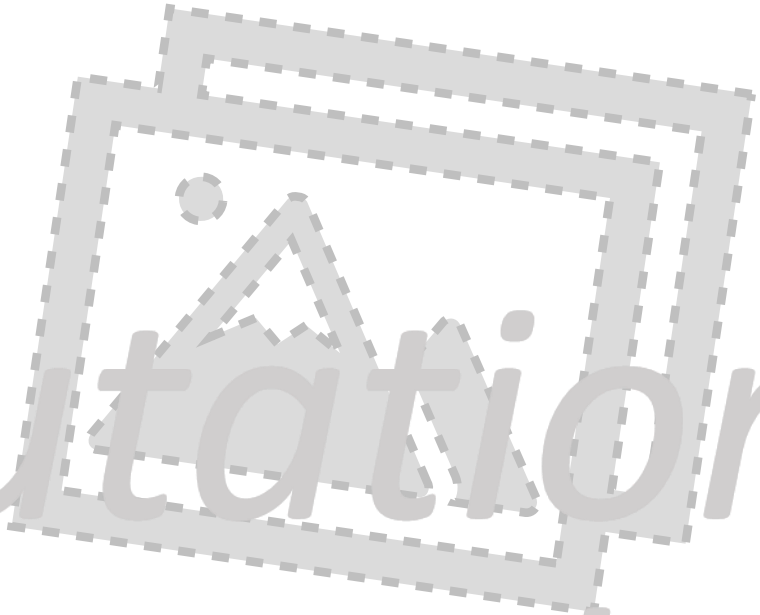
LAB06

Unet & DDPM

计算摄影学 2025春夏

2025/4/1



Computation Photography

BEFORE

- 本次实验内容需要在学在浙大上提交作业，本次作业计入成绩。
- 提交要求：
  - 将**源代码和实验文档**打包。
  - 不要提交训练好的模型文件，也不要提交数据集。
  - 如有需要，请补充 requirements.txt 。
  - 压缩包名称为 Lab6-学号-姓名.zip/7z。
  - 在**4月15日23:59:59**前提交至学在浙大。

实验任务

- （上周）搭建 Unet 网络
  - 配置深度学习环境
  - 搭建 Unet 网络并进行推理
- **（本周）基于 MNIST 数据集训练扩散模型**
  - 训练 epsilon – prediction 模型
  - **Bonus**：训练 v-prediction 模型

扩散模型与DDPM

Diffusion models are incremental updates where the assembly of the whole gives us the encoder-decoder structure. The transition from one state to another is realized by a denoiser.

扩散模型是由一系列**渐进式更新**组成的，这些更新共同构建形成了**编码器-解码器**结构。

扩散模型中，一个状态向另一个状态的转变过程是通过**去噪器**实现的。

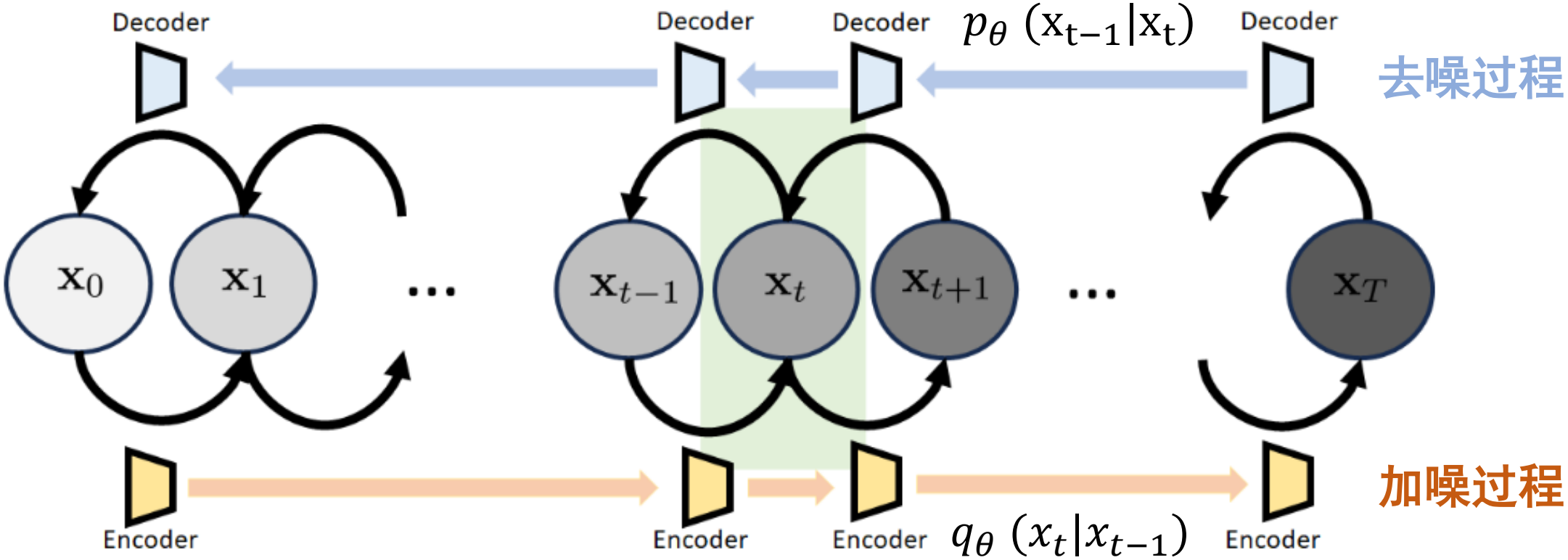
Chan, S. H. (2025). Tutorial on Diffusion Models for Imaging and Vision. arXiv preprint arXiv:2403.18103. https://arxiv.org/abs/2403.18103

扩散模型与DDPM(Cont.)

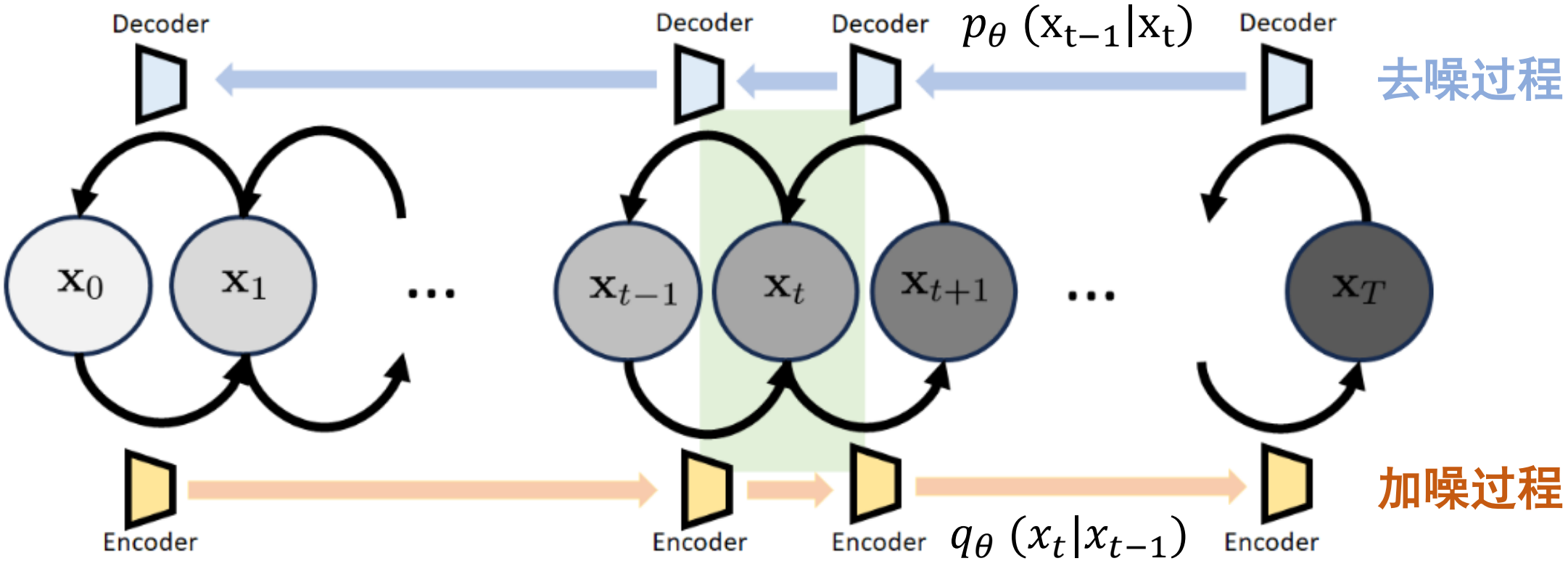
扩散模型以其训练稳定性、生成数据的多样性和高质量而受到广泛关注，是目前生成式模型的前沿技术。在2020年提出的 DDPM (Denoising Diffusion Probabilistic Model) 是最基础的扩散模型。

扩散模型与DDPM(Cont.)

DDPM 的模型结构如下所示：

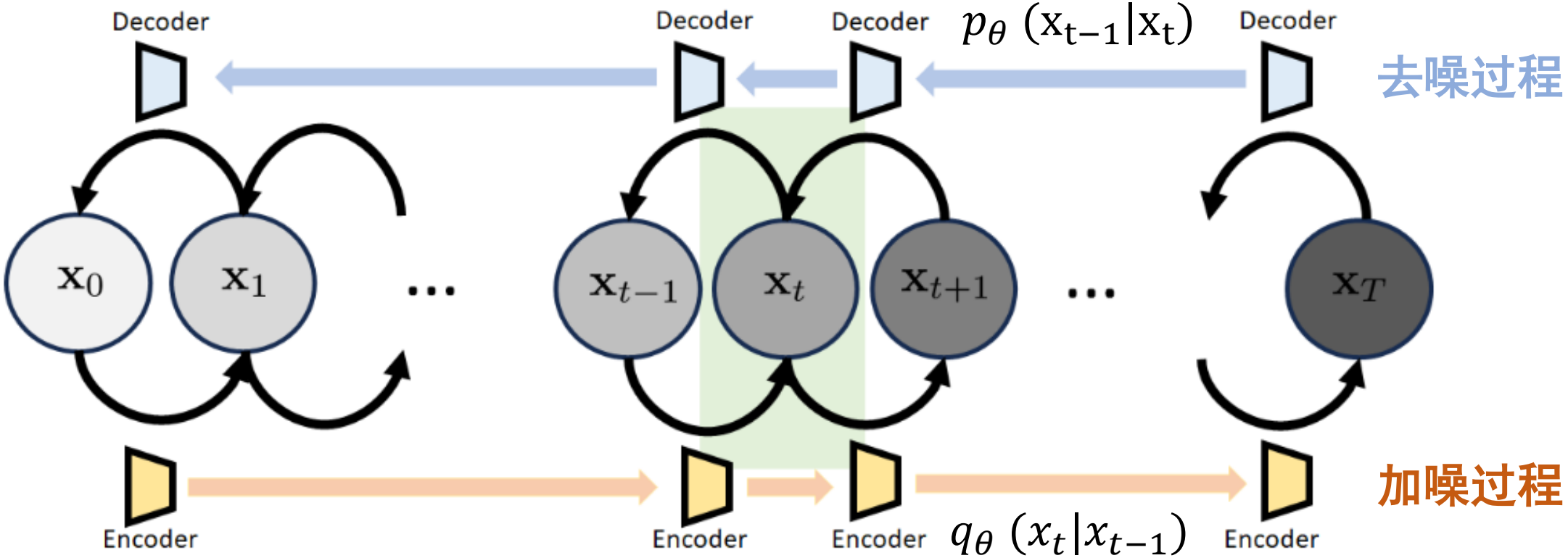


扩散模型与DDPM(Cont.)



其中 x_0 是原始真实图像，逐步添加高斯噪声得到 $x_1, x_2, \ldots, x_T$ ，可以假设有 $x_T \sim \mathcal{N}(0, \mathbf{I})$。

扩散模型与DDPM(Cont.)



那么如果可以预测每一步添加的高斯噪声，就可以从 $\mathcal{N}(0, \mathbf{I})$ 中采样，并逐步去噪得到一张真实的图像 $\widehat{x_0}$ 。

扩散模型的参考文献

为了更好地实现扩散模型，这里有一些参考文献：

- Lilian Weng 的博客：What are Diffusion Models?
- 最初的 DDPM 论文：Denoising Diffusion Probabilistic Models
- DDIM 论文：Denoising Diffusion Implicit Models
- Stanley H. Chan 的教程：Tutorial on Diffusion Models for Imaging and Vision

关键公式

加噪公式：

$$x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$$

其中，

$$q(x_t|x_{t-1}) = \mathcal{N}\left(\sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right)$$

其中 $\alpha_t = 1 - \beta_t$ 是关于 t 单调递减的，$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$，噪声 $\epsilon \sim \mathcal{N}(0, \mathbf{I})$。

关键公式

DDPM 的去噪公式：

x_{t-1} 从 $p\left(x_t|x_{t-1}\right)$ 中采样，该分布近似为高斯分布 $\mathcal{N}\left(\tilde{\mu}_t, \tilde{\beta}_t \mathbf{I}\right)$，其中

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta^{(t)}(x_t)\right)$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{\sqrt{1-\bar{\alpha}_t}}\beta_t$$

关键公式

DDIM 的去噪公式：

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{“ predicted } x_0\text{”}} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\varepsilon_\theta^{(t)}(x_t)}_{\text{“direction pointing to } x_t\text{”}} + \underbrace{\sigma_t \varepsilon_t}_{\text{random noise}}$$

$\epsilon_\theta^{(t)}(x_t)$ 是模型根据 x_t 预测出的噪声，ϵ_t 是随机采样的单位高斯噪声。

关键公式

DDIM 的去噪公式：

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{“ predicted } x_0\text{”}} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\varepsilon_\theta^{(t)}(x_t)}_{\text{“direction pointing to } x_t\text{”}} + \underbrace{\sigma_t\varepsilon_t}_{\text{random noise}}$$

$\sigma_t^2 = \eta\tilde{\beta}_t$是去噪过程中的噪声方差。当

- $\eta = 1$ 时，公式等价于 DDPM 的去噪公式
- $\eta = 0$ 时，就是 DDIM 的去噪公式，不确定项 ϵ_t 不再存在
- $\eta \in (0,1)$时，既不是 DDPM 也不是 DDIM。

当需要加速推断时，可以将公式中的 $t-1$ 替换为 $t-k$。

任务介绍：MNIST

MNIST 数据集 (Mixed National Institute of Standards and Technology database) 是美国国家标准与技术研究院收集整理的大型手写数字数据库，包含 60,000 个示例的训练集以及 10,000 个示例的测试集。

通常可以通过 MNIST 数据集下载链接 下载数据集。torchvision 中集成了包括 MNIST 的经典的数据集， 通过 datasets 的方法可以准备该数据集。

本实验基于 DDPM 建模 MNIST 手写数字的数据分布，从而能够采样生成新的手写数字图片。

DDPM 训练

本实验不要求搭建 U-Net 网络，将提供模块化实现的 unet.py。简要结构如下：

```
class Swish(nn.Module)
class TimeEmbedding(nn.Module)
class ResidualBlock(nn.Module)
class AttentionBlock(nn.Module)
class DownBlock(nn.Module)
class UpBlock(nn.Module)
class MiddleBlock(nn.Module)
class Upsample(nn.Module)
class Downsample(nn.Module)

class UNet(nn.Module):
    def __init__()
    def forward(self, x: torch.Tensor, t: torch.Tensor)
```

DDPM 训练(Cont.)

在实际训练循环中，对于每个 x_0，需要：

1. 随机生成一系列时间 t。
2. 根据 t 生成随机的单位高斯噪声。
3. 对 x_0 加噪得到 x_t 。
4. 使用模型预测从x_0 到 x_t 的噪声 $\epsilon_\theta^{(t)}(x_t)$ 。
5. 将预测的噪声与生成的噪声真值计算 MSE 损失，并通过反向传播更新模型参数。

本实验将提供 train.py，要求补充代码中的 TODOs。

注意实际训练中以批为基础训练，因此在补全代码时候请注意变量的 batch 维度。

DDPM 训练(Cont.)

根据论文 Progressive Distillation for Fast Sampling of Diffusion Models，让模型预测一个速度 v 会比直接预测噪声 ϵ 具有更好的数值稳定性。训练循环即变为：

1. 随机生成一系列时间 t。
2. 根据 t 生成随机的单位高斯噪声。
3. 对 x_0 加噪得到 x_t 。
4. 使用模型预测从x_0 到 x_t 的速度 $v_\theta^{(t)}(x_t)$ 。
5. 将预测的速度与生成的噪声对应的速度真值计算 MSE 损失，并通过反向传播更新模型参数。

DDPM 训练(Cont.)

在实际训练循环中，对于每个 x_0，需要：

1. 随机生成一系列时间 t。
2. 根据 t 生成随机的单位高斯噪声。
3. 对 x_0 **加噪**得到 x_t 。
4. 使用模型预测从x_0 到 x_t 的噪声 $\epsilon_\theta^{(t)}(x_t)$ 。
5. 将预测的噪声与生成的噪声真值计算 MSE 损失，并通过反向传播更新模型参数。

加噪通过类 **DDIMScheduler** 实现。

DDPM 推理

在训练完得到模型后，本实验要求独立完成 infer.py ，能够载入预训练的 epsilon-prediction （BONUS: v-prediction) 模型，推断生成新的手写数字图片。

其中生成新图片，即**采样**，也通过类 **DDIMScheduler** 实现。

DDPM Scheduler

DDIMScheduler 类是本实验的核心内容。本实验将提供 scheduler.py，要求补充代码中的 TODOs：

- 补全 __init__ 中设置 $\beta_t, \alpha_t, \bar{\alpha}_t$ 的代码。
- 根据加噪公式补全 add_noise 方法。
- 根据去噪公式补全 denoise 方法。
- 使用 denoise 方法补全 sample 方法。
- **BONUS：根据论文或博客完成 get_velocity 方法。**

RECALL

$$\alpha_t = 1 - \beta_t \quad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

$$x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$$

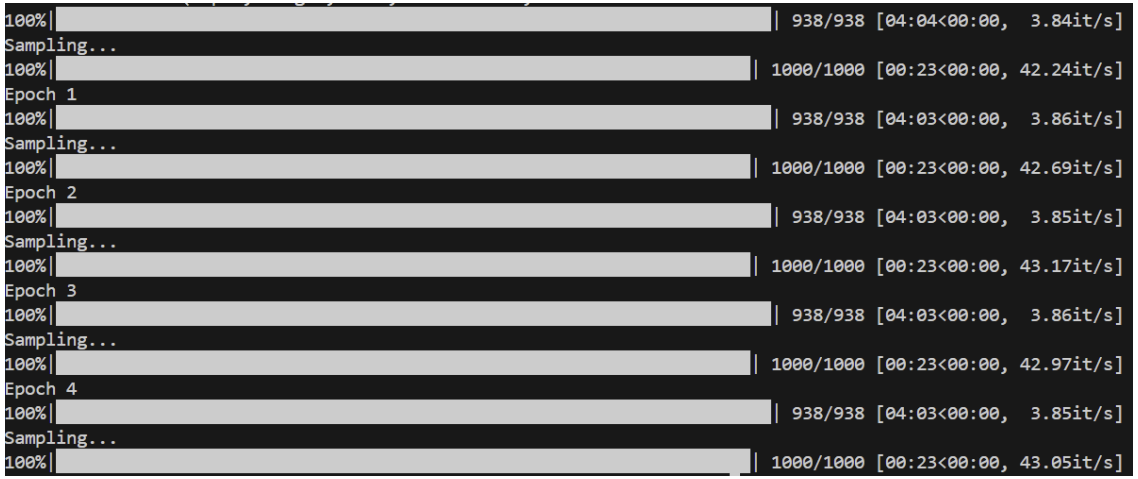
采用 DDIM 去噪公式

DDPM Scheduler(Cont.)

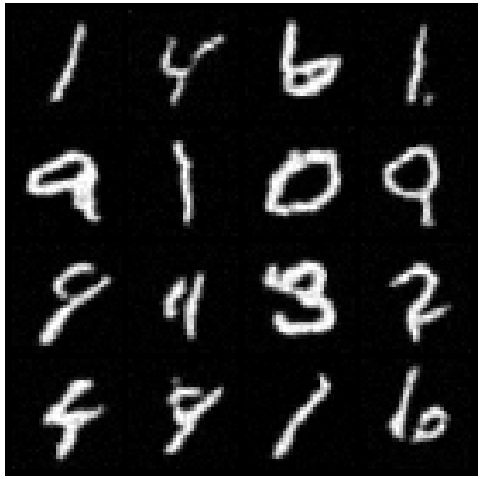
Sample 方法

```
def sample(self, save_path: str, model: UNet, eta: float,
    n_samples: int = 16,
    image_channels: int = 1,
    image_size: int = 32,
    inference_steps: int = 1_000,
    device: torch.device = torch.device('cuda'),
):
    with torch.no_grad():
        x = ...                                          # 创建单位高斯变量
        t_stepsize = self.n_steps // inference_steps     # 指定去噪步数
        for t_ in tqdm.trange(0, inference_steps):       # 逐步去噪
            t = ...                                      # 获得时间步
            x = self.denoise(
                ...
            )
        torchvision.utils.save_image(x.clamp(0., 1.), save_path, nrow=4)
```

期望结果



训练过程



推理结果

思考题

- 模型预测类型对生成效果有什么影响？
- 推断步数对生成效果有什么影响？
- 回忆 DDIM 的去噪公式，其中 $\sigma_t^2 = \eta\tilde{\beta}_t$ 是去噪过程中的噪声方差。那么 η 的变化对生成效果有什么影响？

AGAIN

- 两节课的内容合并在一个压缩包，命名为 Lab6-学号-姓名.zip/7z，在**4月15日23:59:59**前提交至学在浙大。
- 你提交的压缩包应具有如右所示的文件结构。
- 不要提交训练好的模型文件，不要提交数据集。请在报告中尽可能详细地述明实验过程及结果。

```
. Lab6-stuid-name.zip
├── DDPM
│   ├── infer.py
│   ├── scheduler.py
│   ├── train.py
│   └── unet.py
├── UNet
│   ├── infer.py
│   ├── try.py
│   └── unet.py
└── Lab6-stuid-name-report.pdf
```

Questions are welcome

计算摄影学 2025春夏

2025/4/1

特别感谢：2024年计算摄影学助教周健均对本实验的贡献

Computation Photography