DDPM

Denoising Diffusion Probabilistic Models

Mathematical derivation and code analysis

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

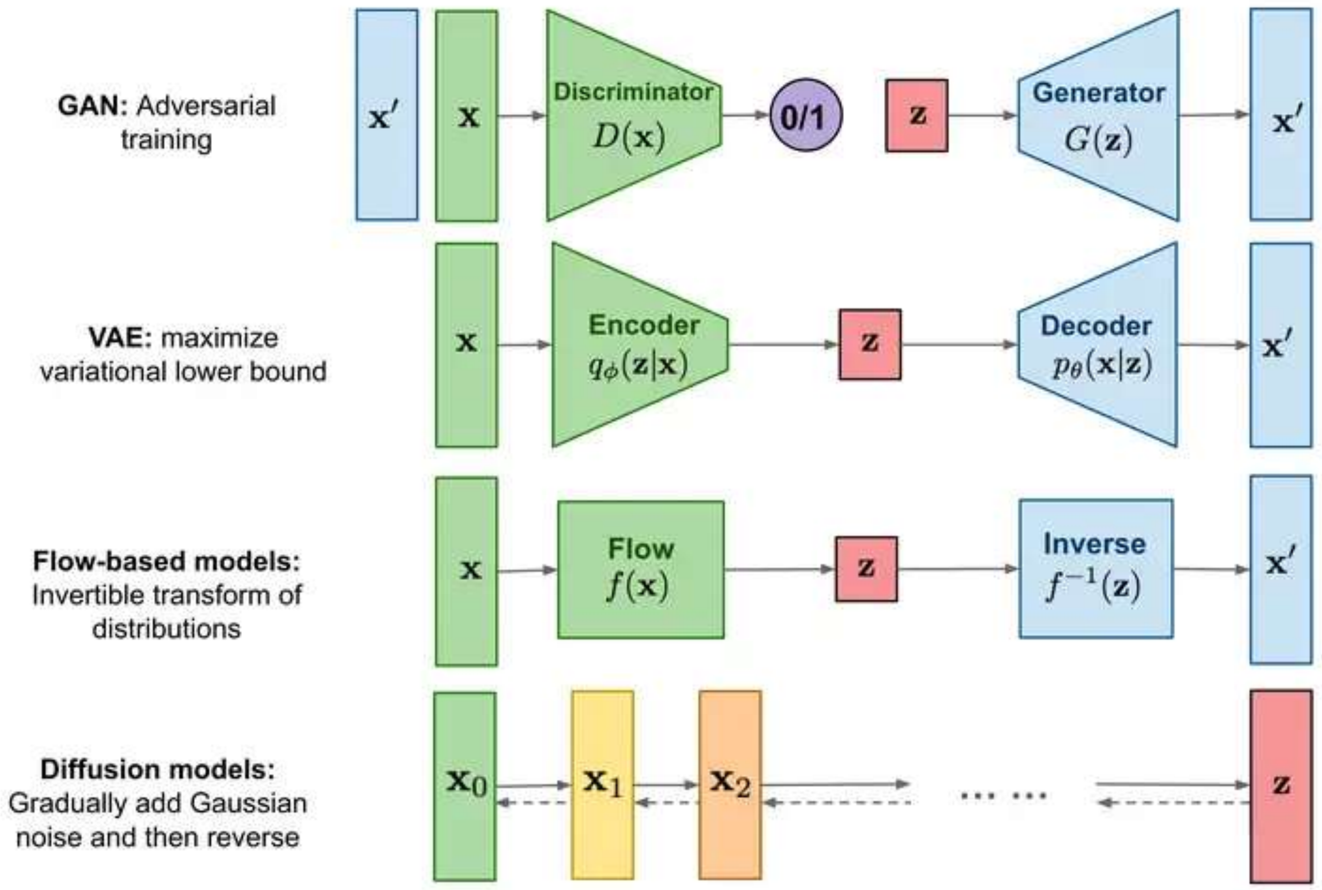
[1] HO J, JAIN AjayN, ABBEEL P. Denoising Diffusion Probabilistic Models.[J]. NeurIPS 2020.

CONTENTS

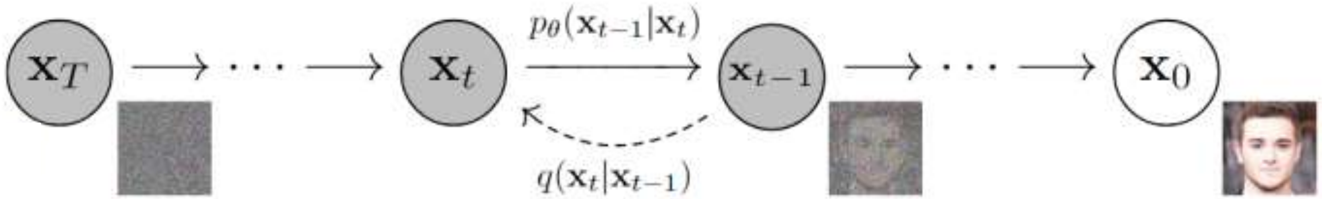
01 Diffusion Model

https://www.bilibili.com/video/BV14c411J7f2?p=1&vd_source=bbb5530c330da8646a6ba53b714a6c0e

生成模型

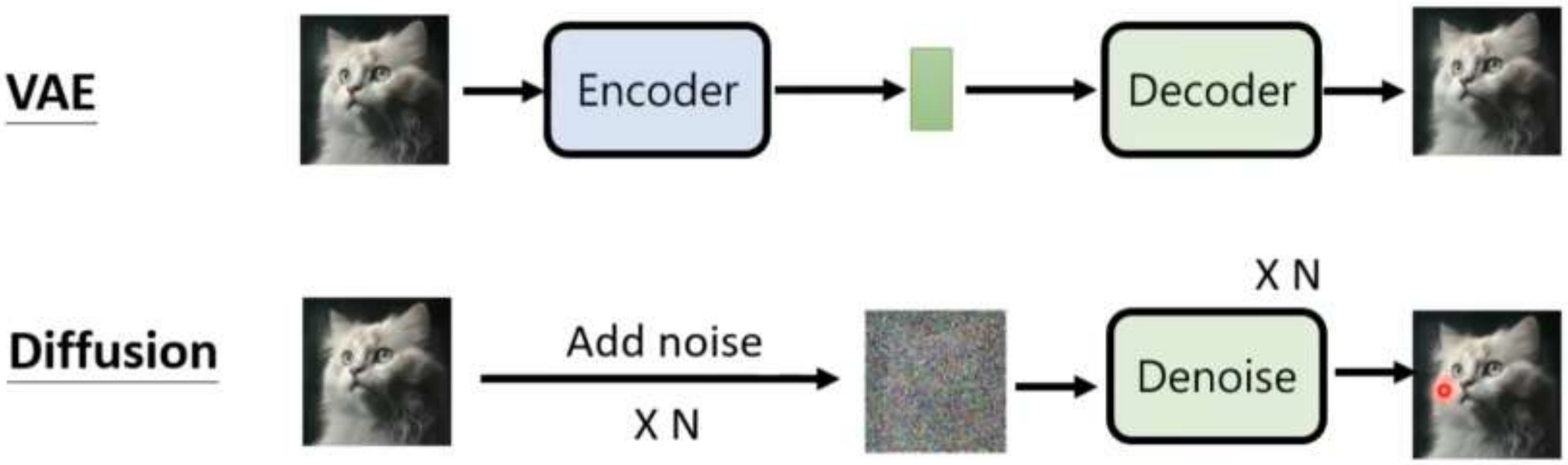

GAN: Adversarial training
x′
x
Discriminator
D(x)
0/1
z
Generator
G(z)
x′
VAE: maximize variational lower bound
x
Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
z
Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
x′
Flow-based models: Invertible transform of distributions
x
Flow
f(x)
z
Inverse
$f^{-1}(\mathbf{z})$
x′
Diffusion models: Gradually add Gaussian noise and then reverse
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_2$
z

扩散模型(DM)



扩散模型：实现从噪声（采样自简单的分布）生成目标数据样本。

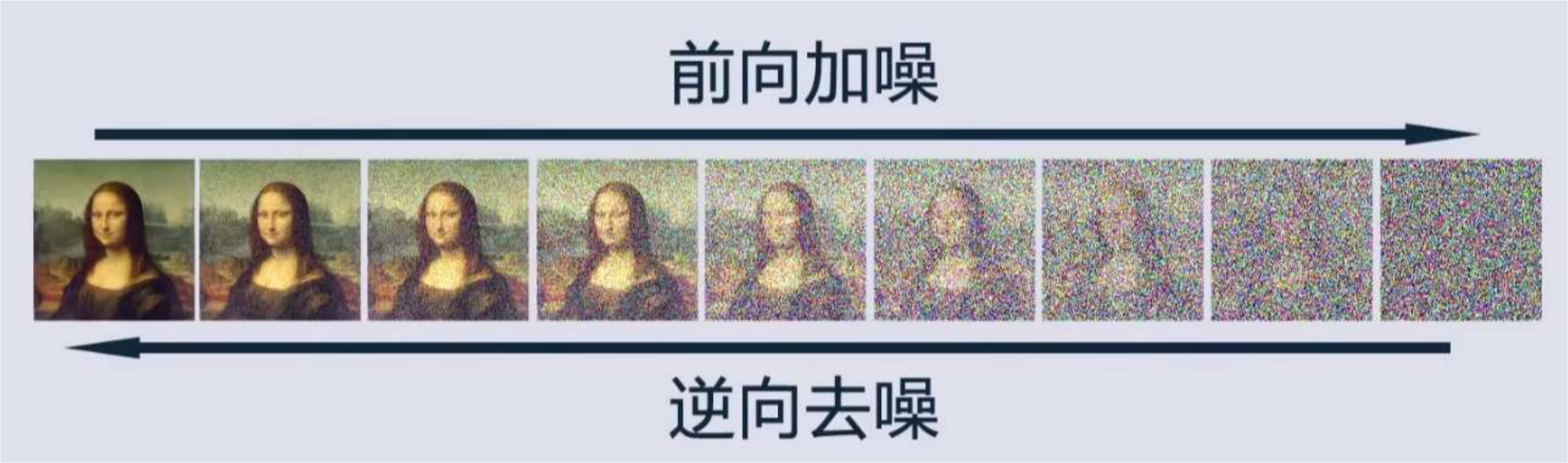
DDPM 优点:	**缺点:**
• 高质量生成 • 稳健的训练过程 • 多样性	• 计算成本高 • 训练过程复杂 • 推理速度慢



一次到位

N 次到位

扩散模型(DM)



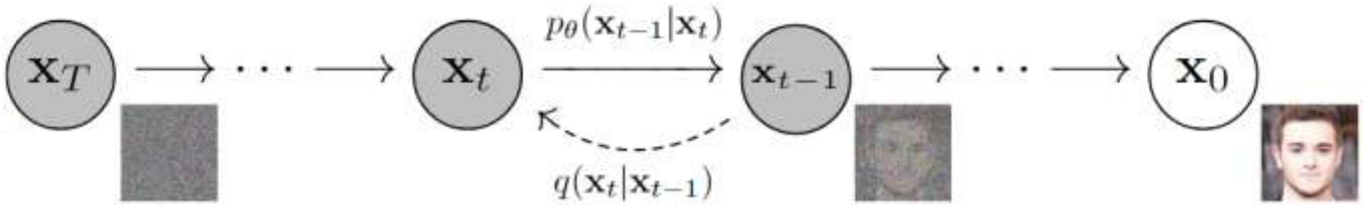
两个重要条件:

- 马尔可夫性质
- 前向反向过程服从高斯分布，变化比较小（利于数学分析）

扩散模型(DM)

$$x_t = \sqrt{\beta_t} \times \epsilon_t + \sqrt{1-\beta_t} \times x_{t-1}$$
$$\epsilon_t \sim N(0,1)$$

$$0 < \beta_1 < \beta_2 < \beta_3 < \beta_{t-1} < \beta_t < 1$$



包括两个过程：

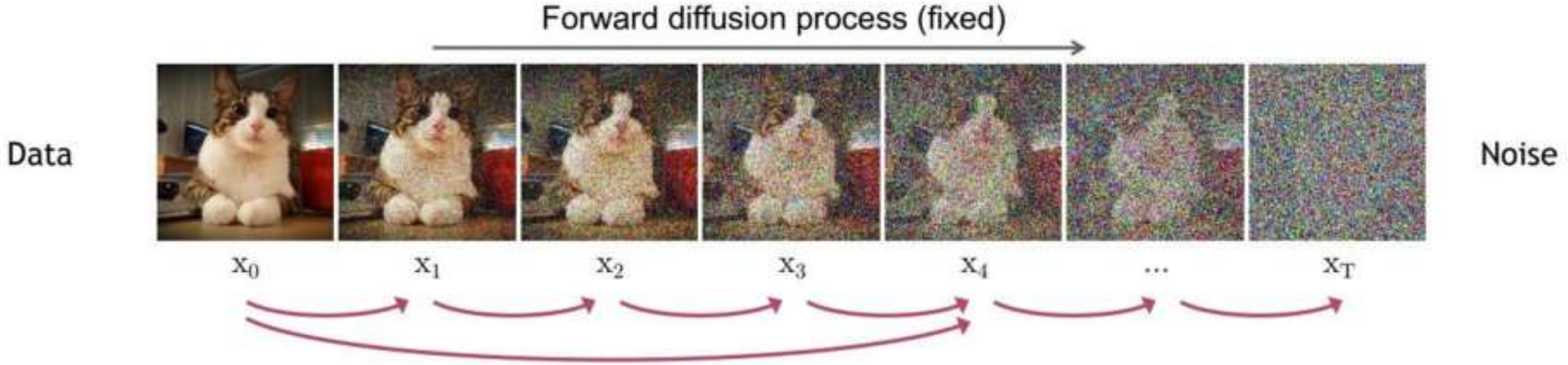
- 前向过程（forward process）

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

- 反向过程（reverse process）

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

Forward



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

$$\alpha_t := 1 - \beta_t \text{ and } \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

$$x_t = \sqrt{1-\alpha_t} \times \epsilon_t + \sqrt{\alpha_t} \times x_{t-1}$$

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

Forward

$$x_t = \sqrt{1-\alpha_t} \times \epsilon_t + \sqrt{\alpha_t} \times x_{t-1}$$

$$x_{t-1} = \sqrt{1-\alpha_{t-1}} \times \epsilon_{t-1} + \sqrt{\alpha_{t-1}} \times x_{t-2}$$

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

$$x_t = \sqrt{a_t(1-a_{t-1})}\epsilon_{t-1} + \sqrt{1-a_t} \times \epsilon_t + \sqrt{a_t a_{t-1}} \times x_{t-2}$$

Forward

$$x_t = \sqrt{a_t(1 - a_{t-1})}\epsilon_{t-1} + \sqrt{1 - a_t} \times \epsilon_t + \sqrt{a_t a_{t-1}} \times x_{t-2}$$

$$\boxed{\begin{gathered} N(\mu_1, {\sigma_1}^2) + N(\mu_2, {\sigma_2}^2) = N(\mu_1 + \mu_2, {\sigma_1}^2 + {\sigma_2}^2) \\ N(0, \alpha_t - \alpha_t\alpha_{t-1}) + N(0, 1 - \alpha_t) = N(0, 1 - \alpha_t\alpha_{t-1}) \end{gathered}}$$

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

Forward

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

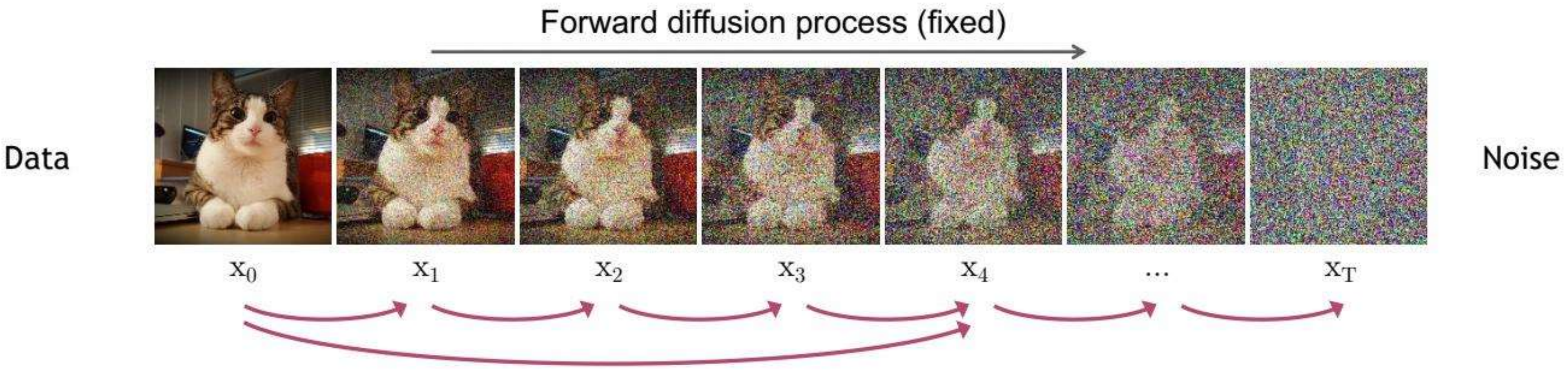
$$x_{t-2} = \sqrt{1 - \alpha_{t-2}} \times \epsilon_{t-2} + \sqrt{\alpha_{t-2}} \times x_{t-3}$$

$$\bar{\alpha}_t = \alpha_1 \alpha_2 \dots \alpha_t$$

$$x_t = \sqrt{1 - \alpha_t \alpha_{t-1} \alpha_{t-2}} \times \epsilon + \sqrt{\alpha_t \alpha_{t-1} \alpha_{t-2}} \times x_{t-3}$$

$$\boxed{x_t = \sqrt{1 - a_t a_{t-1} a_{t-2} a_{t-3} \dots a_2 a_1} \times \epsilon + \sqrt{a_t a_{t-1} a_{t-2} a_{t-3} \dots a_2 a_1} \times x0}$$

Forward

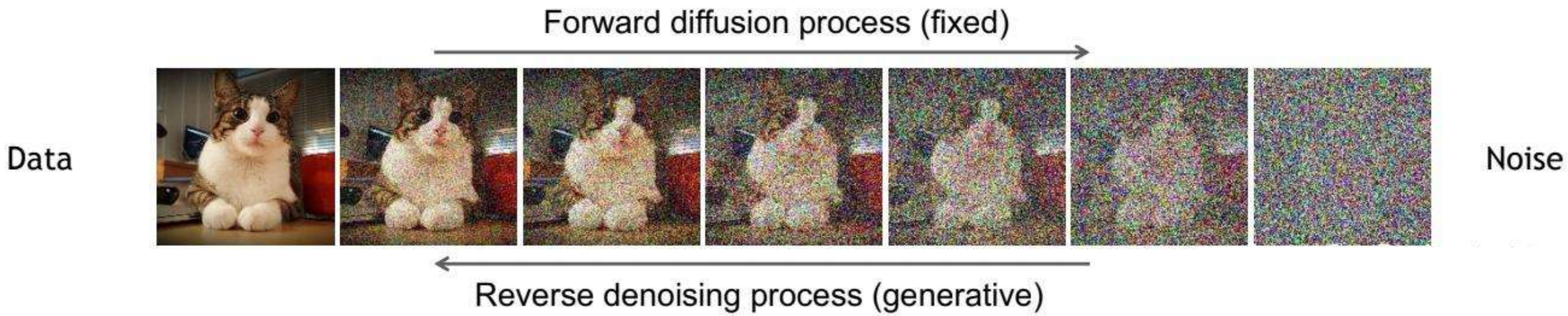


Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

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 $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

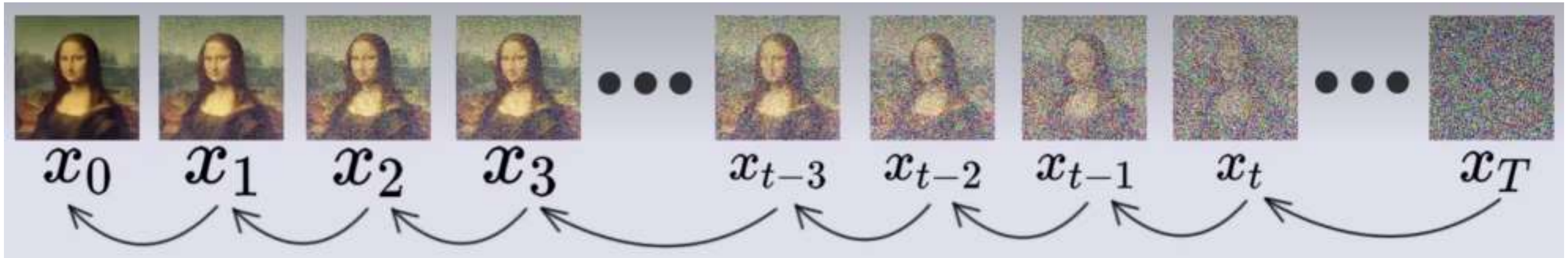
Denoise



$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

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

Denoise



贝叶斯公式

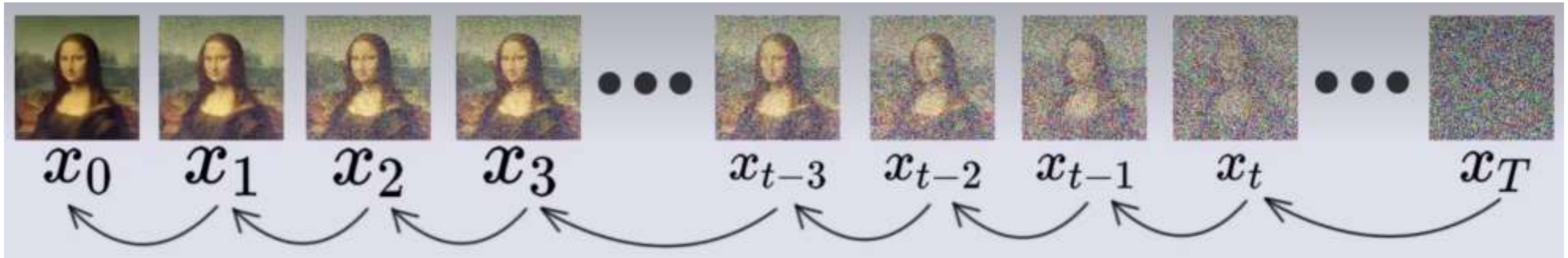
$$\boxed{P(A|B) = \frac{P(B|A)P(A)}{P(B)}}$$

$$P(x_{t-1}|x_t) = \frac{P(x_t|x_{t-1})P(x_{t-1})}{P(x_t)} \qquad = \frac{P(x_t|x_{t-1}, x_0)P(x_{t-1}|x_0)}{P(x_t|x_0)}$$

$$x_t = \sqrt{1-\alpha_t} \times \epsilon_t + \sqrt{\alpha_t} \times x_{t-1}$$
$$x_t = \sqrt{1-\bar{\alpha}_t} \times \epsilon + \sqrt{\bar{\alpha}_t} \times x_0$$
$$x_{t-1} = \sqrt{1-\bar{\alpha}_{t-1}} \times \epsilon + \sqrt{\bar{\alpha}_{t-1}} \times x_0$$

$$N(\sqrt{\alpha_t}x_{t-1}, 1-\alpha_t)$$
$$N(\sqrt{\bar{\alpha}_t}x_0, 1-\bar{\alpha}_t)$$
$$N(\sqrt{\bar{\alpha}_{t-1}}x_0, 1-\bar{\alpha}_{t-1})$$

Denoise



贝叶斯公式

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

$$P(x_{t-1}|x_t) = \frac{P(x_t|x_{t-1})P(x_{t-1})}{P(x_t)} \qquad = \frac{P(x_t|x_{t-1}, x_0)P(x_{t-1}|x_0)}{P(x_t|x_0)}$$

$$\boxed{\begin{aligned} x_t &= \sqrt{1-\alpha_t} \times \epsilon_t + \sqrt{\alpha_t} \times x_{t-1} \\ x_t &= \sqrt{1-\bar{\alpha}_t} \times \epsilon + \sqrt{\bar{\alpha}_t} \times x_0 \\ x_{t-1} &= \sqrt{1-\bar{\alpha}_{t-1}} \times \epsilon + \sqrt{\bar{\alpha}_{t-1}} \times x_0 \end{aligned}}$$

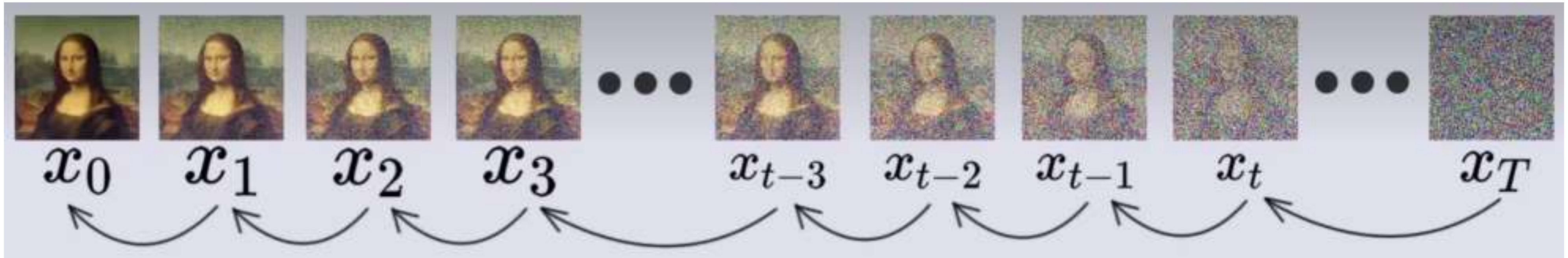
$$P(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, x_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-a_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{a_t}x_{t-1})^2}{1-a_t}\right]}$$

$$P(\boldsymbol{x}_t|\boldsymbol{x}_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{\bar{a}_t}x_0)^2}{1-\bar{a}_t}\right]}$$

$$P(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_{t-1}}} e^{\left[-\frac{1}{2}\frac{(x_{t-1}-\sqrt{\bar{a}_{t-1}}x_0)^2}{1-\bar{a}_{t-1}}\right]}$$

Denoise

贝叶斯公式



$$P(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, x_0) = \frac{\frac{1}{\sqrt{2\pi}\sqrt{1-a_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{a_t}x_{t-1})^2}{1-a_t}\right]} \frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_{t-1}}} e^{\left[-\frac{1}{2}\frac{(x_{t-1}-\sqrt{\bar{a}_{t-1}}x_0)^2}{1-\bar{a}_{t-1}}\right]}}{\frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{\bar{a}_t}x_0)^2}{1-\bar{a}_t}\right]}}$$

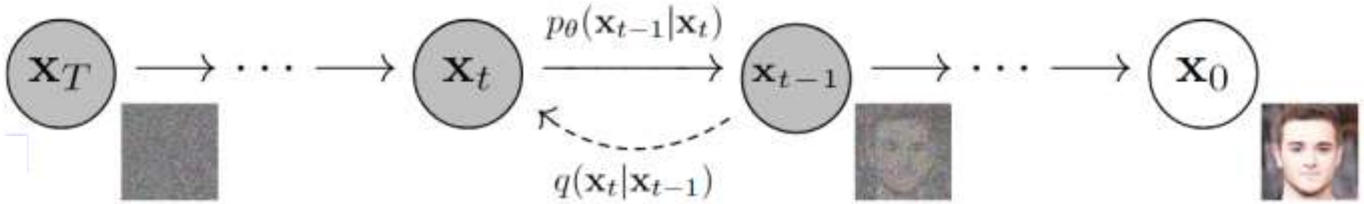
$$\boxed{\begin{aligned} x_t &= \sqrt{1-\alpha_t} \times \epsilon_t + \sqrt{\alpha_t} \times x_{t-1} \\ x_t &= \sqrt{1-\bar{\alpha}_t} \times \epsilon + \sqrt{\bar{\alpha}_t} \times x_0 \\ x_{t-1} &= \sqrt{1-\bar{\alpha}_{t-1}} \times \epsilon + \sqrt{\bar{\alpha}_{t-1}} \times x_0 \end{aligned}}$$

$$P(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, x_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-a_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{a_t}x_{t-1})^2}{1-a_t}\right]}$$

$$P(\boldsymbol{x}_t|\boldsymbol{x}_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_t}} e^{\left[-\frac{1}{2}\frac{(x_t-\sqrt{\bar{a}_t}x_0)^2}{1-\bar{a}_t}\right]}$$

$$P(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0) = \frac{1}{\sqrt{2\pi}\sqrt{1-\bar{a}_{t-1}}} e^{\left[-\frac{1}{2}\frac{(x_{t-1}-\sqrt{\bar{a}_{t-1}}x_0)^2}{1-a_{t-1}}\right]}$$

Denoise



方差:

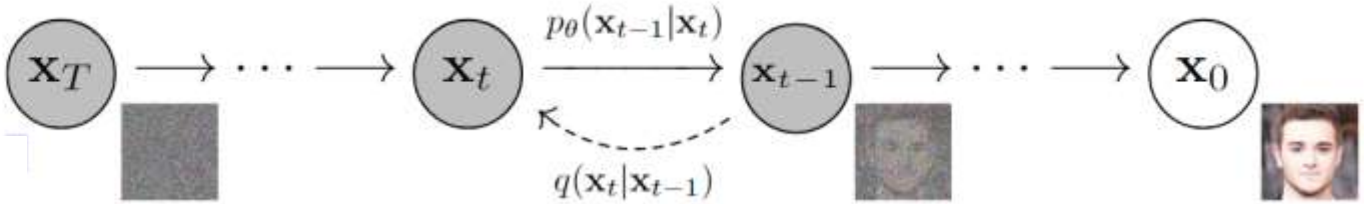
$$\tilde{\beta}_t = 1/\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right) = 1/\left(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}\right) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

均值:

$$\begin{aligned}\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) &= \left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)/\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right) \\ &= \left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t \\ &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0\end{aligned}$$

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

Denoise



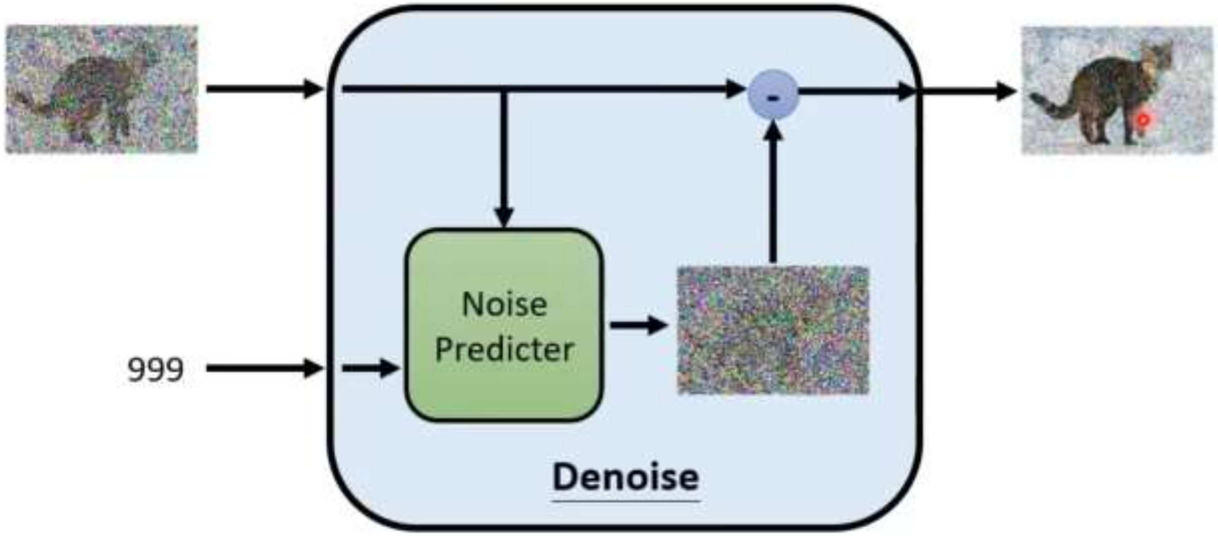
$$\mathbf{x_t}(\mathbf{x_0}, \epsilon) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \quad \text{where } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

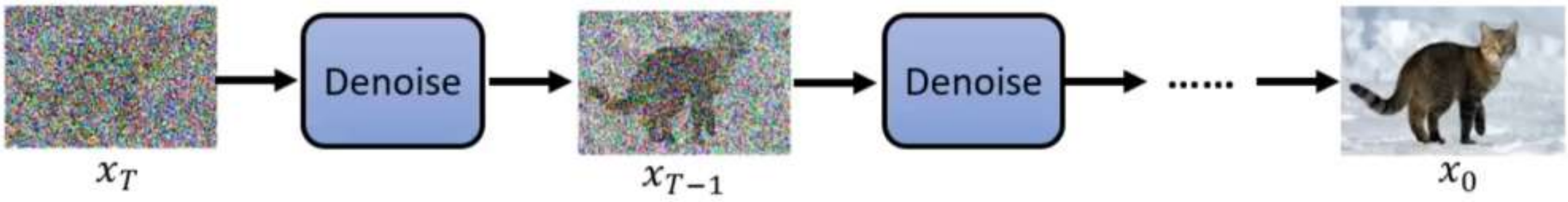
$$\mathbf{x_0} = \frac{\mathbf{x_t}(\mathbf{x_0}, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}}$$

Network

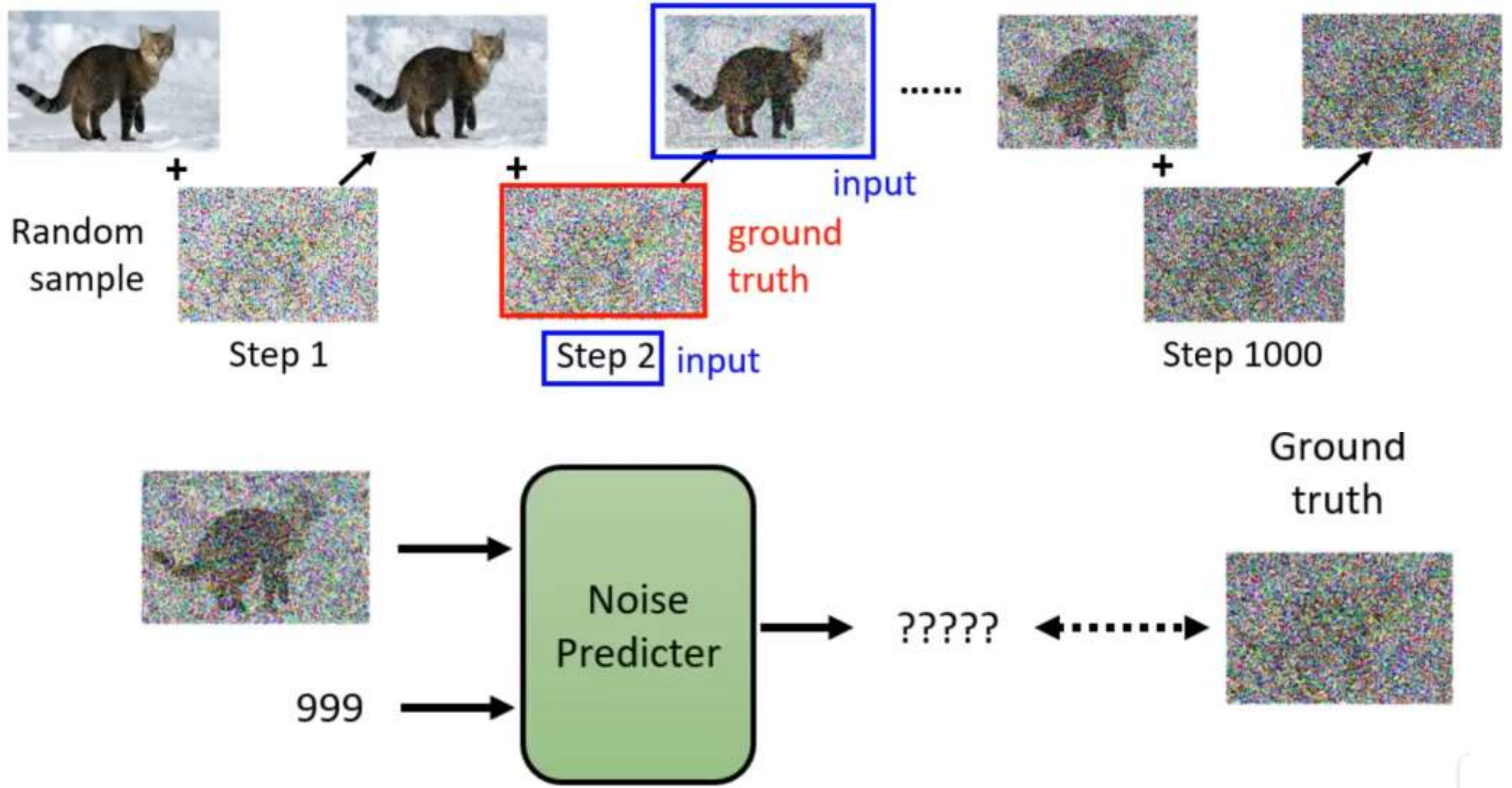
$$\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 \Longrightarrow \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x_t}(\mathbf{x_0}, \epsilon) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boxed{\epsilon}\right)$$

Denoise





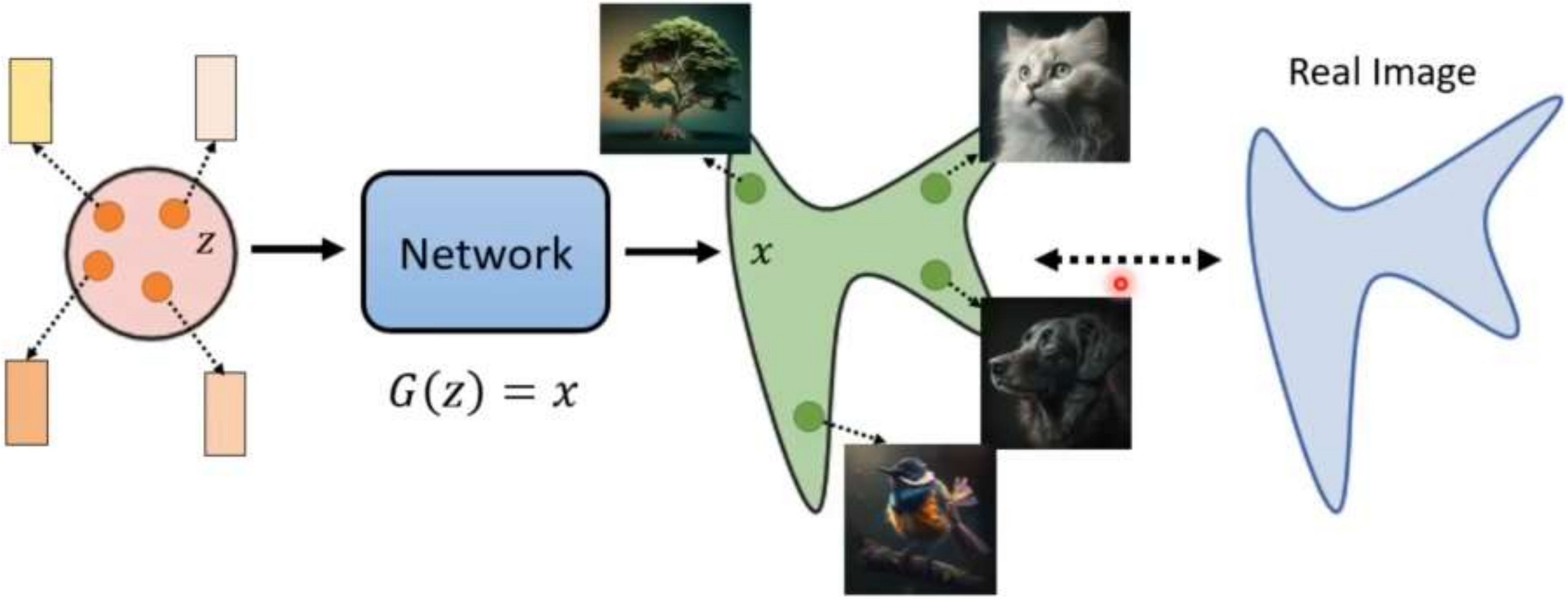
DDPM



input
Random sample
ground truth
Step 1
Step 2
input
Step 1000
Ground truth
Noise Predicter
999
?????

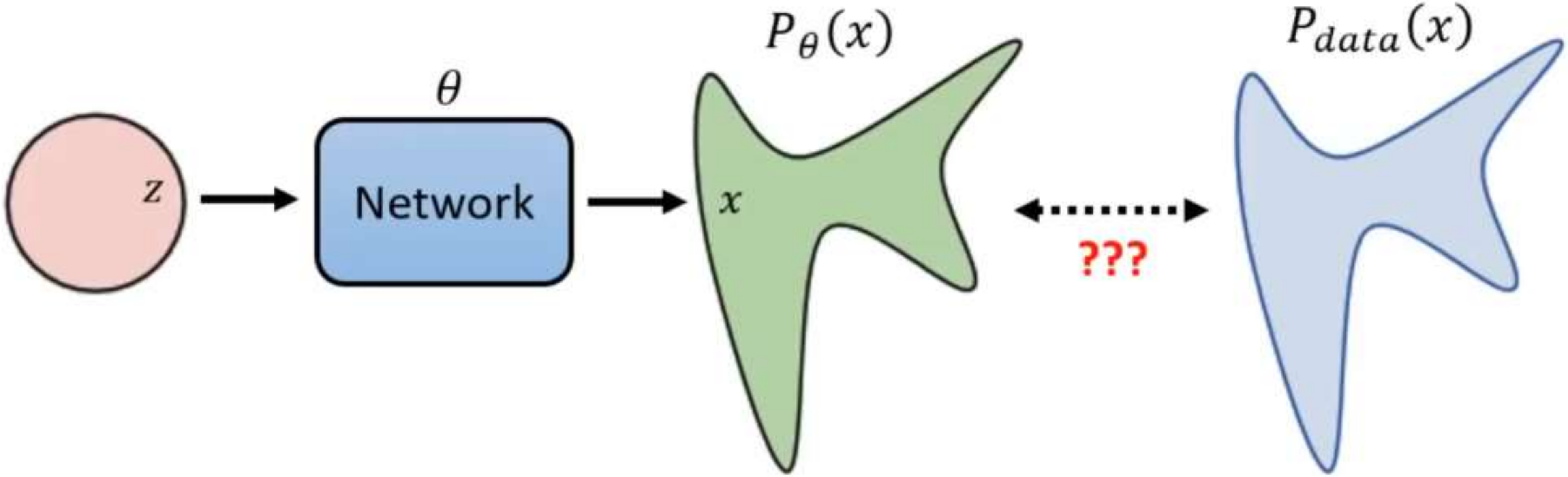
Optimization objectives

影像生成模型共同的目标



Optimization objectives

Maximum Likelihood Estimation



Sample $\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$

compute $P_\theta(x^i)$

$$\theta^* = arg \max_\theta \prod_{i=1}^{m} P_\theta(x^i)$$

Optimization objectives

Sample $\{x^1, x^2, \dots, x^m\}$ from $P_{data}(x)$

$$\theta^* = arg\max_{\theta} \prod_{i=1}^{m} P_\theta(x^i) \quad = arg\max_{\theta} log \prod_{i=1}^{m} P_\theta(x^i)$$

$$= arg\max_{\theta} \sum_{i=1}^{m} logP_\theta(x^i) \approx arg\max_{\theta} E_{x \sim P_{data}}[logP_\theta(x)]$$

(not related to θ)

$$= arg\max_{\theta} \int_x P_{data}(x) logP_\theta(x) dx \quad - \int_x P_{data}(x) logP_{data}(x) dx$$

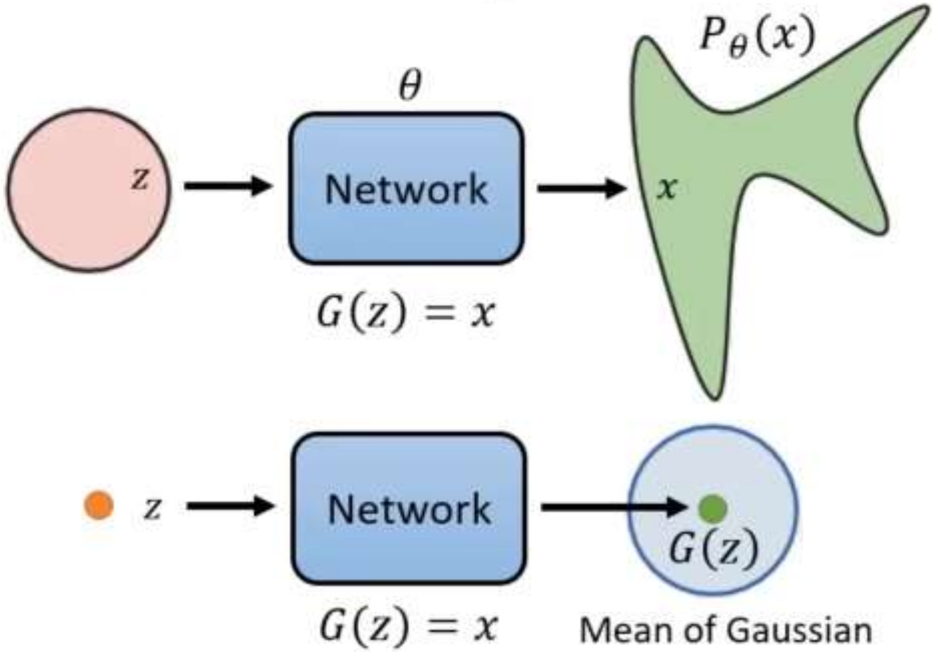
Difference between P_{data} and P_θ

$$= arg\max_{\theta} \int_x P_{data}(x) log \frac{P_\theta(x)}{P_{data}(x)} dx \quad = arg\min_{\theta} KL(P_{data} || P_\theta)$$

最大似然估计等价于最小 KL 散度

Optimization objectives

VAE: Compute $P_\theta(x)$



$$P_\theta(x) = \int_z P(z)P_\theta(x|z)dz$$

$$P_\theta(x|z) \quad \propto exp(-\|G(z) - x\|_2)$$

Optimization objectives

VAE： Lower bound of $logP(x)$

$$logP_\theta(x) = \int_z q(z|x)logP(x)dz$$

$q(z|x)$ can be any distribution

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{P(z|x)}\right)dz = \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\right)dz + \int_z q(z|x)log\left(\frac{q(z|x)}{P(z|x)}\right)dz$$

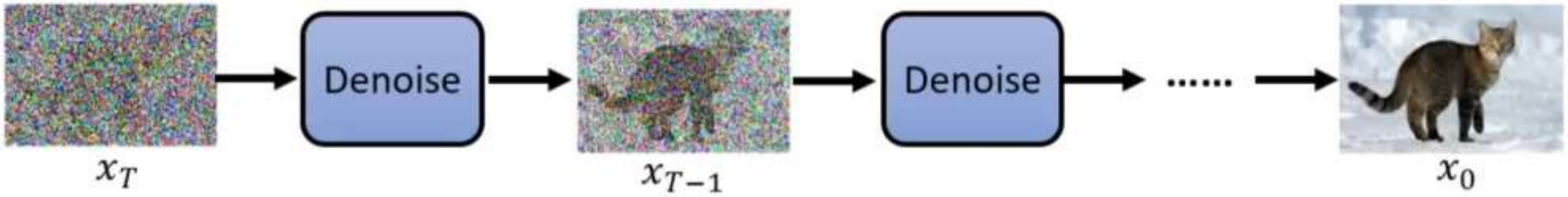
$$\boxed{KL\big(q(z|x)||P(z|x)\big)} \geq 0$$

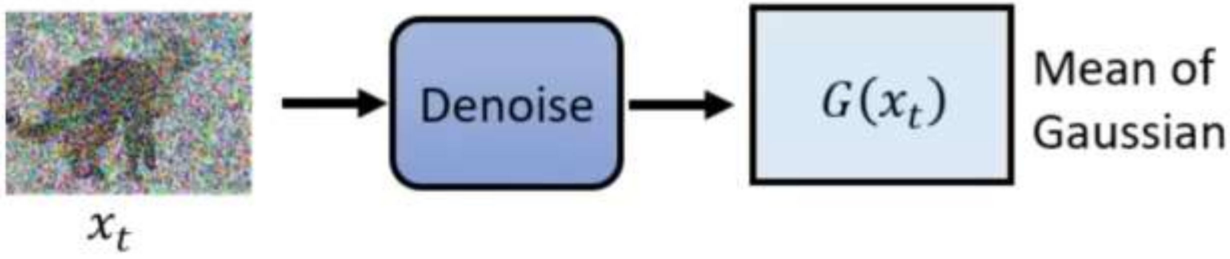
$$\geq \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\right)dz = \mathrm{E}_{q(z|x)}[log\left(\frac{P(x,z)}{q(z|x)}\right)]$$

Encoder

Optimization objectives

DDPM: Compute $P_\theta(x)$

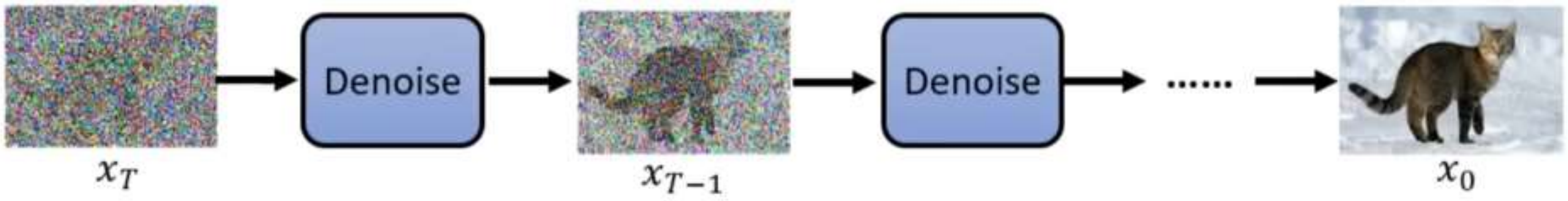




$$P_\theta(x_{t-1}|x_t) \quad \propto exp(-\|G(z) - x_{t-1}\|_2)$$

$$P_\theta(x_0) = \int_{x_1:x_T} P(x_T)P_\theta(x_{T-1}|x_T)...P_\theta(x_{t-1}|x_t)...P_\theta(x_0|x_1)dx_1 : x_T$$

Optimization objectives



$$P_\theta(x_0) = \int_{x_1:x_T} P(x_T)P_\theta(x_{T-1}|x_T) \dots P_\theta(x_{t-1}|x_t) \dots P_\theta(x_0|x_1)dx_1:x_T$$

$$\begin{aligned}
\log p_\theta(\mathbf{x}_0) &= \log \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \\
&= \log \int \frac{p_\theta(\mathbf{x}_{0:T}) q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} d\mathbf{x}_{1:T} \\
&\geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} [\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}]
\end{aligned}$$

应用 Jensen 不等式： $f(\mathbb{E}[X]) \geq \mathbb{E}[f(X)]$

$$\log \int \frac{p_\theta(\mathbf{x}_{0:T}) q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)} d\mathbf{x}_{1:T} \geq \int q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) \log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)} d\mathbf{x}_{1:T}$$

Optimization objectives

VAE Maximize $logP_\theta(x) \longrightarrow$ Maximize $E_{q(z|x)}[log\left(\frac{P(x,z)}{q(z|x)}\right)]$

Encoder

DDPM Maximize $logP_\theta(x_0) \longrightarrow$ Maximize $E_{q(x_1:x_T|x_0)}[log\left(\frac{P(x_0:x_T)}{q(x_1:x_T|x_0)}\right)]$

Forward process

真实后验分布：

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2\mathbf{I})$$

估计分布：

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

Optimization objectives

到变分下界（ELBO）

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

对于网络训练来说，其训练目标为 VLB 取负

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

$$= \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))\right]}_{L_{t-1}} - \underbrace{\mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}$$

最终的优化目标共包含 T + 1 项，其中 L_0 可以看成是原始数据重建，优化的是负对数似然，L_0 可以用估计的 $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_\theta(\mathbf{x}_1, 1), \boldsymbol{\Sigma}_\theta(\mathbf{x}_1, 1))$ 来构建一个离散化的 decoder 来计算 （见 DDPM 论文 3.3 部分）

Optimization objectives

$$\log p(\boldsymbol{x}) = \log \int p(\boldsymbol{x}_{0:T}) d\boldsymbol{x}_{1:T} \tag{34}$$

$$= \log \int \frac{p(\boldsymbol{x}_{0:T}) q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} d\boldsymbol{x}_{1:T} \tag{35}$$

$$= \log \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \tag{36}$$

$$\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \tag{37}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{\prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{38}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) \prod_{t=1}^{T-1} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{39}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \prod_{t=1}^{T-1} p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) \prod_{t=1}^{T-1} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{40}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1})} \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \prod_{t=1}^{T-1} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{41}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1})} \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\sum_{t=1}^{T-1} \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{42}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1})} \right] + \sum_{t=1}^{T-1} \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{43}$$

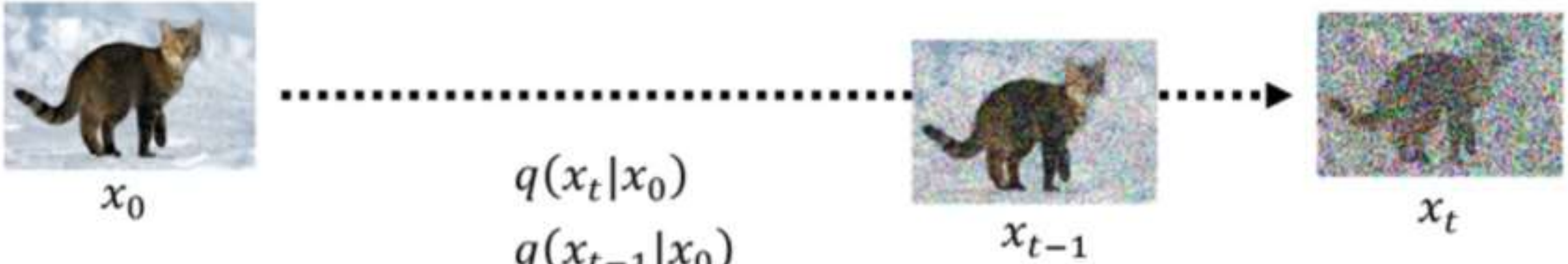
$$= \mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{T-1}, \boldsymbol{x}_T|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1})} \right] + \sum_{t=1}^{T-1} \mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{44}$$

$$= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) \parallel p(\boldsymbol{x}_T)) \right]}_{\text{prior matching term}} - \sum_{t=1}^{T-1} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})) \right]}_{\text{consistency term}} \tag{45}$$

https://arxiv.org/pdf/2208.11970

Optimization objectives

$$E_{q(x_t|x_0)}\left[KL\left(q(x_{t-1}|x_t,x_0)||P(x_{t-1}|x_t)\right)\right]$$



$q(x_t|x_0)$

$q(x_{t-1}|x_0)$

$q(x_t|x_{t-1})$

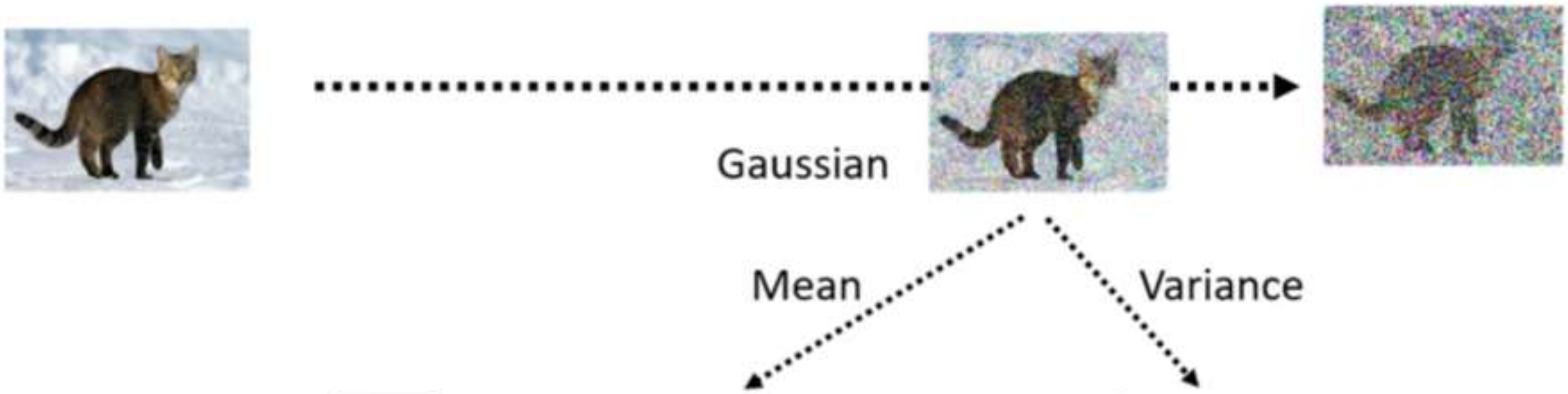
$q(x_{t-1}|x_t,x_0)$

$$= \frac{q(x_{t-1},x_t,x_0)}{q(x_t,x_0)} = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)q(x_0)}{q(x_t|x_0)q(x_0)} = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}$$

$$\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$

Optimization objectives

$$\mathrm{E}_{q(x_t|x_0)}\left[KL\left(q(x_{t-1}|x_t,x_0)||P(x_{t-1}|x_t)\right)\right]$$



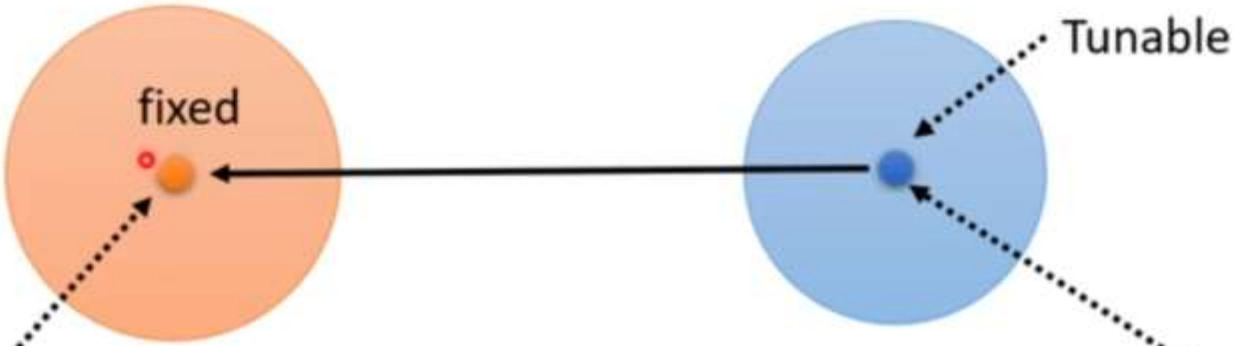
$$\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t x_0 + \sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t}{1-\bar{\alpha}_t}$$

$$\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t I$$

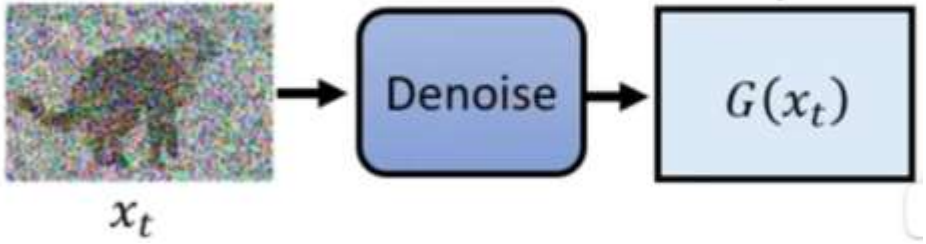
Optimization objectives

$$\sum_{t=2}^{T} \boxed{\mathrm{E}_{q(x_t|x_0)}[KL(q(x_{t-1}|x_t, x_0)||P(x_{t-1}|x_t))]}$$

How to minimize
KL divergence?



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



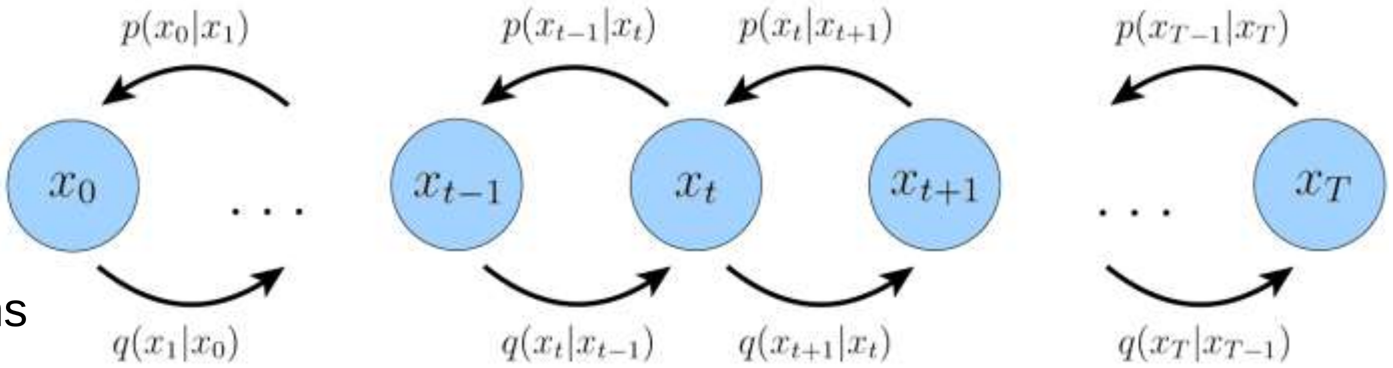
$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon$$

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

$$\frac{x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon}{\sqrt{\bar{\alpha}_t}} = x_0$$

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

Optimization objectives



KL Divergence between two Gaussian distributions

$$\boxed{D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu}_x,\boldsymbol{\Sigma}_x)\parallel\mathcal{N}(\boldsymbol{y};\boldsymbol{\mu}_y,\boldsymbol{\Sigma}_y))=\frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_y|}{|\boldsymbol{\Sigma}_x|}-d+\mathrm{tr}(\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_x)+(\boldsymbol{\mu}_y-\boldsymbol{\mu}_x)\right]}$$

固定的方差：$\boldsymbol{\Sigma}_\theta(\mathbf{x}_t,t)=\sigma_t^2\mathbf{I}$

$$\begin{aligned}
D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) &= D_{\mathrm{KL}}(\mathcal{N}(\mathbf{x}_{t-1};\tilde{\boldsymbol{\mu}}(\mathbf{x}_t,\mathbf{x}_0),\sigma_t^2\mathbf{I})\parallel\mathcal{N}(\mathbf{x}_{t-1};\boldsymbol{\mu}_\theta(\mathbf{x}_t,t),\sigma_t^2\mathbf{I})) \\
&= \frac{1}{2}(n+\frac{1}{\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0)-\boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2-n+\log 1) \\
&= \frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0)-\boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2
\end{aligned}$$

那么优化目标 L_{t-1} 即为：

$$L_{t-1}=\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0)-\boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2\right]$$

Objective	IS	FID
$\tilde{\mu}$ prediction (baseline)		
L, learned diagonal Σ	7.28±0.10	23.69
L, fixed isotropic Σ	8.06±0.09	13.22
$\|\tilde{\mu}-\tilde{\mu}_\theta\|^2$	–	–
ϵ prediction (ours)		
L, learned diagonal Σ	–	–
L, fixed isotropic Σ	7.67±0.13	13.51
$\|\tilde{\epsilon}-\epsilon_\theta\|^2$ (L_{simple})	**9.46±0.11**	**3.17**

Optimization objectives

$$\mathbf{x_t}(\mathbf{x_0}, \epsilon) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \quad \text{where } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

用预测噪声代替预测均值

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0}\left(\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\right]\right)$$

$$= \mathbb{E}_{\mathbf{x}_0, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x_t}(\mathbf{x_0}, \epsilon), \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x_t}(\mathbf{x_0}, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon\right)\right) - \boldsymbol{\mu}_\theta(\mathbf{x_t}(\mathbf{x_0}, \epsilon), t)\right\|^2\right]$$

$$= \mathbb{E}_{\mathbf{x}_0, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[\frac{1}{2\sigma_t^2}\left\|\left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x_t}(\mathbf{x_0}, \epsilon) + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x_t}(\mathbf{x_0}, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon\right)\right) - \boldsymbol{\mu}_\theta(\mathbf{x_t}(\mathbf{x_0}\right.\right.$$

$$= \mathbb{E}_{\mathbf{x}_0, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x_0}, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right) - \boldsymbol{\mu}_\theta(\mathbf{x_t}(\mathbf{x_0}, \epsilon), t)\right\|^2\right]$$

Optimization objectives

$$\mathbf{x_{t-1}} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x_t} - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x_t}, t)\right) + \sigma_t \mathbf{z}$$

$$\boldsymbol{\mu}_\theta(\mathbf{x_t}(\mathbf{x_0}, \epsilon), t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x_0}, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\big(\mathbf{x}_t(\mathbf{x_0}, \epsilon), t\big)\right)$$

$$\begin{aligned}
L_{t-1} &= \mathbb{E}_{\mathbf{x}_0, \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x_0}, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right) - \boldsymbol{\mu}_\theta(\mathbf{x_t}(\mathbf{x_0}, \epsilon), t)\right\|^2\right] \\
&= \mathbb{E}_{\mathbf{x}_0, \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon - \epsilon_\theta\big(\mathbf{x}_t(\mathbf{x_0}, \epsilon), t\big)\right\|^2\right] \\
&= \mathbb{E}_{\mathbf{x}_0, \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon - \epsilon_\theta\big(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\big)\right\|^2\right]
\end{aligned}$$

去掉权重系数

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

Algorithm 1 Training

repeat

$\mathbf{x}_0 \sim q(\mathbf{x}_0)$

$t \sim \mathrm{Uniform}(\{1, \dots, T\})$

$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

until converged

Algorithm 2 Sampling

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

for $t = T, \dots, 1$ **do**

$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$$

end for

return $\mathbf{x}_0$

Train

Algorithm 1 Training

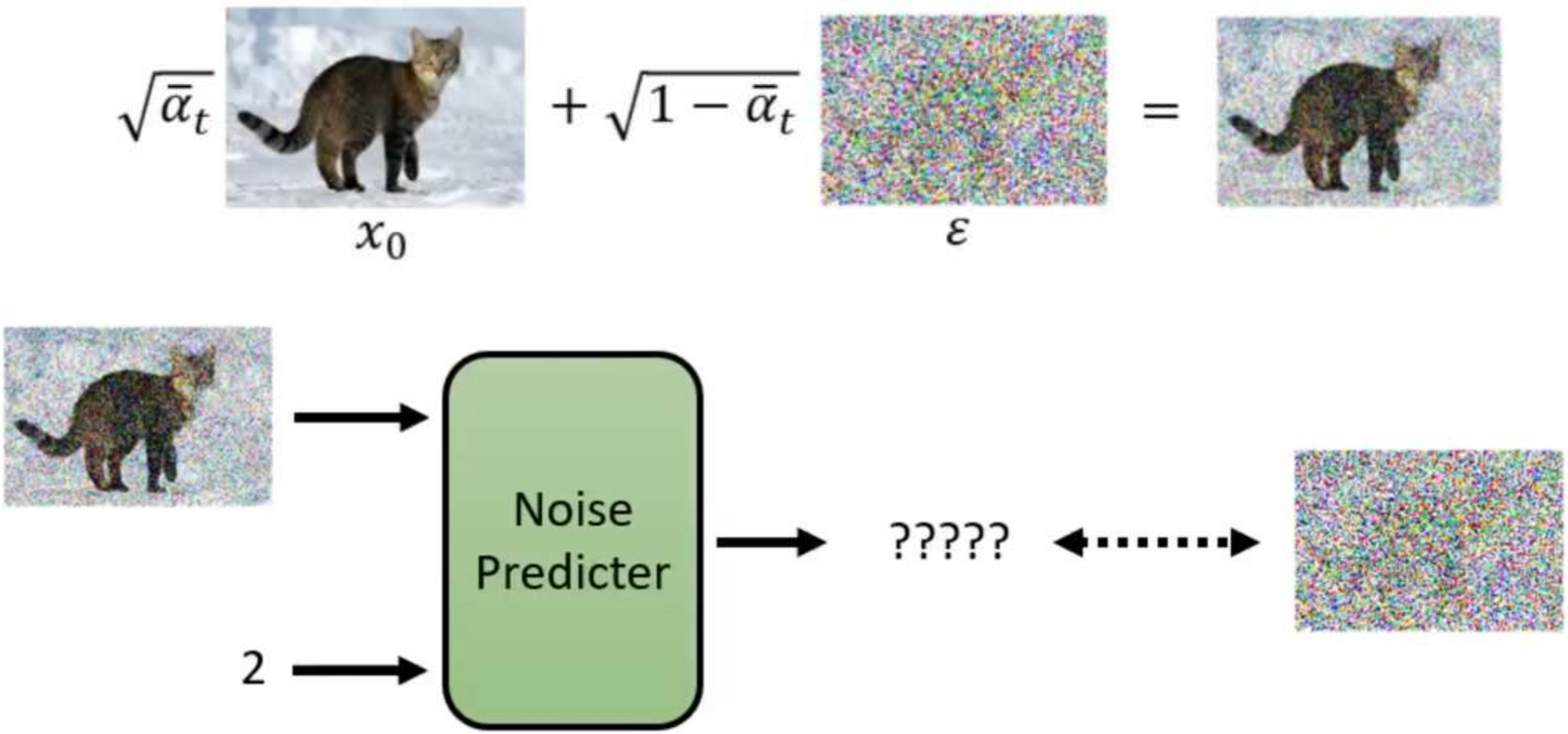
repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ ← sample clean image
  $t \sim \text{Uniform}(\{1, \dots, T\})$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ← sample a noise
  Take gradient descent step on

$$\nabla_\theta \left\| \underset{\text{Target Noise}}{\boldsymbol{\epsilon}} - \underbrace{\boldsymbol{\epsilon}_\theta\Big(\underset{\text{Noisy image}}{\boxed{\sqrt{\bar\alpha_t}\mathbf{x}_0 + \sqrt{1-\bar\alpha_t}\boldsymbol{\epsilon}}}, t\Big)}_{\text{Noise predictor}} \right\|^2$$

until converged

$$\xrightarrow[\text{smaller}]{\bar\alpha_1, \bar\alpha_2, \dots\ \bar\alpha_T}$$

Train



$\sqrt{\bar{\alpha}_t}$
$+ \sqrt{1-\bar{\alpha}_t}$
$=$
x_0
ε
Noise Predicter
2
?????

Inference

x_T

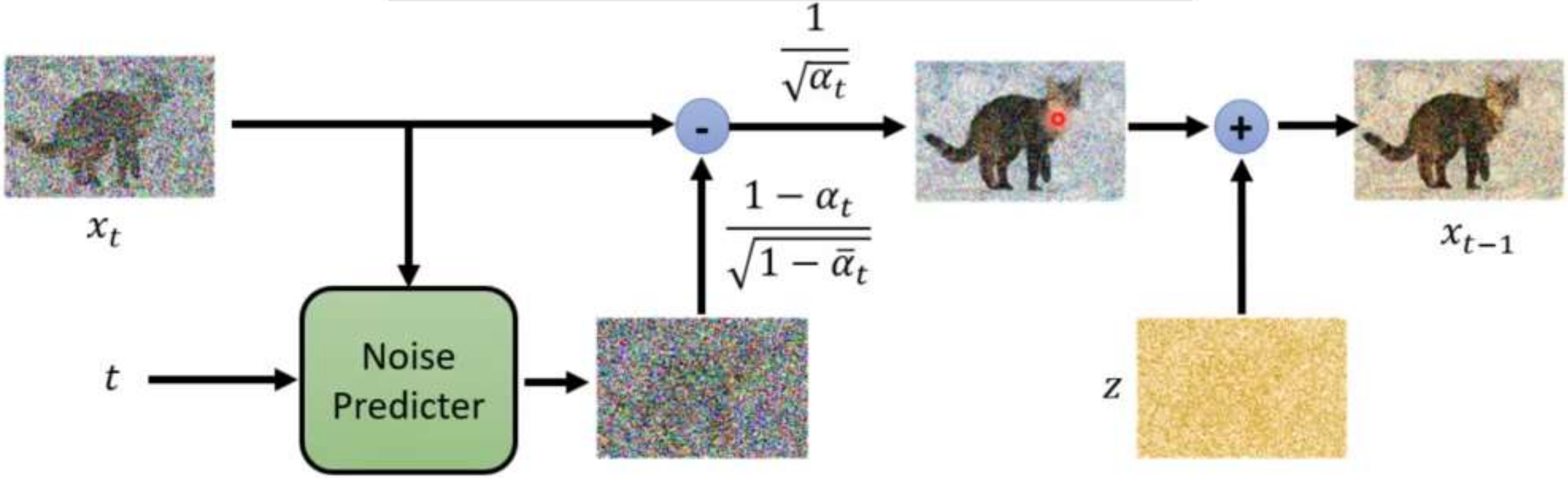
Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\boxed{\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
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** $\mathbf{x}_0$

sample a noise?!

$\bar{\alpha}_1, \bar{\alpha}_2, \dots \bar{\alpha}_T$

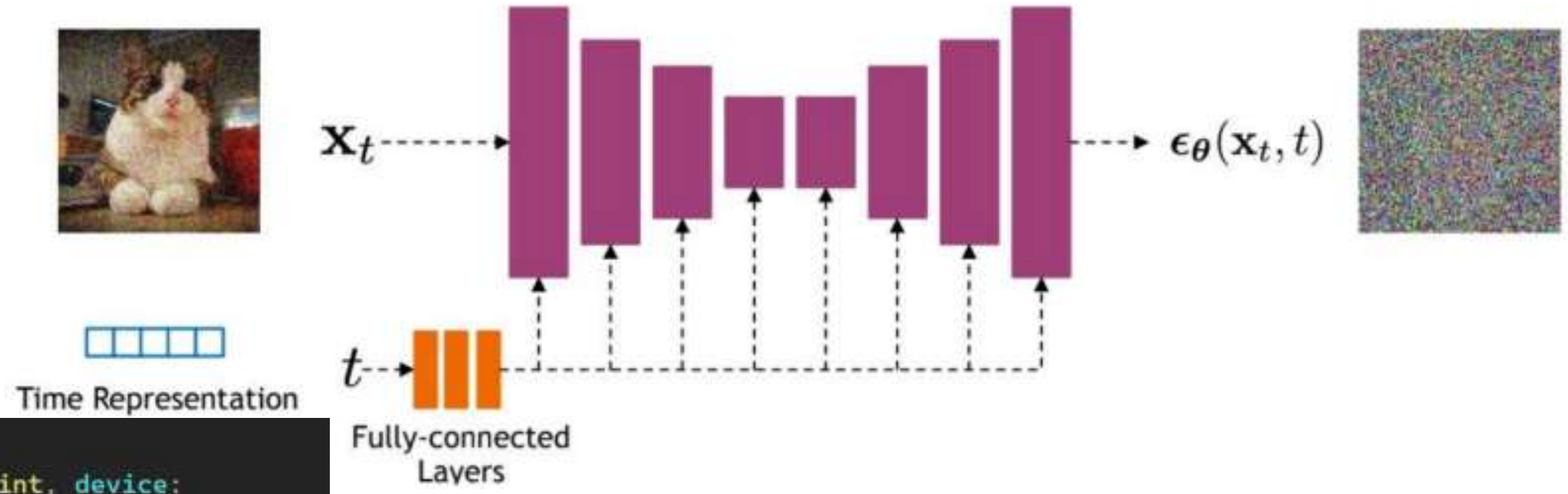
$\alpha_1, \alpha_2, \dots \alpha_T$



02

Code explanation

Train




```python
class DenoiseDiffusion:
    def __init__(self, eps_model: nn.Module, n_steps: int, device:
torch.device):
        """
        Params:
            eps_model: UNet去噪模型
            n_steps: 训练总步数T
            device: 训练所用硬件
        """
        super().__init__()
        # 定义UNet架构模型
        self.eps_model = eps_model
        # 人为设置超参数beta，满足beta随着t的增大而增大，同时将beta搬运到训练硬件上
        self.beta = torch.linspace(0.0001, 0.02, n_steps).to(device)
        # 根据beta计算alpha
        self.alpha = 1. - self.beta
        # 根据alpha计算alpha_bar
        self.alpha_bar = torch.cumprod(self.alpha, dim=0)
        # 定义训练总步长
        self.n_steps = n_steps
        # sampling中的sigma_t
        self.sigma2 = self.beta
```

$$\alpha_t := 1 - \beta_t$$

$$\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

https://zhuanlan.zhihu.com/p/655568910

Train

```python
def train(self):
    """
    单epoch训练DDPM
    """
    # 遍历每一个batch
    for data in monit.iterate('Train', self.data_loader):
        # step数+1（tracker是自定义类，详情参见github完整代码）
        tracker.add_global_step()
        data = data.to(self.device)
        # 每个batch开始时，梯度清0
        self.optimizer.zero_grad()
        # self.diffusion即为DenoiseModel实例，执行forward，计算loss
        loss = self.diffusion.loss(data)
        # 计算梯度
        loss.backward()
        # 更新
        self.optimizer.step()
        # 保存loss，用于后续可视化之类的操作
        tracker.save('loss', loss)
```

Algorithm 1 Training

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

Train

```python
    def loss(self, x0: torch.Tensor, noise: Optional[torch.Tensor] = None):
        """
        Params:
            x0: 来自训练数据的干净的图片
            noise: diffusion process中随机抽样的噪声epsilon~N(0, I)
        Return:
            loss: 真实噪声和预测噪声之间的loss
        """

        batch_size = x0.shape[0]
        # 随机抽样t
        t = torch.randint(0, self.n_steps, (batch_size,), device=x0.device,
dtype=torch.long)

        # 如果为传入噪声，则从N(0, I)中抽样噪声
        if noise is None:
            noise = torch.randn_like(x0)

        # 执行Diffusion process，计算xt
        xt = self.q_sample(x0, t, eps=noise)
        # 执行Denoise Process，得到预测的噪声epsilon_theta
        eps_theta = self.eps_model(xt, t)

        # 返回真实噪声和预测噪声之间的mse loss
        return F.mse_loss(noise, eps_theta)
```

Algorithm 1 Training

1: **repeat**

2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$

3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$

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

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

Test

```python
    def q_xt_x0(self, x0: torch.Tensor, t: torch.Tensor) ->
Tuple[torch.Tensor, torch.Tensor]:
        """
        Diffusion Process的中间步骤，根据x0和t，推导出xt所服从的高斯分布的mean和var
        Params:
            x0: 来自训练数据的干净的图片
            t: 某一步time_step
        Return:
            mean: xt所服从的高斯分布的均值
            var: xt所服从的高斯分布的方差
        """
        mean = gather(self.alpha_bar, t) ** 0.5 * x0
        var = 1 - gather(self.alpha_bar, t)

        return mean, var

    def q_sample(self, x0: torch.Tensor, t: torch.Tensor, eps:
Optional[torch.Tensor] = None):
        """
        Diffusion Process，根据xt所服从的高斯分布的mean和var，求出xt
        Params:
            x0: 来自训练数据的干净的图片
            t: 某一步time_step
        Return:
            xt: 第t时刻加完噪声的图片
        """
        if eps is None:
            eps = torch.randn_like(x0)

        mean, var = self.q_xt_x0(x0, t)
        return mean + (var ** 0.5) * eps
```

Algorithm 1 Training

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

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

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$$

Sampling

```python
def sample(self):
    """
    利用当前模型，将一张随机高斯噪声(xt)逐步还原回x0,
    x0将用于评估模型效果（例如FID分数）
    """
    with torch.no_grad():
        # 随机抽取n_samples张纯高斯噪声
        x = torch.randn([self.n_samples, self.image_channels,
self.image_size, self.image_size],
                        device=self.device)

        # 对每一张噪声，按照sample公式，还原回x0
        for t_ in monit.iterate('Sample', self.n_steps):
            t = self.n_steps - t_ - 1
            x = self.diffusion.p_sample(x, x.new_full((self.n_samples,), t,
dtype=torch.long))

        # 保存x0
        tracker.save('sample', x)
```

Algorithm 2 Sampling

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

Sampling

```python
def p_sample(self, xt: torch.Tensor, t: torch.Tensor):
    """
    Sampling, 当模型训练好之后，根据x_t和t，推出x_{t-1}
    Params:
        x_t: t时刻的图片
        t: 某一步time_step
    Return:
        x_{t-1}: 第t-1时刻的图片
    """

    # eps_model: 训练好的UNet去噪模型
    # eps_theta: 用训练好的UNet去噪模型，预测第t步的噪声
    eps_theta = self.eps_model(xt, t)

    # 根据Sampling提供的公式，推导出x_{t-1}
    alpha_bar = gather(self.alpha_bar, t)
    alpha = gather(self.alpha, t)
    eps_coef = (1 - alpha) / (1 - alpha_bar) ** .5
    mean = 1 / (alpha ** 0.5) * (xt - eps_coef * eps_theta)
    var = gather(self.sigma2, t)
    eps = torch.randn(xt.shape, device=xt.device)

    return mean + (var ** .5) * eps
```

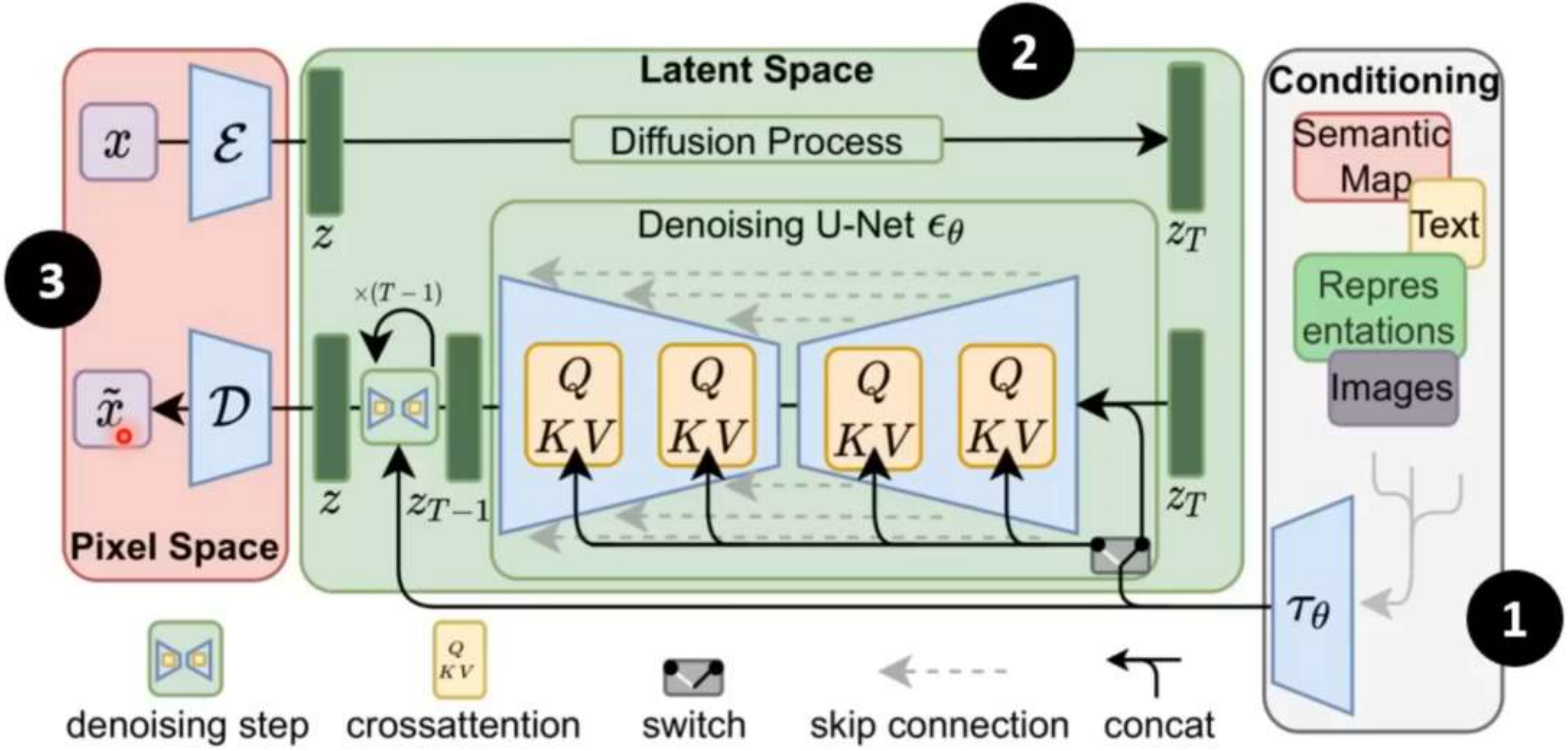
Algorithm 2 Sampling

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

$$\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$$

Stable Diffusion



THANKS