Week 10

# Diffusion Models

[ECEA0649/ECE40049] Deep Learning for Image Processing | Spring 2026

Kihyun Na (Research Professor)
BK21 AI Education and Research Group &
Institute for Information and Communication Technology,
Handong Global University

HANDONG GLOBAL UNIVERSITY

# Curriculum

** Adjusted based on survey results*

| | |
|---|---|
| **Wk 1** | OT + Introduction |
| **Wk 2** | DL Fundamentals Review |
| **Wk 3** | CNN |
| **Wk 4** | From Sequence Modeling to Transformer |
| **Wk 5** | Transformer in Vision |
| **Wk 6** | Detection & Segmentation |
| **Wk 7** | Self-Supervised Learning |
| **Wk 8** | Review Literacy + Role Explanation |
| **Wk 9** | Foundation Models (CLIP, SAM) + Paper #1 |
| **Wk 10** | Diffusion Models + Paper #2 (Today) |
| **Wk 11** | Conditional Generation + Paper #3 |
| **Wk 12** | Vision-Language Models + Paper #4 |
| **Wk 13** | VLM Applications + Paper #5 |
| **Wk 14** | Video Understanding + Paper #6 |
| **Wk 15** | Embodied AI & Robot Vision + Paper #7 |
| **Wk 16** | Miniconference (Final Project) |

Lecture  Lecture + Paper  Miniconference

# Today's Agenda

*Wk 10 = Foundations | Wk 11 = Practice & Conditional Generation*

**01** **Generative Models** — 10 min

What is generation? Discriminative vs Generative, density, model families

**02** **VAE & GAN** — 25 min

Latent variables, ELBO derivation, adversarial training, why diffusion won

**03** **Diffusion Models** — 20 min

Forward/reverse, DDPM training, ELBO → $L_{simple}$, DDIM, U-Net

**04** **Preview** — 5 min

DDIM, applications, what's coming in Wk 11

Part 1

# Generative Models

*What does it mean to "generate"?*

# Discriminative vs Generative

*Generation is the inverse of representation learning*

## Discriminative (Wk 2–9)

$$Learn\ p(y|x)$$

Input: image
Output: label, box, mask

"What is in this image?"

Data → Representation
(abstraction)

All of Wk 3–9

## Generative (Wk 10–11)

$$Learn\ p(x)\ or\ p(x|y)$$

Input: noise (+ condition)
Output: image

"Create a new image"

Representation → Data
(concretization)

Key trick: turn generation into a sequence of supervised problems

# Generative vs Discriminative Models

**Discriminative Model**: Learn a probability distribution p(y|x)

**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)

**Data:** x

**Label:** y

Cat

**Density Function**

p(x) assigns a positive number to each possible x; higher numbers mean x is more likely.

Density functions are **normalized:**

$$\int_X p(x)dx = 1$$

Different values of x **compete** for density

# Generative vs Discriminative Models

**Discriminative Model**: Learn a probability distribution p(y|x)

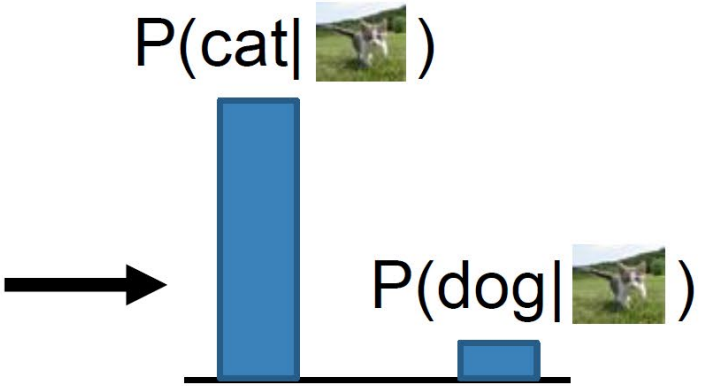
**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)

**Data:** x

**Label:** y

Cat




$$\int_X p(x)dx = 1$$

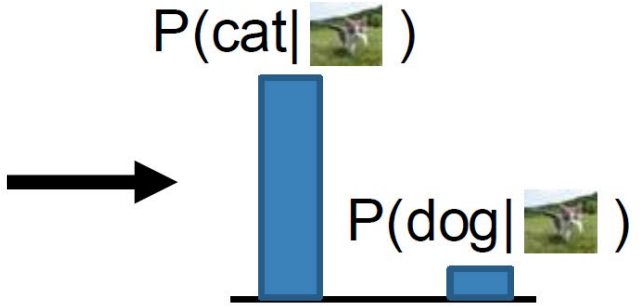
Different values of x **compete** for density

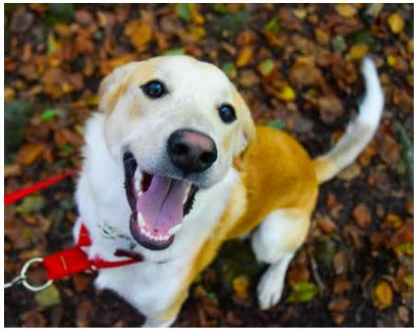
# Generative vs Discriminative Models

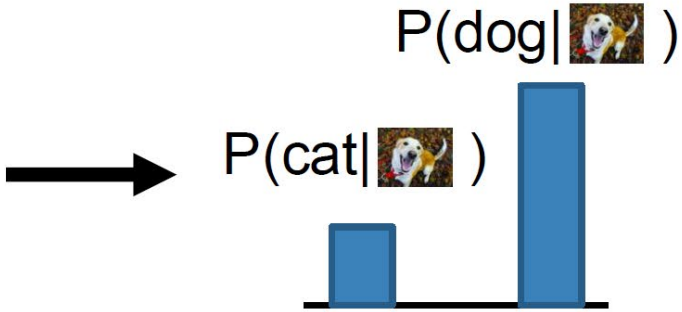
**Discriminative Model**: Learn a probability distribution p(y|x)

**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)





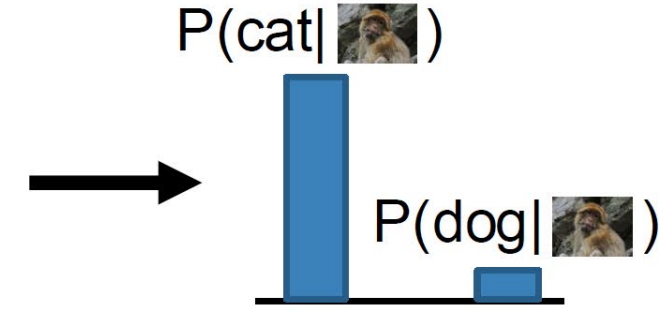

Possible **labels** for each image compete for probability.
No competition between **images**

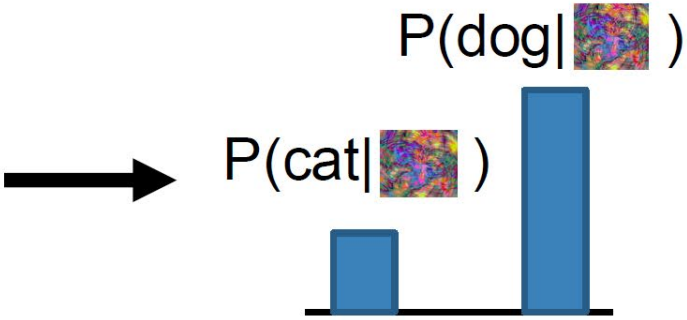
# Generative vs Discriminative Models

**Discriminative Model**: Learn a probability distribution p(y|x)

**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)



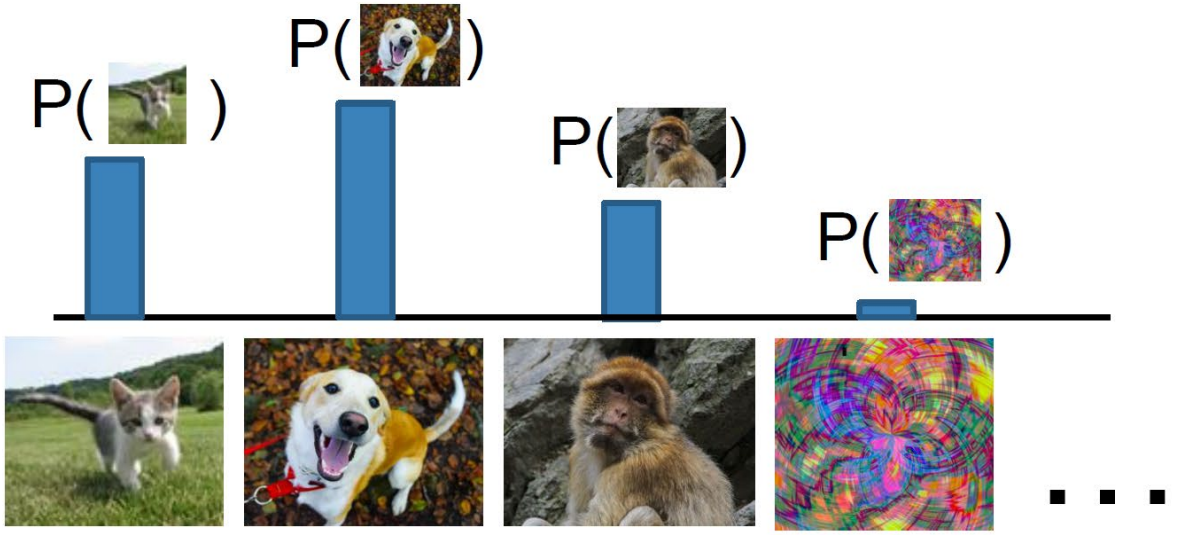

No way to handle unreasonable inputs; must give a label distribution for all possible inputs

# Generative vs Discriminative Models

**Discriminative Model**: Learn a probability distribution p(y|x)

**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)




All possible **images** compete for probability mass

Model can “reject” unreasonable inputs by giving them small probability mass

# Generative vs Discriminative Models

**Discriminative Model**: Learn a probability distribution p(y|x)

**Generative Model**: Learn a probability distribution p(x)

**Conditional Generative Model**: Learn p(x|y)

Recall **Bayes' Rule**:

$$P(x \mid y) = \frac{P(y \mid x)}{P(y)} P(x)$$

Discriminative Model

Conditional Generative Model

Prior over labels

(Unconditional) Generative Model

We can build a conditional generative model from other components … but not common in practice

# Taxonomy of Generative Models

*Direct likelihood (AR, Flow) / Approximate (VAE, Diffusion) / Implicit (GAN)*

### Autoregressive

Explicit, tractable

$$p(x) = \prod p(x_i | x_{\{<i\}})$$

PixelCNN, GPT

Exact likelihood
but sequential
= slow generation

### VAE

Explicit, approximate

Encoder $q(z|x)$
Decoder $p(x|z)$
ELBO lower bound

Fast but blurry
(LDM borrows this
autoencoder idea)

### GAN

Implicit density

Generator vs
Discriminator
Adversarial game

Sharp outputs
but unstable training
& mode collapse

### Flow

Explicit, tractable

Invertible transforms
Exact likelihood
RealNVP, Glow

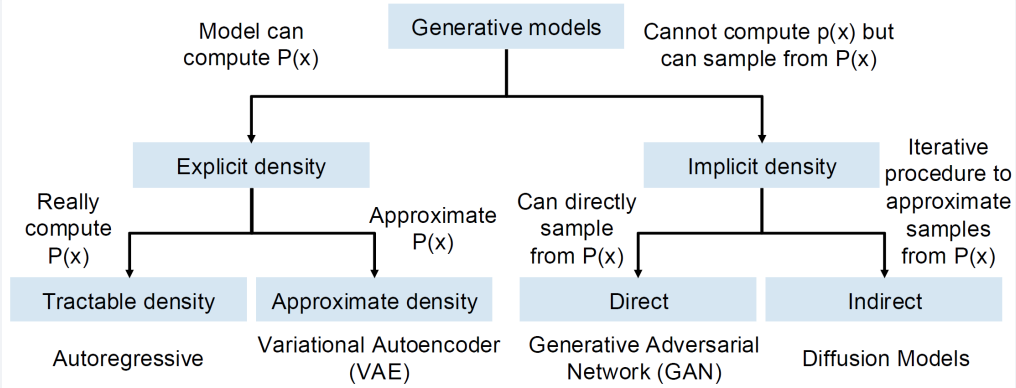
Architecture constraints
→ Inspired Flow Matching
(Wk 11)

### Diffusion

Iterative approximate

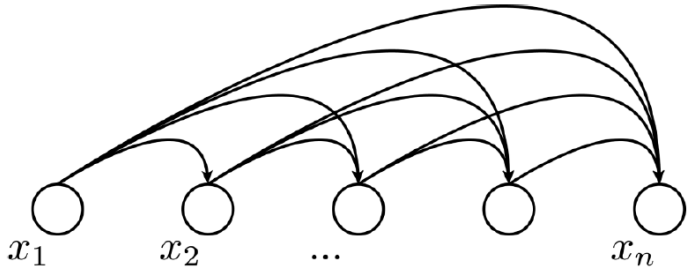
Denoise from noise
DDPM, Score-based

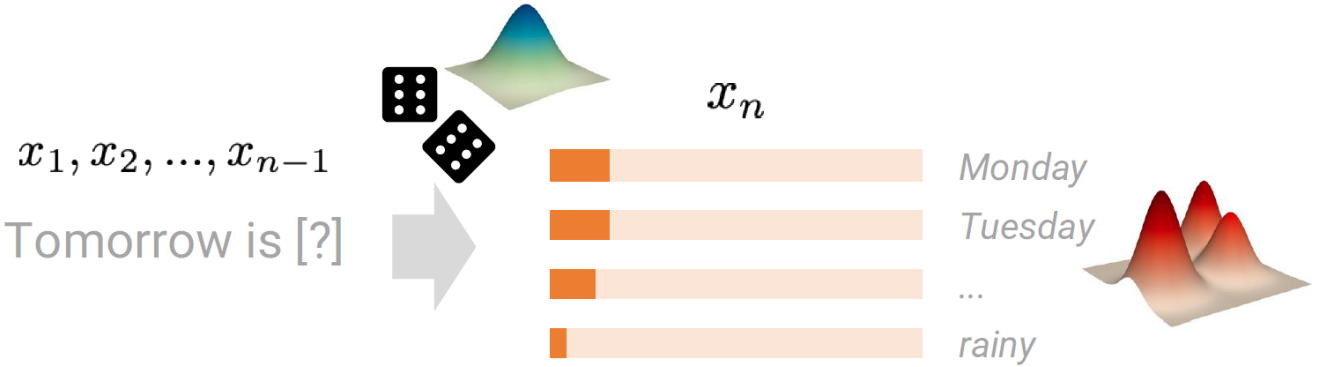
High quality
stable training
diverse outputs
= today's focus



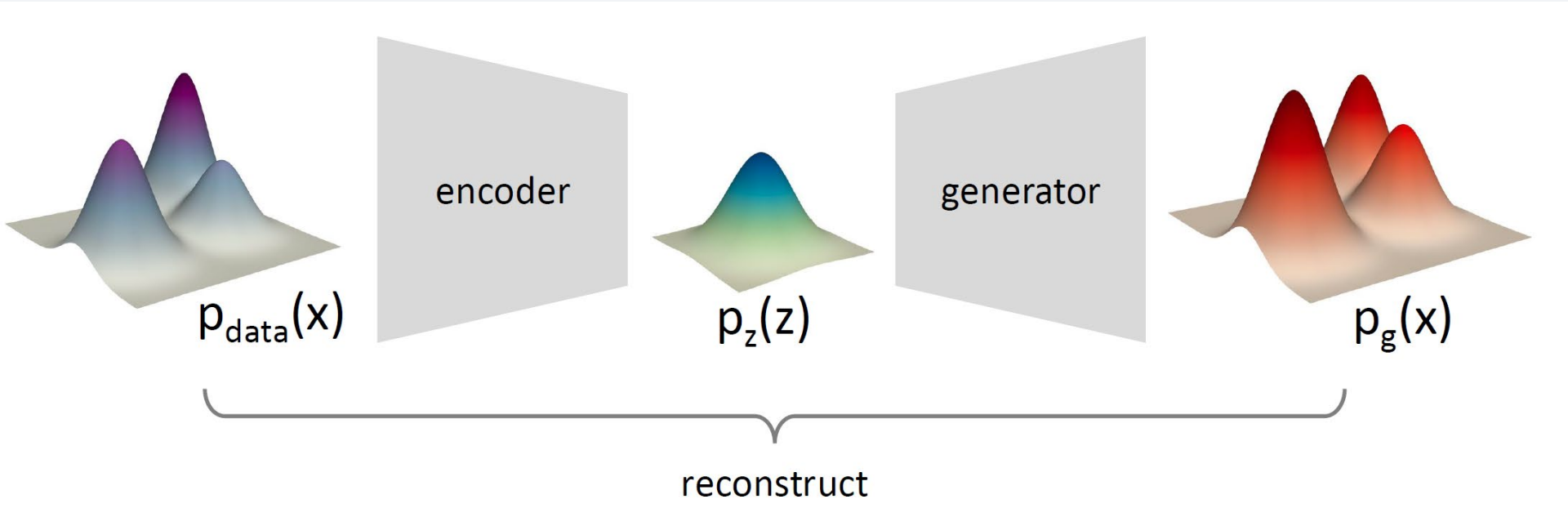
# Autoregressive Models (AR)

$$p(x_1, x_2, \ldots, x_n) = p(x_1)p(x_2 \mid x_1)\ldots p(x_n \mid x_1, x_2, \ldots, x_{n-1})$$

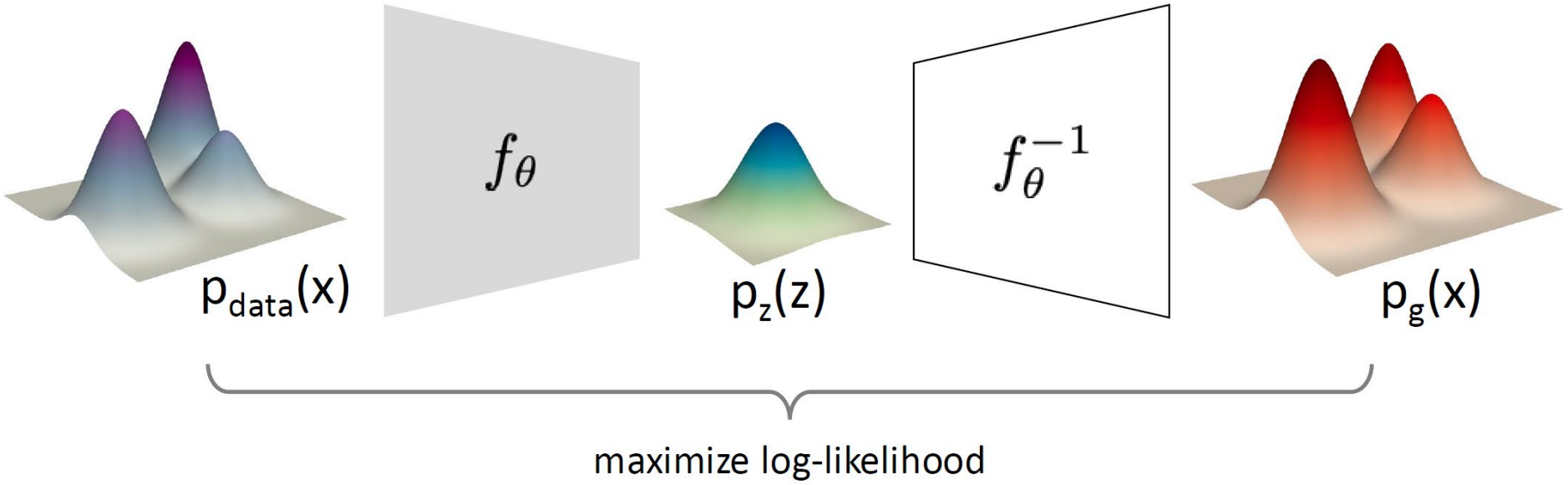


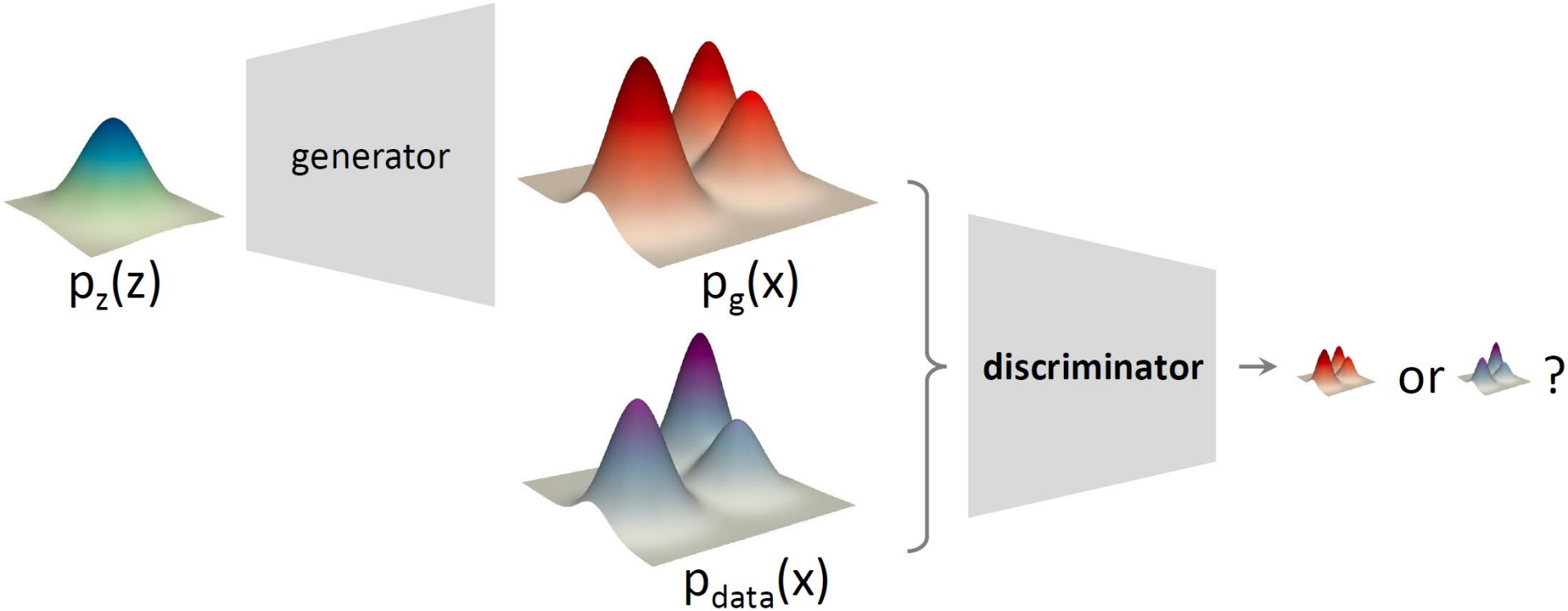
# Variational Autoencoder (VAE)



# Normalizing Flows



# Generative Adversarial Net (GAN)



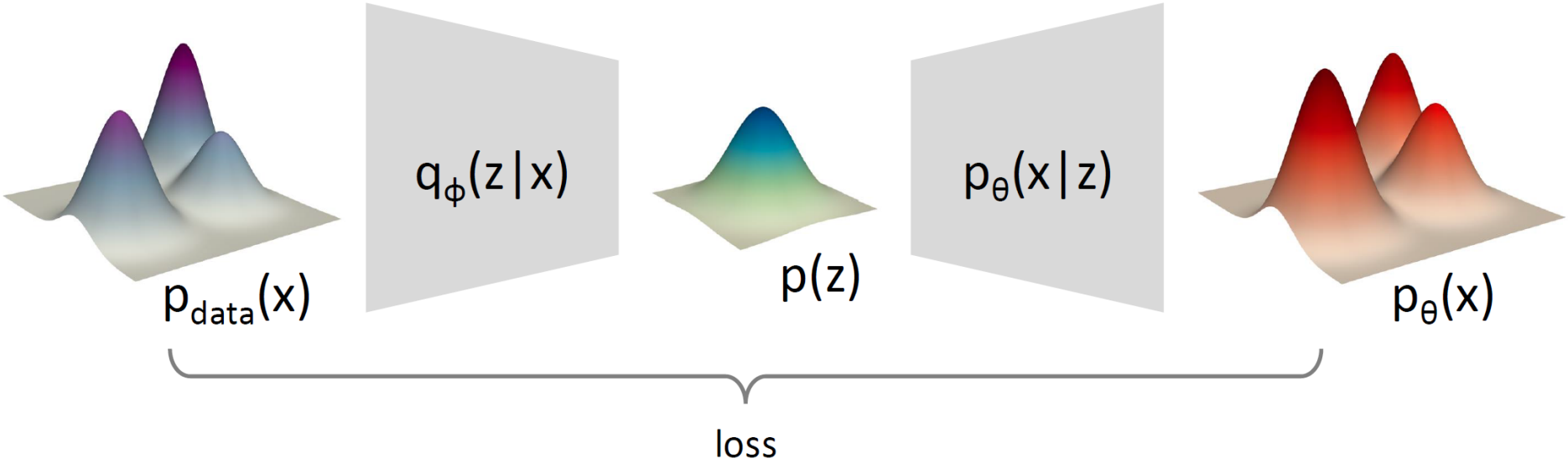
Part 2

# VAE & GAN

*Two classical approaches to generation*

# Variational Autoencoder

- **Encoder**: data to latent, parameterized by $\phi$
- **Decoder**: latent to data, parameterized by $\theta$



# Maximum Likelihood Estimation (MLE)

Minimize KL-Divergence:

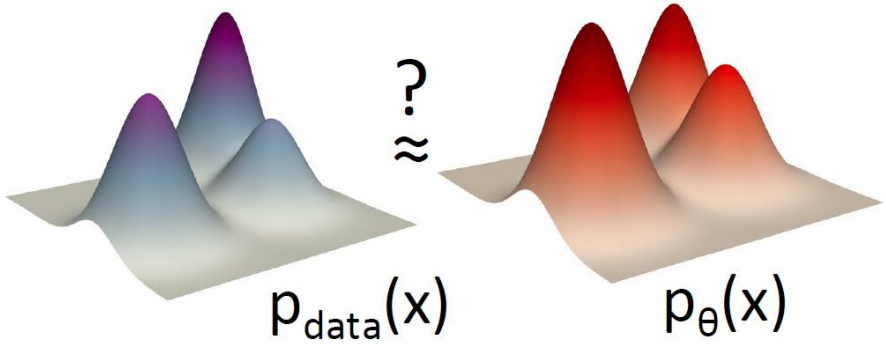
$$\min_\theta \; \mathcal{D}_{\mathrm{KL}}( \, p_{data} \, || \, p_\theta \, )$$

⇒ **Maximize** likelihood:

$$\max_\theta \; \mathbb{E}_{x \sim p_{data}} \log p_\theta(x)$$

tl; dr

$$\begin{aligned}
& \arg\min_\theta \; \mathcal{D}_{\mathrm{KL}}( \, p_{data} \, || \, p_\theta \, ) \\
= \; & \arg\min_\theta \; \sum_x p_{data}(x) \log \frac{p_{data}(x)}{p_\theta(x)} \\
= \; & \arg\min_\theta \; \sum_x -p_{data}(x) \log p_\theta(x) + const \\
= \; & \arg\max_\theta \; \sum_x p_{data}(x) \log p_\theta(x) \\
= \; & \arg\max_\theta \; \mathbb{E}_{x \sim p_{data}} \log p_\theta(x)
\end{aligned}$$



# Maximum Likelihood Estimation (MLE)

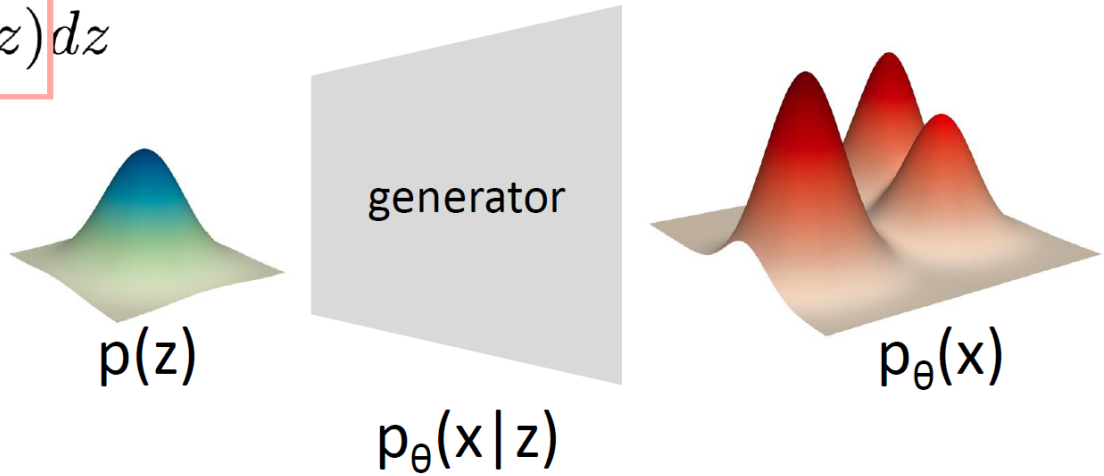
We want to maximize $\mathbb{E}_{x \sim p_{data}} \log p_\theta(x)$

with $p_\theta(x)$ represented as:

$$p_\theta(x) = \int_z p_\theta(x|z) p(z) dz$$

**Two** sets of unknowns:

- We need to optimize for θ
- We can't control "true" p(z)




We choose a simple prior $p(z)$, but the marginal likelihood $p_\theta(x)$ and true posterior $p_\theta(z|x)$ are intractable

Idea: introduce a "controllable" distribution $q_\phi(z)$

# Evidence Lower Bound (ELBO)

- Rewrite log-likelihood using latent z

$$
\begin{aligned}
& \log p_\theta(x) \\
= & \int_z q(z) \log p_\theta(x) dz \\
= & \int_z q(z) \log \left( \frac{p_\theta(x|z) p_\theta(z)}{p_\theta(z|x)} \right) dz \\
= & \int_z q(z) \log \left( \frac{p_\theta(x|z) p_\theta(z)}{p_\theta(z|x)} \frac{q(z)}{q(z)} \right) dz \\
= & \int_z q(z) \left( \log p_\theta(x|z) + \log \frac{p_\theta(z)}{q(z)} + \log \frac{q(z)}{p_\theta(z|x)} \right) dz \\
= & \mathbb{E}_{z \sim q(z)} \left[ \log p_\theta(x|z) \right] - \mathcal{D}_{\mathrm{KL}} \left( q(z) || p_\theta(z) \right) + \mathcal{D}_{\mathrm{KL}} \left( q(z) || p_\theta(z|x) \right)
\end{aligned}
$$

- valid for any distribution q(z)
- Bayes' rule
- just algebra
- just algebra

# Evidence Lower Bound (ELBO)

intractable $\log p_\theta(x) - \mathcal{D}_{\mathrm{KL}}\left(q(z)||p_\theta(z|x)\right)$ intractable

$= \mathbb{E}_{z\sim q(z)}\left[\log p_\theta(x|z)\right] - \mathcal{D}_{\mathrm{KL}}\left(q(z)||p_\theta(z)\right)$

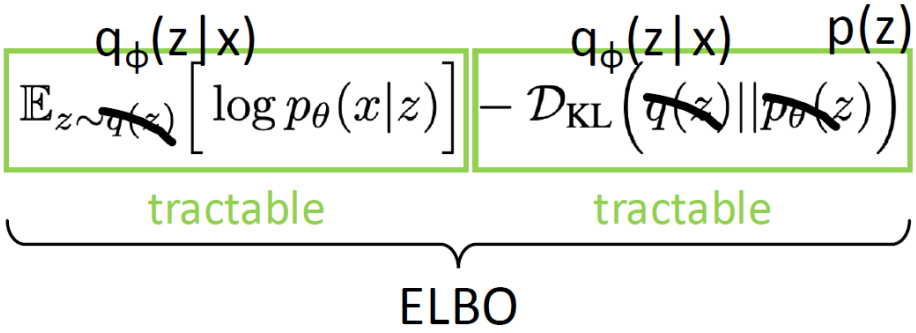
tractable tractable

ELBO

**ELBO**:

- It's lower bound of $\log p_\theta(x)$
- This formulation holds for <u>any</u> distribution q(z)

# Evidence Lower Bound (ELBO)




*__Note__: ELBO is not specific to VAE.

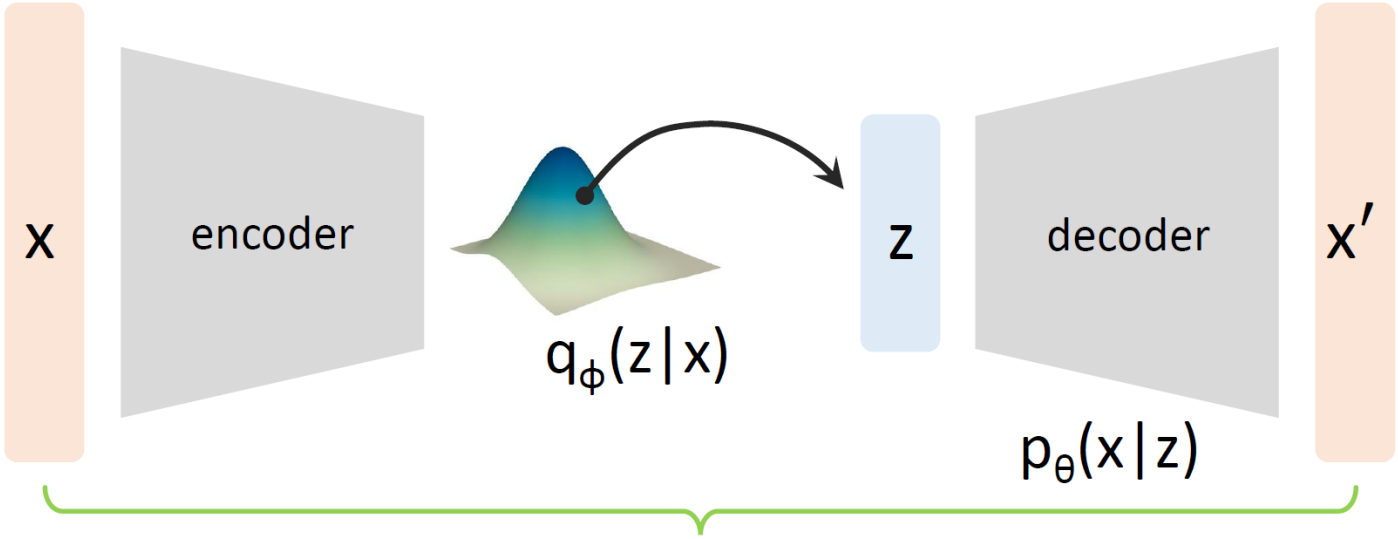
**ELBO**:

- It's lower bound of $\log p_\theta(x)$
- This formulation holds for <u>any</u> distribution q(z) , so ...
- we can parameterize q(z) by $q_\phi(z|x)$
- and let $p_\theta(z)$ be a fixed, known prior p(z) (e.g., Gaussian)

# VAE using ELBO

Maximize ELBO ⇒ minimize:

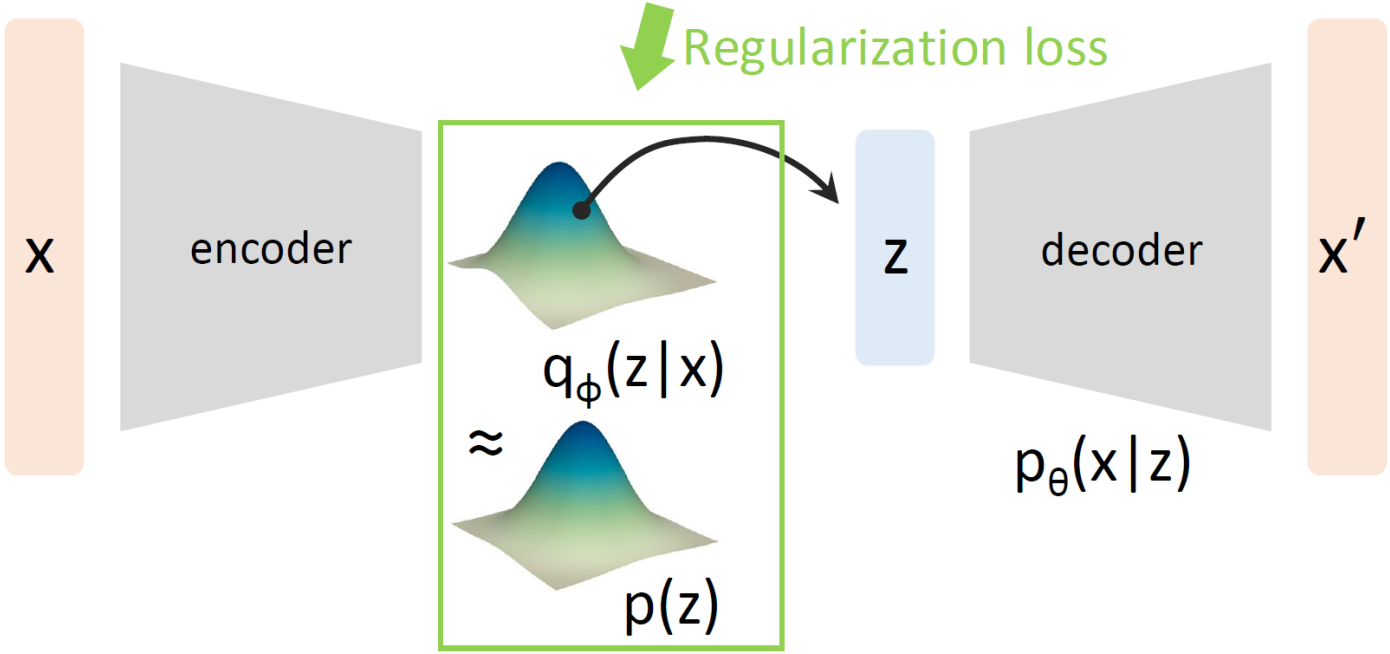
$$\mathcal{L}_{\theta,\phi}(x) = -\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x|z)\right] + \mathcal{D}_{\mathrm{KL}}\left(q_\phi(z|x)||p(z)\right)$$



# VAE using ELBO

Maximize ELBO ⇒ minimize:

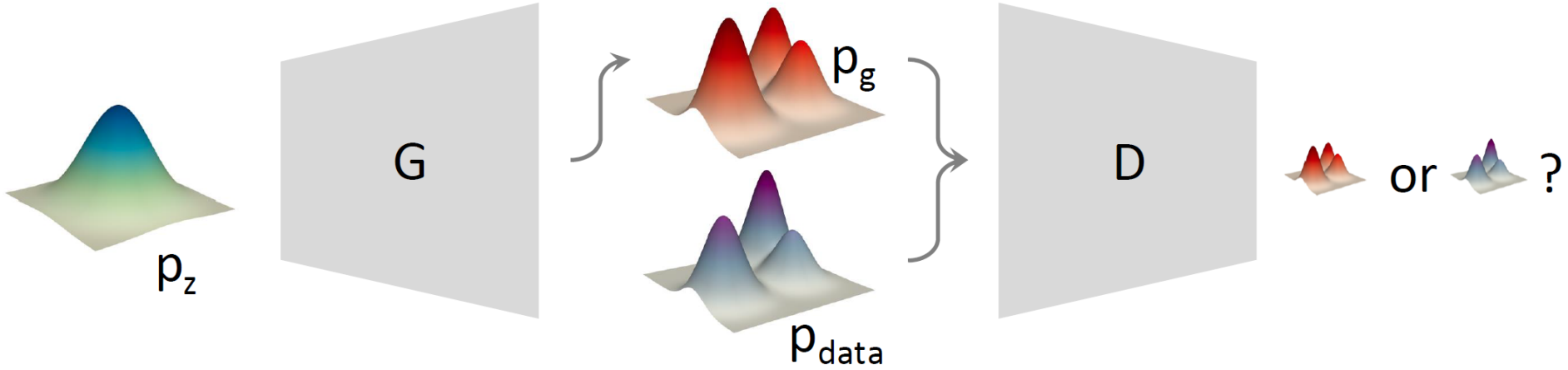
$$\mathcal{L}_{\theta,\phi}(x) = -\mathbb{E}_{z\sim q_\phi(z|x)}\Big[\log p_\theta(x|z)\Big] + \mathcal{D}_{\mathrm{KL}}\Big(q_\phi(z|x)||p(z)\Big)$$



# Generative Adversarial Networks

$$\min_G \max_D \mathcal{L}(D, G) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

min-max process



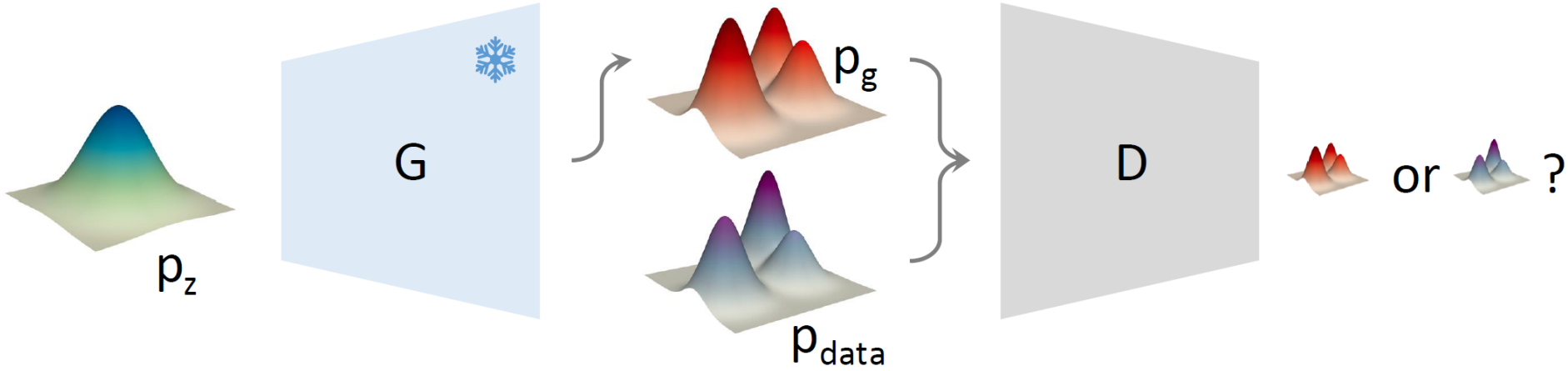
# Adversarial Objective: D-step

$$\max_D \mathcal{L}(D) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

push to 1 push to 0

D-step: fix G, optimize D

- D : classify real or fake
- binary logistic regression (sigmoid + cross-entropy)



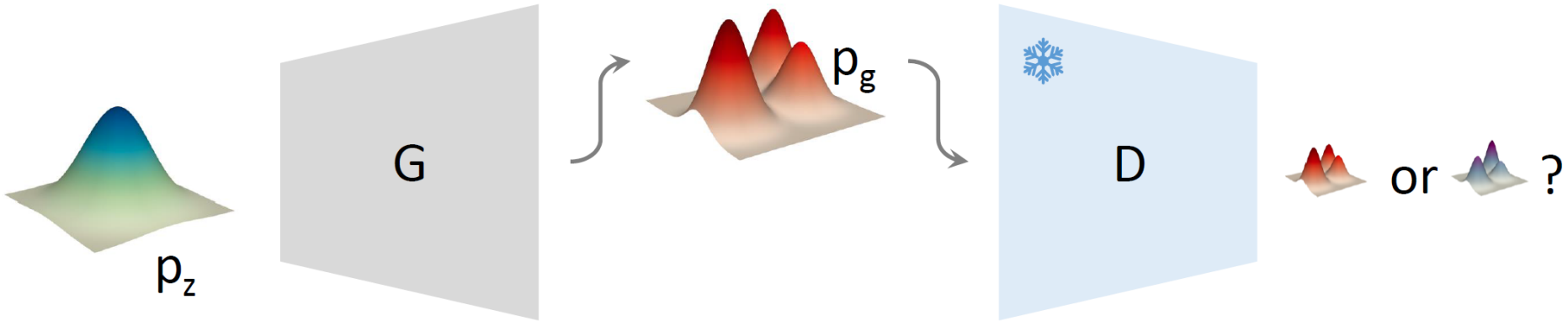
# Adversarial Objective: G-step

$$\min_G \mathcal{L}(G) = \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

push to 1

G-step: fix D, optimize G

- generate fake data such that D classifies it as “real”
- G to “confuse” D



# Adversarial Objective: G-step

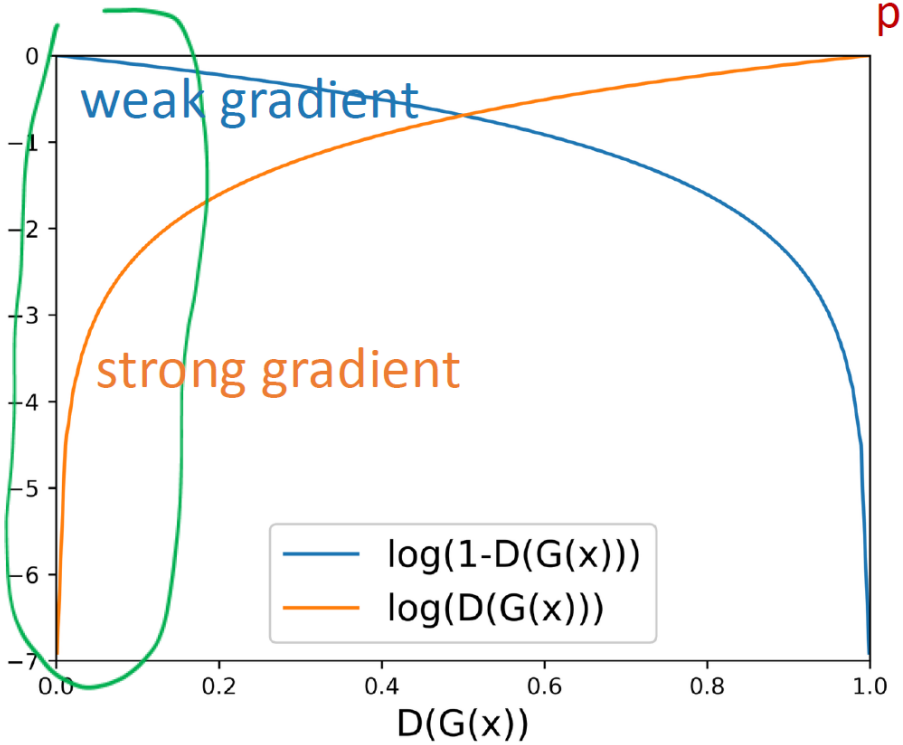
a "flip" trick:

max
$\min_G \mathcal{L}(G) = \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$

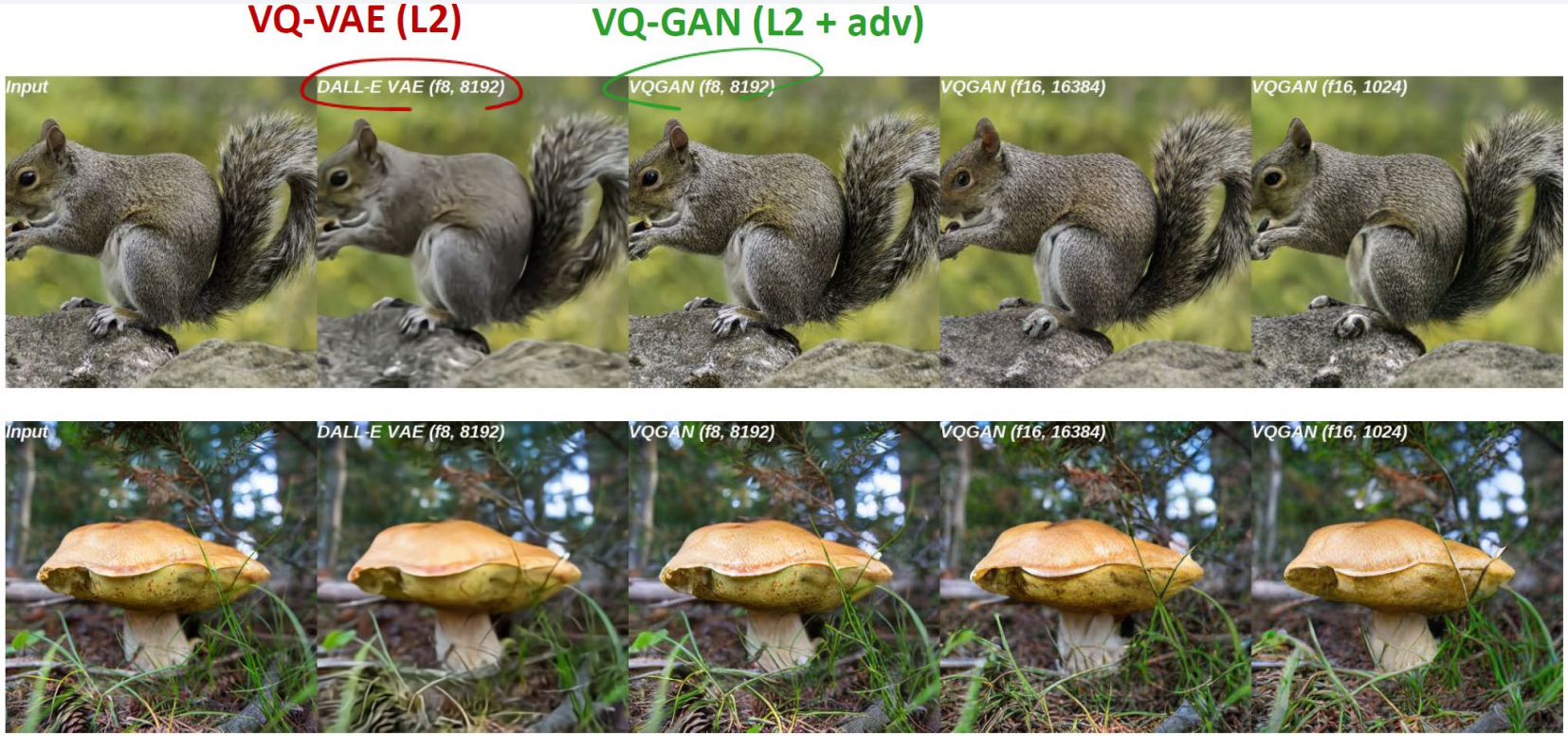
push to 1

Early in training:

- G is poor
- D(G) is near 0



# Example: VQ-VAE to VQ-GAN




Esser, et al., "Taming Transformers for High-Resolution Image Synthesis", CVPR 2021

# Why Diffusion Won

*The quality-stability-diversity trilemma, resolved*

## GAN (2014–2021)

Training: adversarial (unstable)
Quality: sharp
Diversity: mode collapse
Likelihood: none
Inference: 1 step (fast)

Dominated for 7 years
but required expert tuning

Legacy: adversarial loss
(SRGAN, VQ-GAN)

## Diffusion (2020–)

Training: MSE regression (stable)
Quality: equal or better
Diversity: full mode coverage
Likelihood: ELBO available
Inference: multi-step (slow → DDIM)

Simple objective, reliable training
Scales with compute

Now: SD, DALL-E, Sora, FLUX

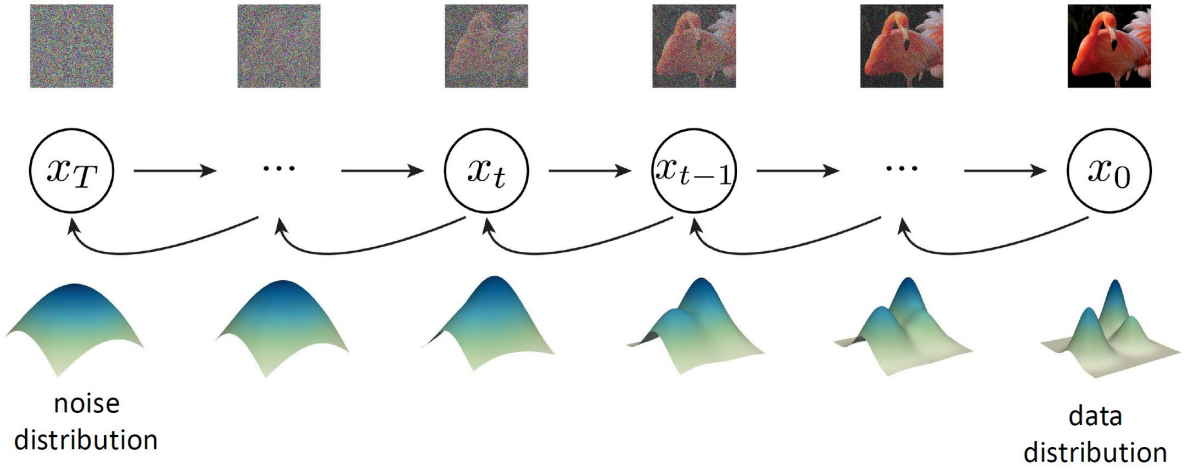
*GAN is not dead — its fast inference makes it valuable for distilling diffusion models (ICML 2024). But new systems are all diffusion/flow-based.*

Part 3

# Diffusion Models

*Learning to reverse the noise*

# Diffusion Models



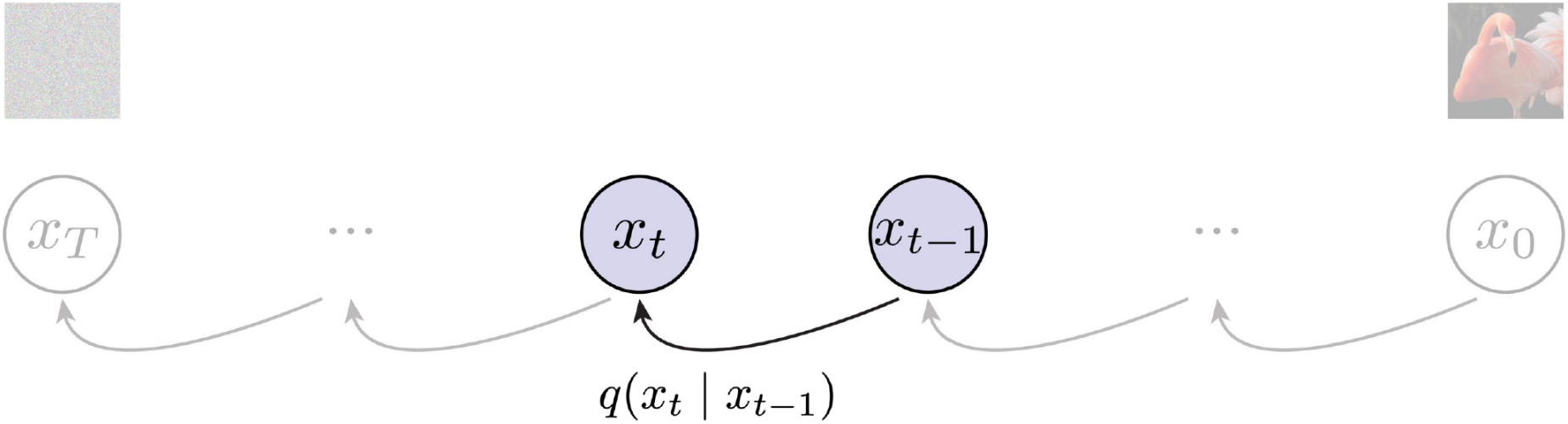

**Forward process**
- add noise to data

**Reverse process**
- learn to denoise

**Training objective**
- from Hierarchical VAE to L2 loss

**Noise Conditional Network**
- represent distributions

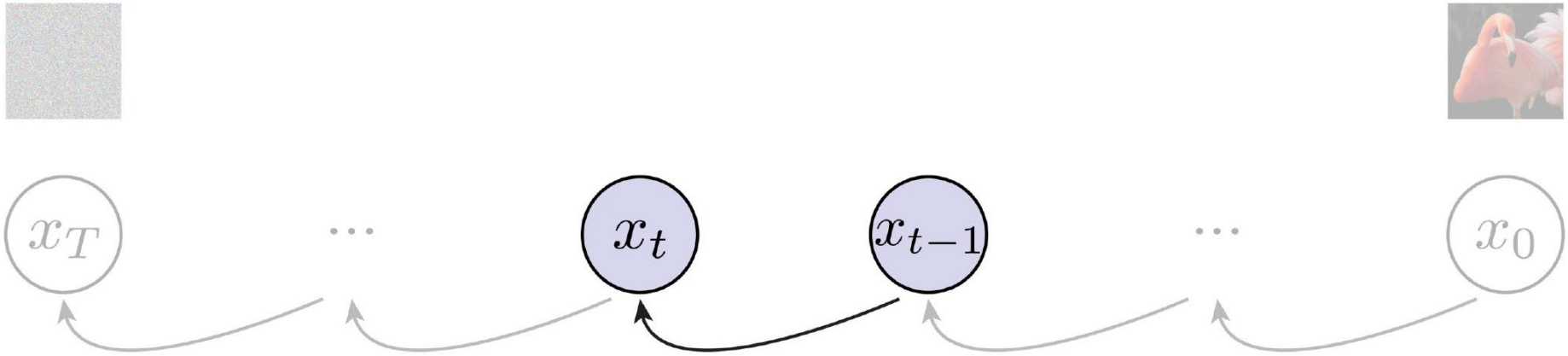
# Diffusion: Forward Process




tl; dr:

- pre-defined conditional distributions
- Gaussian w/ controllable mean/std
- easy to sample from

# Diffusion: Forward Process

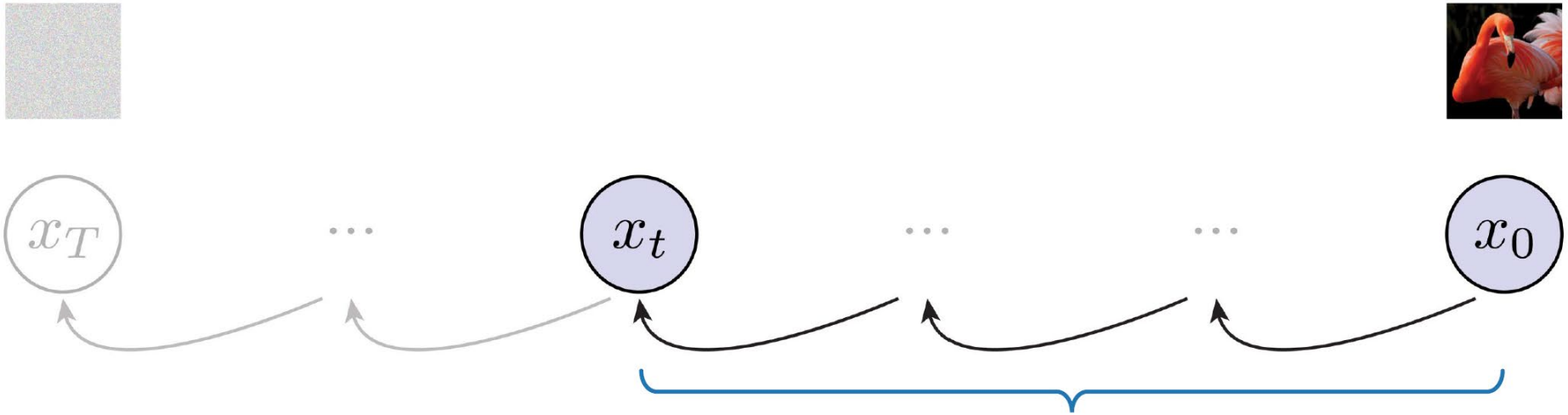



$$q(x_t \mid x_{t-1})$$

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

mean of $x_t$ var of $x_t$

# Diffusion: Forward Process



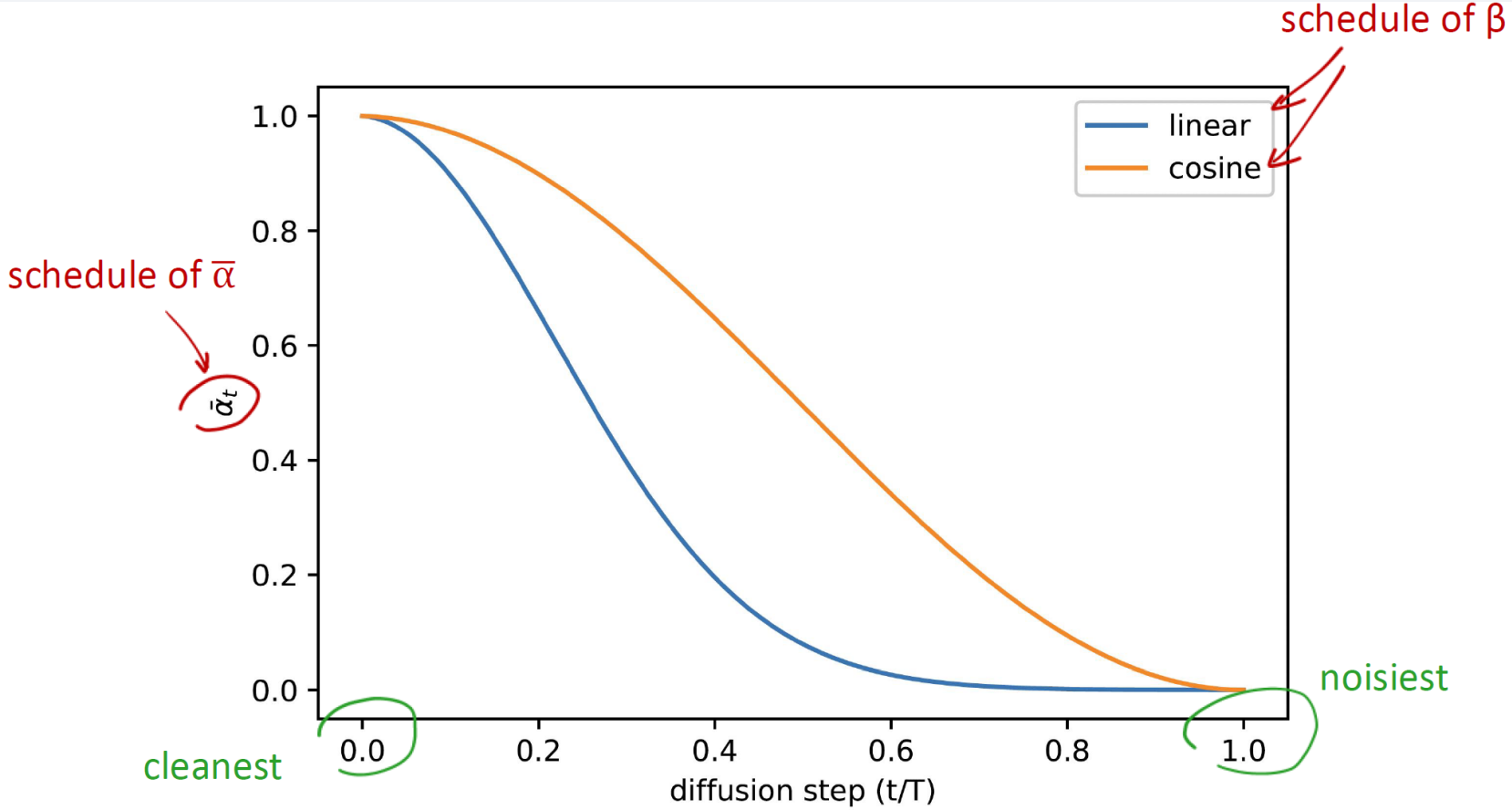

- sampling without iterating
- $x_t$ from $x_0$ in closed form

$$q(x_t \mid x_0) = \mathcal{N}(x_t \mid \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t)\mathbf{I})$$

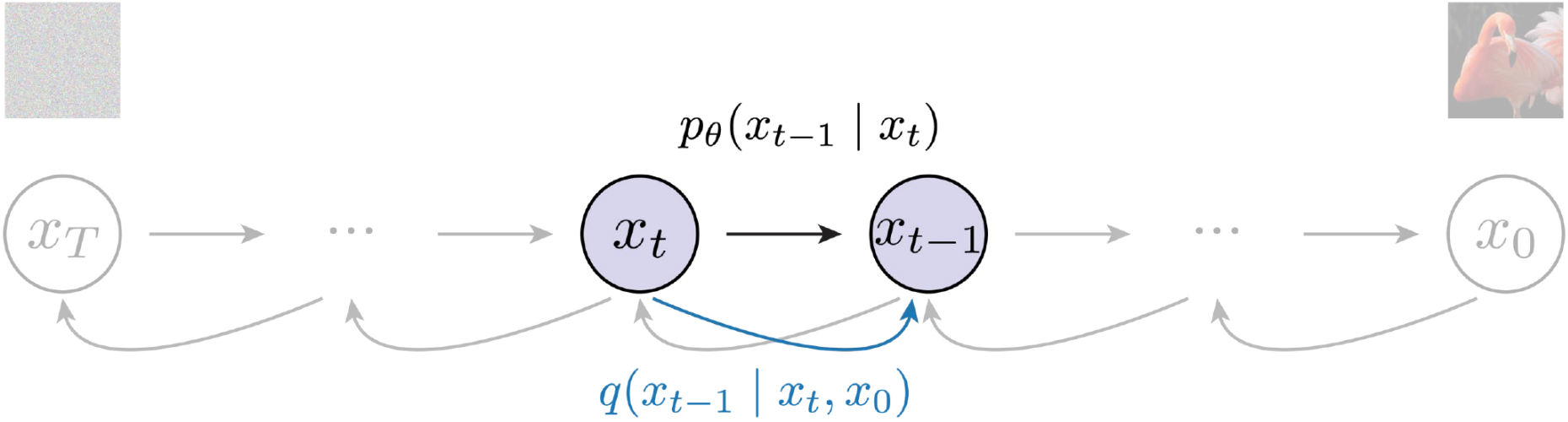
schedule derived from β

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

# Diffusion: Noise Schedule

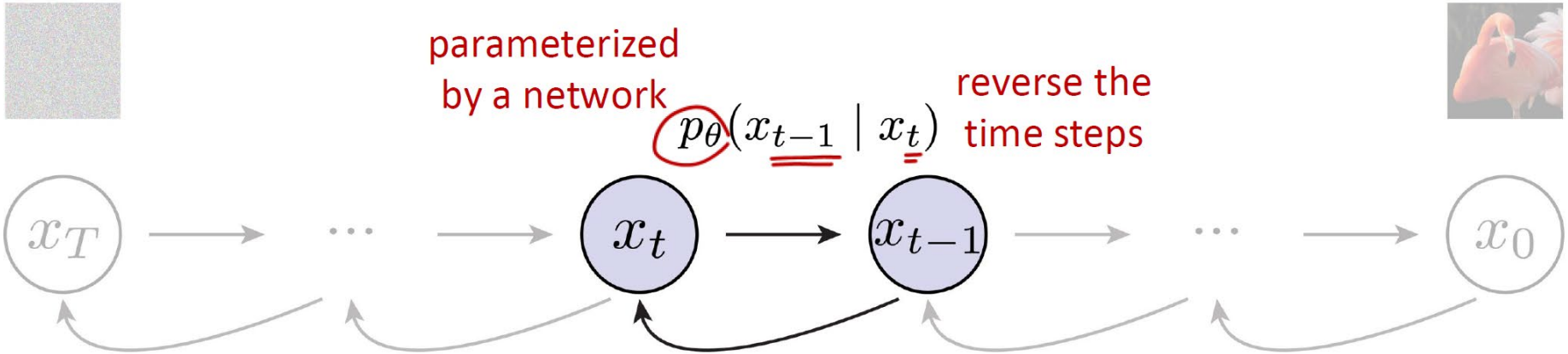


# Diffusion: Reverse Process



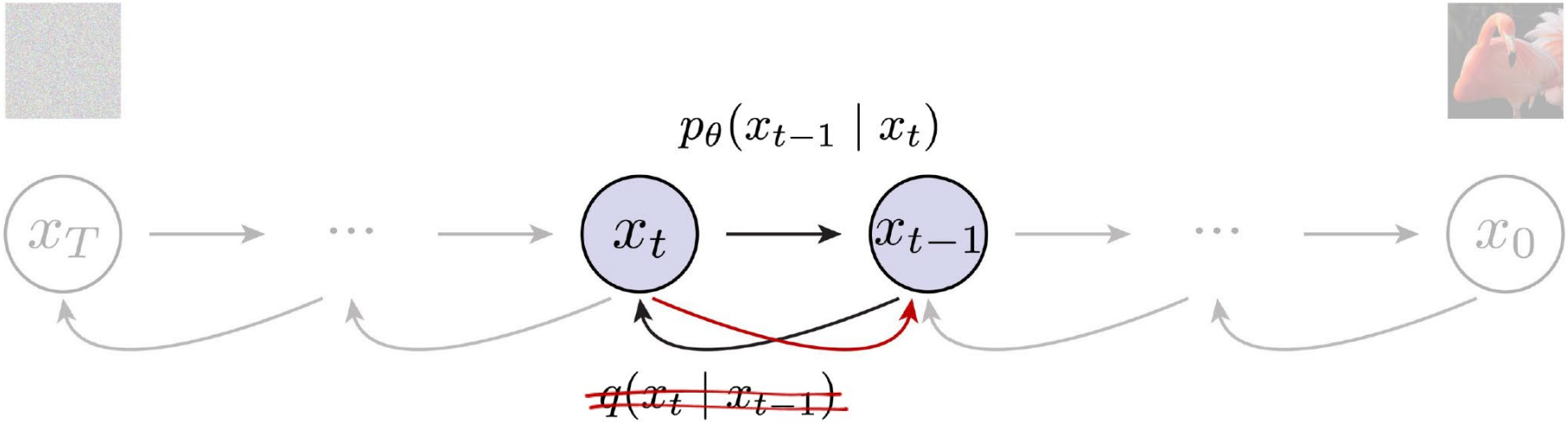

- **tl; dr**: a known Gaussian
- we want to learn it by $p_\theta$
- we can represent $p_\theta$ by a Gaussian
- minimize KL divergence

# Diffusion: Reverse Process



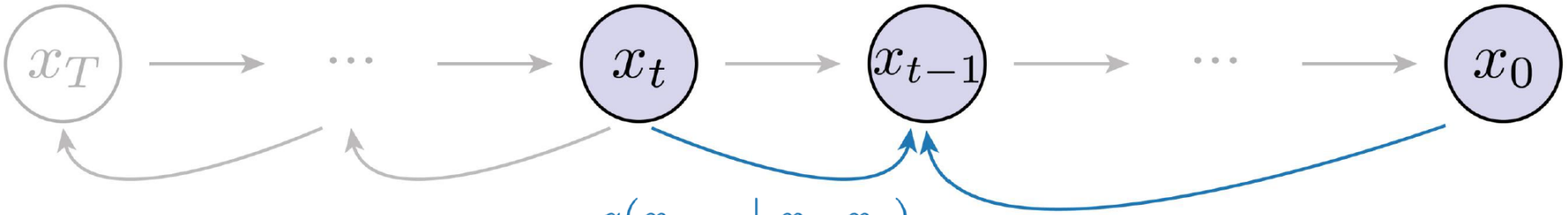

- known
- but not our target

# Diffusion: Reverse Process




- our target
- but unknown

# Diffusion: Reverse Process




$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1} \mid \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I})$$

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

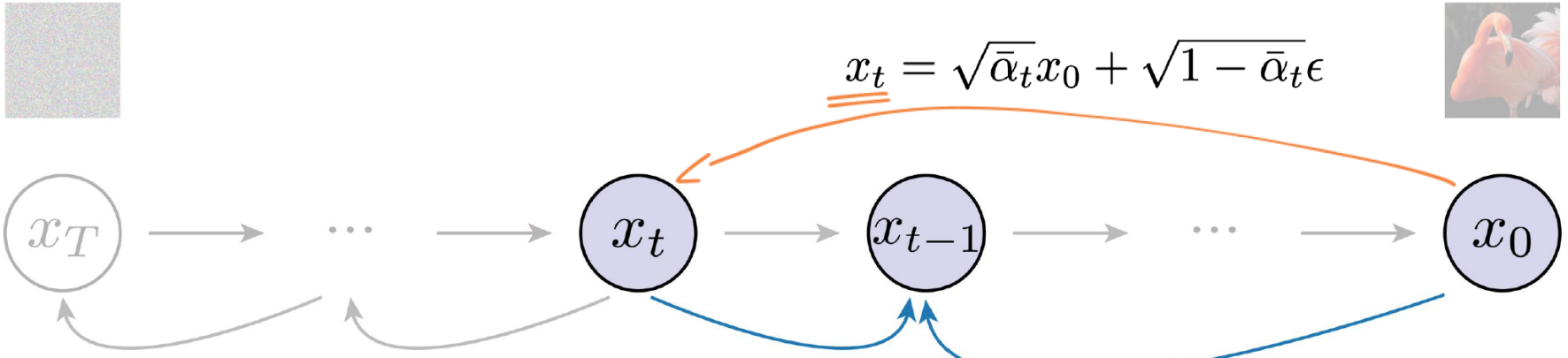
mean

linear combination

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

var

# Diffusion: Reverse Process




$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1} \mid \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I})$$

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \alpha_t} x_0 + \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t$$

mean

$$= \frac{1}{\sqrt{\bar{\alpha}_t}} \left( x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon \right)$$

"noise"

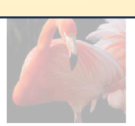
$$\tilde{\beta}_t := \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t$$

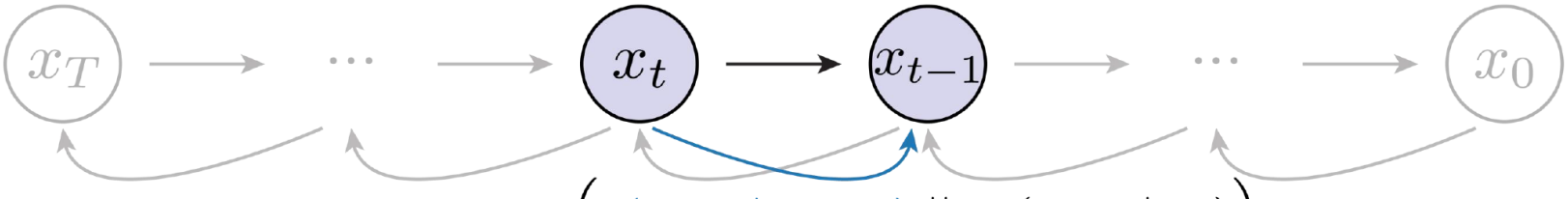
var

# Diffusion: Reverse Process

$D_{KL}$ of two Gaussians is like L2 loss: (pset 1)

$$D_{KL}(\mathcal{N}_1 \parallel \mathcal{N}_2) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + \|\mu_1 - \mu_2\|^2}{2\sigma_2^2} - \frac{1}{2}$$






$$D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \,||\, p_\theta(x_{t-1} \mid x_t)\right)$$

$$\mathcal{N}(x_{t-1} \mid \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I})$$

$$\mathcal{N}(x_{t-1} \mid \mu_\theta(x_t), \sigma_t^2 \mathbf{I})$$

learn

preset

$$\frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

"noise"

estimate noise

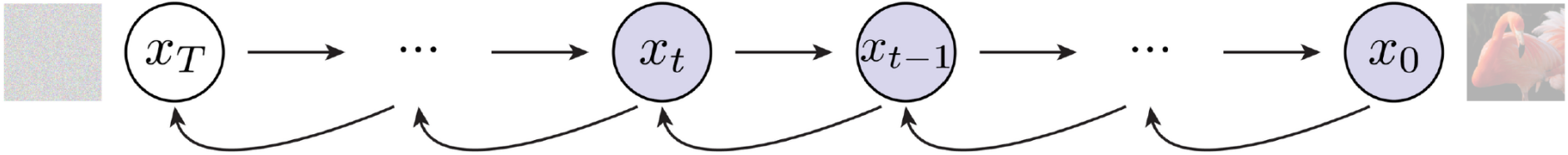
# Training Objective

$q_\phi(z|x)$ $q_\phi(z|x)$ $p(z)$

$$\mathbb{E}_{z\sim q(z)}\left[\log p_\theta(x|z)\right] - \mathcal{D}_{\mathrm{KL}}\left(q(z)||p_\theta(z)\right)$$

tractable tractable

ELBO




- variational lower bound
- like ELBO

$$\mathcal{L}_{\mathrm{VLB}} := \mathcal{L}_T + \mathcal{L}_{T-1} + \ldots + \mathcal{L}_0$$

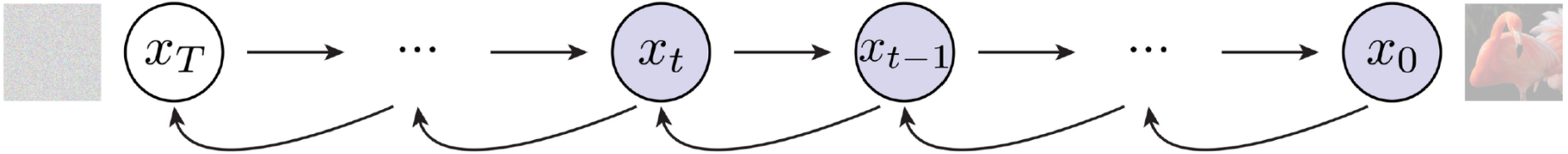
$$\mathcal{L}_T := D_{\mathrm{KL}}\left(q(x_T \mid x_0) \,||\, p_\theta(x_T)\right)$$

$$\mathcal{L}_{t-1} := D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \,||\, p_\theta(x_{t-1} \mid x_t)\right)$$

$$\mathcal{L}_0 := -\log p_\theta(x_0 \mid x_1)$$

if it's one-step, this is ELBO.

# Diffusion: Training Objective + ELBO




- variational lower bound
- like ELBO

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

$$\mathcal{L}_{\text{VLB}} := \mathcal{L}_T + \mathcal{L}_{T-1} + \ldots + \mathcal{L}_0$$

$$\mathcal{L}_T := D_{\text{KL}}\Big(q(x_T \mid x_0) \,||\, p_\theta(x_T)\Big)$$

$$\mathcal{L}_{t-1} := D_{\text{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,||\, p_\theta(x_{t-1} \mid x_t)\Big)$$

$$\mathcal{L}_0 := -\log p_\theta(x_0 \mid x_1)$$

# DDPM: Architecture & Training

*Each denoising step is just a supervised regression problem*

U-Net (Wk 6: encoder-decoder with skip connections)

- ResNet blocks (Wk 3) + self-attention at lower resolutions
- Timestep t: sinusoidal embedding (same as Wk 4 positional encoding)
- Class label (optional): embedding concatenated with timestep

Training loop (= supervised learning!):

1. Sample $x_0$ from data, sample t ~ Uniform(1, T)
2. Sample noise $\varepsilon \sim N(0, I)$
3. Compute $x\_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{(1-\bar{\alpha}_t)} \cdot \varepsilon$
4. Predict $\varepsilon_\theta(x_t, t)$ with the U-Net ← input: $x_t$, target: $\varepsilon$
5. Minimize $|| \varepsilon - \varepsilon_\theta(x_t, t) ||^2$ ← just MSE regression!

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

over $p_{data}$ · over [1, T ] · over $\mathcal{N}(0, \mathbf{I})$

Generation = a sequence of supervised predictions, chained together.

# Score-Based View (Optional)

*Same phenomenon, different mathematics (Song et al., ICLR 2021)*

## DDPM View

Discrete: T=1000 steps

Markov chain

Learn noise predictor $\varepsilon_\theta(x_t, t)$

Sampling: iterative denoising

Forward: add noise step by step

Reverse: remove noise step by step

## Score-Based View

Continuous: SDE over time t ∈ [0,1]

Score function: $\nabla_x \log p_t(x)$

= "which direction has higher density?"

Why this enables sampling:

Follow the score → climb toward data

(Langevin: $x \leftarrow x + \eta \nabla \log p + noise$)

Key equivalence:

$\varepsilon_\theta \propto -\nabla_x \log p_t(x_t)$

Noise prediction = score estimation

*Key Takeaway  This unification (Song et al. 2021) leads to DDIM, probability flow ODE, and Flow Matching (Wk 11).*

Part 4

# Preview

*DDIM, applications, what's coming in Wk 11*

# Sampling: DDPM & DDIM

*From noise to image*

## DDPM Sampling

Start from $\boldsymbol{x}_T \sim \mathcal{N}(0, \mathbf{I})$

For $t = T, T-1, \ldots, 1$:

$$\boldsymbol{x}_{t-1} = (1/\sqrt{\alpha_t}) \cdot (\boldsymbol{x}_t - (1-\alpha_t)/\sqrt{(1-\bar{\alpha}_t)} \cdot \varepsilon_\theta) + \sigma_t \cdot \boldsymbol{z}$$

1000 steps = very slow
(minutes per image)

But: high quality, diverse

## DDIM (Song et al., 2021)

Same trained model!
No retraining needed.

Non-Markovian process:
- Skip steps: 1000 → 20–50
- 50× speedup!

Deterministic ($\sigma = 0$):
- Same noise → same image
- Smooth interpolation

Modern: DPM-Solver (10–20),
UniPC (8–16 steps)

# What Diffusion Models Can Do

*Everything below is built on the foundations we learned today*

### Text-to-Image

DALL-E 2/3
Stable Diffusion
Midjourney
FLUX

"a cat astronaut"
→ photorealistic image

### Image Editing

Inpainting
Super-resolution
Style transfer
SDEdit

Modify existing
images selectively

### Controlled Generation

ControlNet
IP-Adapter
T2I-Adapter

Pose, edge, depth
→ guided output
(Wk 11 topic)

### Video & 3D

Sora (OpenAI)
VEO-3 (Google)
DreamFusion

Diffusion extends
to temporal/spatial
dimensions

### Science

Protein design
(AlphaFold)
Drug discovery
Weather forecast

Beyond images:
any continuous data

# Preview: What's Coming in Wk 11

*From foundations to practice*

| | |
|---|---|
| **Latent Diffusion (LDM)** | Move from pixel space to latent space — 48× fewer dimensions. Stable Diffusion architecture. |
| **Classifier-Free Guidance** | Steer generation toward desired outputs — THE industry standard technique. |
| **$\varepsilon$ / $x_0$ / $v$ Prediction** | Three ways to parameterize the network. v-prediction → modern standard. |
| **Flow Matching** | Simpler math, straighter paths, fewer steps. SD3, FLUX, Sora are all FM-based. |
| **Conditional Generation** | ControlNet, IP-Adapter — from WHAT to generate to HOW to generate. |

*Today's DiT seminar bridges Wk 10 → Wk 11: same diffusion, Transformer backbone → enables scaling to SD3/FLUX.*

# Summary & Resources

## What We Covered

Part 1: Generative Models
- Discriminative vs Generative vs Conditional
- Model families (AR, VAE, GAN, Diffusion, FM)

Part 2: VAE & GAN
- Latent variables, ELBO derivation
- Adversarial training, GAN limitations
- One-step vs multi-step generation

Part 3: Diffusion
- Forward/reverse, DDPM = hierarchical VAE
- ELBO → $L_{simple}$ (noise prediction MSE)
- DDIM: 50× speedup without retraining

## Further Reading

VAE: Kingma & Welling, 2014
GAN: Goodfellow et al., NeurIPS 2014
DDPM: Ho et al., NeurIPS 2020
DDIM: Song et al., ICLR 2021
DiT: Peebles & Xie, ICCV 2023

Lilian Weng, "What are Diffusion Models?"
Kaiming He, MIT 6.7960 Lec 13-16
MIT 6.S184 Course Notes (2026)

**Next Week** Week 11: Diffusion models (2) LDM, Guidance, Flow Matching, … – Lecture + Paper #3 Seminar

# Acknowledgments

*Some slides and figures in this course are adapted from or inspired by the following open resources.*

- **Stanford CS231n: Deep Learning for Computer Vision (Spring 2025)**
  Fei-Fei Li, Ehsan Adeli, et al. | cs231n.stanford.edu
- **MIT 6.S184: Generative AI with Stochastic Differential Equations (2026)**
  Peter Holderrieth and Ezra Erives | https://diffusion.csail.mit.edu/
- **MIT 6.7960: Deep Learning (Fall 2024, Fall 2025)**
  Phillip Isola, Sara Beery, Jeremy Bernstein | ocw.mit.edu/courses/6-7960-deep-learning-fall-2024/, https://deeplearning6-7960.github.io/
- **3rd Workshop on Generative Models for Computer Vision (CVPR 2025)**
  Kaiming He | https://generative-vision.github.io/workshop-CVPR-25/
- **Key papers: DDPM, Improved DDPM, DDIM, DiT, …**