Deep Generative Models

Lecture 7

Roman Isachenko

Moscow Institute of Physics and Technology
Yandex School of Data Analysis

2025, Autumn

Recap of Previous Lecture

Frechet Inception Distance (FID)

For normal distributions $p_{\text{data}}(\mathbf{x}_1) = \mathcal{N}(\boldsymbol{\mu}_{\text{data}}, \boldsymbol{\Sigma}_{\text{data}})$, $p(\mathbf{x}_2) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$:

$$\mathrm{FID}(p_{\text{data}}, p_{\boldsymbol{\theta}}) = W_2^2(p_{\text{data}}, p_{\boldsymbol{\theta}}) = \inf_{\gamma \in \Gamma(p_{\text{data}}, p_{\boldsymbol{\theta}})} \mathbb{E}_{(\mathbf{x}_1, \mathbf{x}_2) \sim \gamma} \|\mathbf{x}_1 - \mathbf{x}_2\|^2$$

$$= \|\boldsymbol{\mu}_{\text{data}} - \boldsymbol{\mu}_{\boldsymbol{\theta}}\|^2 + \mathrm{tr}\left[\boldsymbol{\Sigma}_{\text{data}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}} - 2\left(\boldsymbol{\Sigma}_{\text{data}}^{1/2} \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Sigma}_{\text{data}}^{1/2}\right)^{1/2}\right]$$

- Requires a large sample size for evaluation
- FID computation is relatively slow
- Results are highly dependent on the chosen pretrained classifier
- Relies on the normality assumption

Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017

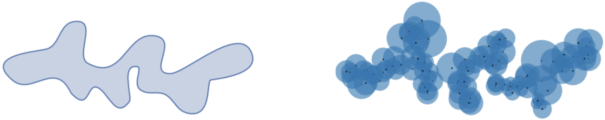
Recap of Previous Lecture

- $\mathcal{S}_{\text{data}} = \{\mathbf{x}_i\}_{i=1}^n \sim p_{\text{data}}(\mathbf{x})$: real samples
- $\mathcal{S}_{\boldsymbol{\theta}} = \{\mathbf{x}_i\}_{i=1}^n \sim p_{\boldsymbol{\theta}}(\mathbf{x})$: generated samples

Define a binary indicator function:

$$\mathbb{I}(\mathbf{x}, \mathcal{S}) = \begin{cases} 1, & \text{if } \exists\, \mathbf{x}' \in \mathcal{S} : \|\mathbf{x} - \mathbf{x}'\|_2 \leq \|\mathbf{x}' - \text{NN}_k(\mathbf{x}', \mathcal{S})\|_2; \\ 0, & \text{otherwise} \end{cases}$$

$$\text{Pr}(\mathcal{S}_{\text{data}}, \mathcal{S}_{\boldsymbol{\theta}}) = \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{S}_{\boldsymbol{\theta}}} \mathbb{I}(\mathbf{x}, \mathcal{S}_{\text{data}}); \quad \text{Rec}(\mathcal{S}_{\text{data}}, \mathcal{S}_{\boldsymbol{\theta}}) = \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{S}_{\text{data}}} \mathbb{I}(\mathbf{x}, \mathcal{S}_{\boldsymbol{\theta}}).$$



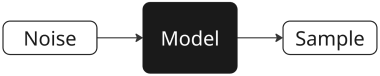
(a) True manifold (b) Approx. manifold

The samples are embedded using a pretrained network, as in FID evaluation.

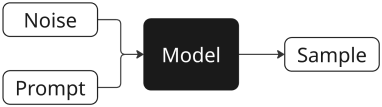
Kynkäänniemi T. et al. Improved precision and recall metric for assessing generative models, 2019

Recap of Previous Lecture

Unconditional Model

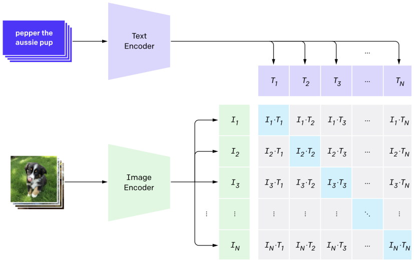


Conditional Model



We require metrics that evaluate not only the quality of generated images, but also their relevance to the prompt.

CLIP Score



Radford A. et al. Learning transferable visual models from natural language supervision, 2021

Recap of Previous Lecture

- No perfect automatic evaluation metric exists
- The most reliable assessment is via human evaluation
- It's important to evaluate a variety of model aspects

Human Evaluation

Аспект	Yandex ART 2.0	Mj 6.1	Mj 6	Ideogram	Recraft	Google Imagen3	Dall-E 3	FLUX	SBER Kandi3.1
Релевантность	0,59	0,58	0,63	**0,45**	**0,51**	**0,50**	**0,50**	**0,54**	0,75
Эстетика	**0,49**	0,55	0,55	**0,51**	**0,51**	0,61	0,61	**0,54**	0,59
Комплексность	**0,44**	0,73	0,70	0,68	0,76	0,75	0,75	0,71	0,74
Дефектность	0,69	0,57	0,68	0,55	0,59	0,63	0,63	**0,50**	0,75
Предпочтение	0,66	0,60	0,69	**0,49**	**0,54**	0,63	0,63	**0,51**	0,84

YandexART 2.5, 2025

Recap of Previous Lecture

Langevin Dynamics

Let $\mathbf{x}_0$ be a random vector. Under mild regularity conditions, samples from the following dynamics will eventually follow $p_{\boldsymbol{\theta}}(\mathbf{x})$ (for sufficiently small η and large l):

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \nabla_{\mathbf{x}_l} \log p_{\boldsymbol{\theta}}(\mathbf{x}_l) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I}).$$

- The density $p_{\boldsymbol{\theta}}(\mathbf{x})$ is the **stationary** distribution of the Markov chain.
- The gradient is taken with respect to $\mathbf{x}$, not $\boldsymbol{\theta}$.
- $\nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x})$ defines a vector field.

Fisher Divergence

$$D_F(p_{\text{data}}, p_{\boldsymbol{\theta}}) = \frac{1}{2}\mathbb{E}_{\pi} \left\| \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Recap of Previous Lecture

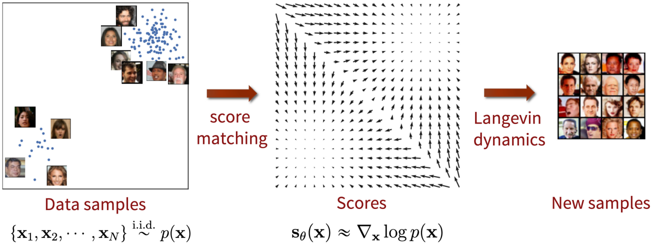
Define **score function** $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x})$.

Training (Score matching)

$$D_F(p_{\text{data}}, p_{\boldsymbol{\theta}}) = \frac{1}{2}\mathbb{E}_{\pi} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Sampling (Langevin Dynamics)

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$$



Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Recap of Previous Lecture

Let us perturb the original data with Gaussian noise $q(\mathbf{x}_\sigma|\mathbf{x}) = \mathcal{N}(\mathbf{x}, \sigma^2 \cdot \mathbf{I})$.

$$q(\mathbf{x}_\sigma) = \int q(\mathbf{x}_\sigma|\mathbf{x}) p_{\text{data}}(\mathbf{x}) d\mathbf{x}.$$

Then the solution of

$$\frac{1}{2}\mathbb{E}_{q(\mathbf{x}_\sigma)} \big\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma) \big\|_2^2 \rightarrow \min_{\boldsymbol{\theta}}$$

satisfies $\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \approx \mathbf{s}_{\boldsymbol{\theta},0}(\mathbf{x}_0) = \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$ if σ is sufficiently small.

Theorem (Denoising Score Matching)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \big\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma) \big\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \big\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x}) \big\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Here, $\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x}) = -\frac{\mathbf{x}_\sigma - \mathbf{x}}{\sigma^2} = -\frac{\boldsymbol{\epsilon}}{\sigma}$. $\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)$ attempts to **denoise** a corrupted sample.

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Outline

1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network
2. Forward Gaussian Diffusion Process
3. Denoising Score Matching for Diffusion
4. Reverse Gaussian Diffusion Process

Outline

1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network

2. Forward Gaussian Diffusion Process

3. Denoising Score Matching for Diffusion

4. Reverse Gaussian Diffusion Process

Outline

1. Score Matching
   - Denoising Score Matching (continued)
   - Noise-Conditioned Score Network
2. Forward Gaussian Diffusion Process
3. Denoising Score Matching for Diffusion
4. Reverse Gaussian Diffusion Process

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \int q(\mathbf{x}_\sigma) h(\mathbf{x}_\sigma) d\mathbf{x}_\sigma$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \int q(\mathbf{x}_\sigma) h(\mathbf{x}_\sigma) d\mathbf{x}_\sigma =$$

$$= \int \left(\int q(\mathbf{x}_\sigma|\mathbf{x}) p_{\text{data}}(\mathbf{x}) d\mathbf{x} \right) h(\mathbf{x}_\sigma) d\mathbf{x}_\sigma = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \int q(\mathbf{x}_\sigma) h(\mathbf{x}_\sigma) d\mathbf{x}_\sigma =$$

$$= \int \left(\int q(\mathbf{x}_\sigma|\mathbf{x}) p_{\text{data}}(\mathbf{x}) d\mathbf{x} \right) h(\mathbf{x}_\sigma) d\mathbf{x}_\sigma = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$

$$= \mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\|^2 + \underbrace{\|\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{\text{const}(\boldsymbol{\theta})} - 2\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma) \right]$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \int q(\mathbf{x}_\sigma) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\frac{\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma)}{q(\mathbf{x}_\sigma)}\right] d\mathbf{x}_\sigma$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \int q(\mathbf{x}_\sigma) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\frac{\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma)}{q(\mathbf{x}_\sigma)}\right] d\mathbf{x}_\sigma =$$
$$= \int \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \left(\int q(\mathbf{x}_\sigma|\mathbf{x})p_{\text{data}}(\mathbf{x})d\mathbf{x}\right)\right] d\mathbf{x}_\sigma$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \int q(\mathbf{x}_\sigma) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\frac{\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma)}{q(\mathbf{x}_\sigma)}\right] d\mathbf{x}_\sigma =$$
$$= \int \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \left(\int q(\mathbf{x}_\sigma|\mathbf{x})p_{\text{data}}(\mathbf{x})d\mathbf{x}\right)\right] d\mathbf{x}_\sigma =$$
$$= \int\int p_{\text{data}}(\mathbf{x}) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma|\mathbf{x})\right] d\mathbf{x}_\sigma d\mathbf{x}$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \int q(\mathbf{x}_\sigma) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \frac{\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma)}{q(\mathbf{x}_\sigma)}\right] d\mathbf{x}_\sigma =$$
$$= \int \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \left(\int q(\mathbf{x}_\sigma|\mathbf{x}) p_{\text{data}}(\mathbf{x}) d\mathbf{x}\right)\right] d\mathbf{x}_\sigma =$$
$$= \int \int p_{\text{data}}(\mathbf{x}) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma|\mathbf{x})\right] d\mathbf{x}_\sigma d\mathbf{x} =$$
$$= \int \int p_{\text{data}}(\mathbf{x}) q(\mathbf{x}_\sigma|\mathbf{x}) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right] d\mathbf{x}_\sigma d\mathbf{x}$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \int q(\mathbf{x}_\sigma) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\frac{\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma)}{q(\mathbf{x}_\sigma)}\right] d\mathbf{x}_\sigma =$$
$$= \int \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \left(\int q(\mathbf{x}_\sigma|\mathbf{x})p_{\text{data}}(\mathbf{x})d\mathbf{x}\right)\right] d\mathbf{x}_\sigma =$$
$$= \int\int p_{\text{data}}(\mathbf{x}) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} q(\mathbf{x}_\sigma|\mathbf{x})\right] d\mathbf{x}_\sigma d\mathbf{x} =$$
$$= \int\int p_{\text{data}}(\mathbf{x})q(\mathbf{x}_\sigma|\mathbf{x}) \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right] d\mathbf{x}_\sigma d\mathbf{x} =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right]$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right]$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right]$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\|^2 - 2\mathbf{s}^T_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right] + \text{const}(\boldsymbol{\theta})$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Theorem

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2}_{h(\mathbf{x}_\sigma)} =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (Continued)

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} h(\mathbf{x}_\sigma) = \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} h(\mathbf{x}_\sigma)$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left[\mathbf{s}_{\boldsymbol{\theta},\sigma}^T(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\right] = \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\mathbf{s}_{\boldsymbol{\theta},\sigma}^T(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right]$$

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma)\|_2^2 =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left[\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma)\|^2 - 2\mathbf{s}_{\boldsymbol{\theta},\sigma}^T(\mathbf{x}_\sigma) \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\right] + \text{const}(\boldsymbol{\theta}) =$$

$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising Score Matching

Original objective:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Denoising Score Matching

Original objective:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Noisy objective:

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}} \log q(\mathbf{x}_\sigma)\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Denoising Score Matching

Original objective:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Noisy objective:

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}} \log q(\mathbf{x}_\sigma)\|_2^2 \to \min_{\boldsymbol{\theta}}$$

This is equivalent to a denoising task:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Denoising Score Matching

Original objective:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Noisy objective:

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}} \log q(\mathbf{x}_\sigma)\|_2^2 \to \min_{\boldsymbol{\theta}}$$

This is equivalent to a denoising task:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\mathcal{N}(0,\mathbf{I})} \left\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x} + \sigma\boldsymbol{\epsilon}) + \frac{\boldsymbol{\epsilon}}{\sigma}\right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Denoising Score Matching

Original objective:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Noisy objective:

$$\mathbb{E}_{q(\mathbf{x}_\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}} \log q(\mathbf{x}_\sigma) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

This is equivalent to a denoising task:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_\sigma|\mathbf{x})} \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_\sigma) - \nabla_{\mathbf{x}_\sigma} \log q(\mathbf{x}_\sigma|\mathbf{x}) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{\mathcal{N}(0,\mathbf{I})} \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x} + \sigma\boldsymbol{\epsilon}) + \frac{\boldsymbol{\epsilon}}{\sigma} \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Langevin Dynamics

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_l) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$$

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Outline

1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network

2. Forward Gaussian Diffusion Process

3. Denoising Score Matching for Diffusion

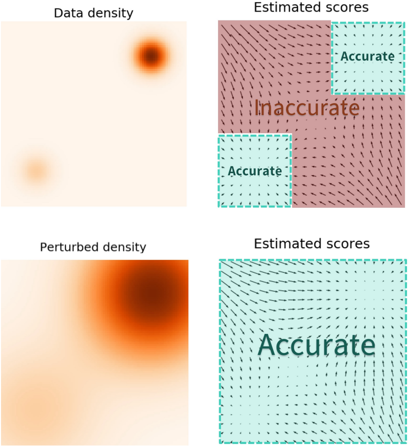
4. Reverse Gaussian Diffusion Process

Denoising Score Matching

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\mathcal{N}(0,\mathbf{I})}\left\|\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}+\sigma\boldsymbol{\epsilon})+\frac{\boldsymbol{\epsilon}}{\sigma}\right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2}\cdot\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x}_l) + \sqrt{\eta}\cdot\boldsymbol{\epsilon}_l$$

- For **small** σ, $\mathbf{s}_{\boldsymbol{\theta},\sigma}(\mathbf{x})$ becomes inaccurate and Langevin dynamics fails to traverse modes
- For **large** σ, robustness in low-density regions is achieved, but the model learns a distribution that is overly corrupted



Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Noise-Conditioned Score Network (NCSN)

- Specify a sequence of noise levels: $\sigma_1 < \sigma_2 < \cdots < \sigma_T$
- Perturb each data point with different noise levels: $\mathbf{x}_t = \mathbf{x} + \sigma_t \epsilon$, so $\mathbf{x}_t \sim q(\mathbf{x}_t)$
- Choose σ_1, σ_T such that:

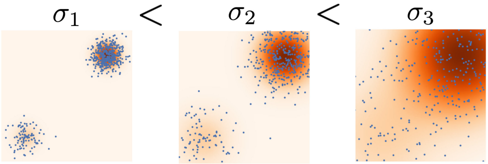
$$q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2 \mathbf{I})$$

Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

- Specify a sequence of noise levels: $\sigma_1 < \sigma_2 < \cdots < \sigma_T$
- Perturb each data point with different noise levels: $\mathbf{x}_t = \mathbf{x} + \sigma_t \epsilon$, so $\mathbf{x}_t \sim q(\mathbf{x}_t)$
- Choose σ_1, σ_T such that:

$$q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2 \mathbf{I})$$



Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

Train the denoising score function $\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t)$ for each noise level σ_t using a unified weighted objective:

$$\sum_{t=1}^{T} \sigma_t^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

Train the denoising score function $\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t)$ for each noise level σ_t using a unified weighted objective:

$$\sum_{t=1}^{T} \sigma_t^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Here, $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2} = -\frac{\boldsymbol{\epsilon}}{\sigma_t}$

Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

Train the denoising score function $\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t)$ for each noise level σ_t using a unified weighted objective:

$$\sum_{t=1}^{T} \sigma_t^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Here, $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2} = -\frac{\boldsymbol{\epsilon}}{\sigma_t}$

Training

1. Sample $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x})$
2. Sample $t \sim U\{1, T\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$
3. Construct noisy image $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}$
4. Evaluate loss $\mathcal{L} = \sigma_t^2 \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) + \frac{\boldsymbol{\epsilon}}{\sigma_t} \right\|^2$

Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

Train the denoising score function $\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t)$ for each noise level σ_t using a unified weighted objective:

$$\sum_{t=1}^{T} \sigma_t^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x})\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Here, $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2} = -\frac{\boldsymbol{\epsilon}}{\sigma_t}$

Training

1. Sample $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x})$
2. Sample $t \sim U\{1, T\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$
3. Construct noisy image $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}$
4. Evaluate loss $\mathcal{L} = \sigma_t^2 \left\| \mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t) + \frac{\boldsymbol{\epsilon}}{\sigma_t} \right\|^2$

How do we sample from such a model?

Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise-Conditioned Score Network (NCSN)

Sampling (Annealed Langevin Dynamics)

- Sample initial point $\mathbf{x}_0 \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I}) \approx q(\mathbf{x}_T)$
- At each noise level, apply L steps of Langevin dynamics:

$$\mathbf{x}_l = \mathbf{x}_{l-1} + \frac{\eta_t}{2} \mathbf{s}_{\boldsymbol{\theta}, \sigma_t}(\mathbf{x}_{l-1}) + \sqrt{\eta_t} \boldsymbol{\epsilon}_l,$$

- Update $\mathbf{x}_0 := \mathbf{x}_L$ and reduce to the next lower σ_t

Song Y. et al. Improved Techniques for Training Score-Based Generative Models, 2020

Noise-Conditioned Score Network (NCSN)

Sampling (Annealed Langevin Dynamics)

- Sample initial point $\mathbf{x}_0 \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I}) \approx q(\mathbf{x}_T)$
- At each noise level, apply L steps of Langevin dynamics:

$$\mathbf{x}_l = \mathbf{x}_{l-1} + \frac{\eta_t}{2} \mathbf{s}_{\boldsymbol{\theta}, \sigma_t}(\mathbf{x}_{l-1}) + \sqrt{\eta_t} \boldsymbol{\epsilon}_l,$$

- Update $\mathbf{x}_0 := \mathbf{x}_L$ and reduce to the next lower σ_t

Song Y. et al. Improved Techniques for Training Score-Based Generative Models, 2020

Outline

1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network

2. Forward Gaussian Diffusion Process

3. Denoising Score Matching for Diffusion

4. Reverse Gaussian Diffusion Process

Forward Gaussian Diffusion Process

Let $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $\beta_t \ll 1$. Define a Markov chain:

$$\mathbf{x}_t = \sqrt{1 - \beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

Let $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $\beta_t \ll 1$. Define a Markov chain:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

Let $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $\beta_t \ll 1$. Define a Markov chain:

$$\mathbf{x}_t = \sqrt{1 - \beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \;\; \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Langevin Dynamics

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \nabla_{\mathbf{x}_l} \log p_{\boldsymbol{\theta}}(\mathbf{x}_l) + \sqrt{\eta}\boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

Let $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $\beta_t \ll 1$. Define a Markov chain:

$$\mathbf{x}_t = \sqrt{1 - \beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Langevin Dynamics

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \nabla_{\mathbf{x}_l} \log p_{\boldsymbol{\theta}}(\mathbf{x}_l) + \sqrt{\eta}\boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$$

$$\mathbf{x}_t = \sqrt{1 - \beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t \approx \left(1 - \frac{\beta_t}{2}\right)\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t =$$

$$= \mathbf{x}_{t-1} + \frac{\beta_t}{2}(-\mathbf{x}_{t-1}) + \sqrt{\beta_t}\boldsymbol{\epsilon}_t$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

Let $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $\beta_t \ll 1$. Define a Markov chain:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Langevin Dynamics

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \frac{\eta}{2} \cdot \nabla_{\mathbf{x}_l} \log p_{\boldsymbol{\theta}}(\mathbf{x}_l) + \sqrt{\eta}\boldsymbol{\epsilon}_l, \quad \boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$$

$$\mathbf{x}_t = \sqrt{1-\beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t \approx \left(1 - \frac{\beta_t}{2}\right)\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t =$$

$$= \mathbf{x}_{t-1} + \frac{\beta_t}{2}(-\mathbf{x}_{t-1}) + \sqrt{\beta_t}\boldsymbol{\epsilon}_t$$

- $\beta_t = \eta$
- $\nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{x}_{t-1}|\boldsymbol{\theta}) = -\mathbf{x}_{t-1} = \nabla_{\mathbf{x}_{t-1}} \log \mathcal{N}(0, \mathbf{I})$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s = \prod_{s=1}^{t}(1-\beta_s)$. Then

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

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s = \prod_{s=1}^{t}(1-\beta_s)$. Then

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

Thus, samples at any timestep t can be generated directly from $\mathbf{x}_0$

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$

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

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s = \prod_{s=1}^t (1-\beta_s)$. Then

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

Thus, samples at any timestep t can be generated directly from $\mathbf{x}_0$

$$\begin{aligned}\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t = \\ &= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t\end{aligned}$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s = \prod_{s=1}^t (1-\beta_s)$. Then

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

Thus, samples at any timestep t can be generated directly from $\mathbf{x}_0$

$$\begin{aligned}\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t = \\ = \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t = \\ = \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + (\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t)\end{aligned}$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s = \prod_{s=1}^t (1-\beta_s)$. Then

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

Thus, samples at any timestep t can be generated directly from $\mathbf{x}_0$

$$\begin{aligned}
\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t =& \\
= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t =& \\
= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + (\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t) =& \\
= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}'_t&
\end{aligned}$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$$
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Statement 1

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s = \prod_{s=1}^t (1-\beta_s)$. Then

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

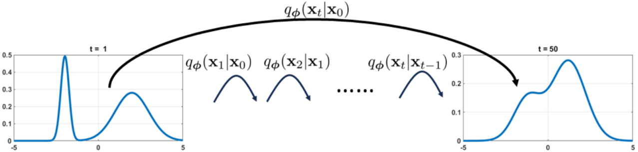
Thus, samples at any timestep t can be generated directly from $\mathbf{x}_0$

$$\begin{aligned}
\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t &= \\
&= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t = \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + (\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t) = \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}'_t = \\
&= \ldots = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})
\end{aligned}$$

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

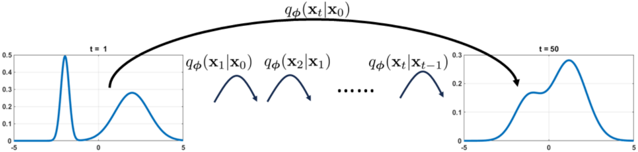
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right); \quad q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$



Chan S. Tutorial on Diffusion Models for Imaging and Vision, 2024
Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

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



Statement 2

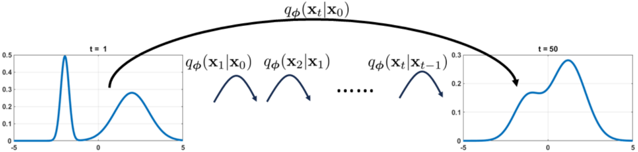
Applying the Markov chain to any distribution $p_{\text{data}}(\mathbf{x})$ yields $\mathbf{x}_\infty \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$, the **stationary** (limiting) distribution:

$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}|\mathbf{x}')p_\infty(\mathbf{x}')d\mathbf{x}'$$

Chan S. Tutorial on Diffusion Models for Imaging and Vision, 2024
Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

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



Statement 2

Applying the Markov chain to any distribution $p_{\text{data}}(\mathbf{x})$ yields $\mathbf{x}_\infty \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$, the **stationary** (limiting) distribution:

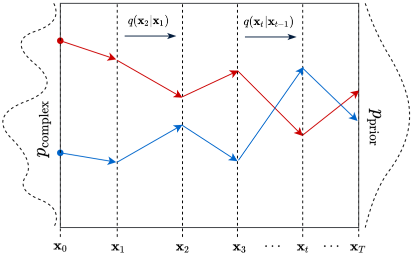
$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}|\mathbf{x}')p_\infty(\mathbf{x}')d\mathbf{x}'$$

$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}_\infty|\mathbf{x}_0)p_{\text{data}}(\mathbf{x}_0)d\mathbf{x}_0 \approx \mathcal{N}(0, \mathbf{I}) \int p_{\text{data}}(\mathbf{x}_0)d\mathbf{x}_0 = \mathcal{N}(0, \mathbf{I})$$

Chan S. Tutorial on Diffusion Models for Imaging and Vision, 2024
Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward Gaussian Diffusion Process

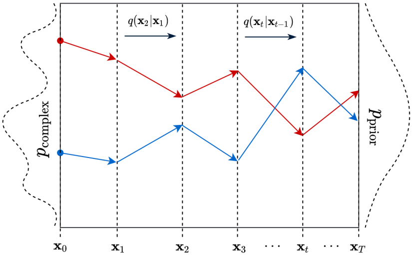
Diffusion describes the migration of particles from regions of high density to those of low density.



Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021

Forward Gaussian Diffusion Process

Diffusion describes the migration of particles from regions of high density to those of low density.

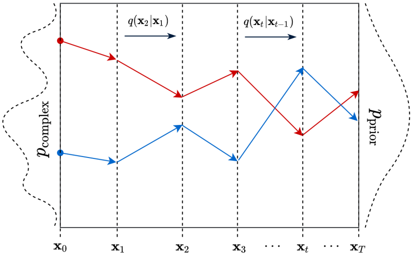


1. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
2. $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, $t \geq 1$
3. After $T \gg 1$ steps: $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$

Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021

Forward Gaussian Diffusion Process

Diffusion describes the migration of particles from regions of high density to those of low density.



1. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
2. $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon$, $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, $t \geq 1$
3. After $T \gg 1$ steps: $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$

If this process can be reversed, we can sample from $p_{\text{data}}(\mathbf{x})$ by starting from noise $p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$.
Our goal now becomes inverting this diffusion.

Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021

Outline

1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network
2. Forward Gaussian Diffusion Process
3. Denoising Score Matching for Diffusion
4. Reverse Gaussian Diffusion Process

Denoising Score Matching

NCSN

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2\mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2\mathbf{I})$$

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2}$$

Denoising Score Matching

NCSN

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2 \mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2 \mathbf{I})$$

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2}$$

Gaussian Diffusion

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \mathbf{I})$$

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

Denoising Score Matching

NCSN

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2\mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2\mathbf{I})$$

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2}$$

Gaussian Diffusion

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \mathbf{I})$$

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

Theorem (Denoising Score Matching)

$$\mathbb{E}_{q(\mathbf{x}_t)} \|\mathbf{s}_{\boldsymbol{\theta},t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Denoising Score Matching

NCSN

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2 \mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \sigma_T^2 \mathbf{I})$$

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t^2}$$

Gaussian Diffusion

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \quad q(\mathbf{x}_1) \approx p_{\text{data}}(\mathbf{x}), \quad q(\mathbf{x}_T) \approx \mathcal{N}(0, \mathbf{I})$$

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

Theorem (Denoising Score Matching)

$$\mathbb{E}_{q(\mathbf{x}_t)} \|\mathbf{s}_{\boldsymbol{\theta},t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)\|_2^2 =$$
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta},t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x})\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Note: This enables applying the NCSN approach with annealed Langevin dynamics to diffusion-based denoising models.

Outline

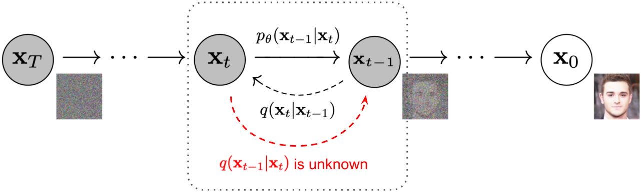
1. Score Matching
   Denoising Score Matching (continued)
   Noise-Conditioned Score Network

2. Forward Gaussian Diffusion Process

3. Denoising Score Matching for Diffusion

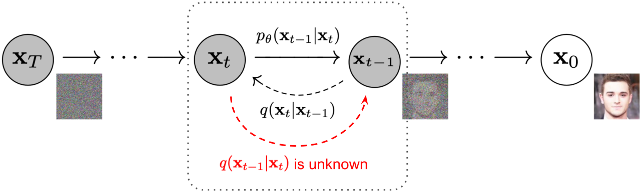
4. Reverse Gaussian Diffusion Process

Reverse Gaussian Diffusion Process



Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process

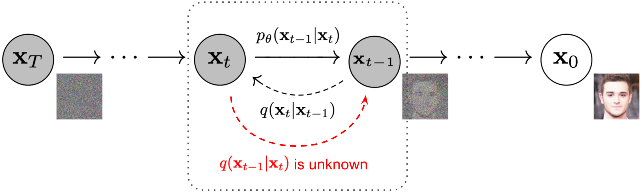


Forward Process

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

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process



Forward Process

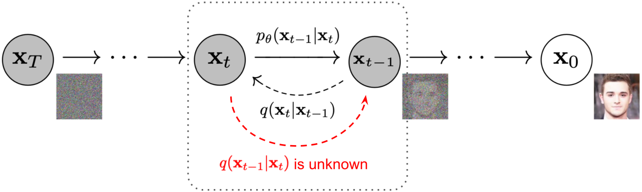
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right)$$

Reverse Process

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)} \approx p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process



Forward Process

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

Reverse Process

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)} \approx p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

$q(\mathbf{x}_{t-1})$ and $q(\mathbf{x}_t)$ are intractable:

$$q(\mathbf{x}_t) = \int q(\mathbf{x}_t|\mathbf{x}_0) p_{\text{data}}(\mathbf{x}_0) d\mathbf{x}_0$$

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)}$$

Theorem (Feller, 1949)

If β_t is sufficiently small, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is Gaussian (thus, diffusion requires $T \approx 1000$ steps for convergence)

Feller W. On the theory of stochastic processes, with particular reference to applications, 1949

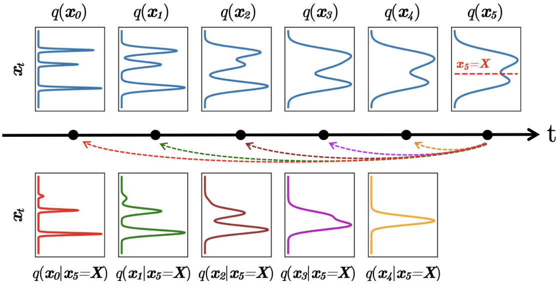
Xiao Z., Kreis K., Vahdat A. Tackling the generative learning trilemma with denoising diffusion GANs, 2021

Reverse Gaussian Diffusion Process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)}$$

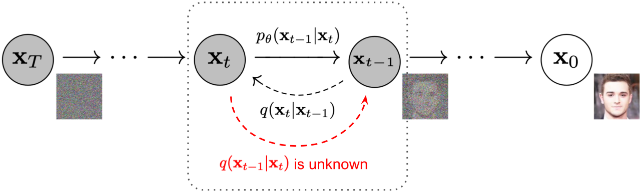
Theorem (Feller, 1949)

If β_t is sufficiently small, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is Gaussian (thus, diffusion requires $T \approx 1000$ steps for convergence)



Feller W. On the theory of stochastic processes, with particular reference to applications, 1949
Xiao Z., Kreis K., Vahdat A. Tackling the generative learning trilemma with denoising diffusion GANs, 2021

Reverse Gaussian Diffusion Process (Ancestral Sampling)



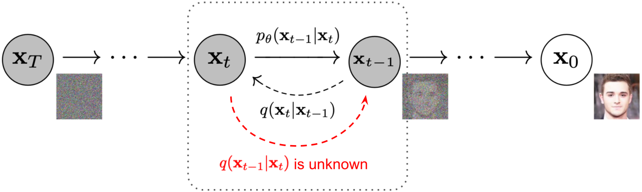
Define the reverse process as:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \approx p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta},t}(\mathbf{x}_t), \boldsymbol{\sigma}^2_{\boldsymbol{\theta},t}(\mathbf{x}_t)\right)$$

Feller's theorem justifies this Gaussian assumption.

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process (Ancestral Sampling)



Define the reverse process as:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \approx p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\boldsymbol{\mu}_{\theta,t}(\mathbf{x}_t), \boldsymbol{\sigma}^2_{\theta,t}(\mathbf{x}_t)\right)$$

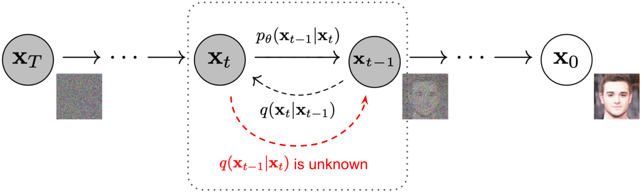
Feller's theorem justifies this Gaussian assumption.

Forward Process

1. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
2. $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}$
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process (Ancestral Sampling)



Define the reverse process as:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \approx p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta},t}(\mathbf{x}_t), \boldsymbol{\sigma}^2_{\boldsymbol{\theta},t}(\mathbf{x}_t)\right)$$

Feller's theorem justifies this Gaussian assumption.

Forward Process

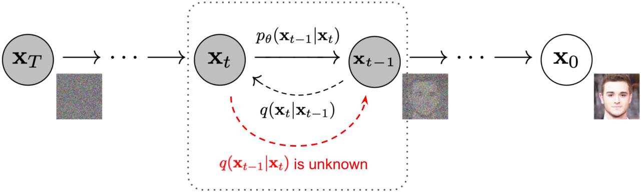
1. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
2. $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}$
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$

Reverse Process

1. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$
2. $\mathbf{x}_{t-1} = \boldsymbol{\sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t)\boldsymbol{\epsilon} + \boldsymbol{\mu}_{\boldsymbol{\theta},t}(\mathbf{x}_t)$
3. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$

Weng L. What are Diffusion Models?, blog post, 2021

Reverse Gaussian Diffusion Process (Ancestral Sampling)



Define the reverse process as:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \approx p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\boldsymbol{\mu}_{\theta,t}(\mathbf{x}_t), \boldsymbol{\sigma}^2_{\theta,t}(\mathbf{x}_t)\right)$$

Feller's theorem justifies this Gaussian assumption.

Forward Process

1. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
2. $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}$
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$

Reverse Process

1. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$
2. $\mathbf{x}_{t-1} = \boldsymbol{\sigma}_{\theta,t}(\mathbf{x}_t)\boldsymbol{\epsilon} + \boldsymbol{\mu}_{\theta,t}(\mathbf{x}_t)$
3. $\mathbf{x}_0 = \mathbf{x} \sim p_{\text{data}}(\mathbf{x})$

Note: The forward process is non-learnable, i.e., it does not involve trainable parameters

Weng L. What are Diffusion Models?, blog post, 2021

Summary

- Denoising score matching minimizes the Fisher divergence on corrupted samples, making the divergence estimable via sampling
- The noise-conditioned score network leverages a range of noise levels and annealed Langevin dynamics to learn the score function and enable sampling
- The Gaussian diffusion process is a Markov chain that incrementally corrupts data with carefully structured Gaussian noise
- Denoising score matching, together with Langevin dynamics, can be applied to the Gaussian diffusion process
- The reverse process reconstructs data from noise samples, although its precise form is intractable