Denoising Diffusion Probabilistic Models

Minh Hoang - Christopher Won

December 19, 2025

Introduction

- Denoising Diffusion Probabilistic Models (DDPMs) are a class of deep generative models—machine learning systems trained to synthesize new data that closely resembles their training distribution.
- The method consists of two phases: (1) a **forward diffusion process** that progressively adds Gaussian noise to data over many timesteps until the signal becomes pure noise, and (2) a **reverse process**, learned by a neural network, that iteratively denoises random noise to generate new samples.

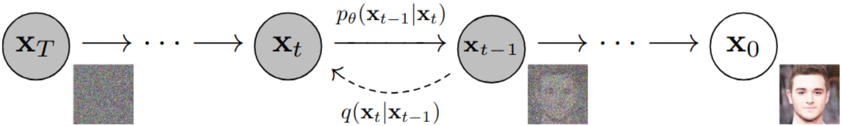


Figure: forward diffusion process and reverse process

Forward Diffusion

- Given an image $\mathbf{x}_0 \in \mathbb{R}^d$, the *forward process* iteratively adds noise to create a sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ via the recurrence for the measurable function f :

$$\mathbf{x}_t = \sqrt{1-\beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\,\boldsymbol{\epsilon}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\epsilon}_t) \tag{1}$$

  Here, each $\boldsymbol{\epsilon}_t \overset{i.i.d}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, and $\beta_t \in (0,1)$ is a pre-specified scalar called the variance of the added noise in the update at step t. The collection $\{\beta_t\}_{t=1}^T$ is increasing.

- Given a data distribution $\mathbf{x}_0 \in \mathbb{R}^d \sim q(\mathbf{x}_0)$, the forward Markov process generates a sequence of random variables $\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_T$ with transition kernel $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$. The joint distribution of $\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_T$ conditioned on $\mathbf{x}_0$, denoted as $q(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{x}_0)$ is

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) := q(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \tag{2}$$

  where $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I})$

Forward Diffusion

- Analytical form of $q(\mathbf{x}_t \mid \mathbf{x}_0)$ for all $t \in \{0, 1, \cdots, T\}$. Specifically, denoting $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=0}^{t} \alpha_s$, we have

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}) \tag{3}$$

- Given $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, we can easily obtain a sample of $\mathbf{x}_t$ by sampling a Gaussian vector $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and applying the transformation yields

$$\mathbf{x}_t \overset{(d)}{=} \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon} \tag{4}$$

When $\bar{\alpha}_T \approx 0$, $\mathbf{x}_T$ is almost Gaussian in distribution, so we have

$$q(\mathbf{x}_T) \approx p_{\text{prior}}(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}),$$

with $q(\mathbf{x}_T) = p_{\text{prior}}(\mathbf{x}_T)$ in the limit as $T \to \infty$. Here, p_{prior} denotes the *prior distribution* over the final latent variable $\mathbf{x}_T$, which is conventionally chosen to be a standard isotropic Gaussian.

Reverse Process

- **Goal**: Sample from the data distribution $q(\mathbf{x}_0)$ by reversing a noising process.
- **Problem**: The true reverse transition $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ requires $q(\mathbf{x}_0)$ and involves intractable integrals:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \int q(\mathbf{x}_{t-1} \mid \mathbf{x}_0) q(\mathbf{x}_0) d\mathbf{x}_0}{\int q(\mathbf{x}_t \mid \mathbf{x}_0) q(\mathbf{x}_0) d\mathbf{x}_0}$$

  $\rightarrow$ We don't know $q(\mathbf{x}_0)$ (p_{complex}), and the integrals are impossible to compute.
- **Solution**: Approximate the reverse process with a neural network:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$$

  The network learns to denoise step-by-step using only $\mathbf{x}_t$ and t.
- **Reverse process** is a Markov chain:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t),$$

  with $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Methodology

Definition (KL divergence)

Let $(\mathcal{X}, \mathcal{F})$ be a measurable space and $\mathbb{P}$ and $\mathbb{Q}$ are its probability measures. If both $\mathbb{P}$ and $\mathbb{Q}$ are both absolutely continuous with respect to the Lebesgue measure μ, then there exist densities p and q respectively such that

$$D_{\mathrm{KL}}(\mathbb{Q} \parallel \mathbb{P}) = \int_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)} \mu(dx), \quad \text{when } \mathbb{Q} \ll \mathbb{P}$$

In Generative Modeling, we want to learn a model $p_\theta(\mathbf{x}_0)$ that approximates the real data complex distribution $q(\mathbf{x}_0)$, so that we can generate new samples of data. One can show that

$$\arg\min_\theta \mathcal{D}_{\mathrm{KL}}\left(q(\mathbf{x}_0) \mid p_\theta(\mathbf{x}_0)\right) = \arg\max_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0)\right]$$

This shows minizing the KL-divergence is equivalent to performing the MLE:

$$\theta^* = \arg\max_\theta \mathbb{E}_{q(\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0)\right]$$

- However, the marginal likelihood

$$p_\theta(\mathbf{x}_0) = \int_{\mathbf{x}_{1:T}} p_\theta(\mathbf{x}_{0:T})\, d\mathbf{x}_{1:T}.$$

is intractable: the latent trajectory $\mathbf{x}_{1:T}$ is high-dimensional, and the neural reverse process prevents closed-form integration. Thus, $\log p_\theta(\mathbf{x}_0)$ cannot be computed directly, but we can optimize a variational lower bound for $\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)]$.

Theorem (Evidence lower bound (ELBO) on log likelihood of DDPM)

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \\
&\leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}\mid\mathbf{x}_0)}\Bigg[\mathcal{D}_{KL}\left(q(\mathbf{x}_T \mid \mathbf{x}_0)\|p(\mathbf{x}_T)\right) + \sum_{t=2}^{T} \mathcal{D}_{KL}\left(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\right) \\
&\qquad - \log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)\Bigg]
\end{aligned}$$

Theorem (Evidence lower bound (ELBO) on log likelihood of DDPM)

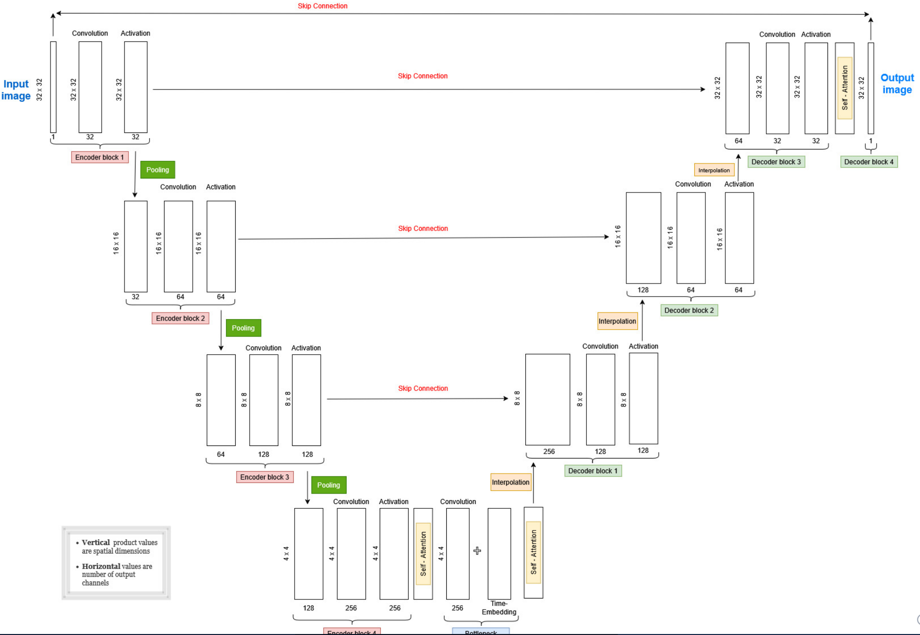
$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right]$$
$$\leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\Bigg[\mathcal{D}_{KL}\left(q(\mathbf{x}_T \mid \mathbf{x}_0)\|p(\mathbf{x}_T)\right) + \sum_{t=2}^{T}\mathcal{D}_{KL}\left(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\right)$$
$$- \log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)\Bigg]$$

- To simplify the training loss, the model is trained with the simple loss function to minimize:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t\sim\mathcal{U}[1,T],\mathbf{x}_0\sim q(\mathbf{x}_0),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon},\ t\right)\right\|^2\right]$$

Results

- We fix the forward process variances $\{\beta_t\}_{t=1}^{T}$ to constants that increase linearly from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$ across all models in our experiments.
- The reverse (denoising) process is modeled using a U-Net architecture. We train all models with the Adam optimizer using a constant learning rate of 10^{-3}.


Skip Connection
Input image
Output image
Convolution
Activation
Pooling
Interpolation
Self - Attention
Time-Embedding
32 x 32
16 x 16
8 x 8
4 x 4
Encoder block 1
Encoder block 2
Encoder block 3
Encoder block 4
Bottleneck
Decoder block 1
Decoder block 2
Decoder block 3
Decoder block 4
• Vertical product values are spatial dimensions
• Horizontal values are number of output channels

MNIST

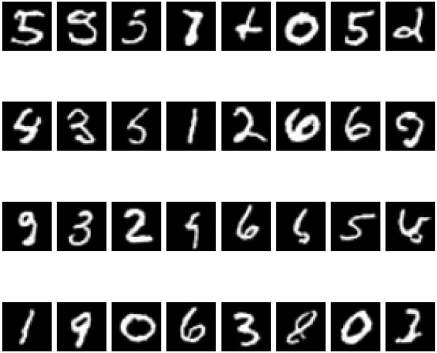


Figure: Diffusion_model_2 generated sample with $T = 500$.

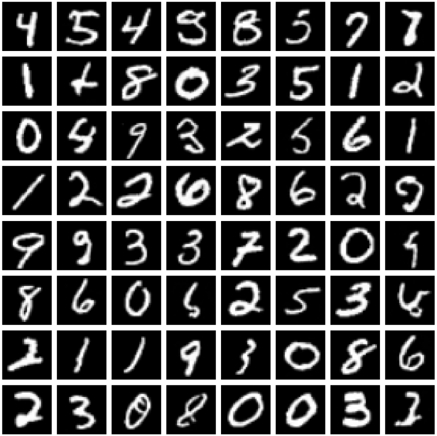


Figure: Diffusion_model_2 comparison with real MNIST samples

CIFAR-10 Reverse Process for Image Generation

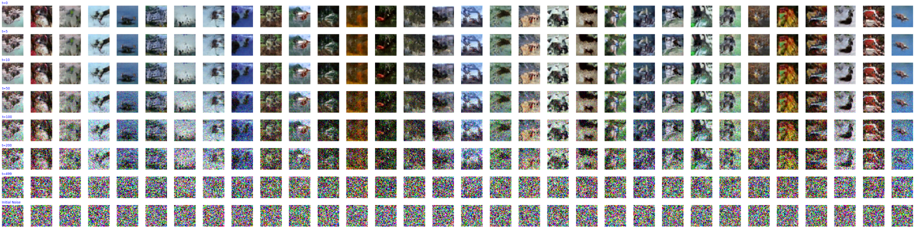


Figure: CIFAR-10_diffusion_model_3 Progressive Generation

CIFAR-10

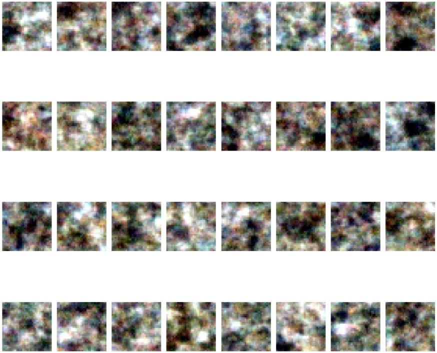


Figure: CIFAR-10_diffusion_model_1 generated samples with $T = 500$.

Figure: Real vs Generated samples comparison from Figure 7

CIFAR-10

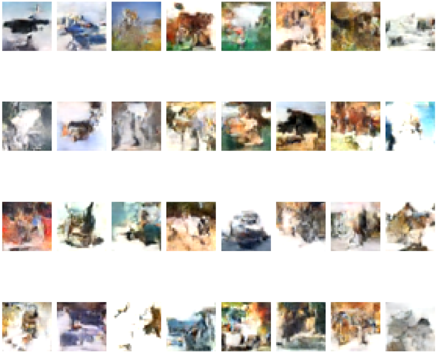


Figure: CIFAR-10_diffusion_model_2 generated samples with $T = 1000$.

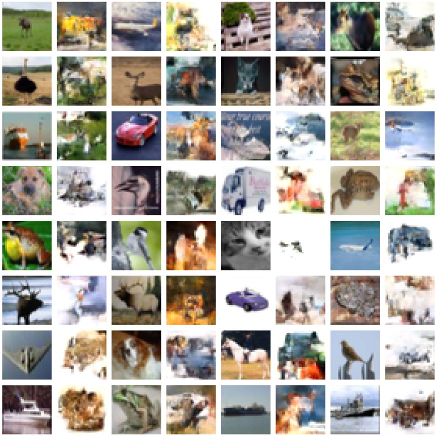


Figure: Real vs Generated samples comparison from Figure 9

CIFAR-10

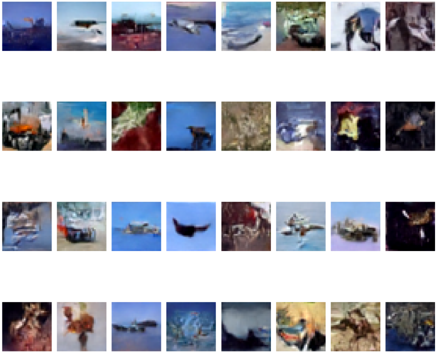


Figure: CIFAR-10_diffusion_model_3 generated samples with $T = 1000$.

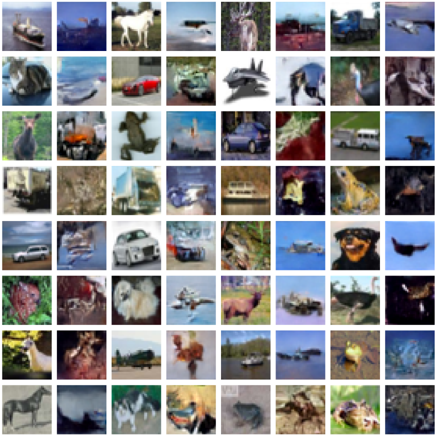


Figure: Real vs Generated samples comparison from Figure 11

Thank you for listening

References

Jonathan Ho, Ajay Jain, and Pieter Abbeel. (2020). Denoising Diffusion Probabilistic Models. arXiv preprint arXiv:2006.11239

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv preprint arXiv:1503.03585.