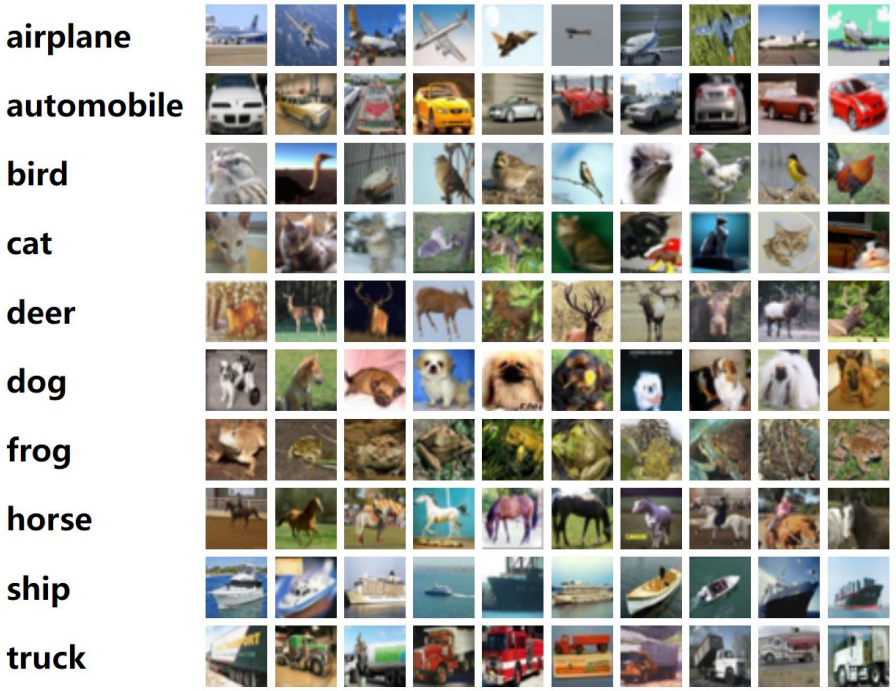
Assignment II: Image Generation

GitHub page

Zhiqian Lan (lanzq@connect.hku.hk)

1 Introduction

- What you have learned in Assignment I



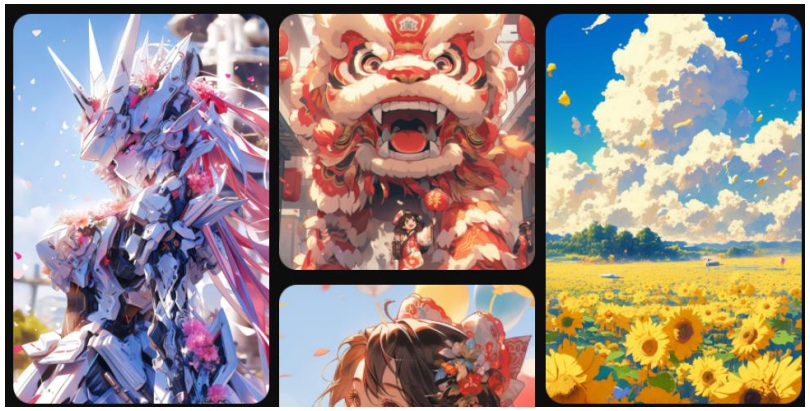
A1: Image Classification
Image → Label

A2: Image Generation
"Cat" - generated samples
Generative model

1 Introduction

- What is a generative model
  - A model learns to capture the underlying data distribution to **generate new data samples** similar to the training data.
- Types of generative model
  - Generative Adversarial Networks (GANs)
  - Variational Autoencoders (VAEs)
  - Diffusion Models: a newer class of generative models

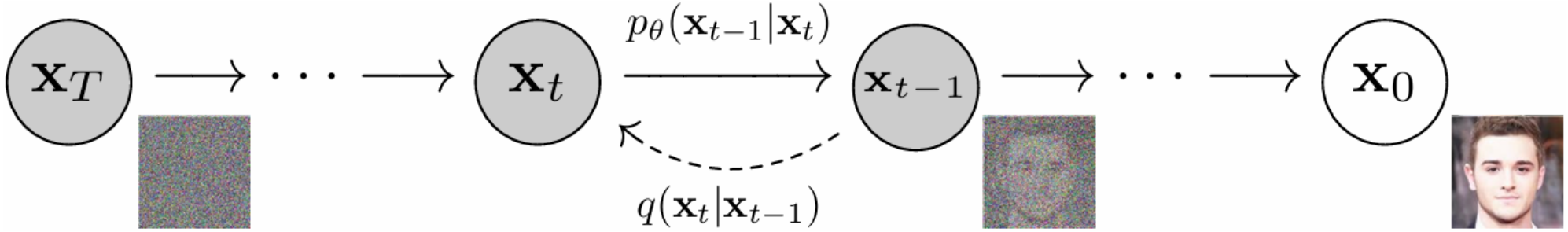
Stable diffusion: text to image



AIGC by Midjourney

1 Introduction

- Denoising Diffusion Probabilistic Models (DDPM)

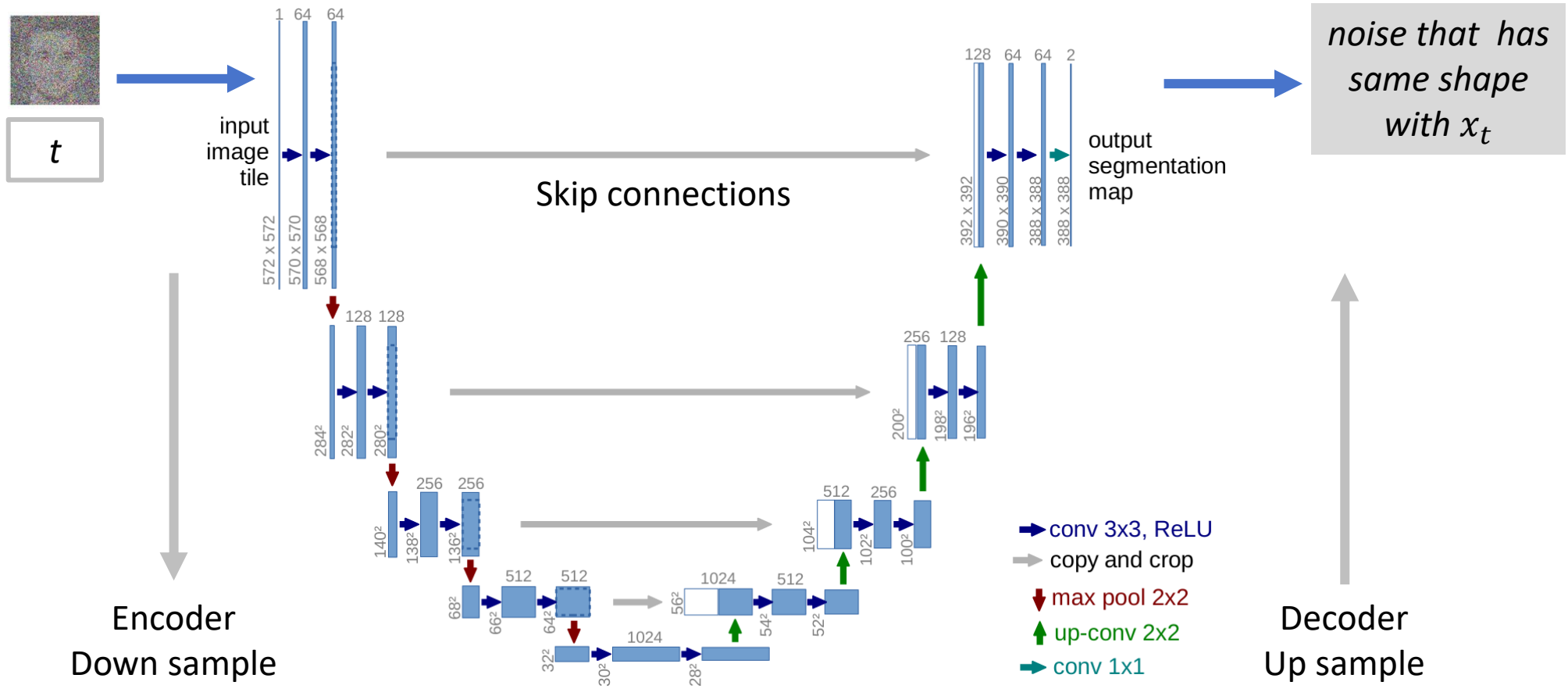


  - Forward process: gradually **adds** noises to an image over several timesteps, converting a **clean image** into almost **pure Gaussian noise**.
  - Reverse process: denoise the noisy samples step by step, reconstructing the original data. Specifically, to **predict the noise that needs to be subtracted** from the noisy input at each step

How can we formulate the model ϵ_θ?

1 Introduction

- U-Net architecture for reverse diffusion modeling



2 Working on the Assignment

- This assignment, in one word, is to train a DDPM with U-Net architecture on the **MNIST handwritten digits** dataset.

- MNIST dataset:

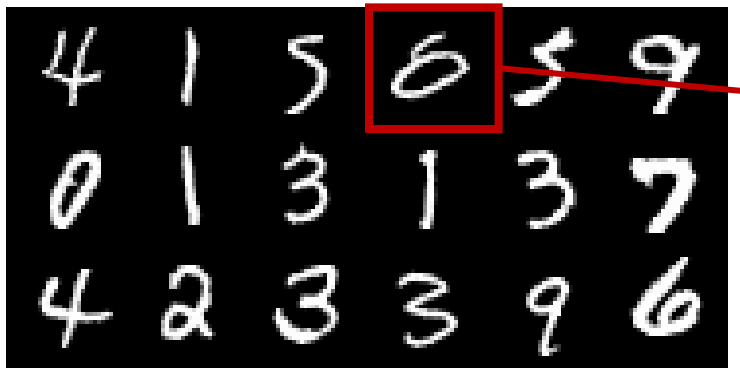
2.1 Task 1: Fill in the blank (70%)

We have provided you with the initial codebase, which is defined in three Python files:

- **model.py** - Contains the definition of the diffusion model.
- **unet.py** - Contains the definition of the U-Net model that handles the denoising process.
- **train_mnist.py** - The main training script. Run `python train_mnist.py` to start training and evaluating the model.

- There are ***5*** code blocks in *model.py* and *unet.py* that require you to complete them.
- Successfully implementing the code and showcasing the generated handwritten digits in the report will earn full marks.

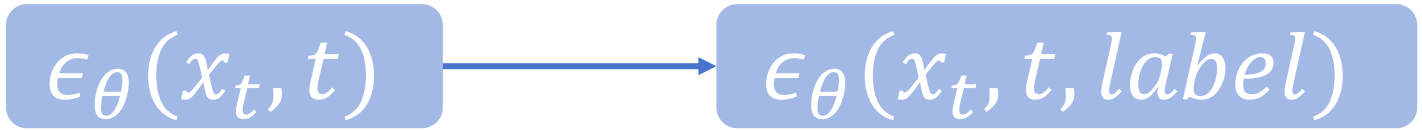
An example



It is **acceptable** to have a few generated samples like this.

2.2 Task 2: Conditional generation (10%)

- This task involves extending the existing code to implement conditional generation, that is, to enable the **generation of images corresponding to specific digits**. The implementation should be detailed in your final report.



Unconditional
Task 1

Conditional
Task 2

- You may look into the handing of time step input, and think about how to cope with the input of *label* in a similar way.

2.3 Task 3: Write a report (20%)

- No more than 2 pages
- Your report should be structured into three main sections: **Introduction, Method, and Experiment**
- It should highlight the effectiveness of the image generation, showcasing outputs for all digits from 0 to 9.
- Your analysis is important. [examples]

3 Files to submit

1. **Final Report** (PDF, up to 2 pages)

2. **Codes**
   a) 3 python files: model.py, unet.py, train_mnist.py

   b) README.txt if you added some python files.

   c) One .ipynb file is also acceptable if you find it more convenient to use Jupyter notebook (Pls copy and paste code into a notebook by yourself)

3. Model Weights

4 Important Dates

- Oct. 6, 2024 (Sun.): The assignment release.

- Nov. 10, 2024 (Sun.): Submission deadline (23:59 GMT+8).

Late submission policy:

- 10% for late assignments submitted within 1 day late.
- 20% for late assignments submitted within 2 days late.
- 50% for late assignments submitted within 7 days late.
- 100% for late assignments submitted after 7 days late.

5 Need More Support?

- For any questions about the assignment which potentially are common to all students, you shall first look for related resources as follows,
  - ✓ We encourage you to use GitHub Issues of this repository.
  - ✓ Or if you prefer online doc: Discussion doc.

- For any other private questions, please contact Zhiqian Lan via email.