Deep Learning (1470)

Randall Balestriero

Class 17: Diffusion Models

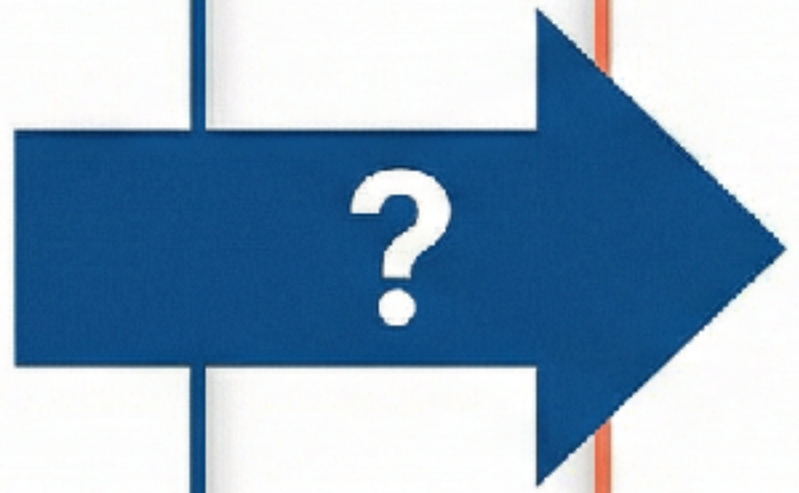
Recap!

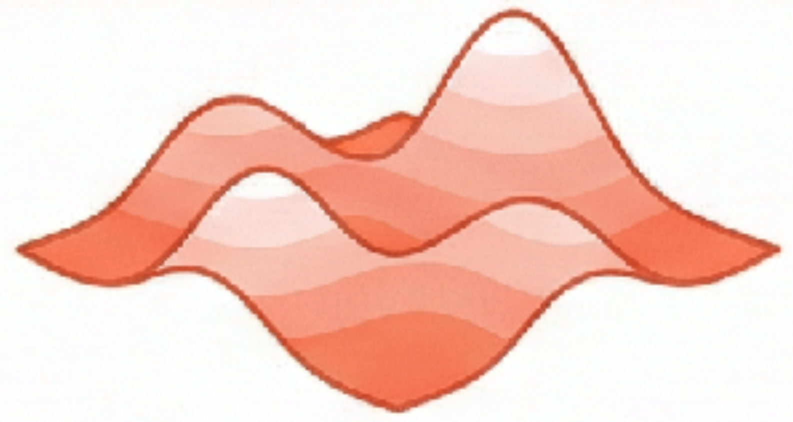
THE GENERATIVE MODELING CHALLENGE

The Problem

What We Have

Training samples from unknown distribution



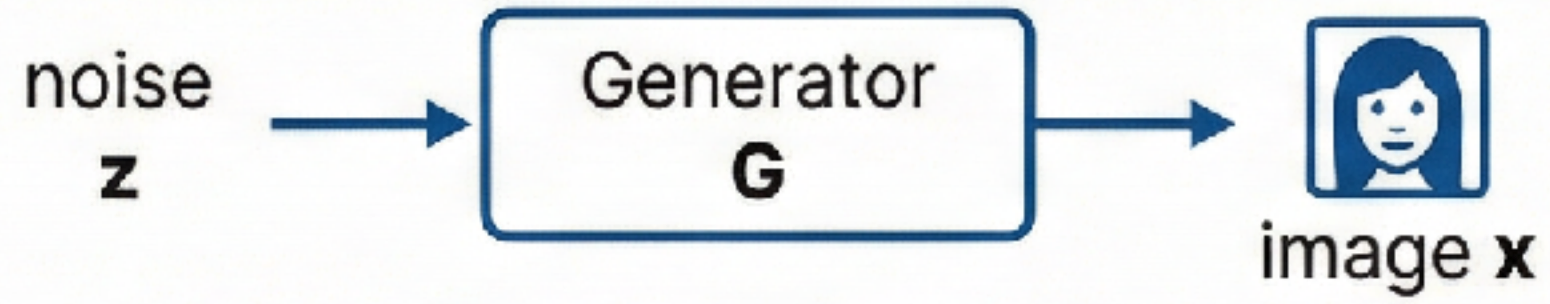


What We Want

Learn **p(x)** to generate NEW samples

Two Approaches

Implicit Models

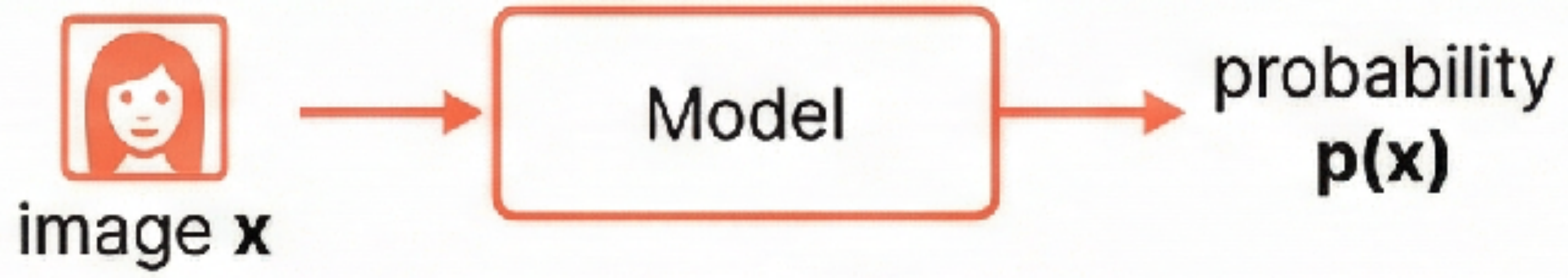


- Learn transformation directly
- Example: GANs

✓ **Pros**
Fast sampling

✗ **Cons**
Unstable training, mode collapse

Explicit Models



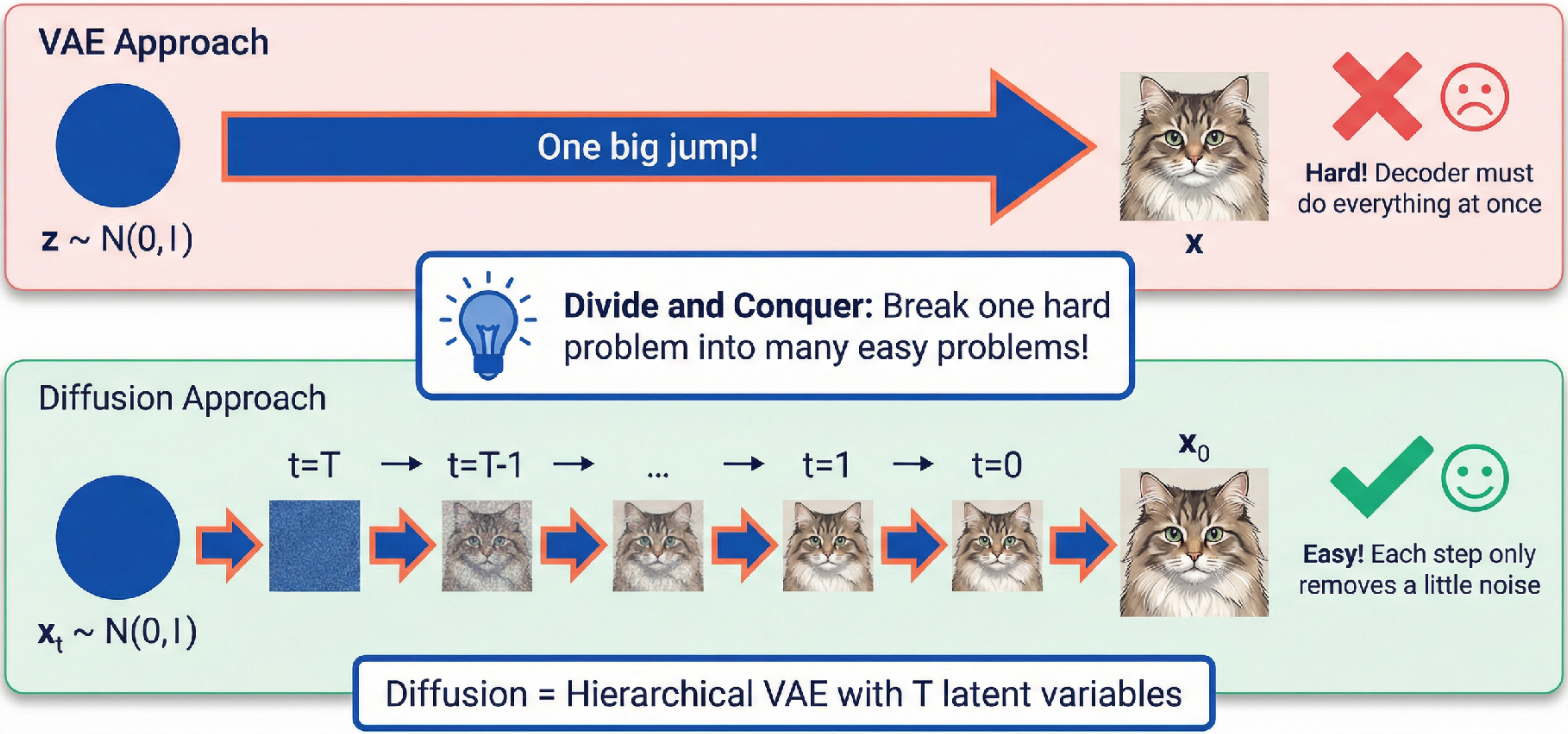
- Learn the distribution
- Examples: VAEs, Diffusion

✓ **Pros**
Stable training, likelihood

✗ **Cons**
Can be slow

We'll focus on this!

THE **KEY INSIGHT:** Many Small Steps


VAE Approach
$z \sim N(0,I)$
One big jump!
x
Hard! Decoder must do everything at once
Diffusion Approach
$x_t \sim N(0,I)$
$t=T \rightarrow t=T-1 \rightarrow \dots \rightarrow t=1 \rightarrow t=0$
x_0
Easy! Each step only removes a little noise

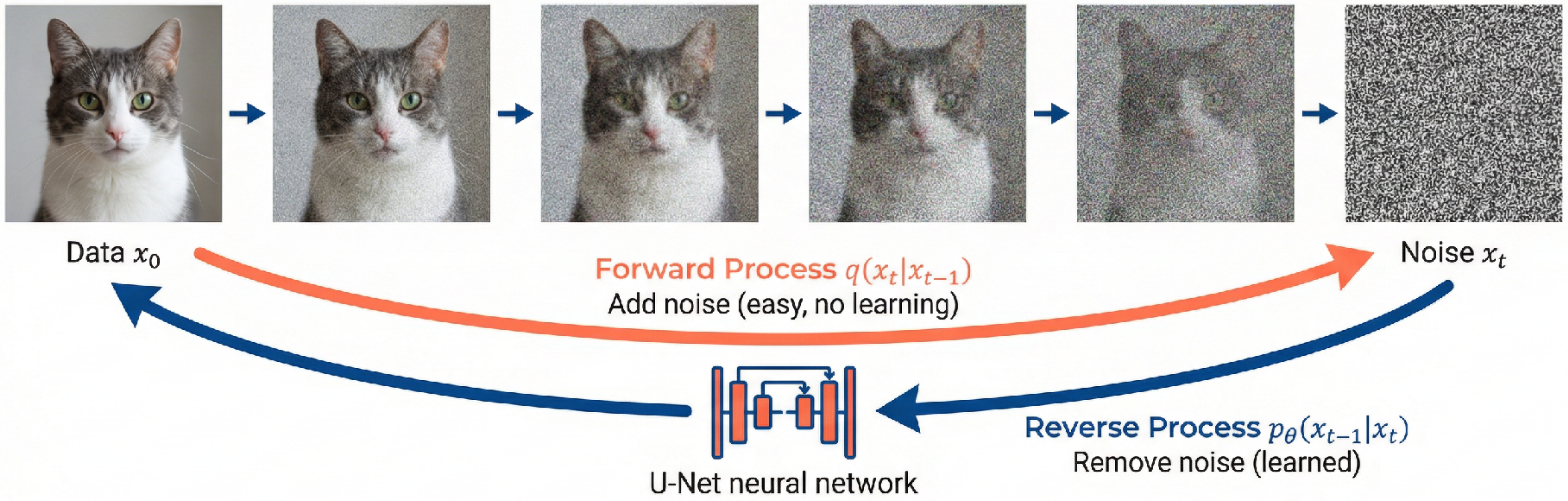
Divide and Conquer: Break one hard problem into many easy problems!

Diffusion = Hierarchical VAE with T latent variables

Any Idea?

DIFFUSION MODELS

From Noise to Data

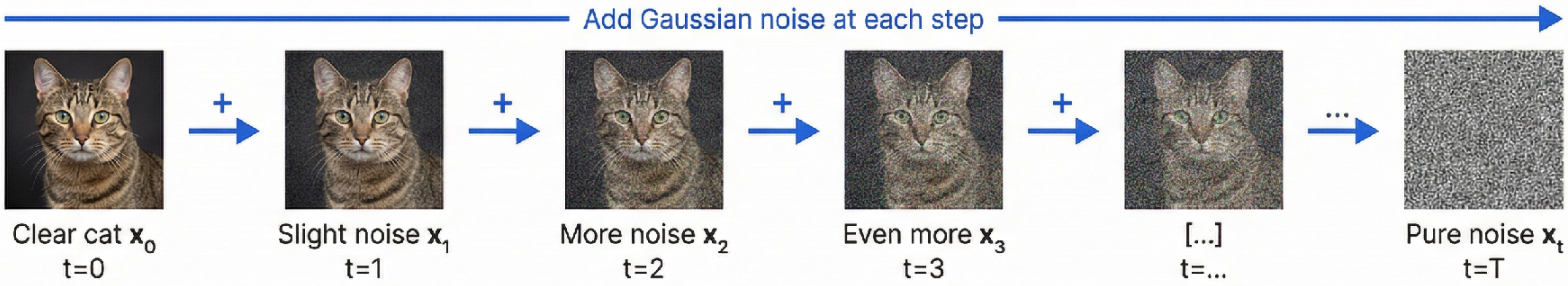

Data x_0
Noise x_t
Forward Process $q(x_t|x_{t-1})$
Add noise (easy, no learning)
Reverse Process $p_\theta(x_{t-1}|x_t)$
Remove noise (learned)
U-Net neural network

Iterative
- Many small steps

Learnable
- Neural network denoiser

Powerful
- State-of-the-art generation

THE FORWARD PROCESS: Adding Noise


Add Gaussian noise at each step
+
+
+
+
...
Clear cat $\mathbf{x}_0$ t=0
Slight noise $\mathbf{x}_1$ t=1
More noise $\mathbf{x}_2$ t=2
Even more $\mathbf{x}_3$ t=3
[...] t=...
Pure noise $\mathbf{x}_t$ t=T

Step-by-Step

One Step

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\sqrt{1-\beta_t}\cdot\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right)$$

$$\longrightarrow$$

Shrink signal, add noise

Direct Jump

Shortcut

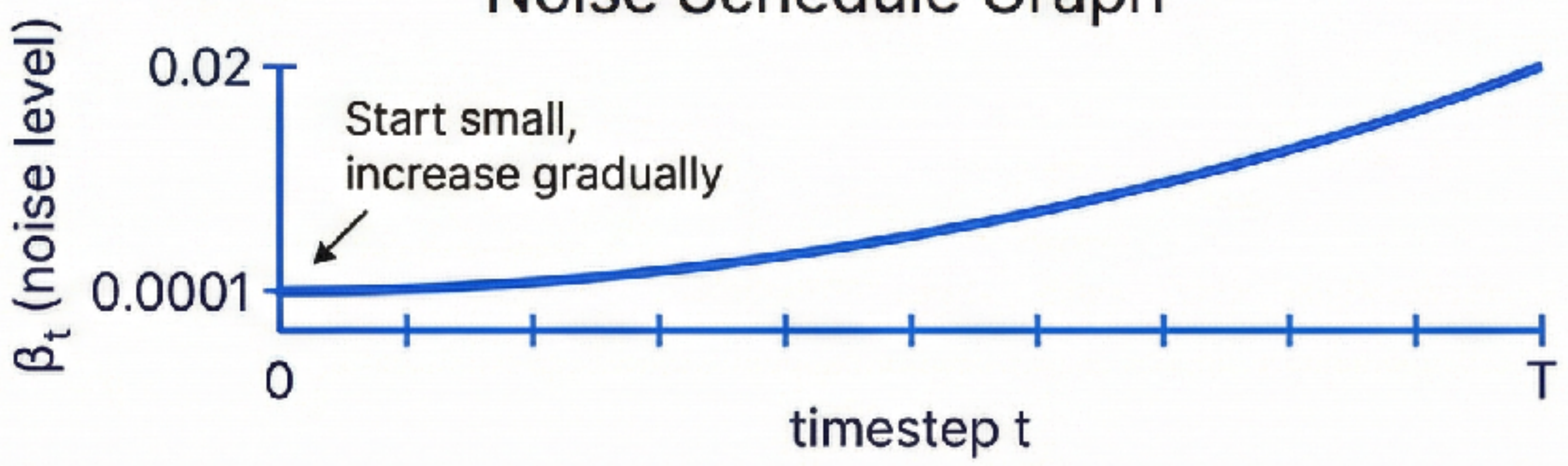
$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\cdot\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$

$$\mathbf{x}_0 \longrightarrow \mathbf{x}_t$$

Jump to any timestep directly!

where $\bar{\alpha}_t = \prod_i (1-\beta_i)$

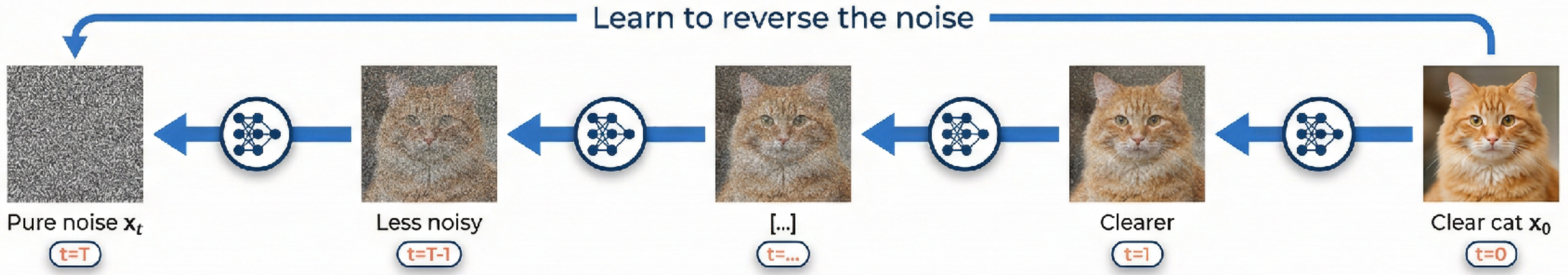
Noise Schedule Graph



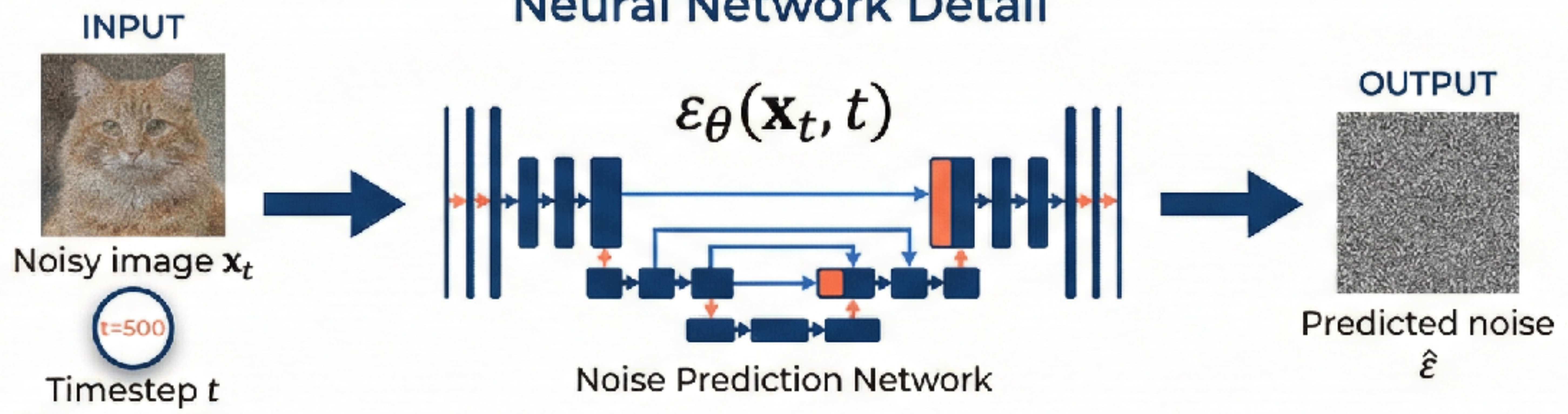
Key Point

✓ No learning required - just math!

THE REVERSE PROCESS: Learning to Denoise



Neural Network Detail


INPUT
Noisy image $\mathbf{x}_t$
t=500
Timestep t
$\varepsilon_\theta(\mathbf{x}_t, t)$
Noise Prediction Network
OUTPUT
Predicted noise $\hat{\varepsilon}$

The Denoising Formula

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_\theta\right) + \sigma_t\mathbf{z}$$

Remove predicted noise

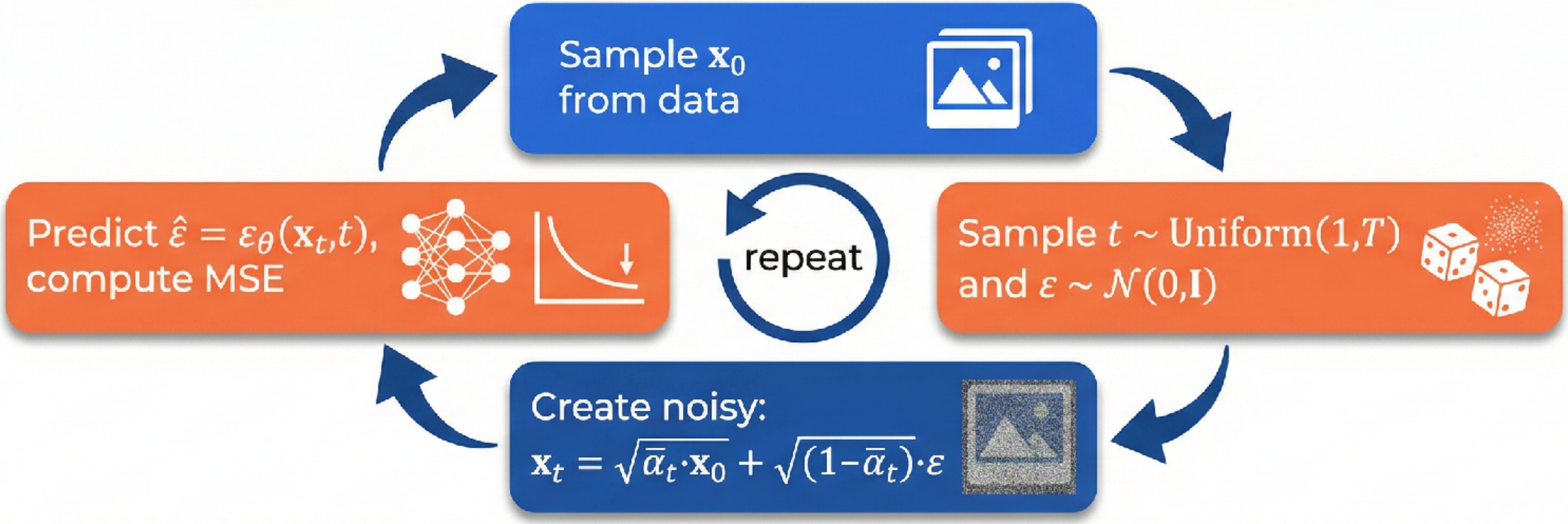
Scale appropriately

Add small fresh noise

TRAINING: The Simple Denoising Objective

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

Just predict the noise that was added!



Sample $\mathbf{x}_0$ from data
Sample $t \sim \mathrm{Uniform}(1,T)$ and $\varepsilon \sim \mathcal{N}(0,\mathbf{I})$
Create noisy: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\cdot\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\cdot\varepsilon$
Predict $\hat{\varepsilon} = \varepsilon_\theta(\mathbf{x}_t,t)$, compute MSE
repeat

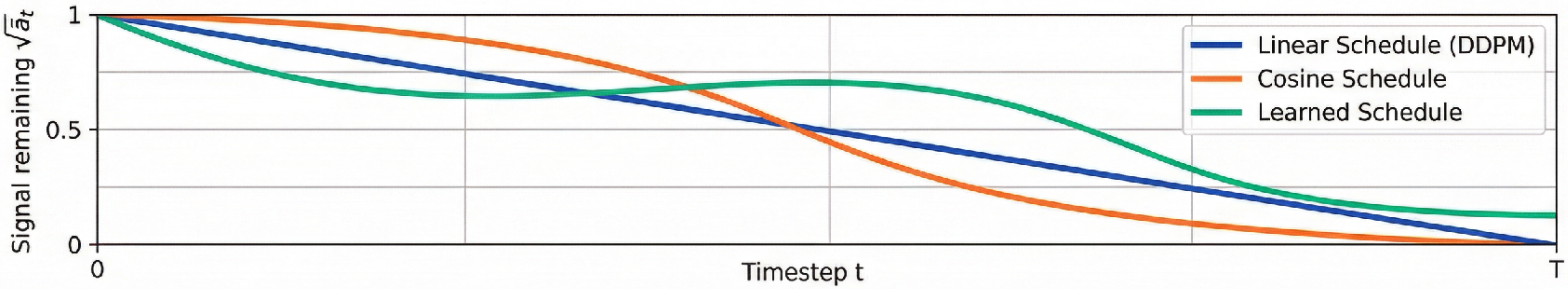
Simple
Just MSE regression

Stable
No adversarial training

Effective
State-of-the-art results

How to schedule alpha?

NOISE SCHEDULES: How Fast to Add Noise?



Linear

Uniform noise addition
DDPM (2020)

Cosine

Preserves structure longer
Improved DDPM (2021)

Learned

Optimized during training
VDM (2021)

Cosine schedule keeps images recognizable longer → better training signal!

Is Generation Fast?

DDIM: Faster Sampling (Song et al., 2020)

THE PROBLEM

DDPM: 1000 steps

Slow generation

→

DDIM: 50 steps

Same quality, 20x faster!

THE KEY INSIGHT

Key Insight: Remove the randomness!

DDPM (Stochastic)

$$\mathbf{x}_{t-1} = \mu(\mathbf{x}_t, t) + \sigma_t \cdot \mathbf{z}$$

Random noise each step

→

DDIM (Deterministic)

$$\mathbf{x}_{t-1} = f(\mathbf{x}_t, \hat{\mathbf{x}}_0, t)$$

No randomness

10-50x Faster

Deterministic

Reproducible

Interpolation

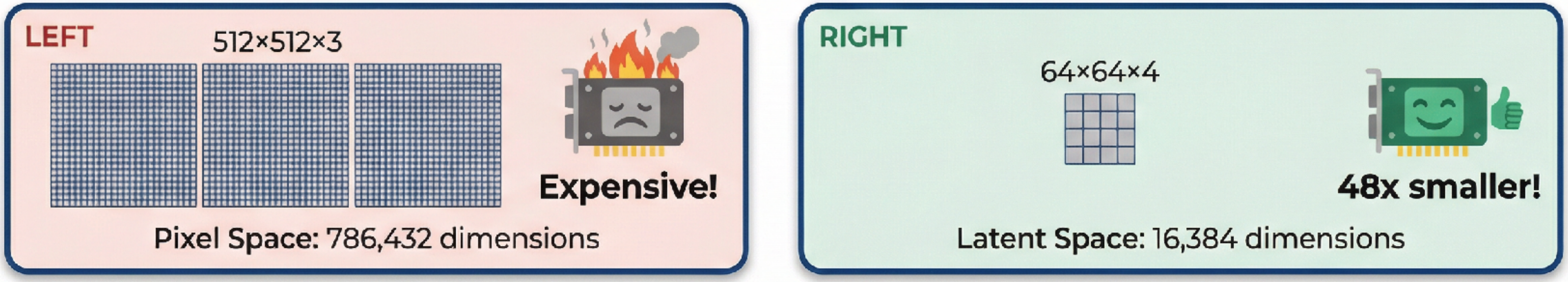
Smooth latents

Uses the SAME trained model - no retraining needed!

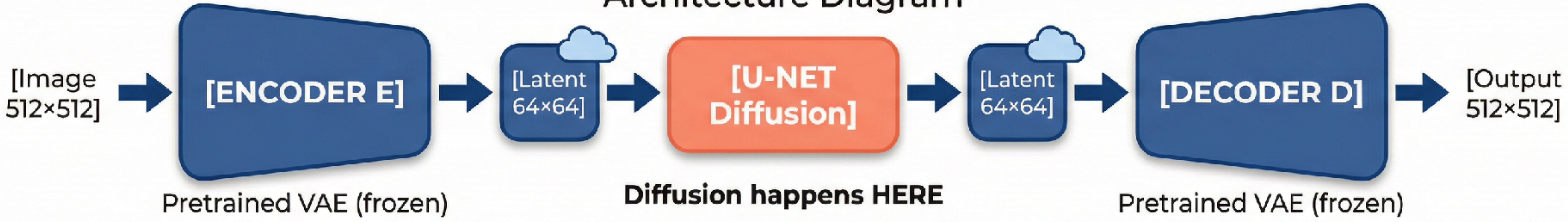
Is Generation Fast Now?

LATENT DIFFUSION: Stable Diffusion **(Rombach et al., 2022)**

The Problem


LEFT
512×512×3
Expensive!
Pixel Space: 786,432 dimensions
RIGHT
64×64×4
48x smaller!
Latent Space: 16,384 dimensions

Architecture Diagram



Efficient
Train on 1 GPU

High Quality
Perceptually same

Flexible
Easy conditioning

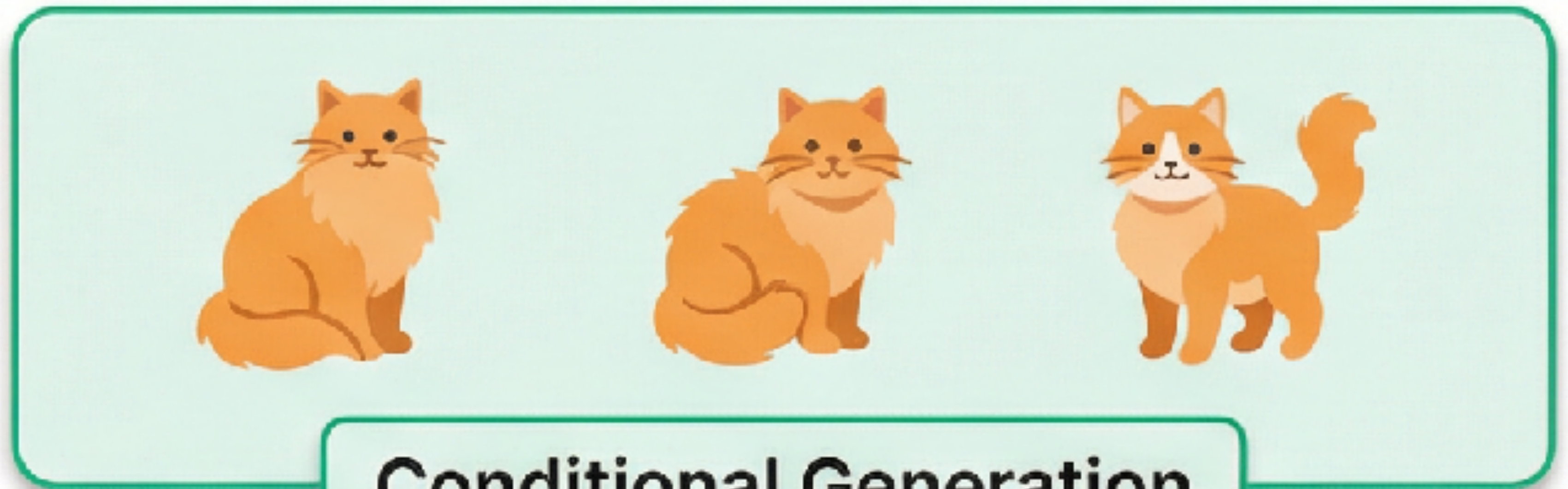
This architecture powers Stable Diffusion!

CLASSIFIER-FREE GUIDANCE: Controlling Generation

TOP SECTION
The Problem

Unconditional Generation

No control over output



Conditional Generation

Generate what we want!

MIDDLE SECTION
The Method

$$\tilde{\varepsilon} = \varepsilon_\theta(\mathbf{x}_t, \emptyset) + w \cdot (\varepsilon_\theta(\mathbf{x}_t, \mathbf{c}) - \varepsilon_\theta(\mathbf{x}_t, \emptyset))$$

Unconditional $\varepsilon_\theta(\mathbf{x}_t, \emptyset)$

Base prediction

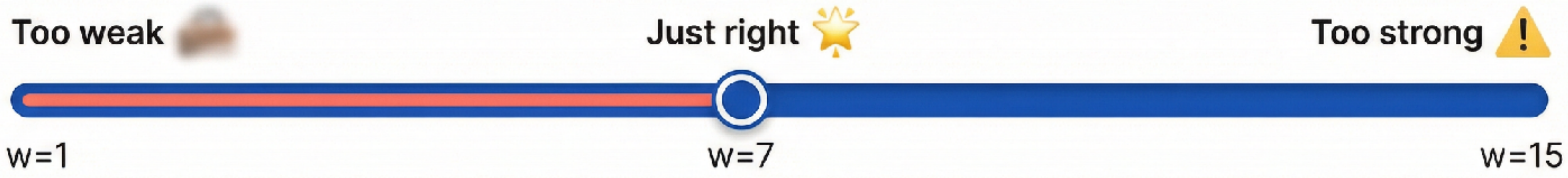
Conditional $\varepsilon_\theta(\mathbf{x}_t, \mathbf{c})$

With text prompt

Guidance w

Strength of control

BOTTOM SECTION
Guidance Scale



Key insight Train with random prompt dropout → one model does both!

THE SCORE FUNCTION: A Deeper Connection

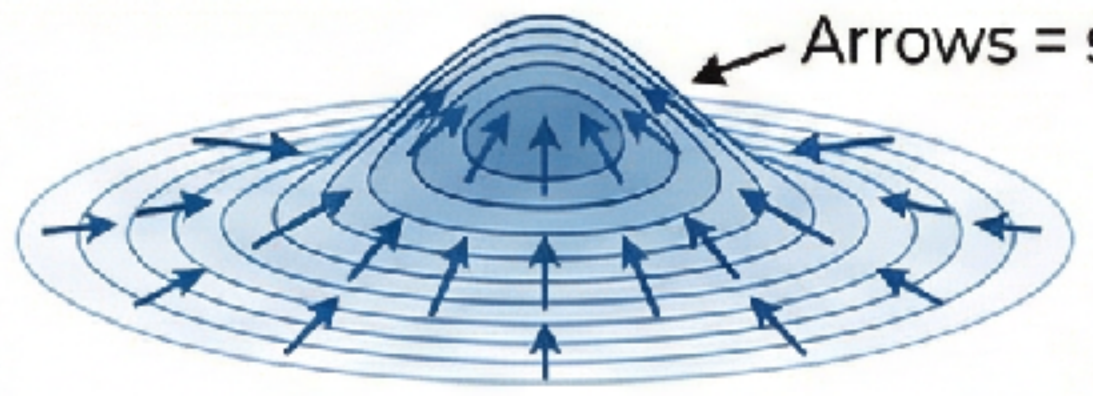
Undergraduate Deep Learning Class

What is the Score?

Score Function

$$\nabla_{\mathbf{x}} \log p(\mathbf{x})$$

- Gradient of log-probability
- Direction toward higher density



Point uphill toward data!

The Key Connection

Noise Prediction = Score Estimation

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}_t|\mathbf{x}_0) = \frac{-\varepsilon}{\sqrt{1-\bar{\alpha}_t}}$$

Diffusion View

- Predict noise ε

=

Score View

- Estimate $\nabla \log p$

Why This Matters

Theory

Rigorous foundation

SDEs

Continuous-time version

Unified

Many methods connected

OPEN QUESTIONS: Food for Thought

Why does denoising work?
What structure is the network learning about natural images?

How fast can we go?
1000 → 50 → 4 → 1 step?
What's the limit?

What's the best latent space?
Pixels? VAE? Something new?

Discrete or continuous time?
Steps vs SDEs - which is optimal?

Beyond images?
Video, 3D, audio, molecules, robotics...

See you on Wednesday!