Lab 13: Diffusion Models

Department of Computer Science,
National Tsing Hua University, Taiwan
2024

Outline

- Introduction to Diffusion Models
- Denoising Diffusion Probabilistic Models(DDPM)
- Assignment

Outline

- Introduction to Diffusion Models
- Denoising Diffusion Probabilistic Models(DDPM)
- Assignment

Diffusion Models

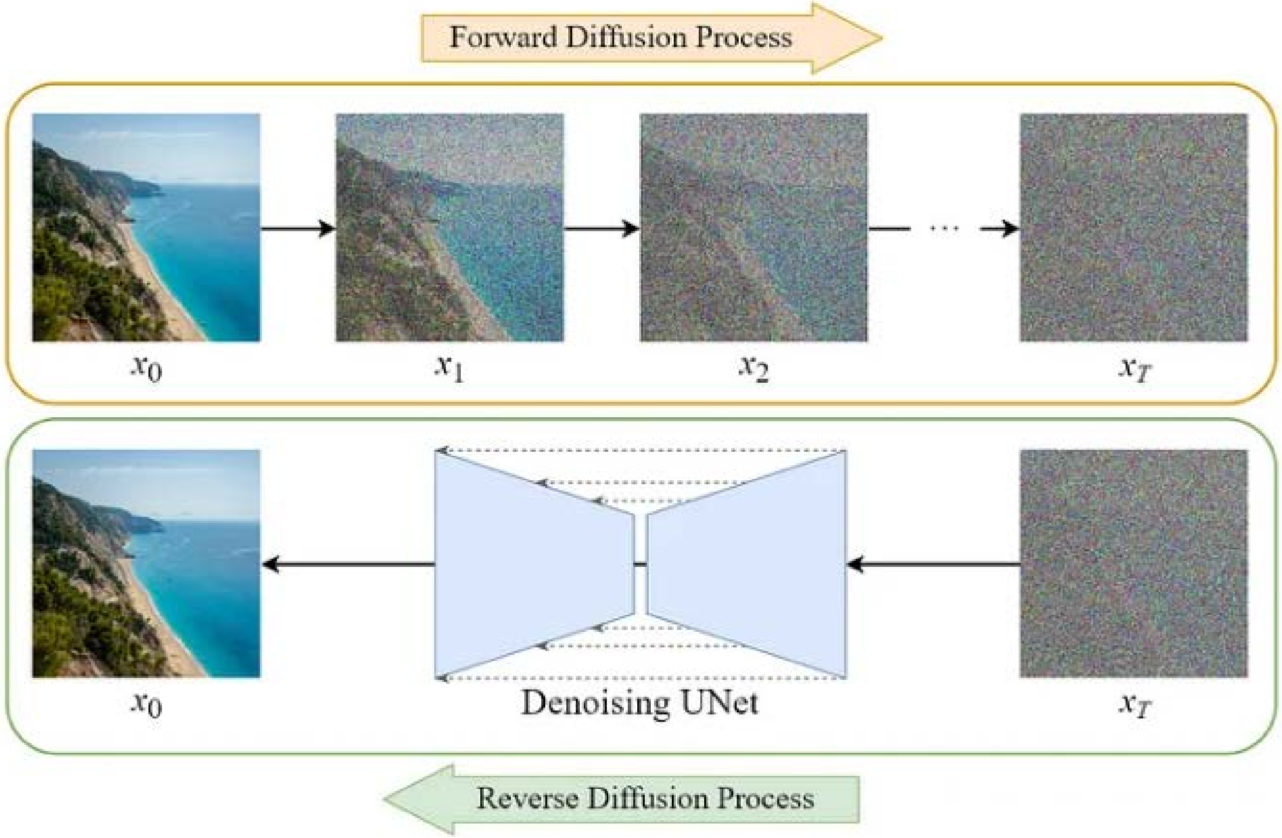
- Models like VAEs, GANs, and **flow-based models** proved to be a great success in generating high-quality content, especially images.
- Diffusion Models are a class of generative models inspired by thermodynamics that has proven to be better than previous approaches.
- They use **forward** and **reverse**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

Flow-based models

- **Invertibility**: Flow-based Models provide a framework for data transformation via invertible mappings, gradually converting simple distributions (e.g., Gaussian) into complex ones (e.g., image distributions).
- **Incremental Transitions**: This approach supports layered transformations, ensuring that each step introduces small and controllable changes.

Forward and Reverse Processes

- Forward Process: Gradually add noise to input data.
- Reverse Process: Step-by-step denoising to reconstruct original data.



Outline

- Introduction to Diffusion Models
- Denoising Diffusion Probabilistic Models(DDPM)
- Assignment

Markov Chains in DDPM

- Forward Process:
  - Progressive Noise Addition: DDPM uses Markov chains to iteratively add Gaussian noise to the data, transforming it step-by-step from structured data into pure noise.
  - Each step depends only on the previous step, following the property of Markov chains.
- **Key Point**: The Markov chain ensures that each step is controlled and reversible, which is critical for the reverse process.

Markov Chains in DDPM

- Reverse Process:
  - Step-by-Step Denoising: The reverse process in DDPM is modeled as an inverse Markov chain.
  - Starting from pure noise , the model predicts the noise at each step to denoise progressively.
- **Key Point**: The Markov chain structure provides a stable and consistent framework for generating high-quality images, though it requires many steps to complete the process

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:
  - Use the MSE loss function to predict noise, combined with EMA for weight updates.

Outline

- Encoder-Decoder model
- Attention-based
- Assignment

Improvements in DDIM

- **Deterministic Mapping in the Reverse Process**:
- DDIM eliminates the stochastic nature of Markov chains in the reverse process.
- Instead, it introduces a new derivation, transforming the reverse process into a deterministic mapping.

Improvements in DDIM

- **Diffusion Schedule**
- controls how much noise is added at each step during the forward process.
- Defined by parameters βt (noise increment) and αt (data retention).
- Ensures smooth transition from clean data to pure noise.
- Helps the model effectively learn the reverse process (denoising).

Improvements in DDIM

- **Efficiency Gains**:
- This design allows DDIM to reduce the number of reverse steps significantly, as it no longer relies on random sampling.
- Markov chains require a large number of steps to ensure the continuity of generation, while DDIM directly performs deterministic calculations with larger step sizes, achieving faster generation.

Requirements

- 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.