Denoising Diffusion Probabilistic Models

김현수

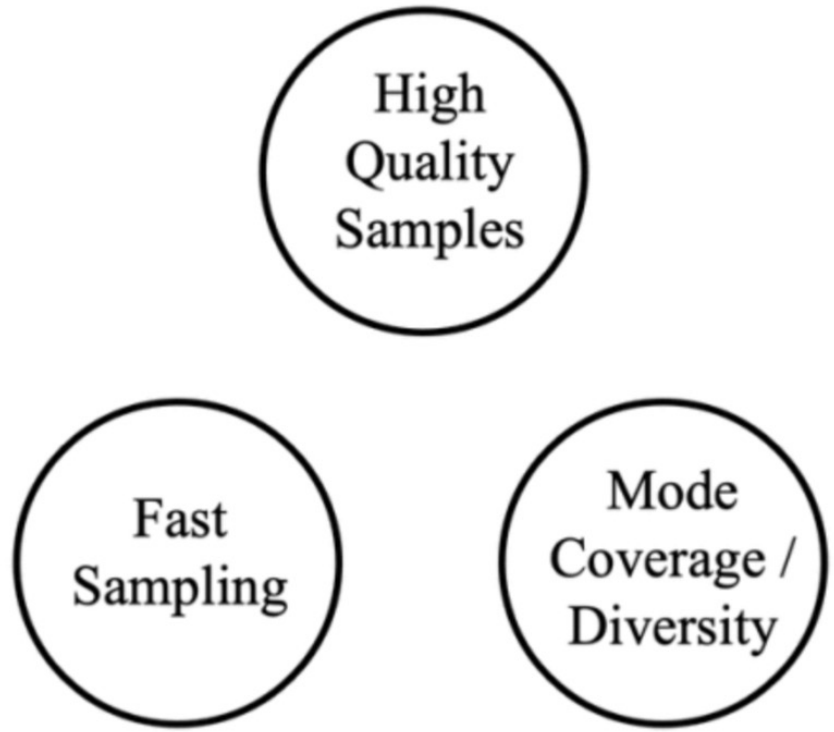
Denoising Diffusion Models

Emerging as powerful generative models, outperforming GANs

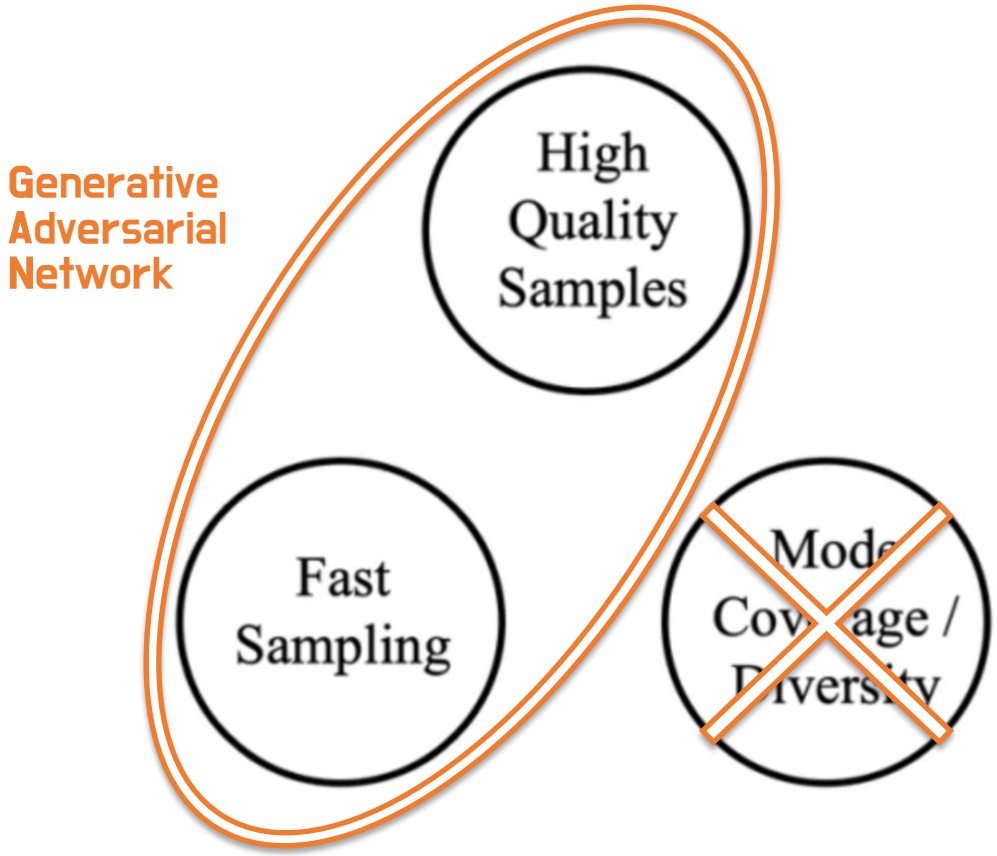
Diffusion Models Beat GANs on Image Synthesis, 2021

Cascaded Diffusion Models for High Fidelity Image Generation, 2021

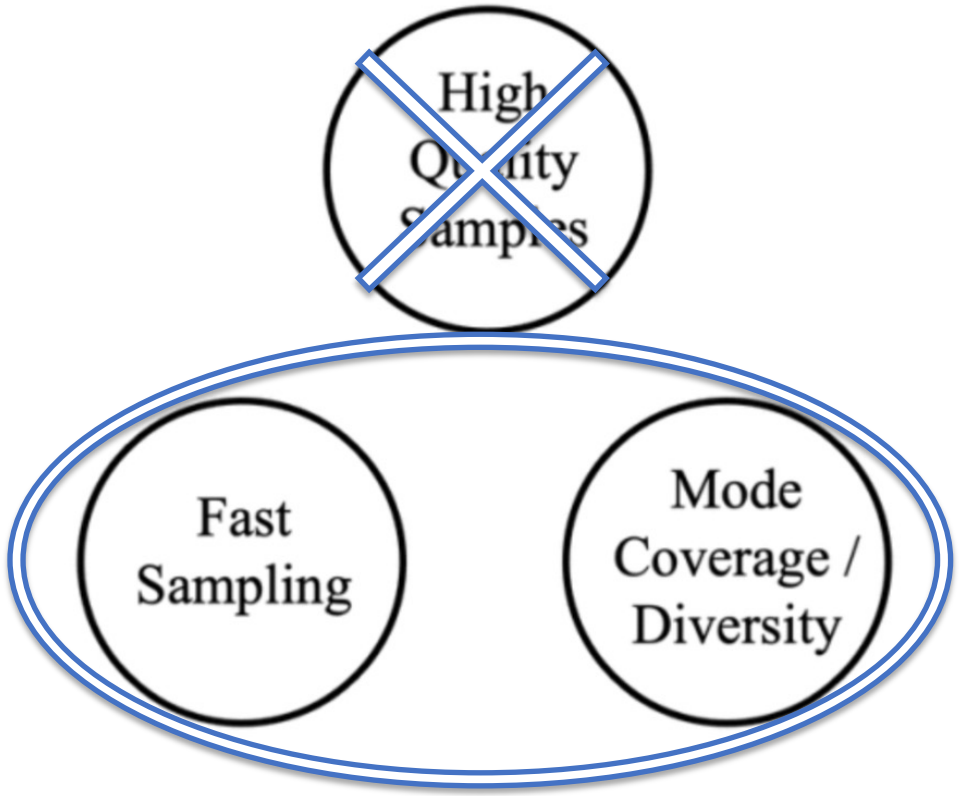
The Generative Learning Trilemma



The Generative Learning Trilemma

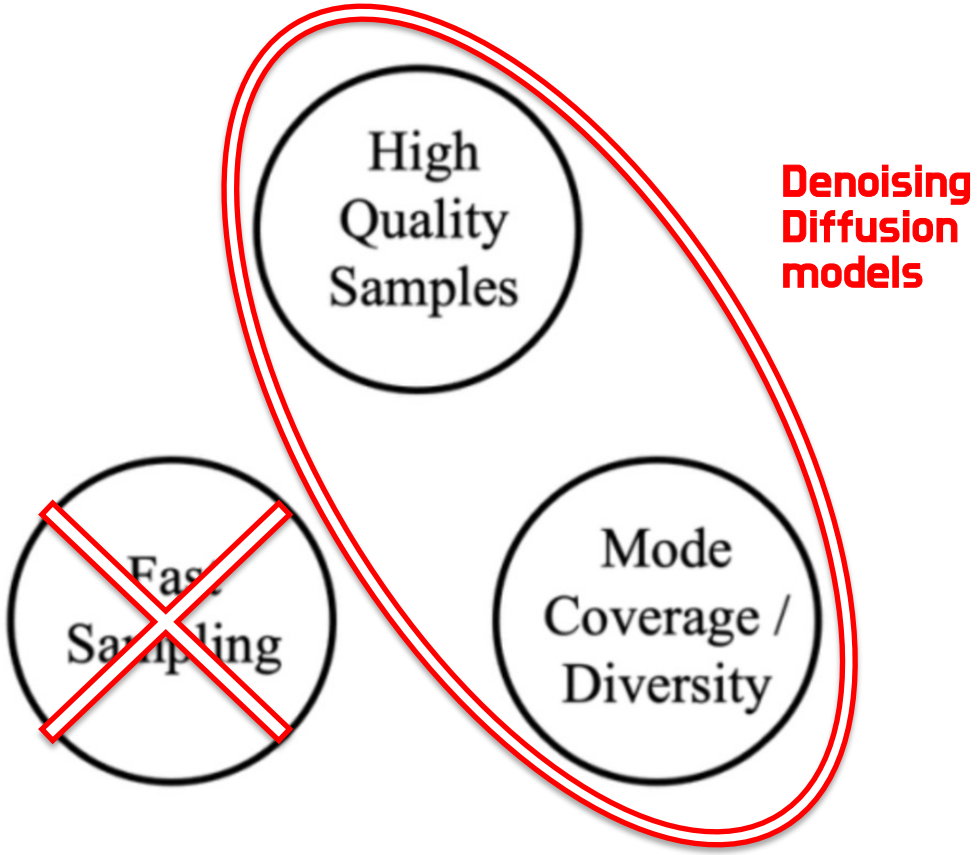


The Generative Learning Trilemma

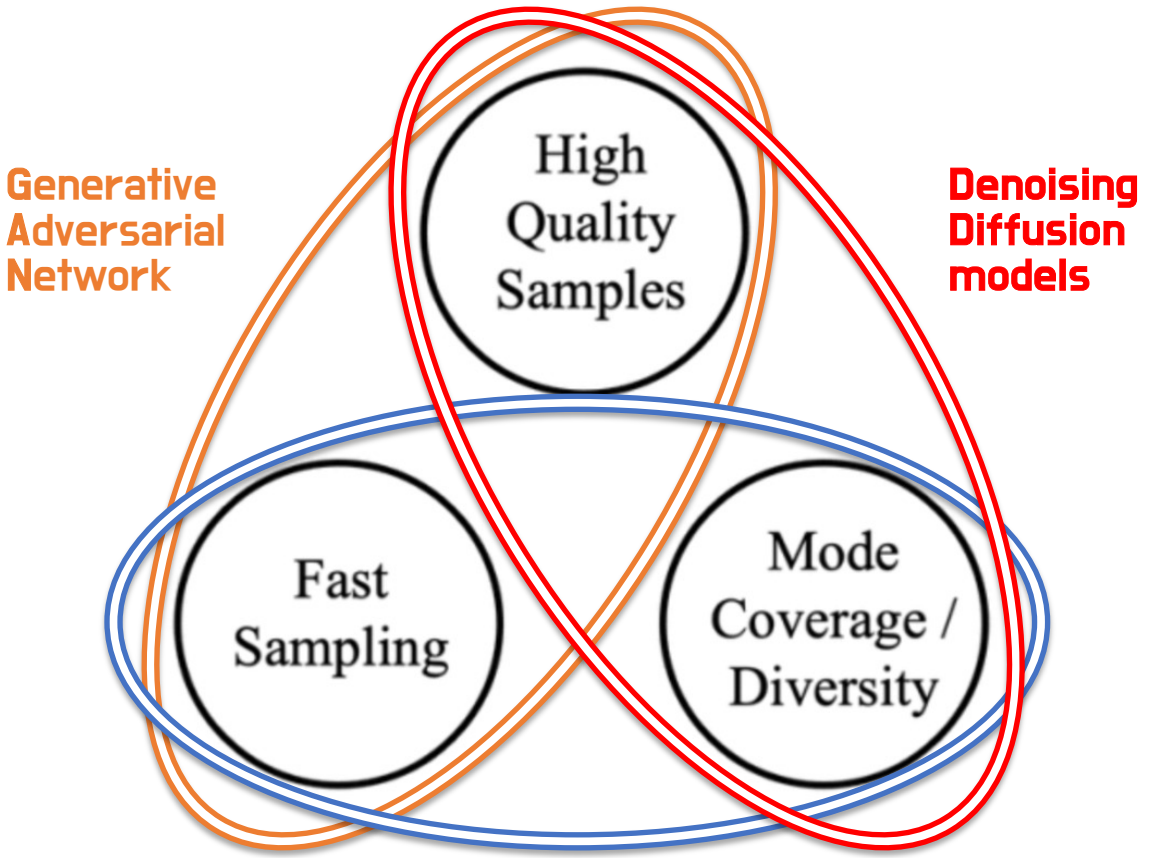


Variational Autoencoders,
Normalizing Flows

The Generative Learning Trilemma



The Generative Learning Trilemma

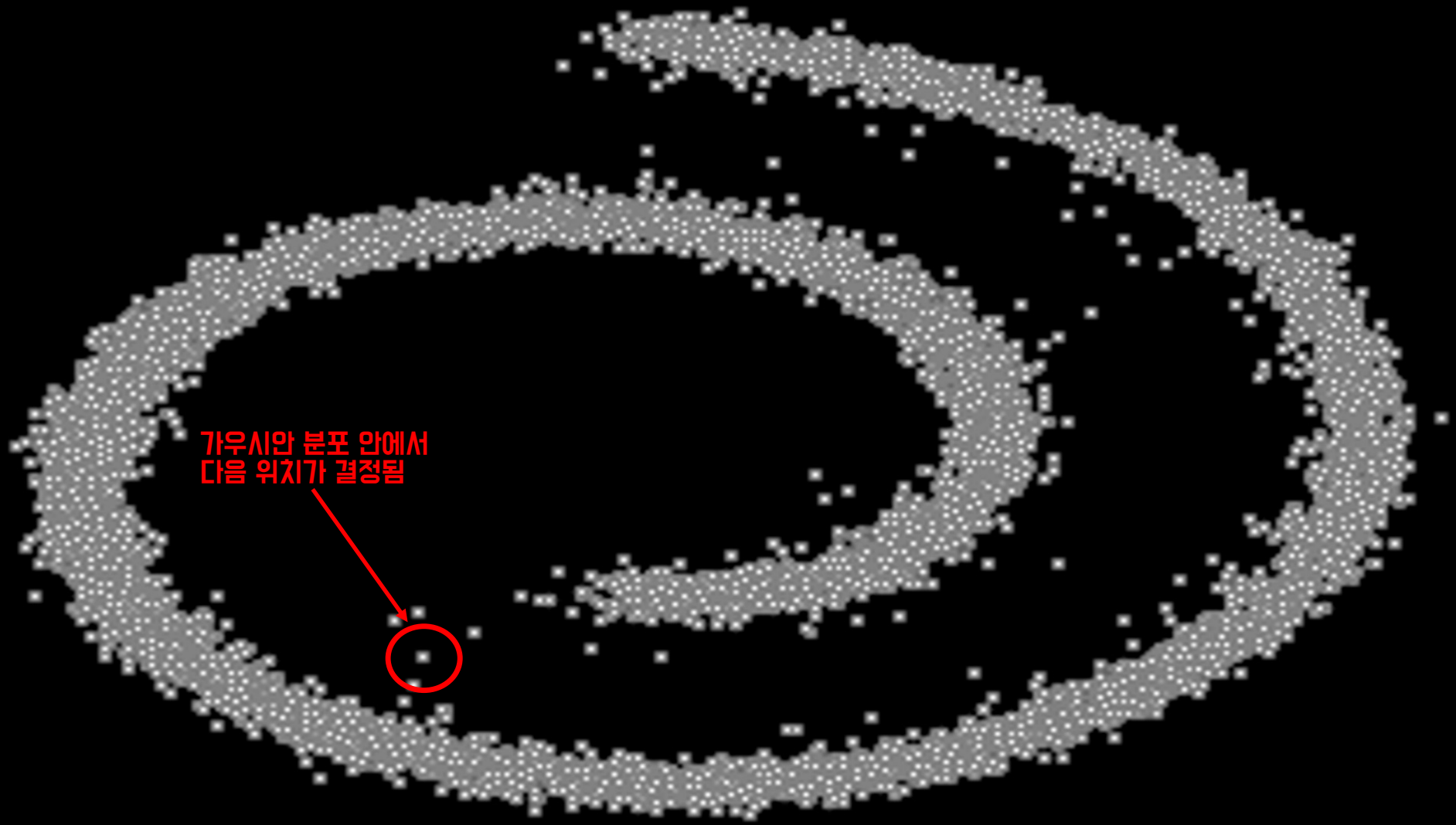


Denoising Diffusion Probabilistic Models

확산을 제거하는 확률적 모델

Denoising Diffusion Probabilistic Models

확산을 제거하는 확률적 모델

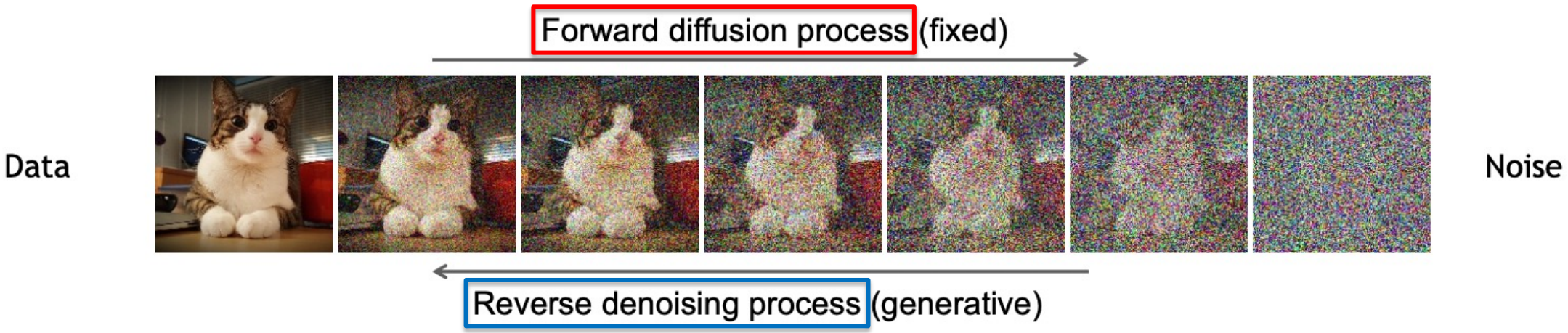

가우시안 분포 안에서
다음 위치가 결정됨

Denoising Diffusion Models

Learning to generate by denoising

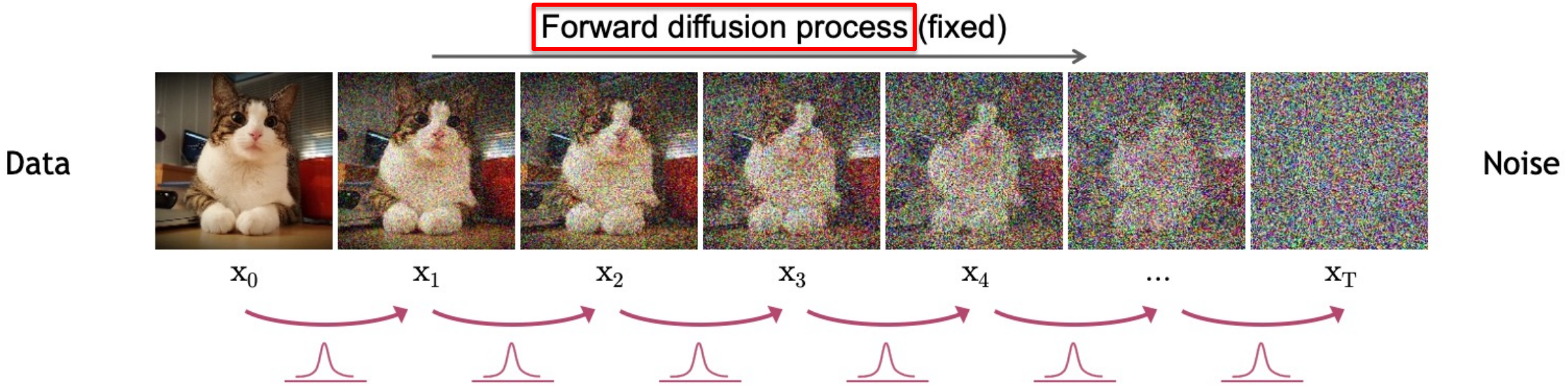
Denoising diffusion models consist of two processes:

- Forward diffusion process that gradually adds noise to input → 수학
- Reverse denoising process that learns to generate data by denoising → 학습(딥러닝)



Forward (Diffusion) Process

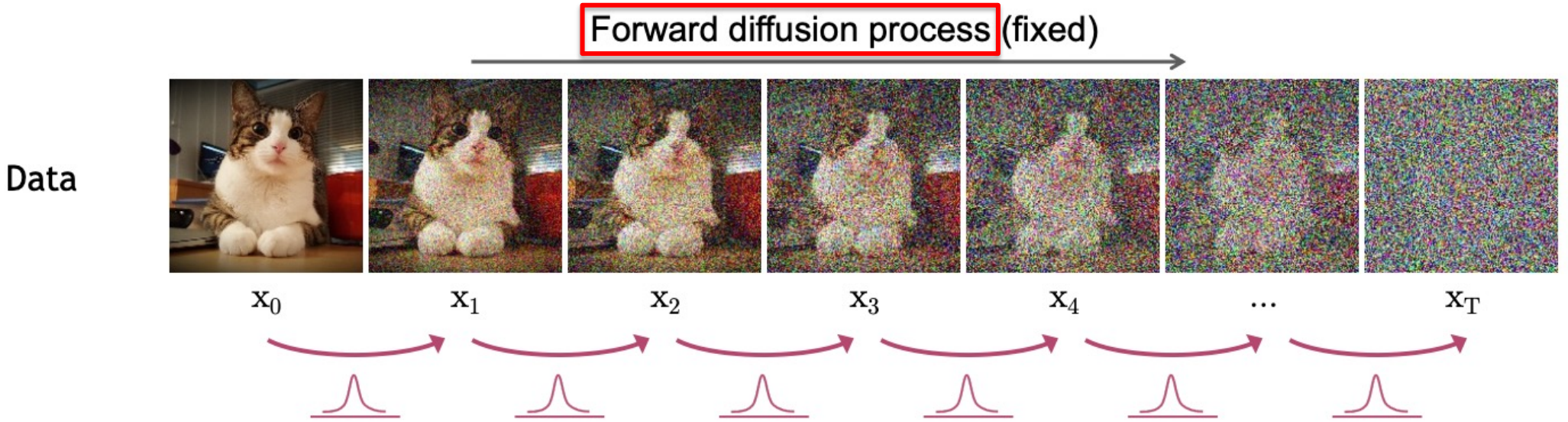
The formal definition of the forward process in T steps:



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad \text{(joint)}$$

Forward (Diffusion) Process

The formal definition of the forward process in T steps:



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad \text{(joint)}$$

$$q(X_t \mid X_{t-1}) = N(X_t; \sqrt{1-\beta_t}\, X_{t-1}, \beta_t I)$$

μ, σ는 고정시키고 ϵ만 정규분포에서 sampling → Why? → Stochastic node를 stochastic + deterministic한 부분으로 분해시켜 deterministic한 부분으로 backpropagation을 흐르게 하자.

$$= \sqrt{1-\beta_t}\, X_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1} \quad \text{(by reparametrization trick)}$$

$$(\epsilon \sim N(0, I))$$

$$\mathrm{Var}(q(X_t \mid X_{t-1})) = \mathrm{Var}\left(\sqrt{1-\beta_t}\, X_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1}\right)$$

$$\mathrm{Var}(aX+bY) = a^2 \mathrm{Var}(X) + b^2 \mathrm{Var}(Y) + 2ab\,\mathrm{Cov}(X,Y)$$

X, Y는 서로 독립이므로 $\mathrm{Cov}(X,Y) = 0$

$$= (1-\beta_t)\,\mathrm{Var}(X_{t-1}) + \beta_t\,\mathrm{Var}(\epsilon_{t-1})$$

$\mathrm{Var}(X_{t-1}) = 1$, $\mathrm{Var}(\epsilon_{t-1}) = 1$

$$= (1-\beta_t) + \beta_t = 1$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad \text{(joint)}$$

The formal definiti

* Markov Property

$$q(X_{t+1} \mid X_t) = q(X_{t+1} \mid X_1, X_2, \cdots, X_t)$$

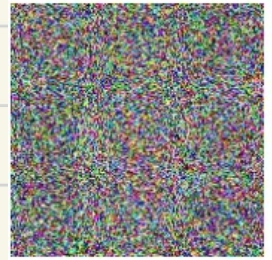
$$\prod_{t=1}^{T} q(X_t \mid X_{t-1}) = \frac{q(X_1, X_0)}{q(X_0)} \cdot \frac{q(X_2, X_1)}{q(X_1)} \cdot \cdots \cdot \frac{q(X_T, X_{T-1})}{q(X_{T-1})}$$

$$= \frac{q(X_1, X_0)}{q(X_0)} \cdot \frac{q(X_2, X_1, X_0)}{q(X_1, X_0)} \cdot \cdots \cdot \frac{q(X_T, X_{T-1}, \cdots, X_1, X_0)}{q(X_{T-1}, \cdots, X_0)}$$

$$= \frac{q(X_0, X_1, \cdots, X_{T-1}, X_T)}{q(X_0)}$$

$$= q(X_{1:T} \mid X_0)$$

Data



$\mathbf{x}_T$

Noise

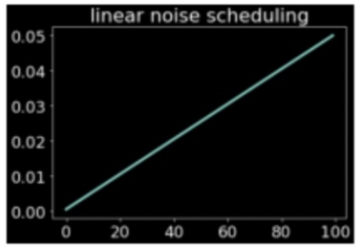
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

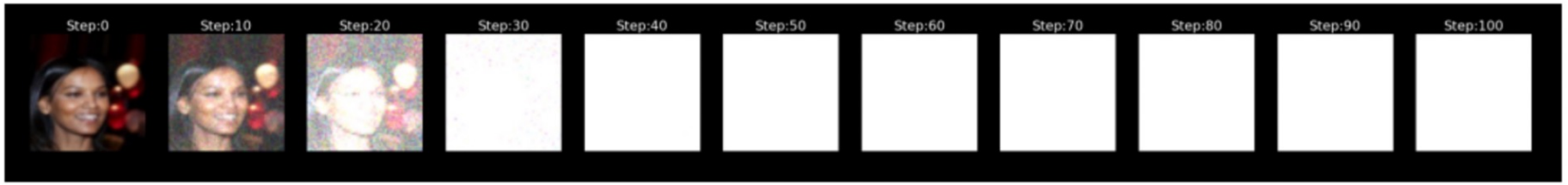
(joint)

$$q(X_{1:T} \mid X_0) = \frac{q(X_0, X_1, \cdots, X_T)}{q(X_0)} \quad ???$$

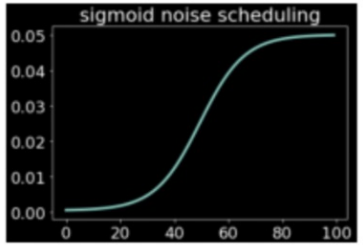
$$\prod_{t=1}^{T} q(X_t \mid X_{t-1}) = \frac{q(X_1, X_0)}{q(X_0)} \cdot \frac{q(X_2, X_1)}{q(X_1)} \cdot \cdots \cdot \frac{q(X_T, X_{T-1})}{q(X_{T-1})}$$

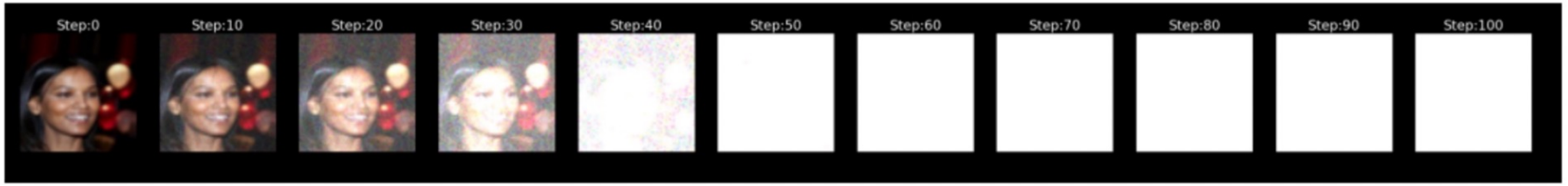
✓ Linear scheduling



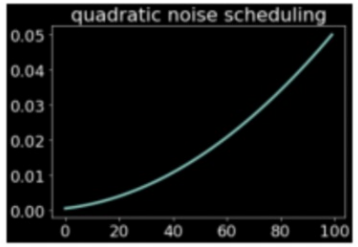


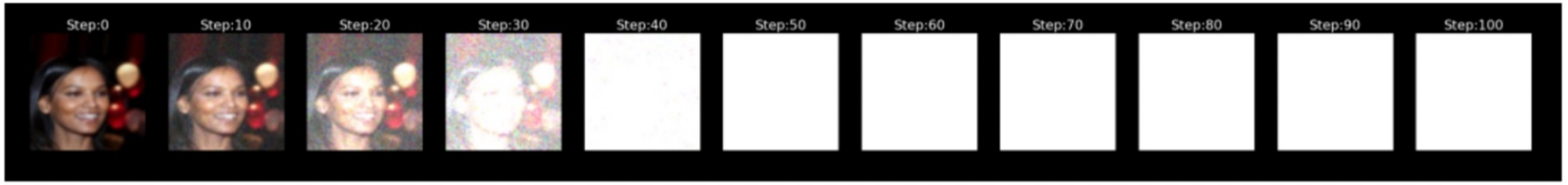
✓ Sigmoid scheduling





✓ Quadratic scheduling

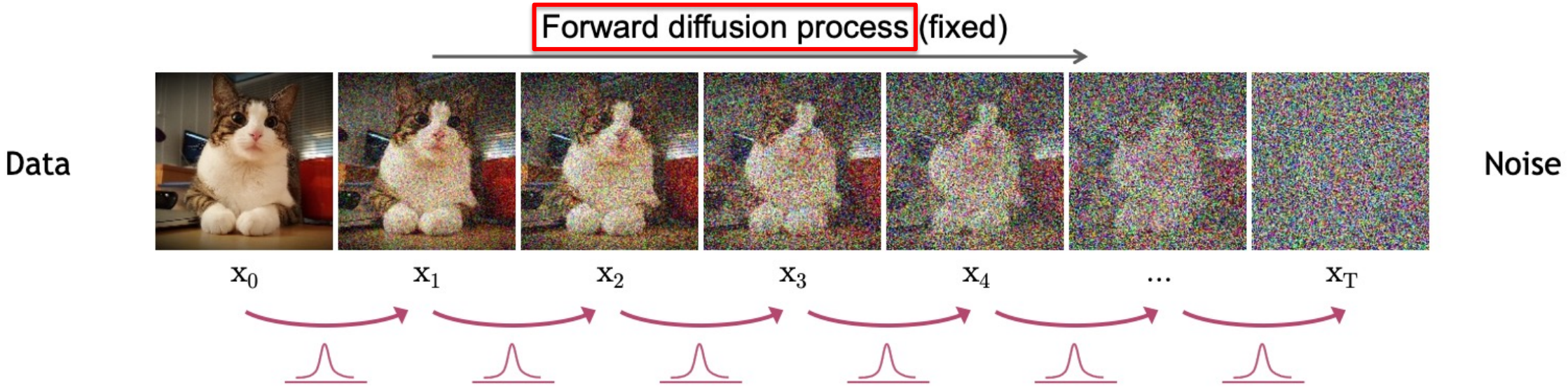




$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad \text{(joint)}$$

Forward (Diffusion) Process

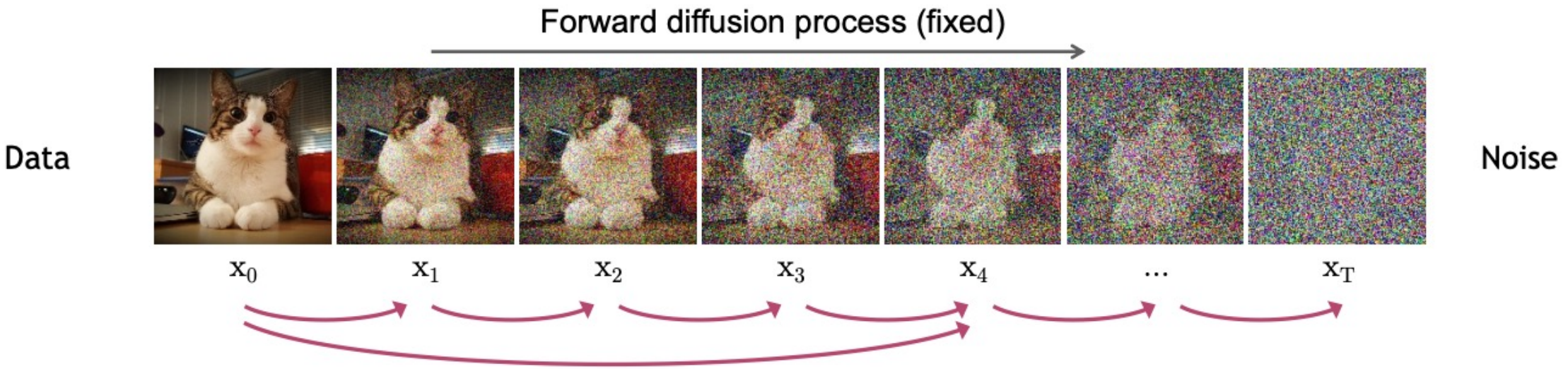
The formal definition of the forward process in T steps:



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad \text{(joint)}$$

→ 한 방에 정리한다면?

Forward (Diffusion) Process



Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \to 0$ and $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Let, $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1}$$

$$= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \epsilon_{t-1}$$

$$= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}}\, \epsilon_{t-2}\right) + \sqrt{1-\alpha_t}\, \epsilon_{t-1}$$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t}\cdot\sqrt{\alpha_t}\cdot\epsilon_{t-2} + \sqrt{1-\alpha_t}\,\epsilon_{t-1}$$

$$= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\,\epsilon$$

$$\cdots$$

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

* $\epsilon_{t-1}, \epsilon_{t-2}, \cdots \sim N(0, I)$
* Joint Distribution (two Gaussians)

$A \sim N(0, \sigma_1^2 I)$, $B \sim N(0, \sigma_2^2 I)$

$\longrightarrow A + B \sim N(0, (\sigma_1^2 + \sigma_2^2) I)$

$$\therefore q(x_t \mid x_0) = N(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}) I)$$

Define $\bar{\alpha}_t = \prod_{s=1}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

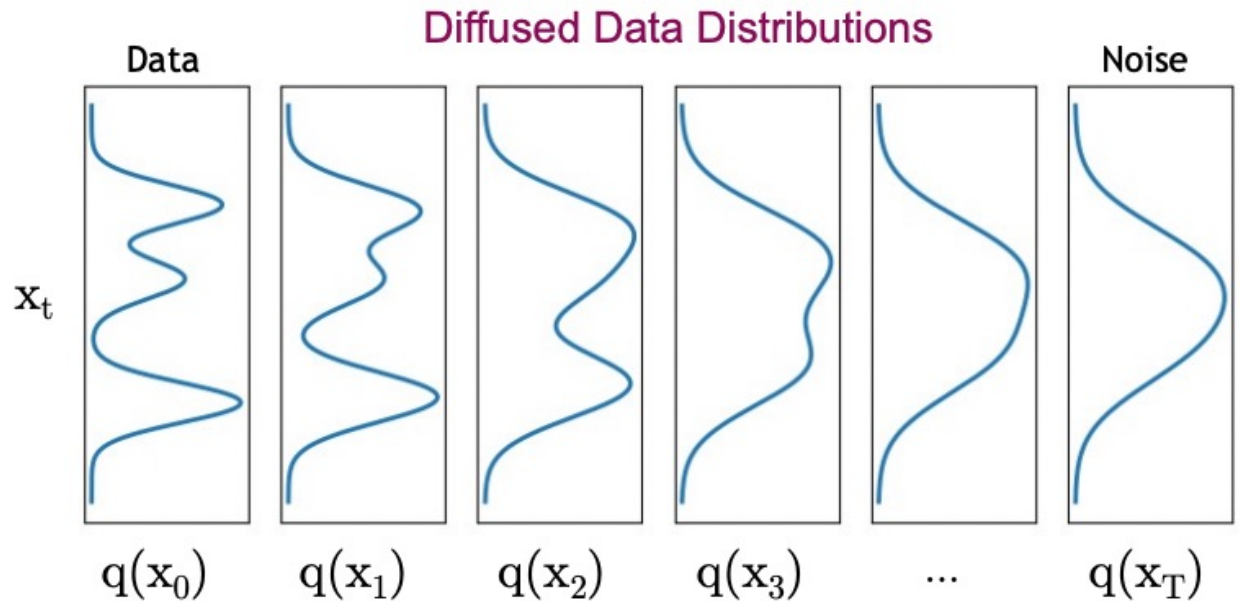
β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \to 0$ and $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

What happens to a distribution in the forward diffusion?

So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$?

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.



We can sample $\mathbf{x}_t \sim q(\mathbf{x}_t)$ by first sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and then sampling $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ (i.e., ancestral sampling).

What happens to a distribution in the forward diffusion?

DEFINITION

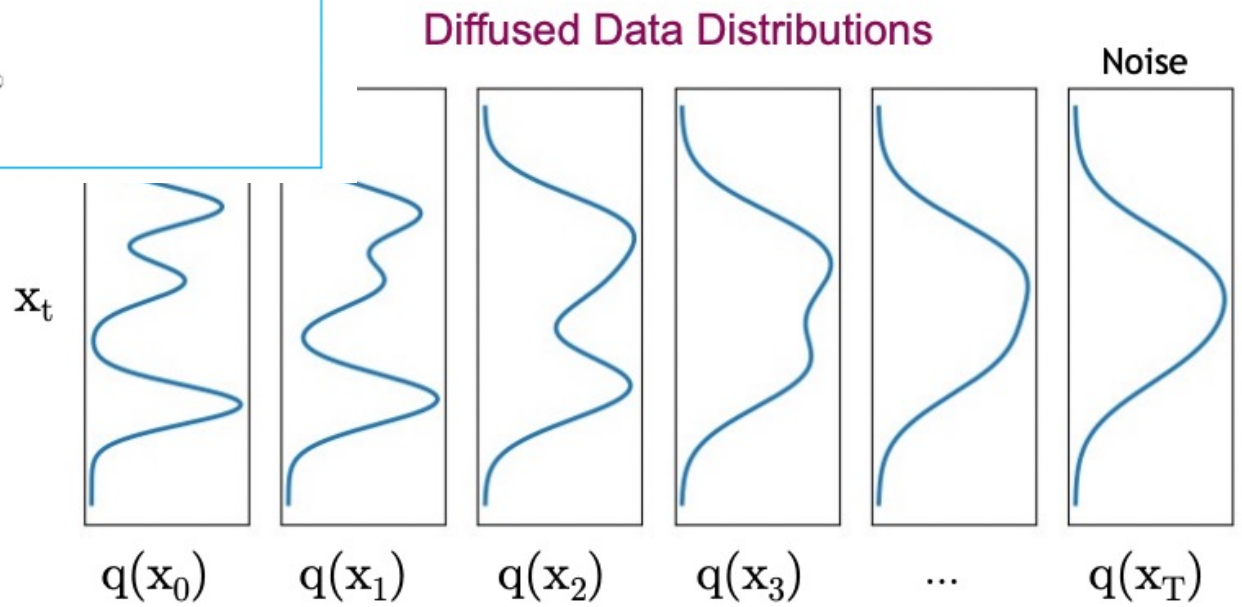
The **marginal probability density functions** of X and Y, denoted by $f_X(x)$ and $f_Y(y)$, respectively, are given by

$$f_X(x) = \int_{-\infty}^{\infty} f(x, y)\, dy \qquad \text{for } -\infty < x < \infty$$

$$f_Y(y) = \int_{-\infty}^{\infty} f(x, y)\, dx \qquad \text{for } -\infty < y < \infty$$

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}}\, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}}\, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}}\, d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.



We can sample $\mathbf{x}_t \sim q(\mathbf{x}_t)$ by first sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and then sampling $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ (i.e., ancestral sampling).

Generative Learning by Denoising

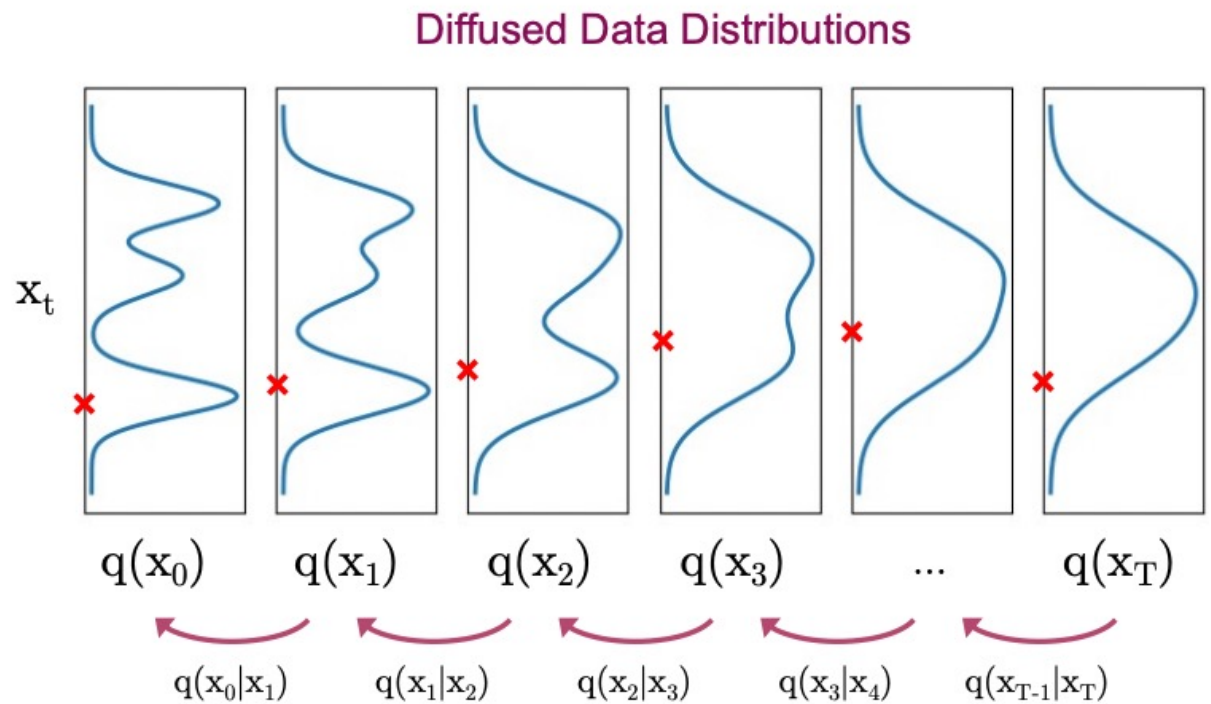
Recall, that the diffusion parameters are designed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}$

True Denoising Dist.



In general, $q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto q(\mathbf{x}_{t-1})q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is intractable.

Can we approximate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$? Yes, we can use a Normal distribution if β_t is small in each forward diffusion step.

Generative Learning by Denoising

Recall, that the diffusion parameters are designed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}$

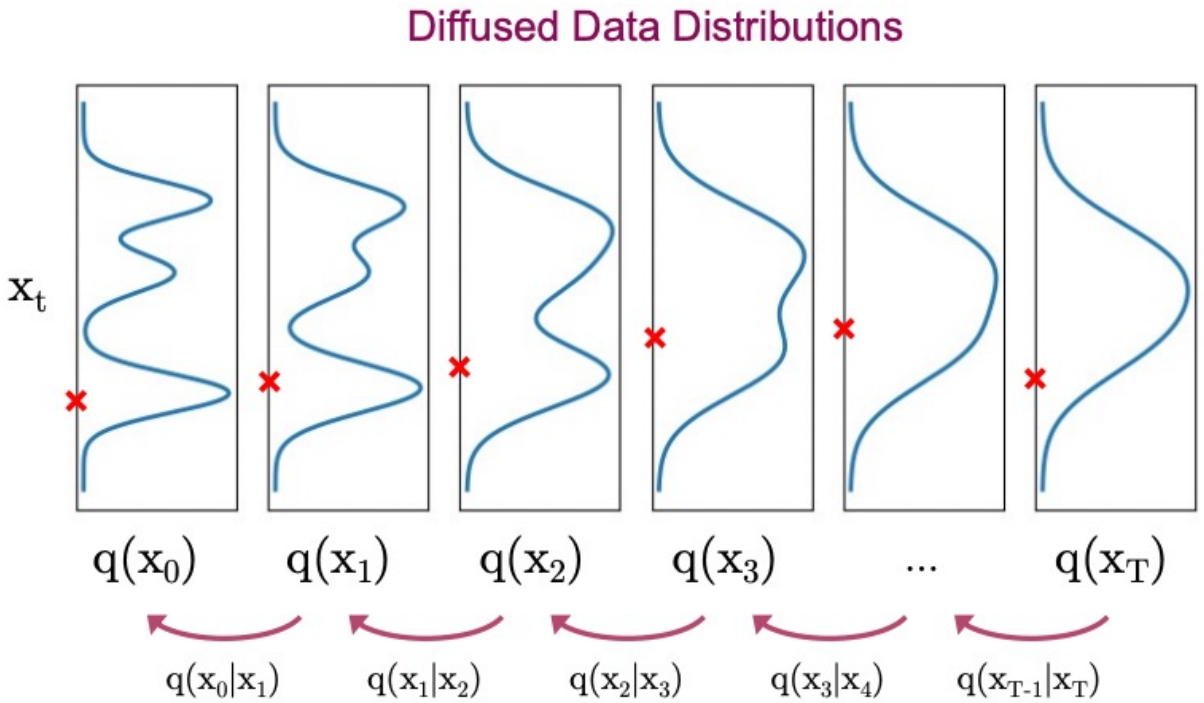
True Denoising Dist.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

(Bayes rule)

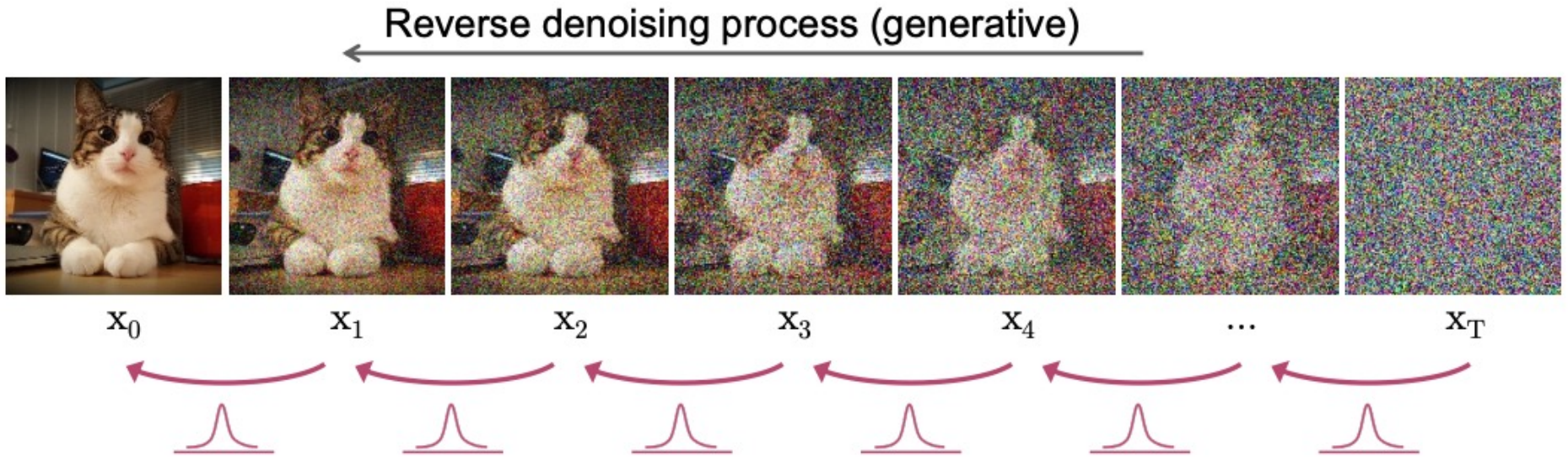
In general, $q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto q(\mathbf{x}_{t-1})q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is intractable.

Can we approximate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$? Yes, we can use a Normal distribution if β_t is small in each forward diffusion step.



Reverse Process

Formal definition of forward and reverse processes in T steps:



$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\mu_\theta(\mathbf{x}_t, t)}, \sigma_t^2\mathbf{I})$$

Trainable network
(U-net, Denoising Autoencoder)

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

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Sohl-Dickstein et al. ICML 2015 and Ho et al. NeurIPS 2020 show that:

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}\underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1))}_{L_0}\right]$$

where $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ is the tractable posterior distribution:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1};\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}),$$

$$\text{where } \tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \text{ and } \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

Learning Denoising Model

Variational upper bound

* Goal : VAE 학습 data likelihood $P_\theta(x) = P_{model}(x;\theta)$를 maximize

⇒ Negative Log Likelihood (NLL)을 사용 : $-\log P_\theta(x)$를 minimize

$$-\log P_\theta(x_0) \le -\log P_\theta(x_0) + D_{KL}(q(X_{1:T}|X_0) \| P_\theta(X_{1:T}|X_0))$$

$$= -\log P_\theta(x_0) + E_{X_{1:T}\sim q(X_{1:T}|X_0)}\left[\log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{1:T}|X_0)}\right]$$

$$= -\log P_\theta(x_0) + E_{X_{1:T}\sim q(X_{1:T}|X_0)}\left[\log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})/P_\theta(X_0)}\right]$$

$$= -\log P_\theta(x_0) + E_q\left[\log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})} + \log P_\theta(x_0)\right]$$

$$= E_q\left[\log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})}\right]$$

* KL Divergence

$$D_{KL}(P\|Q) = E_{x\sim P}\left[\log \frac{P(x)}{Q(x)}\right]$$

$$= E_{x\sim P}\left[-\log \frac{Q(x)}{P(x)}\right] \ge 0$$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}),$$

$$\text{where } \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \text{ and } \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

For training, we

lers:

[Sohl-Dickstein et

where $q(\mathbf{x}_{t-1}|\mathbf{x}_t$

$$= E_{q(x_{0:T})}\left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right]$$

* Reverse Process : $p_\theta(x_{0:T}) := p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$

Forward Process : $q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1})$

$$= E_q\left[\log \frac{\prod_{t=1}^{T} q(x_t|x_{t-1})}{p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=1}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_t|x_{t-1}, x_0)}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right] \quad \text{(by Markov property)}$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{1}{p_\theta(x_{t-1}|x_t)} \cdot \frac{q(x_{t-1}|x_t, x_0)\, q(x_t|x_0)}{q(x_{t-1}|x_0)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} + \sum_{t=2}^{T} \log \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} + \log\left(\frac{q(x_2|x_0)}{q(x_1|x_0)} \times \frac{q(x_3|x_0)}{q(x_2|x_0)} \times \cdots \times \frac{q(x_T|x_0)}{q(x_{T-1}|x_0)}\right) + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= E_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_T|x_0)}{q(x_1|x_0)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= E_q\left[\log \frac{q(x_T|x_0)}{p_\theta(x_0|x_1)\, p_\theta(x_T)} + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)}\right]$$

$$= E_q\left[\log \frac{q(x_T|x_0)}{p_\theta(x_T)} + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} - \log p_\theta(x_0|x_1)\right]$$

$$= E_q\left[D_{KL}(q(x_T|x_0)\,\|\,p_\theta(x_T)) + \sum_{t=2}^{T} D_{KL}(q(x_{t-1}|x_t, x_0)\,\|\,p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)\right]$$

L_T L_{t-1} L_0

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Sohl-Dickstein et al. ICML 2015 and Ho et al. NeurIPS 2020 show that:

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}\underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1))}_{L_0}\right]$$

where $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ is the tractable posterior distribution:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1};\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}),$$

$$\text{where } \tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \text{ and } \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

- 정규분포 pdf $f(x) = \frac{1}{\sqrt{2\pi}\cdot\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
- $q(X_t \mid X_0) = N(X_t ; \sqrt{\bar{\alpha}_t} X_0, (1-\bar{\alpha}_t) I)$
- $q(X_t \mid X_{t-1}) := N(X_t ; \sqrt{1-\beta_t} X_{t-1}, \beta_t I)$

$= q(X_t \mid X_{t-1}, X_0)$ (by Markov chain Rule)

$$q(X_{t-1} \mid X_t, X_0) = q(X_t \mid X_{t-1}, X_0) \cdot \frac{q(X_{t-1} \mid X_0)}{q(X_t \mid X_0)} \quad \text{(by Bayes rule)}$$

(Reverse Process)

$$\propto \exp\left(-\frac{1}{2}\left(\frac{(X_t - \sqrt{\alpha_t} X_{t-1})^2}{\beta_t} + \frac{(X_{t-1} - \sqrt{\bar{\alpha}_{t-1}} X_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(X_t - \sqrt{\bar{\alpha}_t} X_0)^2}{1-\bar{\alpha}_t}\right)\right)$$

$$= \exp\left(-\frac{1}{2}\left(\frac{X_t^2 - 2\sqrt{\alpha_t} X_t X_{t-1} + \alpha_t X_{t-1}^2}{\beta_t} + \frac{X_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}} X_0 X_{t-1} + \bar{\alpha}_{t-1} X_0^2}{1-\bar{\alpha}_{t-1}} - \frac{(X_t - \sqrt{\bar{\alpha}_t} X_0)^2}{1-\bar{\alpha}_t}\right)\right)$$

$$= \exp\left(-\frac{1}{2}\left(\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right) X_{t-1}^2 - \left(\frac{2\sqrt{\alpha_t}}{\beta_t} X_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}} X_0\right) X_{t-1} + C(X_t, X_0)\right)\right)$$

$\left(N \sim (\mu, \sigma^2)$의 pdf : $\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left(\frac{1}{\sigma^2} x^2 - \frac{2\mu}{\sigma^2} x + \frac{\mu^2}{\sigma^2}\right)\right)\right)$

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}),$$

$$\text{where } \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \text{ and } \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

$$q(x_{t-1}|x_t,x_0) = N(x_{t-1}; \tilde{\mu}(x_t,x_0), \tilde{\beta}_t I) \text{ 이다.}$$

① $$\tilde{\beta}_t = \frac{1}{\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)} = \frac{1}{\left(\frac{\alpha_t - \alpha_t\bar{\alpha}_{t-1} + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}\right)} = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t \quad \left(\because \alpha_t = 1-\beta_t,\ \bar{\alpha}_t = \prod_{i=1}^{T}\alpha_i\right)$$

② $$\tilde{\mu}(x_t,x_0) = \left(\frac{\sqrt{\alpha_t}}{\beta_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0\right) \Big/ \left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)$$

$$= \left(\frac{\sqrt{\alpha_t}}{\beta_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0\right)\cdot\left(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t\right) \quad (\text{by } ①)$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\cdot\beta_t}{1-\bar{\alpha}_t}x_0$$

$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \Leftrightarrow x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon)$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\left(\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_t)\right)$$

$$= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}\right)x_t - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\epsilon_t$$

$$= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\alpha_t}}\right)x_t - \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\epsilon_t$$

$$= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\alpha_t}}\right)x_t - \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}\epsilon_t$$

$$= \frac{\sqrt{\alpha_t}\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1}) + \sqrt{\bar{\alpha}_{t-1}} - \alpha_t(\sqrt{\bar{\alpha}_{t-1}})}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}x_t - \frac{1}{\sqrt{\alpha_t}}\frac{1}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\epsilon_t$$

$$= \frac{\alpha_t(\sqrt{\bar{\alpha}_{t-1}})(1-\bar{\alpha}_{t-1}) + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\bar{\alpha}_{t-1}}\sqrt{\alpha_t}}x_t - \frac{1}{\sqrt{\alpha_t}\sqrt{\bar{\alpha}_{t-1}}}\sqrt{\bar{\alpha}_{t-1}}\frac{(1-\alpha_t)}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t$$

$$= \frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}x_t - \frac{1}{\sqrt{\alpha_t}}\frac{(1-\alpha_t)}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t\right)$$

Parameterizing the Denoising Model

Since both $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ and $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ are Normal distributions, the KL divergence has a simple form:

$$L_{t-1} = D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)||^2\right] + C$$

Recall that $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$. Ho et al. NeurIPS 2020 observe that:

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

They propose to represent the mean of the denoising model using a *noise-prediction* network:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta(\mathbf{x}_t, t)\right)$$

With this parameterization

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0\sim q(\mathbf{x}_0),\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{\beta_t^2}{2\sigma_t^2(1-\beta_t)(1-\bar{\alpha}_t)}||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon}_{\mathbf{x}_t}, t)||^2\right] + C$$

Training Objective Weighting

Trading likelihood for perceptual quality

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \Big[\underbrace{\frac{\beta_t^2}{2\sigma_t^2(1-\beta_t)(1-\bar{\alpha}_t)}}_{\lambda_t} ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\, \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, t)||^2 \Big]$$

The time dependent λ_t ensures that the training objective is weighted properly for the maximum data likelihood training.

However, this weight is often very large for small t's.

Ho et al. NeurIPS 2020 observe that simply setting $\lambda_t = 1$ improves sample quality. So, they propose to use:

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

For more advanced weighting see Choi et al., Perception Prioritized Training of Diffusion Models, CVPR 2022.

Summary

Training and Sample Generation

Algorithm 1 Training

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

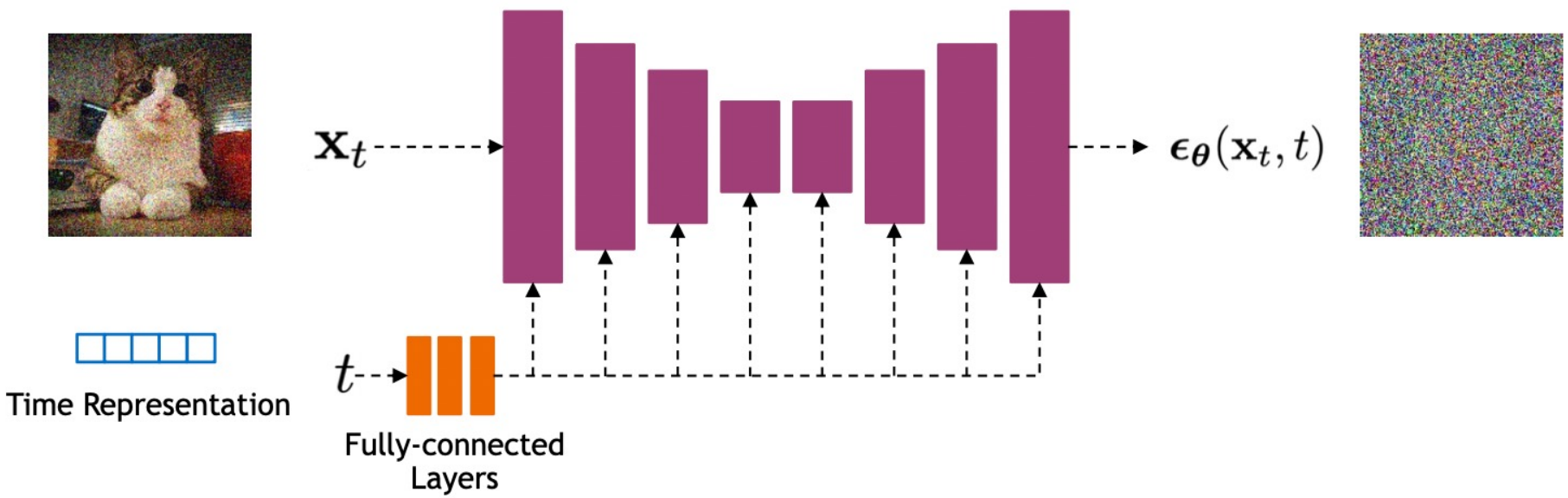
Algorithm 2 Sampling

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

Implementation Considerations

Network Architectures

Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$



Time representation: sinusoidal positional embeddings or random Fourier features.

Time features are fed to the residual blocks using either simple spatial addition or using adaptive group normalization layers. (see Dharivwal and Nichol NeurIPS 2021)

Diffusion Parameters

Noise Schedule

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

Data

Noise

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$$

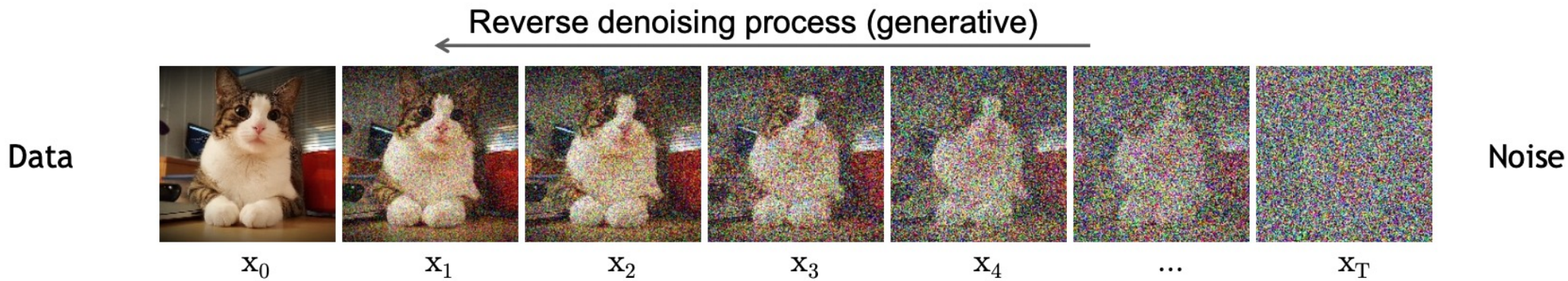
Above, β_t and σ_t^2 control the variance of the forward diffusion and reverse denoising processes respectively.

Often a linear schedule is used for β_t, and σ_t^2 is set equal to β_t.

Kingma et al. NeurIPS 2022 introduce a new parameterization of diffusion models using signal-to-noise ratio (SNR), and show how to learn the noise schedule by minimizing the variance of the training objective.

We can also train σ_t^2 while training the diffusion model by minimizing the variational bound (Improved DPM by Nichol and Dhariwal ICML 2021) or after training the diffusion model (Analytic-DPM by Bao et al. ICLR 2022).

Content-Detail Tradeoff



The weighting of the training objective for different timesteps is important!

Summary

Denoising Diffusion Probabilistic Models

In this part, we reviewed denoising diffusion probabilistic models.

The model is trained by sampling from the forward diffusion process and training a denoising model to predict the noise.

We discussed how the forward process perturbs the data distribution or data samples.

The devil is in the details:

- Network architectures
- Objective weighting
- Diffusion parameters (i.e., noise schedule)

See "Elucidating the Design Space of Diffusion-Based Generative Models" by Karras et al. for important design decisions.

Reference

- https://cvpr2022-tutorial-diffusion-models.github.io/
- https://minibatchai.com/diffusion/generative-models/text2image/2022/06/17/Diffusion.html
- https://lilianweng.github.io/posts/2021-07-11-diffusion-models/#nice
- https://happy-jihye.github.io/diffusion/diffusion-1/

감사합니다