Lab 13: Diffusion Models

Department of Computer Science

National Tsing Hua University, Taiwan

2025

Outline

- Introduction to diffusion model
- Denoise Diffusion Probabilistic Model
- Assignment

Outline

- Introduction to diffusion model
- Denoise Diffusion Probabilistic Model
- Assignment

Introduction to Diffusion Models

- Models like VAEs, GANs, and flow-based models proved to be great success in generating high-quality content, especially images.
- Diffusion Models are a class of generative models inspired by thermodynamics that has proven to be better than previous approach
- They use **forward** and **reverse** process to generate high-quality images
- Applications include image inpainting, super resolution, image generation, etc.

Introduction to Diffusion Models

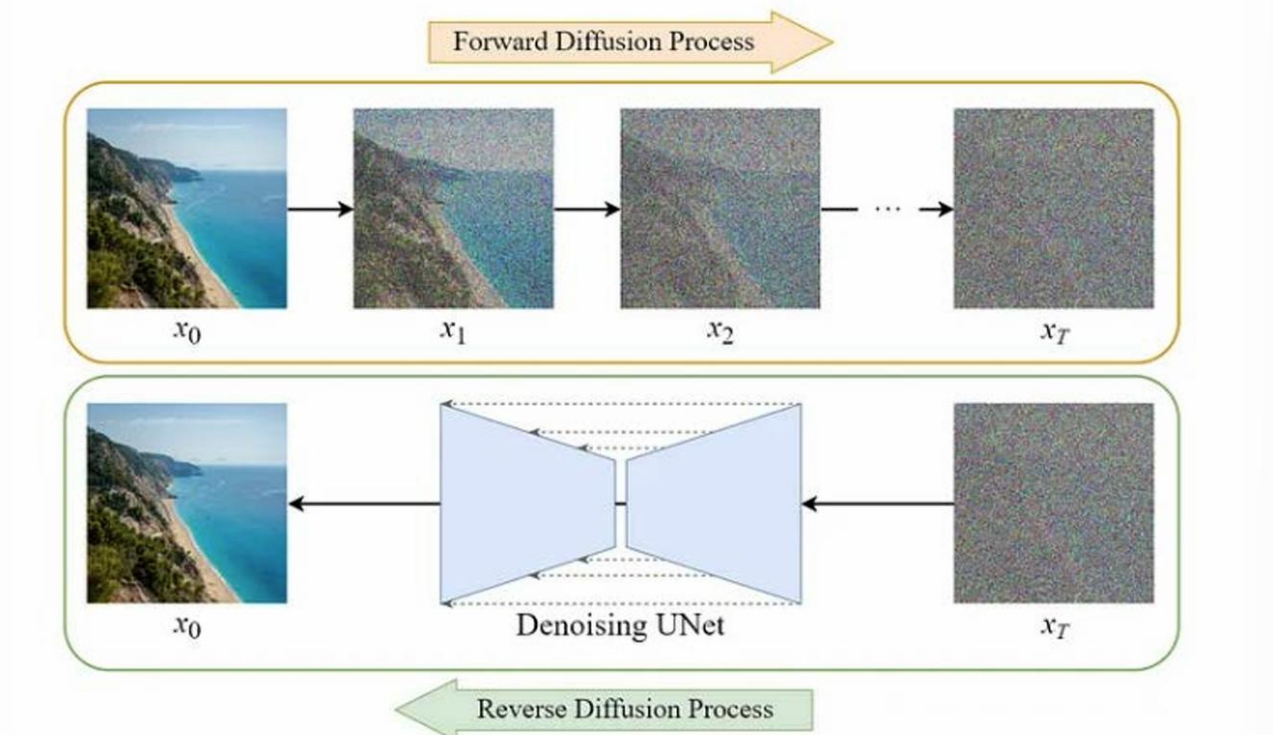
Flow-based models

- **Invertibility:**
  - Flow-based models learn **invertible transformations** that map data x to latent variables z.
  - These transformations convert a **complex distribution** (e.g., natural images) into a **simple distribution** (usually an isotropic Gaussian).
  - Because each step is invertible, the model can also map the Gaussian latent back to the original data distribution for generation
- **Incremental Transitions**
  - This approach supports layered transformations, ensuring that each step introduces small and controllable change.

Introduction to Diffusion model

Diffusion models

- **Forward process**
  - Gradually add noise to input data
- **Reverse process**
  - Step-by-step denoising to reconstruct original data



Outline

- Introduction to diffusion model
- Denoise Diffusion Probabilistic Model
- Assignment

Denoise Diffusion Probabilistic Model (DDPM)

Forward process (Diffusion process):

- Defined as a Markov chains parameterized by a variance schedule $\beta_1, ..., \beta_T$

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$
$$\Rightarrow x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0,1)$$

- Each step depends only on the previous step, following the property of Markov chains.
- A notable property $q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$

$$\Rightarrow x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0,1)$$

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

DDPM

Reverse process:

- They assume a revers process is also a Markov chain with Gaussian transitions: $q(x_{t-1}|x_t) \approx p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \sigma_t^2 I)$

$$\Rightarrow x_{t-1} = \mu_\theta(x_t, t) + \sigma_t z, \quad z \sim \mathcal{N}(0, 1)$$

- Originally, the model is train to minimize the variational lower bound. But the DDPM derivation show that predicted mean can be rewritten into predicted noise

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}})\epsilon_\theta(x_t, t)$$

- Therefore, training simplifies to minimizing the mean squared error between the true noise add in the forward process and the predicted noise $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, 1)$
- We can generate image by

$$q(x_T)p_\theta(x_{T-1}|x_T), ..., p_\theta(x_{t-1}|x_t), ..., p_\theta(x_0|x_1), \quad q(x_T) \sim \mathcal{N}(0, 1)$$

DDPM

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

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **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** $\mathbf{x}_0$

DDPM

Key Modules in the implementation:

- Gaussian Diffusion Utilities:
  - Manage the scheduling and progression of noise
  - Support data processing for both forward and reverse processes.
- Network Architecture:
  - Based on an optimized U-Net structure, designed to suit the characteristics of diffusion models
- Training:
  - Using the MSE loss function to predict noise, combined with **EMA** for weight updates

Outline

- Introduction to diffusion model
- Denoise Diffusion Probabilistic Model
- Assignment

Assignent

Requirements

1. **Diffusion Schedule**
   - Complete the TODO sections in `diffusion_schedule`
   - Implement continuous version of the **cosine schedule**
2. **DDIM reverse process**
   - Complete the TODO section in `denoise` and `reverse_diffusion`
   - Implement the deterministic **reverse process of DDIM**.
3. Briefly summarize what you did and explain the performance results

Assignment

Diffusion Schule

- Controls how much noise is added at each step during the forward process.
- Defined by parameters β_t (noise increment) and α_t (data retention)
- Ensures smooth transition from clean data to pure noise
- Helps the model effectively learn the reverse process (denoising)
- They are many kind of schedule, linear, cosine…

Assignment

Cosine schedule

- Recall that noisy image is compute by $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, 1)$
- When x_0 is normalized image (mean is 0 and variance is 1)
- Since x_0 and ϵ are independent, $\mathrm{Var}(x_t) = \bar{\alpha}_t \cdot 1 + (1 - \bar{\alpha}_t) \cdot 1 = 1$
- Define
  - $\sqrt{\bar{\alpha}_t}$ as the **signal rate(t)**
  - $\sqrt{1 - \bar{\alpha}_t}$ as the **noise rate(t),**
  - then we have $\big(\mathrm{signal\_rate}(t)\big)^2 + \big(\mathrm{noise\_rate}(t)\big)^2 = 1$. This condition matches the unit circle which can be express as $\cos^2\theta(t) + \sin^2\theta(t) = 1$
- Therefore, to decide the signal and noise rate at t is decide and angle $\theta(t)$
- Given Max single rate s_{max}, at $t = 0 \Rightarrow \theta_{start} = \arccos(s_{max})$
- Given Min single rate s_{min}, at $t = 1 \Rightarrow \theta_{end} = \arccos(s_{min})$
- For any $t \in [0, 1]$, we have

$$\theta(t) = (1 - t)\theta_{start} + t\theta_{end}$$

$$\mathrm{signal\_rate}(t) = cos\big(\theta(t)\big), \quad \mathrm{noise\_rate}(t) = \sin\big(\theta(t)\big)$$

Assignment

Bottleneck of DDPM:

- The length T of the forward process is an important hyperparameter in DDPM
- From a variational perspective, a large T allows the reverse process to be close to a Gaussian
- This motivates the choice of large T value, such as $T = 1000$
- However, as all T iteration have to be performed sequentially, to obtain a sample x_0 from DDPM is much slower than sampling from other deep generative models.

Assignment

Denoising Diffusion Implicit Models (DDIM)

- Re-design the reverse process of DDPM into a **deterministic, non-Markovian** mapping
- Since DDIM removes the stochastic noise term in the DDPM reverse process, sampling becomes fully deterministic.
- Therefore, **non-fixed** and **larger step sizes** can be used, enabling faster sampling with fewer steps.

https://arxiv.org/abs/2010.02502

Assignment

Denoise Diffusion Implicit models (DDIM)

- They found that changing the reverse transition to

$$q_\sigma(x_{t-1}|x_t, x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\frac{x_t-\sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}}, \sigma_t^2 I)$$

  a new family of Gaussian conditionals indexed by σ.
- The marginal $q_\sigma(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$ is same as DDPM. Therefore, the same noise predictor $\epsilon_\theta(x_t, t)$ trained in DDPM can be used directly
- Note that using the DDPM forward process $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0,1)$ we have

$$x_0 \approx f_\theta(x_t, t) = \frac{x_t-\sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t,t)}{\sqrt{\bar{\alpha}_t}}, \quad \frac{x_t-\sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}} \approx \epsilon_\theta(x_t, t) \qquad (1)$$

- Plugging (1) into the DDIM reverse transition:

$$q_\sigma(x_{t-1}|x_t, f_\theta(x_t,t)) = \mathcal{N}(\sqrt{\bar{\alpha}_{t-1}}f_\theta(x_t,t) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(x_t,t), \sigma_t^2 I)$$

- So the update rule becomes $\Rightarrow x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}f_\theta(x_t,t) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(x_t,t) + \sigma_t^2 z$
- When $\sigma = 0$, we have deterministic reverse process

$$\Rightarrow x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}f_\theta(x_t,t) + \sqrt{1-\bar{\alpha}_{t-1}}\cdot\epsilon_\theta(x_t,t)$$

Assignment

Requirement

- Complete the TODO sections in the following code, including `diffusion_schedule`, `denoise` and `reverse_diffusion`.
- Briefly summarize what you did and explain the performance results.
- This assignment does not specify a clear model loss threshold.
- However, please train your model to the extent that it can generate images with clearly recognizable flowers.