High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach et al., CVPR 2022

CSD715: Special Topic in AI

Jayin Khanna

Shiv Nadar Institution of Eminence

Outline

DDPMs

- Introduction
- Diffusion Models
- Forward and Reverse Processes
- Training Objective

Challenges with DDP

- Limitations of Pixel-Space Diffusion

Latent Diffusion Models

- LDM Overview
- Autoencoder (Stage 1)
- Diffusion in Latent Space (Stage 2)
- Computational Efficiency

Extensions & Results

- Conditioning via Cross-Attention
- Model Pipeline
- Results and Trade-offs

Generative Modelling

The Setup

- Dataset of images: $x_1, x_2, \ldots, x_n$
- Samples from an unknown distribution $p_{\text{data}}(x)$
- We do *not* know $p_{\text{data}}(x)$ —only samples

The Goal

- Learn $p_\theta(x) \approx p_{\text{data}}(x)$
- Generate new samples:

$$x \sim p_\theta(x)$$

Why is Generative Modelling Hard?

- Images lie in extremely high-dimensional space
- Example:

$$256 \times 256 \times 3 = 196{,}608 \text{ dimensions}$$

- Real images form a tiny, complex manifold
- Explicitly modeling $p_{\text{data}}(x)$ is intractable

Key Question

Can a neural network learn to sample from $p_{\text{data}}(x)$ using only data?

Diffusion Models (DDPMs)

Main Idea

- Start from clean image x_0
- Add noise step-by-step:

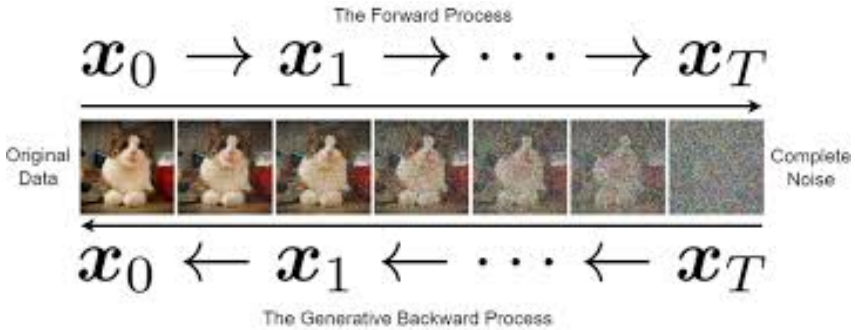
$$x_0 \to x_1 \to \cdots \to x_T$$

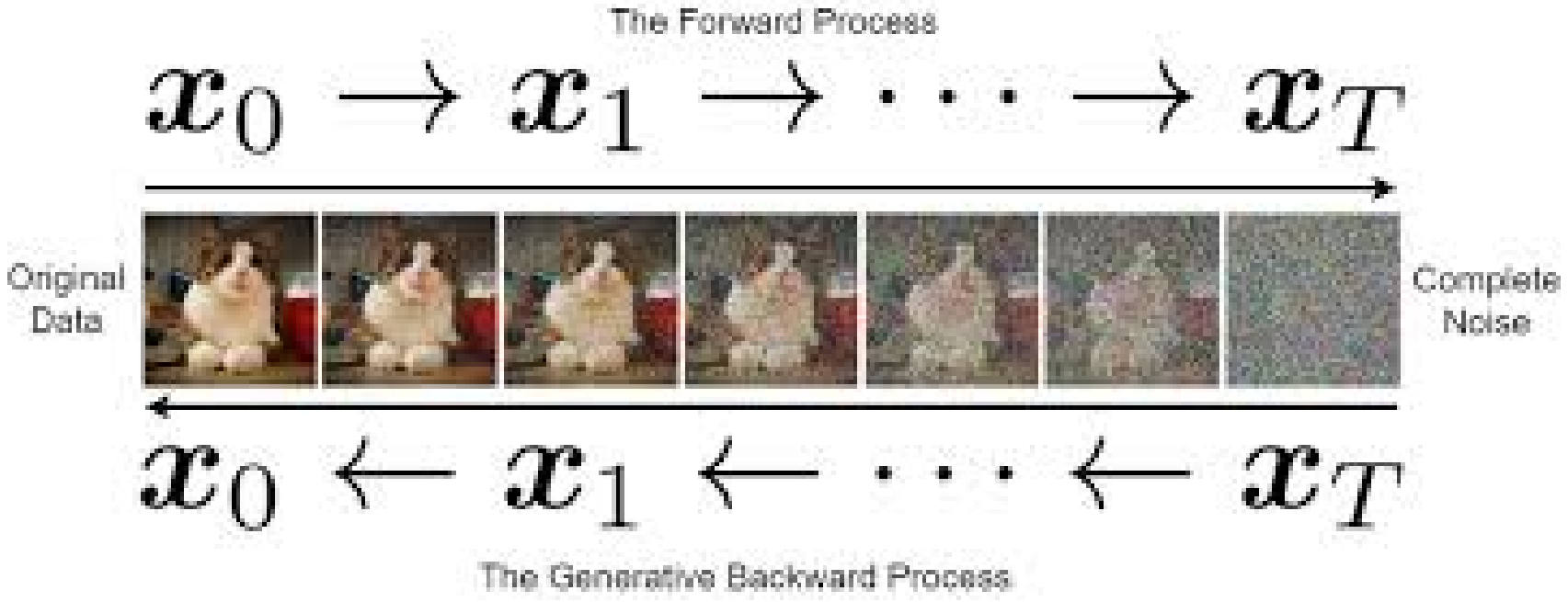
- $x_T \sim \mathcal{N}(0, I)$ (pure noise)
- Learn to reverse the process

Generation

1. Sample $x_T \sim \mathcal{N}(0, I)$
2. Denoise step-by-step
3. Obtain x_0

Forward $\to$ *Noise* *Reverse* $\to$ *Image*




The Forward Process
$x_0 \rightarrow x_1 \rightarrow \cdots \rightarrow x_T$
Original Data
Complete Noise
$x_0 \leftarrow x_1 \leftarrow \cdots \leftarrow x_T$
The Generative Backward Process

Forward Process

Forward encoding:

The forward process is fixed — no learning involved. It defines how noise is added via a schedule β_t. We go from data to a Gaussian distribution i.e. noise

How can we do this?:

- At each step t, add Gaussian noise:

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

- β_t is the **noise schedule**, increasing over time
- Example: $T = 1000$, $\beta_1 = 0.0001$, $\beta_T = 0.02$
- Closed form:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

Reverse denoising:

The reverse exists mathematically but requires the full data distribution.

$$p(x_0) = \int p(x_T) \prod_{t=1}^{T} p(x_{t-1} \mid x_t)$$

We approximate it using a neural network:

$$p(x_{t-1} \mid x_t) := q(x_{t-1} \mid x_t, x_\theta(x_t, t))$$

$$= \mathcal{N}\left(x_{t-1} \,\middle|\, \mu_\theta(x_t, t),\ \sigma^2_{t|t-1} I\right)$$

Key idea: Learn to denoise step-by-step

From ELBO to Denoising Objective

Reparameterization

Instead of modeling the full distribution, we parameterize the mean using a neural network:

$$x_\theta(x_t, t)$$

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Noise prediction parameterization

$$\epsilon_\theta(x_t, t) = \frac{x_t - \alpha_t x_\theta(x_t, t)}{\sigma_t}$$

Rewriting reconstruction term

$$\|x_0 - x_\theta(\alpha_t x_0 + \sigma_t \epsilon, t)\|^2$$

$$= \|\epsilon - \epsilon_\theta(\alpha_t x_0 + \sigma_t \epsilon, t)\|^2$$

Final DDPM Loss

ELBO Decomposition

$$\mathcal{L}(q) = \mathbb{E}_{q(x_1|x_0)}\left[\log p_\theta(x_0|x_1)\right] - D_{\mathrm{KL}}\left(q(x_T|x_0) \,\|\, p(x_T)\right)$$
$$- \sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}\left[D_{\mathrm{KL}}\left(q(x_{t-1}|x_t, x_0) \,\|\, p_\theta(x_{t-1}|x_t)\right)\right]$$

We optimize a variational lower bound on the data likelihood

DDPM Training Objective

Key Idea

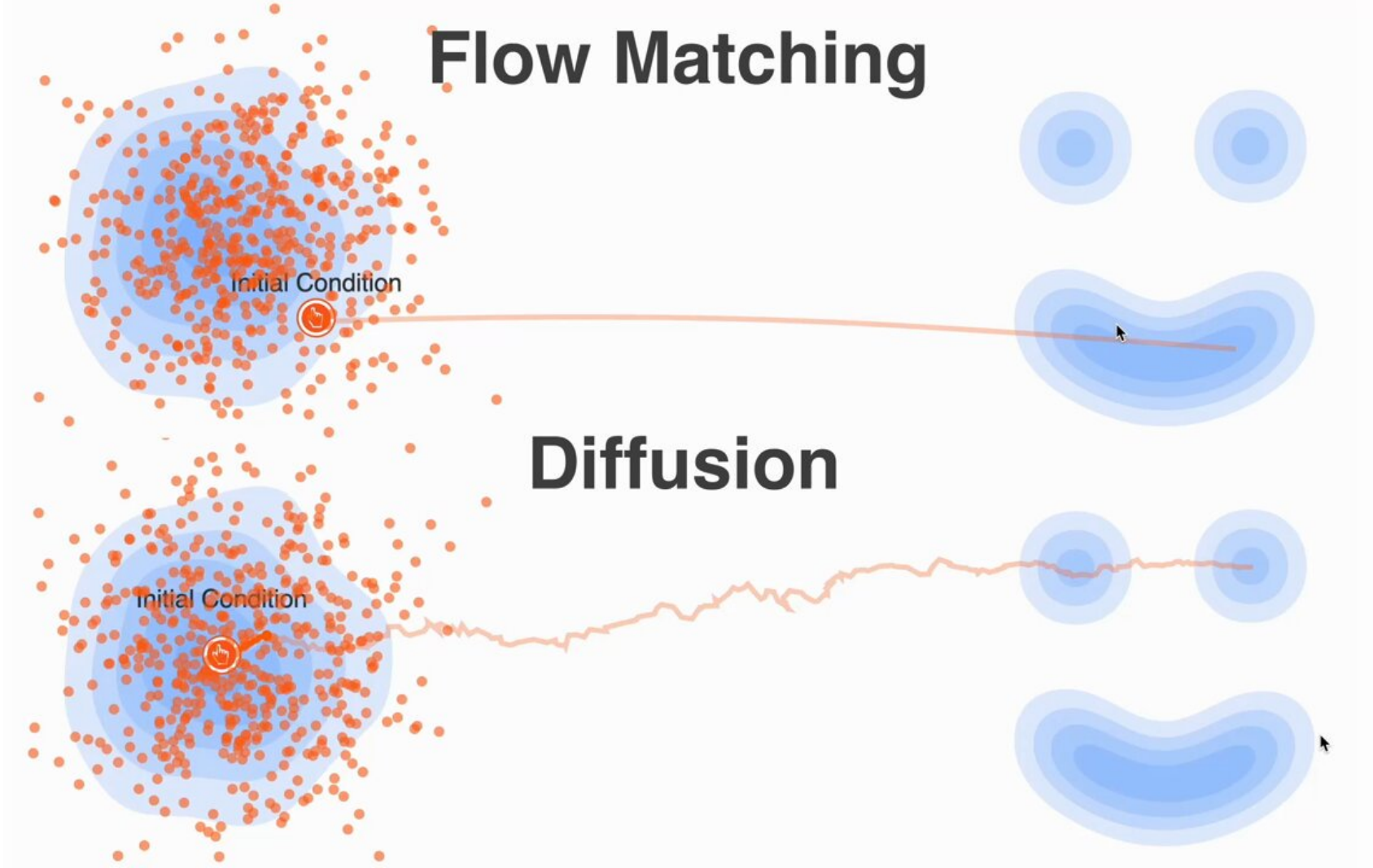
Instead of modeling the full distribution, we train the model to **predict the noise**.

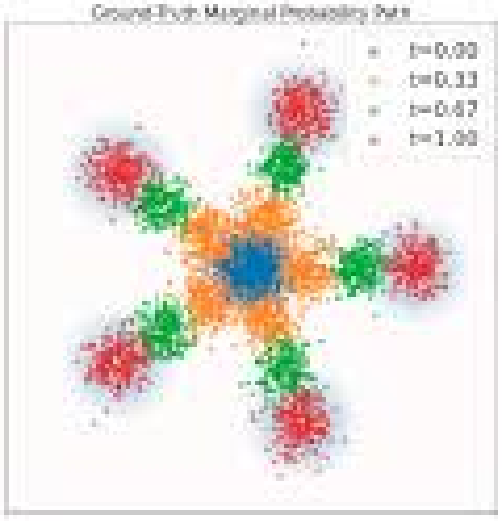
Simplified Loss

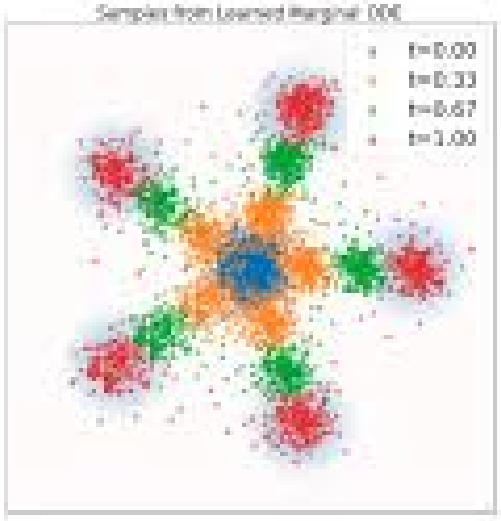
$$\mathcal{L} = \mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$$

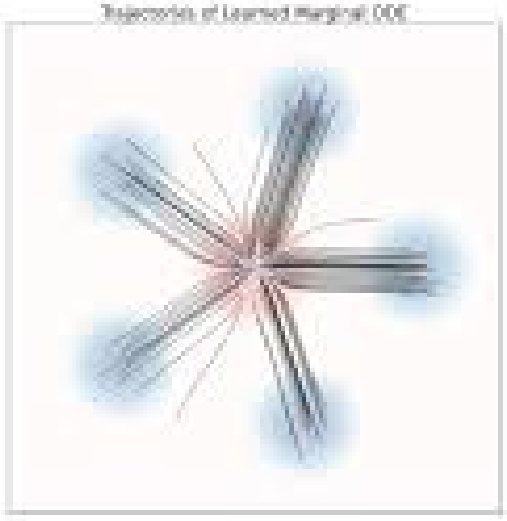
Where:

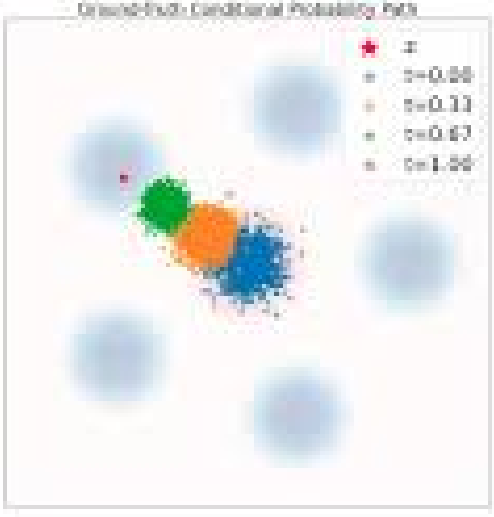
- $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon$
- $\epsilon \sim \mathcal{N}(0, I)$

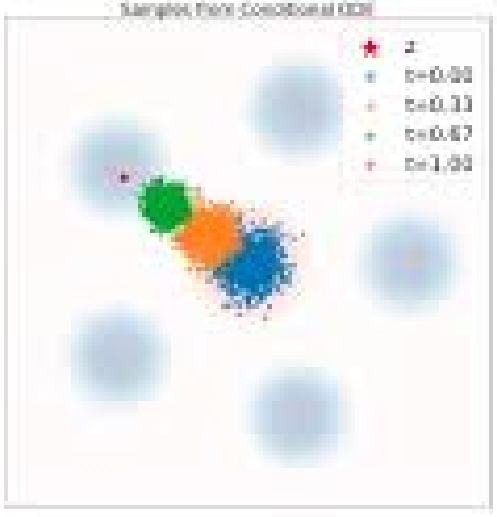

Flow Matching
Initial Condition
Diffusion
Initial Condition

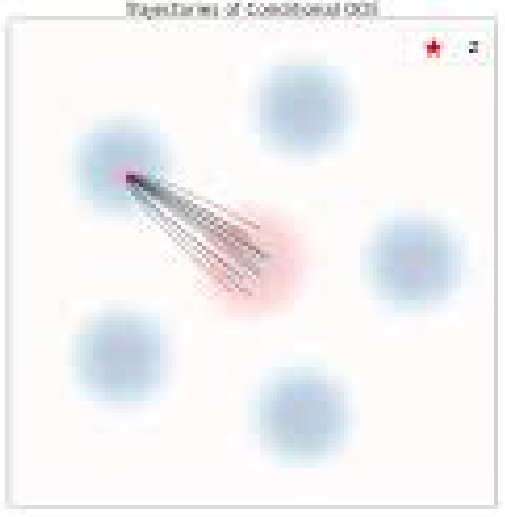

Ground-Truth Marginal Probability Path
t=0.00
t=0.33
t=0.67
t=1.00


Samples from Learned Marginal ODE
t=0.00
t=0.33
t=0.67
t=1.00


Trajectories of Learned Marginal ODE


Ground-Truth Conditional Probability Path
z
t=0.00
t=0.33
t=0.67
t=1.00


Samples from Conditional ODE
z
t=0.00
t=0.33
t=0.67
t=1.00


Trajectories of Conditional ODE
z

Dimensionality Problem

Why Pixel-Space Diffusion Fails

- Images live in high-dimensional space
- Example:
  - $64 \times 64 \times 3 = 12{,}288$
  - $256 \times 256 \times 3 = 196{,}608$
  - $1024 \times 1024 \times 3 = 3{,}145{,}728$
- Attention operates on $N = h \times w$ tokens
- Cost:

$$\mathcal{O}(N^2) = \mathcal{O}((h \cdot w)^2)$$

Attention Scaling

Res	N	N^2	Memory
32^2	1K	1M	4 MB
64^2	4K	16M	67 MB
128^2	16K	268M	1.07 GB
256^2	65K	4.3B	17 GB
512^2	262K	68B	275 GB

Training Cost

- ADM (ImageNet):
  - 916 V100-days
- 50K samples:
  - $\sim$5 days on A100

Takeaway

Latent Diffusion Models (LDM)

Main Idea

Compress $\rightarrow$ Diffusion in latent space $\rightarrow$ Decode to image space

Stage 1: Autoencoder

- $x \rightarrow E \rightarrow z \rightarrow D \rightarrow \tilde{x}$
- z is 4–$8\times$ smaller than x
- $\tilde{x}$ is perceptually similar to x
- E and D are frozen after training

Stage 2: Diffusion in Latent Space

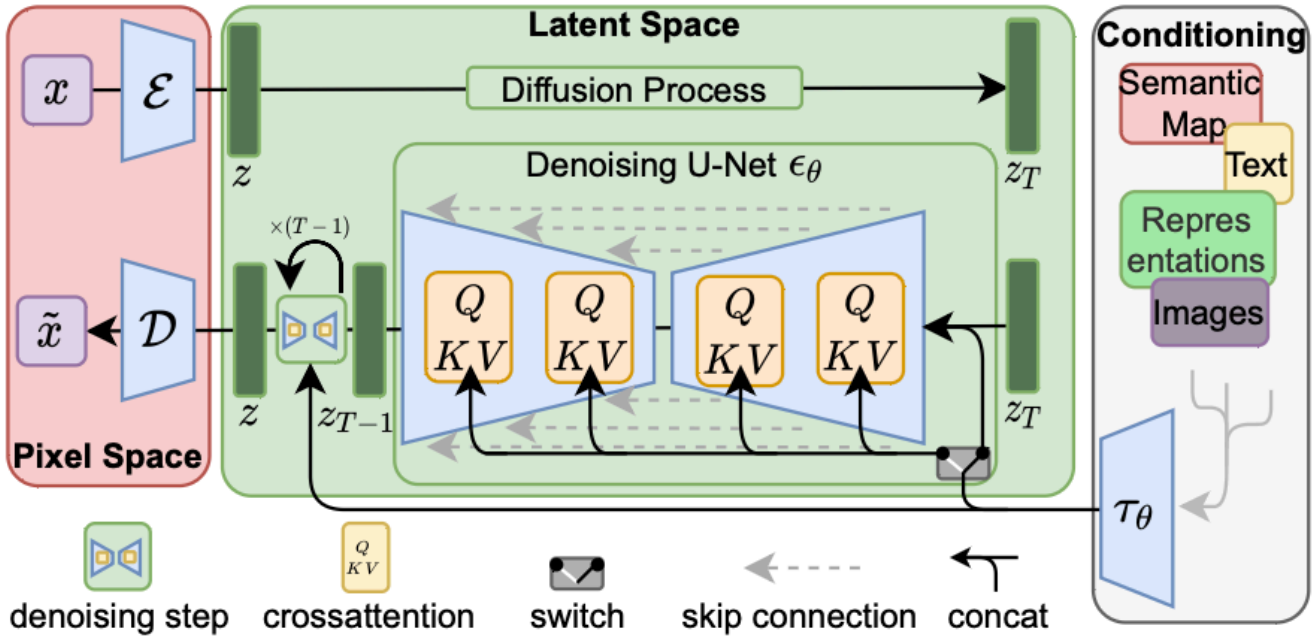
- $z \rightarrow z_T \sim \mathcal{N}(0, I)$
- Learn reverse process:

$$z_T \rightarrow z_0$$

- Decode once:

$$z_0 \rightarrow D \rightarrow \tilde{x}$$

LDM Architecture



Key Idea

Diffusion operates in latent space while conditioning is injected via cross-attention.

The Main idea

The autoencoder is trained with three complementary losses — each fixing a specific failure mode that the others cannot address.

The full objective

$$L_{\text{Autoencoder}} = \min_{E,D} \max_{\psi} \Big(L_{\text{rec}}(x, D(E(x))) - L_{\text{adv}}(D(E(x))) + \log D_{\psi}(x) + L_{\text{reg}}(x; E, D) \Big)$$

Loss 1 —Perceptual Reconstruction

$$L_{\text{rec}} = L_{L1}(x, \tilde{x}) + \lambda_{\text{perc}} \|\phi_l(x) - \phi_l(\tilde{x})\|^2$$

ϕ_l = activations at layer l of frozen VGG network

Loss 2 —Adversarial L_{adv} (GAN training)

Decoder tries to fool:

$$D_\psi(\tilde{x}) \to 1$$

Discriminator learns:

$$D_\psi(x) \to 1, \quad D_\psi(\tilde{x}) \to 0$$

A patch discriminator classifies 70×70 local patches independently — providing spatially precise feedback.

Loss 3 —Regularization $L_{\mathbf{KL}}$

KL-reg: push $q(z|x)$ toward $\mathcal{N}(0,1)$

$$L_{\text{KL}} = \frac{1}{2}\sum_i \left(\mu_i^2 + \sigma_i^2 - 1 - \log \sigma_i^2\right)$$

weight $\lambda_{\text{KL}} = 10^{-6}$ —near zero, just prevents degenerate scales

Role of Each Loss

- $L_{\text{rec}} \rightarrow$ global structure, semantic content, no blurring
- $L_{\text{adv}} \rightarrow$ local texture realism, sharpness, patch-level detail
- $L_{\text{KL}} \rightarrow$ latent space numerical stability for diffusion training

Training

$$x \to E \to z \to D \to \tilde{x} \to \text{VGG} \to L_{\text{perc}} \to D_{\psi} \to L_{\text{adv}}$$

$$z \to L_{\text{KL}}$$

Inference

$$x \to E \to z \to \text{diffusion model} \to z \to D \to \tilde{x}$$

VGG and D_{ψ} completely absent

Conditioning via Cross-Attention

The Main idea

A single conditioning mechanism — cross-attention into the UNet — handles text, class labels, layouts, and images without any architectural changes.

The full objective

$$L_{\text{LDM}} := \mathbb{E}_{\epsilon(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$$

Two new components

- y: conditioning input (text, class label, bounding boxes)
- $\tau_\theta(y)$: encoder mapping y to a sequence of vectors

Both τ_θ and ϵ_θ are trained jointly end-to-end.

Cross-Attention Mechanism

Encode the condition

- Text prompt $\rightarrow$ BERT-style transformer $\rightarrow 77 \times 1280$
- Class label $\rightarrow$ embedding $\rightarrow 1 \times 512$
- Bounding boxes $\rightarrow$ small transformer $\rightarrow M \times 512$
- Image alignment $\rightarrow$ identity (concatenate) $\rightarrow$ spatial

Inject via cross-attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V$$

$$Q = W_Q^{(i)} \cdot \phi_i(z_t), \quad K = W_K^{(i)} \cdot \tau_\theta(y), \quad V = W_V^{(i)} \cdot \tau_\theta(y)$$

Complexity with LDM

Computation

Attention matrix: $QK^T \in \mathbb{R}^{N \times M}$ (not $N \times N$)

$$\text{Cost: } \mathcal{O}(N \times M)$$

With $M = 77$, $N = 256$:

$$256 \times 77 = 19{,}712 \quad \text{(negligible)}$$

Understanding Cross-Attention

What the attention scores mean

Each row of

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$$

is a probability distribution over the M condition tokens for one spatial position.

- Cat region $\rightarrow$ high attention on "cat"
- Sofa region $\rightarrow$ high attention on "red", "sofa"
- Sky region $\rightarrow$ attention on context tokens

Modified UNet Attention Block

Input: $\phi_i(z_t) \rightarrow$ LayerNorm

1. **Self-attention:**
   $Q = K = V = \phi_i$
   (spatial positions interact)

2. **MLP:** position-wise feedforward

3. **Cross-attention:**
   $Q = \phi_i, \quad K = V = \tau_\theta(y)$
   (inject conditioning)

Output

Updated ϕ_i carrying both spatial context and conditioning

Results and Discussion

Compression Tradeoff: Choosing f

Key Idea

Downsampling factor f controls the division of work between:

- Autoencoder (perception)
- Diffusion model (generation)

Two Failure Modes

- **Too little compression ($f = 1, 2$):**
  - Latent $\approx$ pixel space
  - Diffusion must learn everything
  - Slow training, poor FID
- **Too much compression ($f = 16, 32$):**
  - Autoencoder loses detail
  - Irrecoverable information loss
  - Quality ceiling limited

Sweet Spot

Compression Tradeoff: Evidence

Same compute, same model —only vary f

Model	FID ()
LDM-1 (pixel)	$\sim$38+
LDM-4 (latent)	much lower

Key Observations

- LDM-4 improves FID significantly under same compute
- Faster inference at all sampling steps

Key Insight

Smaller space + fewer steps = faster AND better

Practical Recommendation

- $f = 4$: best quality (complex datasets)
- $f = 8$: faster, still strong performance

Results: One Framework, Multiple Tasks

ImageNet 256Œ256

Model	FID	Params	Compute
ADM-G	4.59	608M	916 V100-days
LDM-4-G	3.60	400M	∼271 V100-days

Across Tasks

- Unconditional: competitive with StyleGAN2
- Text-to-image: comparable to GLIDE (4Œ fewer params)
- Super-resolution: better FID than SR3
- Inpainting: better FID + diverse outputs

Thank You

Questions?

"The future depends on some graduate student who is deeply suspicious of everything I have said.

— Geoffrey Hinton