Deep Generative Models: Diffusion Models

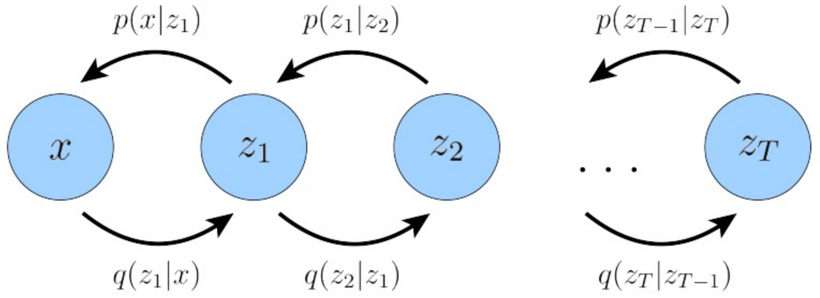
Fall Semester 2025

René Vidal
Director of the Center for Innovation in Data Engineering and Science (IDEAS)
Rachleff University Professor, University of Pennsylvania
Amazon Scholar & Chief Scientist at NORCE

Outline

- Markov Hierarchical Variational Auto Encoders (MHVAEs)
  - Autoregressive Encoder and Autoregressive Decoder of an MHVAE
  - Derivation of the ELBO of an MHVAE
- Diffusion Models as MHVAEs with a Linear Gaussian Autoregressive Latent Space
  - Forward Diffusion Process
  - Reverse Diffusion Process
  - ELBO for Diffusion Models as a particular case of the ELBO for MHVAEs
- **Implementation Details: UNet architecture, Training and Sampling Strategies**
- Application of Diffusion Models
  - Stable Diffusion: Text-Conditioned Diffusion Model
  - ControlNet: Multimodal Control for Consistent Synthesis

Implementation (DDPM)



- The Denoising Diffusion Probabilistic Model (DDPM) fixes the noise variances α_t of the forward process and learns only the backward (denoising) process [Ho et al., 2020].

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

- For the backward process, we use reparameterization of ELBO

$$\log p(x) \geq \underbrace{\mathbb{E}_{q_\phi(x_1|x_0)}[\log p_\theta(x_0 \mid x_1)]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}\left(q_\phi(x_T \mid x_0) \,||\, p_\theta(x_T)\right)}_{\text{prior matching term}} - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q_\phi(x_t|x_0)}\left[D_{\mathrm{KL}}\left(q_\phi(x_{t-1} \mid x_t, x_0) \,||\, p_\theta(x_{t-1} \mid x_t)\right)\right]}_{\text{score matching term}}$$

$$= -\underbrace{\sum_{t=1}^{T} \mathbb{E}_{q_\phi(x_t|x_0)}\left[D_{\mathrm{KL}}\left(q_\phi(x_{t-1} \mid x_t, x_0) \,||\, p_\theta(x_{t-1} \mid x_t)\right)\right]}_{\text{reconstruction term} + \text{score matching term}} = -\sum_{t=1}^{T} \mathbb{E}_{q_\phi(x_t|x_0)}\left[\frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} [\|\epsilon_0 - \hat{\epsilon}_\theta(x_t, t)\|_2^2]\right]$$

$$\boxed{\sigma_q^2(t) = \frac{(1-\alpha_t)(1-\overline{\alpha_{t-1}})}{1-\overline{\alpha_t}}}$$

- For each x, we need to compute T terms in the sum, which is expensive for typical values of T (T= 1000).

SGD over time-steps

- DDPM minimizes the ELBO efficiently by performing SGD over the set of timesteps $[T] = \{1, 2, \dots, T\}$.
- Using linearity of expectation and letting $t \sim \mathrm{Unif}([T])$

$$\log p(x) \geq -D_{\mathrm{KL}}\left(q_\phi(x_T \mid x_0) \,||\, p_\theta(x_T)\right) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(x_t|x_0)}\left[\frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} \|\epsilon_0 - \hat{\epsilon}_\theta(x_t, t)\|_2^2\right]$$

$$= -D_{\mathrm{KL}}\left(q_\phi(x_T \mid x_0) \,||\, p_\theta(x_T)\right) - \mathbb{E}_{q_\phi(x_t|x_0)}\left[\sum_{t=1}^{T} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} \|\epsilon_0 - \hat{\epsilon}_\theta(x_t, t)\|_2^2\right]$$

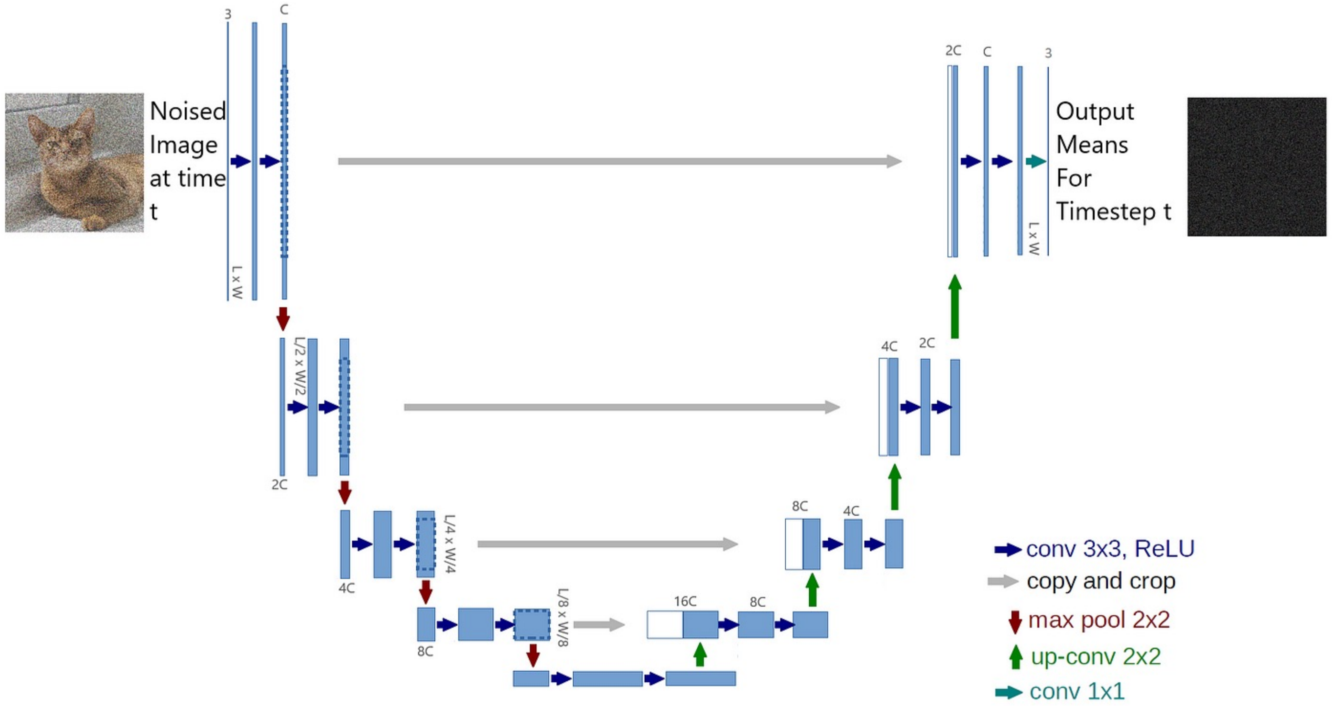
$$= -D_{\mathrm{KL}}\left(q_\phi(x_T \mid x_0) \,||\, p_\theta(x_T)\right) - \mathbb{E}_{q_\phi(x_t|x_0),\, t}\left[\frac{T}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} \|\epsilon_0 - \hat{\epsilon}_\theta(x_t, t)\|_2^2\right]$$

- Thus, the sampling procedure computes only 1 term instead of T terms.
- The term $-D_{\mathrm{KL}}\left(q_\phi(x_T \mid x_0) \,||\, p_\theta(x_T)\right)$ is constant during training, since $q_\phi(x_T \mid x_0)$ is not learnable.
- Thus, the simplified (unweighted) learning objective used in DDPM is

$$\mathbb{E}_{q_\phi(x_t|x_0),\, t}\left[\|\epsilon_0 - \hat{\epsilon}_\theta(x_t, t)\|_2^2\right]$$

Architecture of DDPM

- DDPM uses a U-Net with residual connection and self-attention layers to represent the noise $\overline{\epsilon_\theta}(x_t, t)$.
- Since the parameter $\overline{\epsilon_\theta}(x_t, t)$ depends both on x_t and on time t, a time embedding needs to be provided as input to each unit of the U-Net.

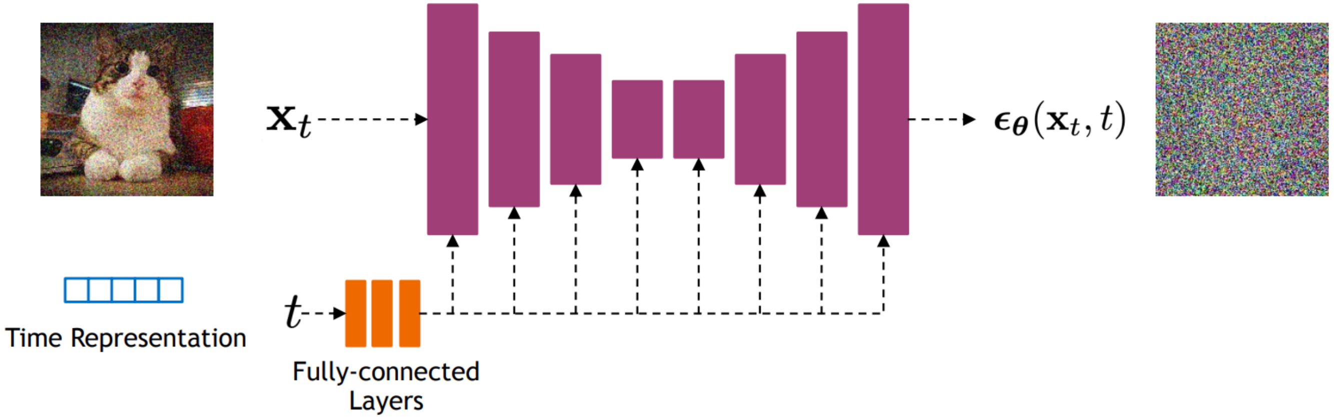


Time Encoding

- The time representation is implemented using sinusoidal positional embeddings.
- Given a time step $t \in [T]$ and an embedding dimension d, the sinusoidal positional embedding $\mathrm{SPE}(t)$ is given for $i = 0, 1, \ldots, \frac{d}{2} - 1$ as

$$\mathrm{SPE}(t)_{2i} = \sin\left(\frac{t}{10^{\frac{6i}{d}}}\right), \mathrm{SPE}(t)_{2i+1} = \cos\left(\frac{t}{10^{\frac{6i}{d}}}\right)$$

- This produces an fixed-length vector $\mathrm{SPE}(t)$ that encodes the time smoothly, since small changes in t cause predictable oscillations in the time embedding.
- The time embedding is given as input to all units of the U-Net.



Training and Sampling in DDPM

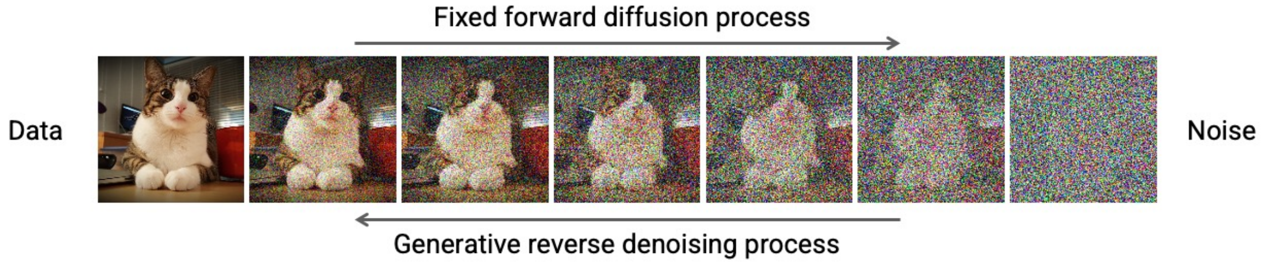
- DDPM implements the training procedure by performing SGD on the set of training images over timesteps.
- The sampling procedure executes iteratively the denoising process from a Gaussian initialization x_T.

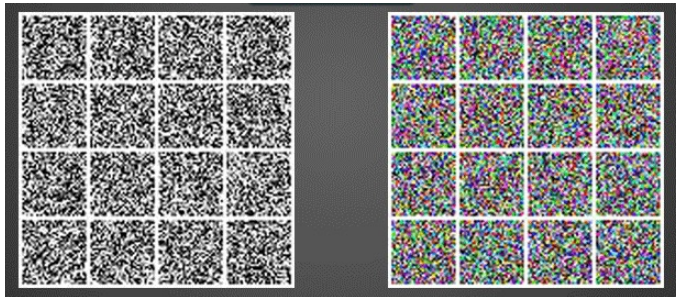
Algorithm 1 Training

1: **repeat**
2: $\quad \mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $\quad t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $\quad$ Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\big(\underbrace{\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}}_{x_t}, t\big) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \underbrace{\boxed{\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)}}_{\mu_\theta(x_t, t)} + \underbrace{\boxed{\sigma_t}}_{\sigma_t = \sqrt{\beta_t}}\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$





DDPM Noise Scheduler

- **Noise Scheduler for Forward Process**: in DDPM, the scheduler refers to how the noise variance ($\beta_t = 1 - \alpha_t$) changes across the diffusion timesteps $t = 1, \ldots, T$.
  - **Linear Schedule**: β_t increases linearly from initial value (10^{-4}) to a maximum value (0.02)

$$\beta_t = \beta_{\text{start}} + t\,\frac{\beta_{\text{end}} - \beta_{\text{start}}}{T - 1}, t = 0, \ldots, T - 1$$

  - **Cosine Schedule**: Uses a cosine function to define β_t, which better preserves signal early in the process and decays more gently near the end. It helps maintain more information in the intermediate steps, improve sample quality, require fewer steps for comparable results.

$$f(t) = \cos\left(\frac{t/T+s}{1+s} \cdot \frac{\pi}{2}\right), t = 0, \ldots, T - 1, \qquad \overline{\alpha}_t = \frac{f(t)}{f(0)}, \qquad \beta_t = 1 - \frac{\overline{\alpha}_t}{\overline{\alpha}_{t-1}}, \qquad s = 0.008.$$

- **Training Data**: For each training image x_0 and timestep t, a noisy image x_t is generated as: $x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\,\epsilon, \quad \epsilon \sim \mathcal{N}(\epsilon; 0, I)$.

Training Curves & Test Metrics

- The Inception Score (IS) and Fréchet Inception Distance (FID) are two common metrics for evaluating the quality of generated images.
- IS measures how realistic and diverse the generated samples are by comparing the distribution of generated samples p_g with the output of an Inceptionv3 network

$$\mathrm{IS} = \exp\left(\mathbb{E}_{x\sim p_g}[D_{KL}(p(y|x)\ ||\ p(y))]\right), p(y|x) = \mathrm{InceptionNet}\,(x)$$

- The FID measures how close the distribution of the generated images with parameters is (μ_g, Σ_g) to the real data distribution with parameters (μ_r, Σ_r)

$$\mathrm{FID} = ||\mu_r - \mu_g||^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_\mathrm{g} - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}\right)$$

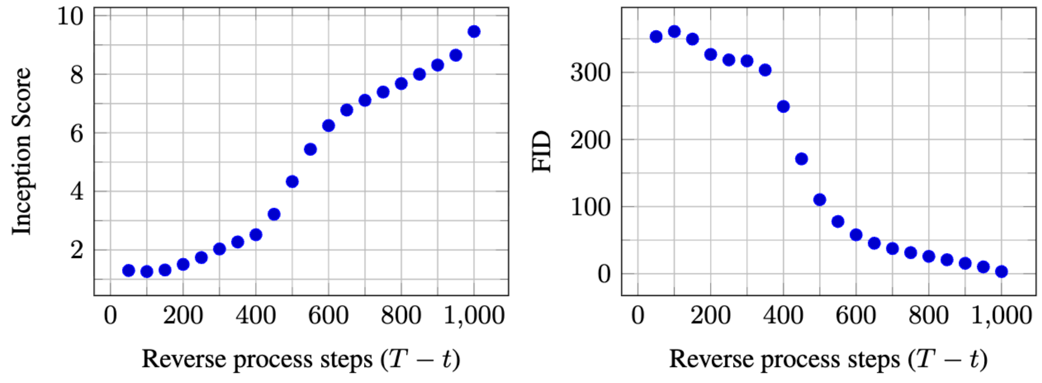


Figure 10: Unconditional CIFAR10 progressive sampling quality over time

Generated Samples of DDPM