CS 486/686
Generative AI: Diffusion & Multimodal

Yuntian Deng

Lecture 23

VAEs · diffusion · text-to-image · CLIP

Learning goals

- Place the **generative model families** on one map.
- Explain the **VAE**: ELBO and the reparameterization trick.
- Derive the **diffusion** forward/reverse process and its loss.
- Explain **guidance** and **latent diffusion** (text-to-image).
- Describe **multimodal** models (CLIP).

Recap: four ways to generate

Autoregressive

Next-token $\prod_t p(x_t \mid x_{<t})$. Powers LLMs (L22); also images/audio.

GAN

Generator vs discriminator. Sharp but unstable — now largely legacy.

VAE

Probabilistic autoencoder; sample a latent code, decode it.

Diffusion

Learn to denoise. Today's state of the art for images & video.

L22 did autoregressive text. Today: VAEs → diffusion → multimodal.

Variational autoencoders (VAEs)

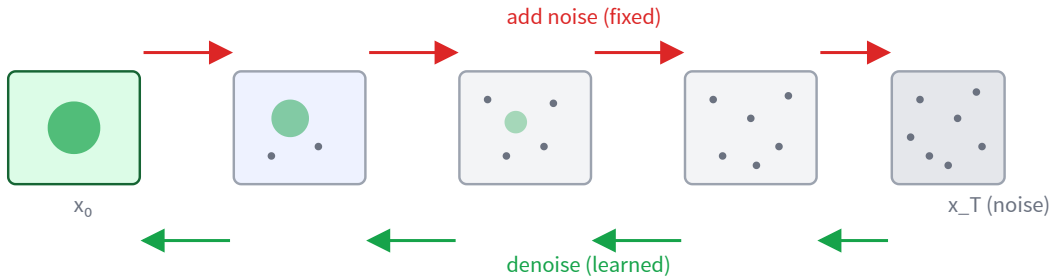
An autoencoder (L17) with a **probabilistic** bottleneck: the encoder outputs a distribution $q_\phi(z \mid x)$; the decoder $p_\theta(x \mid z)$ reconstructs. Train by maximizing the **ELBO**:

$$\log p(x) \;\geq\; \underbrace{\mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x \mid z)\big]}_{\text{reconstruction}} \;-\; \underbrace{\mathrm{KL}\big(q_\phi(z \mid x) \,\|\, p(z)\big)}_{\text{stay near prior } \mathcal{N}(0,I)}$$

- **Reparameterization trick:** sample $z = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon, \varepsilon \sim \mathcal{N}(0, I)$ — so gradients flow through the sampling.
- Generate by sampling $z \sim \mathcal{N}(0, I)$ and decoding. Smooth latent space, but samples can be blurry.

Diffusion: destroy, then learn to restore

Slowly add Gaussian noise until an image becomes pure static (forward). Then train a network to **undo** one step of noise (reverse).



The forward (noising) process

Add a little Gaussian noise at each of T steps, with a schedule $\beta_1, \ldots, \beta_T$:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\big(x_t;\ \sqrt{1-\beta_t}\, x_{t-1},\ \beta_t I\big)$$

Conveniently, we can jump to *any* step in closed form (let $\bar{\alpha}_t = \prod_{s \le t}(1-\beta_s)$):

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I)$$

No learning here — the forward process is fixed. As $t \to T$, x_t becomes pure noise.

The reverse process: learn to denoise

Train a network $\varepsilon_\theta(x_t, t)$ to **predict the noise** that was added. Since $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon$, knowing ε lets us step back toward x_0.

$$\mathcal{L} = \mathbb{E}_{x_0,\,\varepsilon,\,t}\left[\,\left\|\varepsilon - \varepsilon_\theta(x_t, t)\right\|^2\,\right]$$

1. Take a clean x_0, pick a random step t, draw noise ε.
2. Form x_t; ask the network to predict ε.
3. Minimize squared error — a simple regression, no adversarial game.

Sampling: noise → image

`sample()`

1. Start from pure noise $x_T \sim \mathcal{N}(0, I)$.
2. For $t = T, T-1, \ldots, 1$: predict $\varepsilon_\theta(x_t, t)$, remove a bit of noise to get x_{t-1}.
3. Return x_0 — a fresh sample.

DDPM takes many steps (slow, high quality); **DDIM** and distillation cut it to a few.

Steering the output: conditioning & guidance

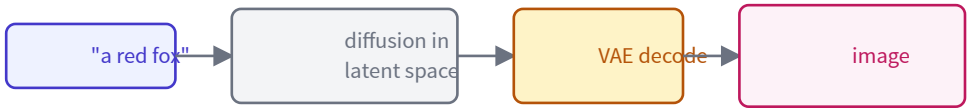
Condition the denoiser on a prompt c (e.g. text): $\varepsilon_\theta(x_t, t, c)$. To make it follow the prompt *more* strongly, use **classifier-free guidance**:

$$\tilde{\varepsilon} = \varepsilon_\theta(x_t, t, \varnothing) + w\left(\varepsilon_\theta(x_t, t, c) - \varepsilon_\theta(x_t, t, \varnothing)\right)$$

- Train with the condition sometimes dropped ($\varnothing$), so the model learns both conditional and unconditional denoising.
- Guidance weight w: higher = more prompt-faithful (but less diverse).

Text-to-image: latent diffusion

Pixels are huge, so run diffusion in a compressed **latent** space (a VAE encodes/decodes), conditioned on a text embedding — this is Stable Diffusion.



Text conditions the denoiser via cross-attention (L21). Latent space makes it fast enough for one GPU.

Multimodal models: a shared space (CLIP)

To connect *text* and *images*, train an image encoder f and text encoder g so matching pairs land close together (contrastive objective, L17):

> maximize $f(\text{image})^\top g(\text{text})$ for true pairs, minimize it for mismatched pairs

- Enables **zero-shot** classification: score an image against text labels.
- Provides the text understanding that guides text-to-image models.
- Same idea scales to **vision-language models** that chat about images.

Frontier: generating interfaces (NeuralOS)

> Push generation further: instead of one image, generate the **next screen** of an operating system given your mouse and keyboard input — a generative model of an entire UI.

- Combines the ideas in this module: sequence modeling + diffusion decoding.
- A glimpse of generative models as **interactive world models**, not just image makers.

Learning goals (recap) — Next: course recap

✓ The **generative model families** on one map.

✓ **VAE**: ELBO + reparameterization.

✓ **Diffusion**: forward/reverse process + noise-prediction loss.

✓ **Guidance**, **latent diffusion**, and **CLIP** / multimodal.

L24: tie the whole course together — recap & exam prep.