差有效果提升). 这样第 i 项可以写作

$$C_1 \mathbb{E} [\|\boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2] + C_2,$$

这里 C_i 是与 θ 无关的常数, $\boldsymbol{\mu}_t$ 表示等式 1 中均值的部分. 进一步,

$$\boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1 - \alpha_t}} \boldsymbol{\epsilon} \right) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t).$$

我们选择只让 $\boldsymbol{\mu}_\theta$ 去预测 $\boldsymbol{\epsilon}_\theta$, 但是采样时不会直接一步算得原始图片, 而是依旧计算 \mathbf{x}_{t-1} . 原论文称这样的训练效果比直接预测均值要好, 并且有朗之万动力学的理论依据, 在这里不做讨论.

2 模型结构及实现

模型基本参考 [2] 中的实现, 以及 [3][7][4] 中提到的一些优化. 不过, 我们并没有进行优化前后的对比试验. 数据集来自于 Kaggle <https://www.kaggle.com/datasets/splcher/animefacedataset/data>, 包含 6 万多张低分辨率的动漫风格人脸.

2.1 噪声方差

在原始论文 [2] 中, 方差从 0.001 至 0.02 线性增加. 但是 [3] 中提到, 用余弦分配在低分辨率图片上会有更好的效果. 具体来说, 我们想让增加噪声的每一时间点, 相对于原始图片的噪声的方差接近与 $\cos^2 x$ 的图像, 这相较于线性增加更加平滑. 每一采样的方差是从这个期望的 $\bar{\alpha}$ 反推出来的. 以下两张图是线性增加, 余弦分配, 和用 sigmoid 分配的 β 和 $1 - \bar{\alpha}$ 的图像. 其中, β 的最大值被限制在了 0.02.

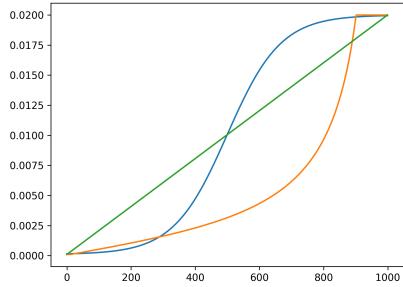


图 2. β . 蓝, 绿, 黄分别对应 sigmoid, 线性, 余弦.
下面是分别用这三种方差为同一张图片加噪的实例.

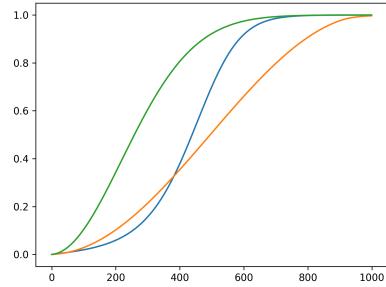


图 4. 用不同的方差加噪声. 从上至下分别为线性, sigmoid, 余弦.

实际实现中, 我们使用了 sigmoid 和余弦. 后文的分析如未特别注明, 都是基于 sigmoid 的.

2.2 神经网络

2.2.1 U-Net

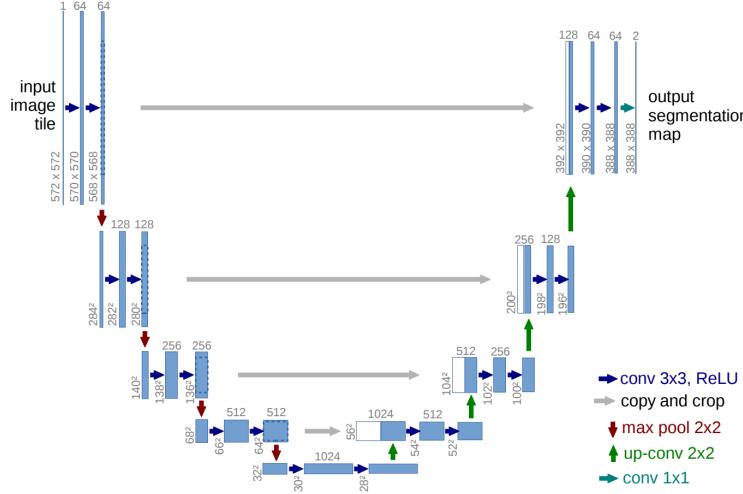


图 5. U-Net 的结构. 图片来自 [5].

由于预测噪声的神经网络输出和输入维度相同, 原论文整体结构使用 U-Net [5]. 课堂上已经介绍过, U-Net 通过直接将高维的输入凭借到同维度的输出, 使得模型容易学习到图像的高频特征的同时参数量不会过于膨胀.

原文中, 作者为 32×32 的图像设计了三层分辨率, 为 256×256 的图像设计了六层. 由于显存限制, 我们为 64×64 的图像设计了三层, 初始通道数 64, 每层通道数量翻倍.

2.2.2 残差网络

与提出 U-Net 的原始论文不同, [2] 中的 U-Net 模仿 PixelCNN++ 在网络中的每一层使用两层残差结构替代普通的卷积, 我们同样如此实现. 残差网络的优势在于梯度反传稳定. 另外不同的一点是, 原论文用 group normalization 代替了 weight normalization, 因为实现简单.

2.2.3 注意力机制

在尺寸为 16×16 的两层残差网络之间, 原论文还加入了一层自注意力 (self attention) 层, 更准确地说就是 transformer. 原文中这一层加在 U-Net 中对应尺寸为 16×16 的层上, 对应于我们的网络结构这恰好是网络的最深一层, 简化了网络设计.

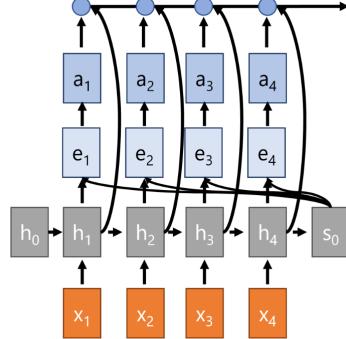


图 6. Transformer 的架构. 图片来自课件.

2.2.4 位置编码

为了编码进时间, 每个残差网络块还要接受一个位置编码 (和相配的全连接层). 这一信息看作 1×1 的卷积施加在了每个残差块的第一个卷积层之后. 原论文使用了 Sinusoidal 位置编码, 即

$$\begin{aligned} p_{i,2j} &= \sin\left(\frac{k}{10000^{\frac{2i}{d}}}\right), \\ p_{i,2j+1} &= \cos\left(\frac{k}{10000^{\frac{2i}{d}}}\right). \end{aligned}$$

可视化展示如下

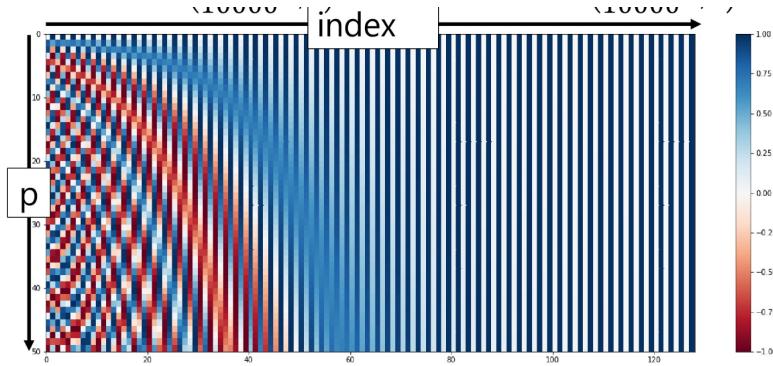


图 7. Sinusoidal 位置编码的可视化展示. 图片来自课件.

2.3 训练超参数

训练使用 Adam 优化器, 学习率 0.0001, batch size 设置为 128.

3 模型效果对比

3.1 生成效果对比

以下是扩散模型训练约 25000 迭代后生成的图片和 GAN 在训练 25000 迭代后生成的图片.



图 8. 扩散模型的生成效果.
以下是在另一门课程中实现的 VAE 生成的图片.



图 9. GAN 的生成效果.



图 10. VAE 的生成效果.

这样比较来看, 扩散模型生成图片的真实度和清晰度与 GAN 不相伯仲, 而 VAE 过于模糊. 不过扩散模型, 实际上还有其他许多生成模型, 存在色彩偏移的问题: 生成的图片均偏向某一颜色. 比如以下扩散模型在稍早迭代时生成的数据明显更浅更红. 在训练的更早时期情况更加极端.



图 11. 扩散模型在训练 15000 迭代后生成的图片.



图 12. 扩散模型在训练 10000 迭代后生成的图片.

[1] 提出了增加网络结构和相应损失函数的方式解决这个问题. 不过在 [6] 中, 作者提到了一个缺

少理论基础却简单有效的方法, 即对训练模型取一个指数移动平均 (*exponential moving average*). 除训练模型 θ 外, 保存一个平均模型 $\bar{\theta}$, 每一迭代后更新 $\bar{\theta} = \eta \bar{\theta} + (1 - \eta) \theta$, 其中 θ 取 0.999. 注意 $\bar{\theta}$ 并不参与训练, 只用作采样生成图像. 根据 [8], 这一做法对 GAN 也有效, 故为求公平, 对 GAN 也同样应用该技巧. 以下分别是扩散模型和 GAN 生成的图片.



图 13. 取移动平均后的扩散模型的生成效果.



图 14. 取移动平均后的 GAN 的生成效果.

可见移动平均除了解决色彩偏移问题外, 也减少了图片中突兀的部分 (artifacts). 再次对比, 发现扩散模型的效果略优于 GAN: 不会生成完全错误的图像, 在细节上有优势. 不过, (我们的) 扩散模型在图像整体结构 (五官和朝向的一致性等) 上做的不够好, 怀疑是迭代不够 (因为 25000 迭代的模型已经有改善, 不过由于移动平均的性质, 平均的模型还未改善) 或噪声不足. 实践角度更致命的缺点是生成图像需要模型进行与加噪次数相同的推理, 也就是说图像生成花费的时间几乎是 GAN 的 1000 倍. 时间中, 生成一个 batch 的图像需要约一分钟.

3.2 训练难度对比

由于 U-Net 的结构远比 DCGAN 复杂, 尤其是有残存网络结构和拼贴结构, 其需要的显存远多于 DCGAN. 扩散模型训练时间略长于 DCGAN, 但是 GAN 的训练参数选择比较微妙, 训练不够稳定. 比如, 若分辨器和生成器的更新频率一致, 那么分辨器的准确率很快变得极高, 并且出现课程实验中的不稳定现象, 虽然从实际效果看并没有出现严重的模式崩溃. 以下是训练中的损失函数曲线和生成的图像.

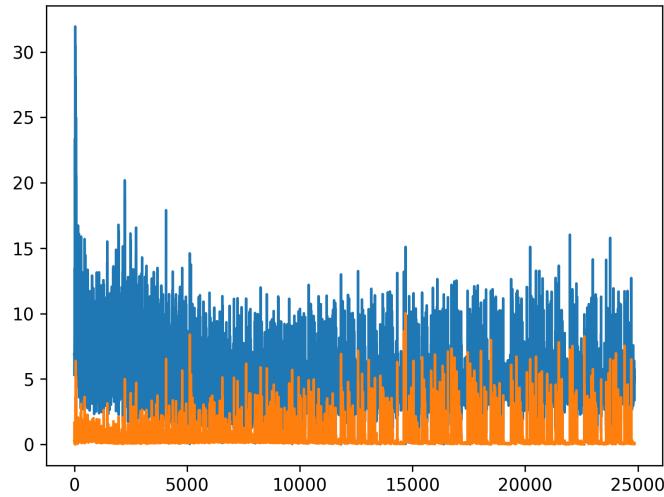


图 15. GAN 训练过程的损失曲线 (1 : 1).

如果频率比例为 5, 虽然分辨器的准确率稳定在百分之 50, 但是生成的图片很模糊, 如下.



图 16. GAN (1 : 5) 的生成效果.

最终选择了比例为 2, 但是从损失曲线来看, 训练后期的准确率仍然过高. 生成图像也显示, 在准确率逼近 1 后的训练效果不明显.

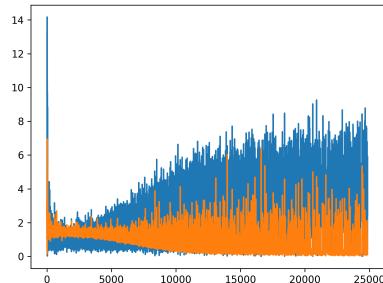


图 17. GAN 训练过程的损失曲线 (1 : 2).

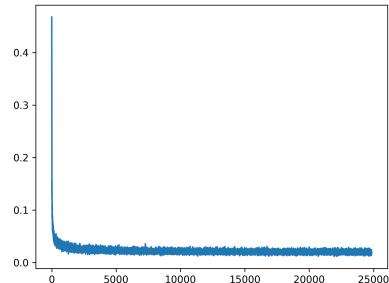


图 18. 扩散模型训练过程的损失曲线.



图 19. GAN 在 15000 迭代的生成效果.



图 20. GAN 在 10000 迭代的生成效果.

但是扩散模型的训练对于超参数就比较稳定, 噪声方差的设置和加噪次数影响不大. 除了上面的损失函数图像外, 以下故事可例证: 我们在第一次训练扩散模型时减少轮数至 500, 但忘记增大方差. 可以预见这会导致采样的噪声不够像正态分布; 而采样的噪声仍然是正态分布, 因此生成的图片

会扭曲.



图 21. 对一张图片做错误的采样.

但是从结果来看, 还是有相当比例的图片可以称为真实.



图 22. 错误的扩散模型的生成效果.



图 23. 附: 用余弦方差生成的正确的效果.

4 总结

我们实现了简单的去噪扩散模型. 在漫画风格人物头像上训练后, 该模型能够生成质量足够好的同样风格图片, 在肉眼下好于 GAN 和 VAE.

参考文献

- [1] Katherine Deck 与 Tobias Bischoff. Easing color shifts in score-based diffusion models. 2023.
- [2] Jonathan Ho, Ajay Jain, 与 Pieter Abbeel. Denoising diffusion probabilistic models. 2020.
- [3] Alex Nichol 与 Prafulla Dharwal. Improved denoising diffusion probabilistic models. 2021.
- [4] Niels Rogge 与 Kashif Rasul. The annotated diffusion model. June 2022.
- [5] Olaf Ronneberger, Philipp Fischer, 与 Thomas Brox. U-net: convolutional networks for biomedical image segmentation. 2015.
- [6] Yang Song 与 Stefano Ermon. Improved techniques for training score-based generative models. 2020.
- [7] Lilian Weng. What are diffusion models? [Lilianweng.github.io](https://lilianweng.github.io), Jul 2021.
- [8] Yasin Yazici, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, 与 Vijay Chandrasekhar. The unusual effectiveness of averaging in gan training. 2018.

图形目录

扩散模型工作的过程. 图片来自 [2].	1
β . 蓝, 绿, 黄分别对应 sigmoid, 线性, 余弦.	3
$1 - \bar{\alpha}$. 蓝, 绿, 黄分别对应 sigmoid, 线性, 余弦.	3
用不同的方差加噪声. 从上至下分别为线性, sigmoid, 余弦.	3
U-Net 的结构. 图片来自 [5].	3
Transformer 的架构. 图片来自课件.	4
Sinusoidal 位置编码的可视化展示. 图片来自课件.	4
扩散模型的生成效果.	5
GAN 的生成效果.	5
VAE 的生成效果.	5

扩散模型在训练 15000 迭代后生成的图片.	5
扩散模型在训练 10000 迭代后生成的图片.	5
取移动平均后的扩散模型的生成效果.	6
取移动平均后的 GAN 的生成效果.	6
GAN 训练过程的损失曲线 (1 : 1).	6
GAN (1 : 5) 的生成效果.	7
GAN 训练过程的损失曲线 (1 : 2).	7
扩散模型训练过程的损失曲线.	7
GAN 在 15000 迭代的生成效果.	7
GAN 在 10000 迭代的生成效果.	7
对一张图片做错误的采样.	8
错误的扩散模型的生成效果.	8
附: 用余弦方差生成的正确的效果.	8