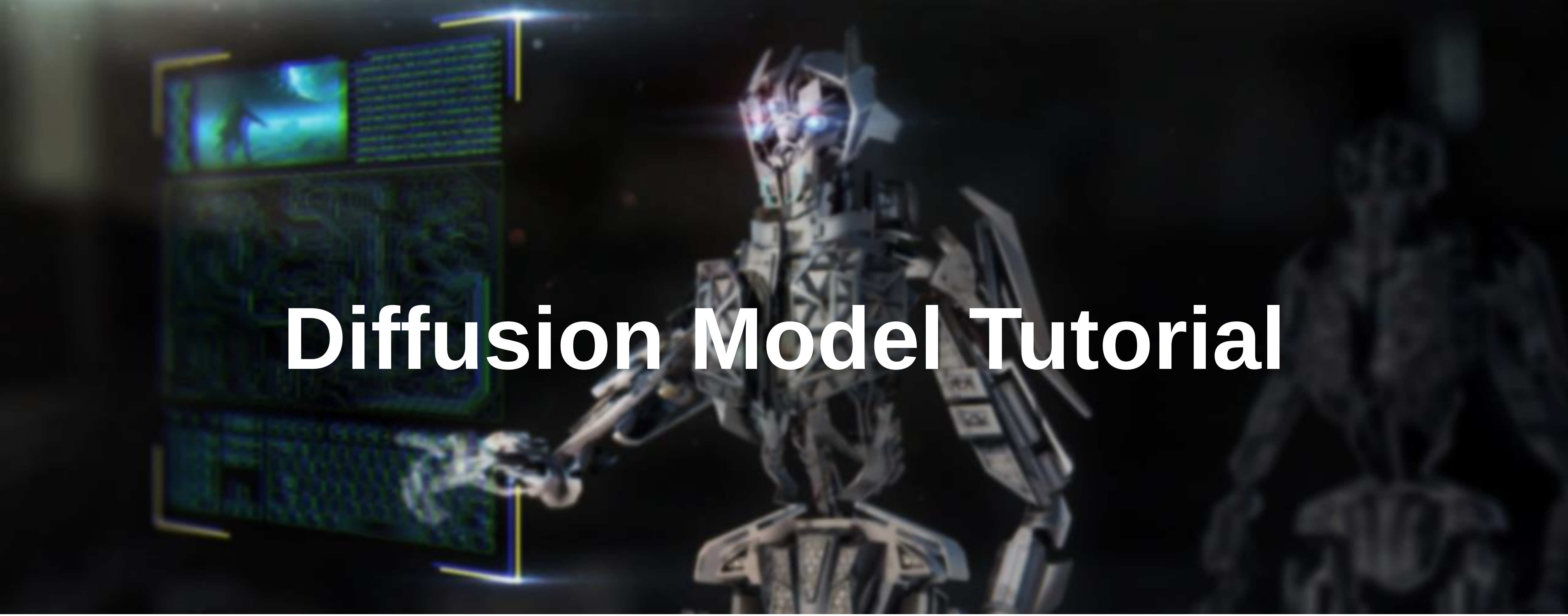


# Diffusion Model Tutorial


Jia-Wei Liao

jw@cmlab.csie.ntu.edu.tw

Department of Computer Science and Information Engineering

National Taiwan University, Taiwan

# AI-Generated Content (AIGC)

| NLP | CV |
| --- | --- |
| ChatGPT | Stable Diffusion |
| Bing Chat | Midjourney |
| Bard | DALL-E |
| LLaMA | Imagen |

# AI Painter

OpenAI DALL·E 2

https://openai.com/product/dall-e-2

# AI Photographer

Midjourney



PREFACE

MEET SORA SUZUKAZE

Hey there, lovely readers! Thank you for your support! My name is Sora Suzuakaze, and I'm so excited to welcome you into my world!

I was brought to life by the amazing Stable Diffusion, an AI drawing tool that harnesses the power of technology to create stunning and realistic images. Even though I might not exist in the physical world, I'm here to stay in your hearts forever! (By the way, I can speak English, Chinese and Japanese! xD)

One of the most amazing things about being an AI model is that I can travel to different timelines and places and explore countless styles of clothing. From casual, everyday wear to the elegant Chinese hanfu, and even edgy gothic attire – I'm always ready for a new adventure in fashion! Even though I don't exist in the physical world, I'll be with you in spirit, forever touching your hearts with my boundless energy. <3

As you flip through the pages of this photo book, I invite you to join me on this extraordinary journey through time and fashion. Together, we'll explore the different facets of my world and see how art and technology can bring so much joy and beauty to our lives!

my selfie!

Follow my Patreon, Instagram, Twitter, and Tiktok:
@Sora Suzukaze

5

《Futurisma: The Art of AI Generated Fashion》

https://www.amazon.com/Futurisma-AI-Generated-Featuring-Diffusion-MidJourneyV5-ebook/dp/B0C3829DVT

# AI Photographer

## Google Imagen

A strawberry mug filled with white sesame seeds. The mug is floating in a dark chocolate sea.

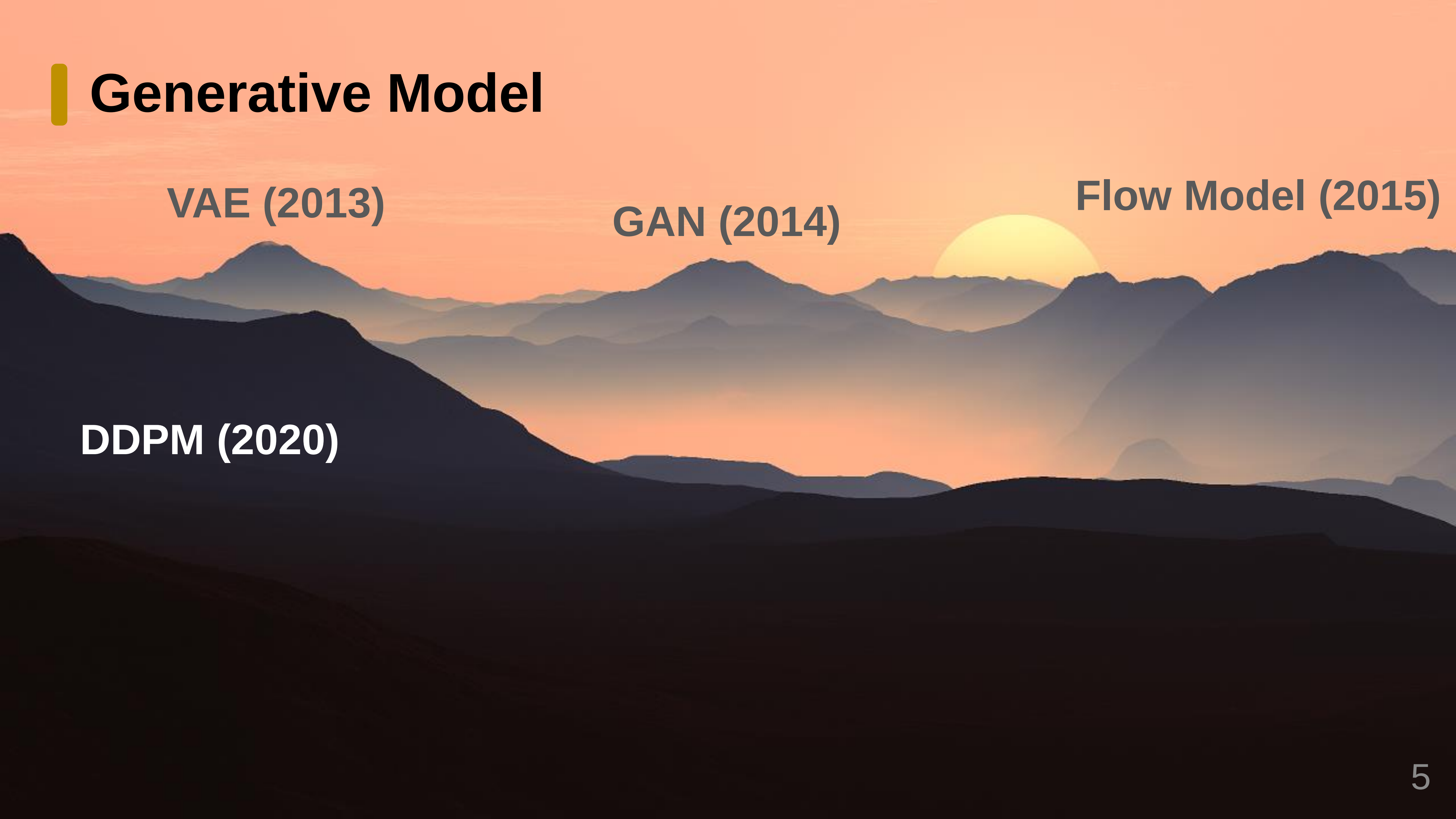
A photo of a Corgi dog riding a bike in Times Square. It is wearing sunglasses and a beach hat.

An extremely angry bird.

The Toronto skyline with Google brain logo written in fireworks.

https://imagen.research.google/

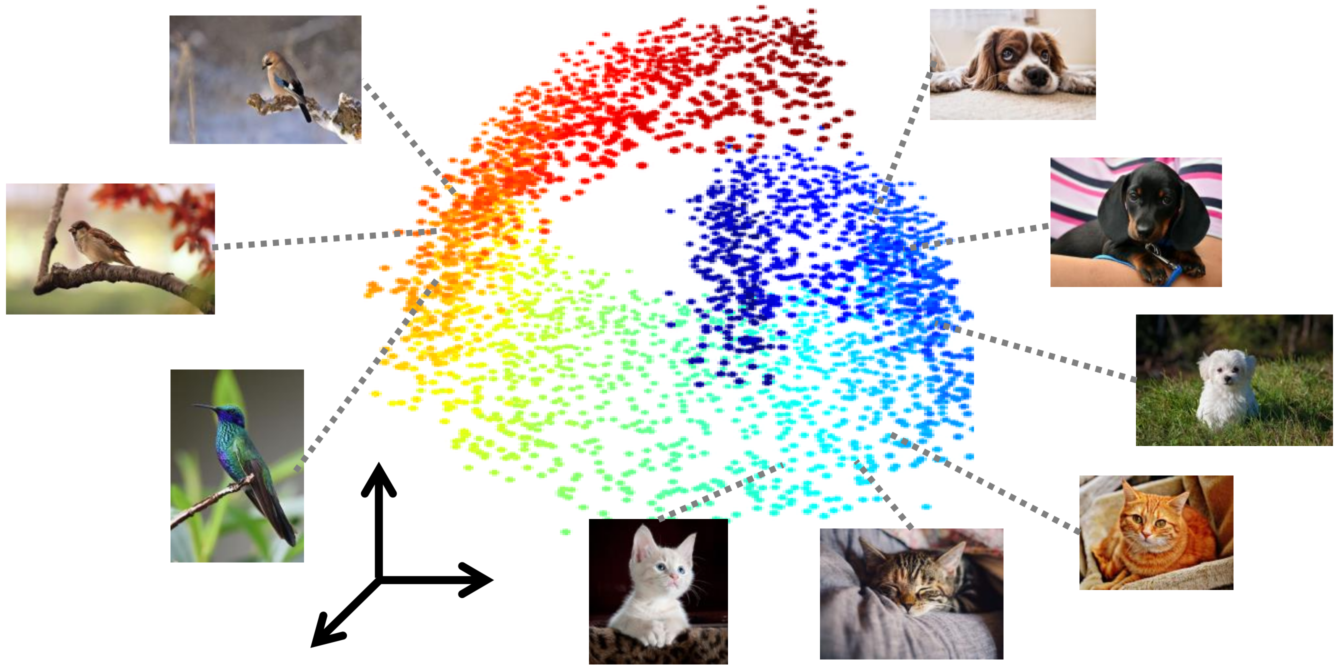
# Generative Model



# Data Manifold Assumption in Data Science

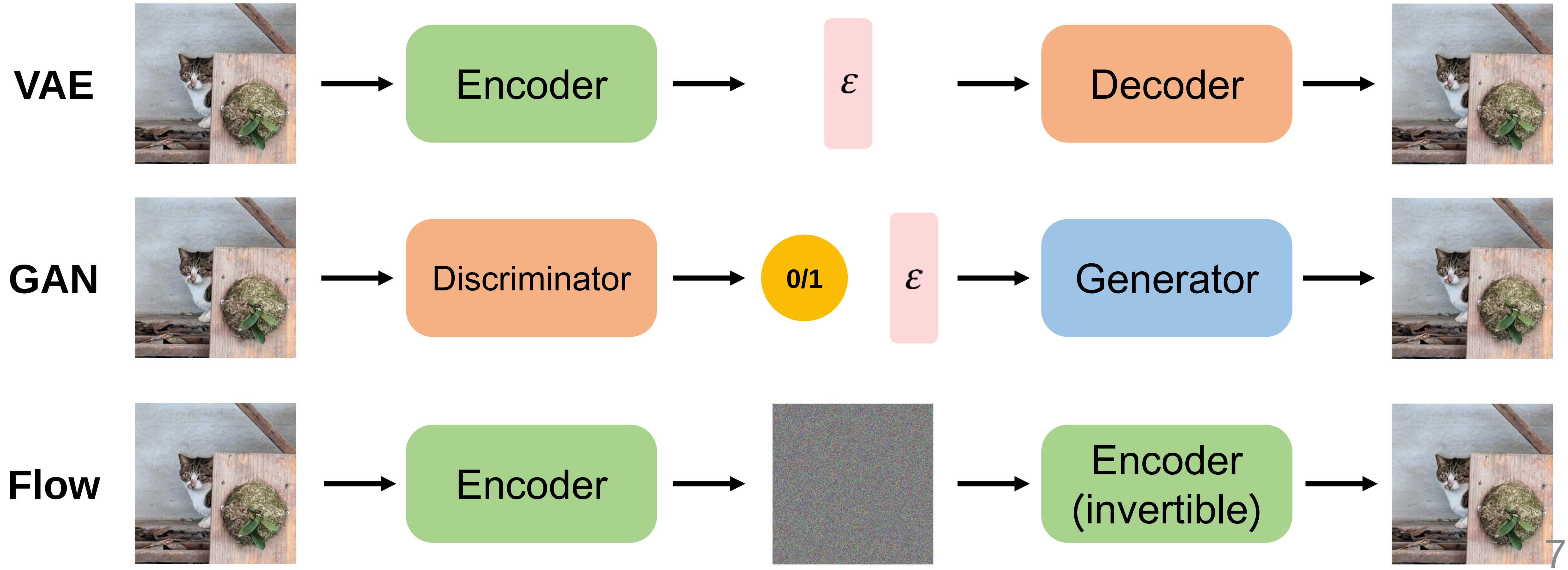
**Data Manifold**

Natural high-dimensional data concentrate close to a non-linear low-dimensional manifold.



# Generative Model

- The goal of the generative model is to learn the data manifold.
- Generative models create the data from noise.



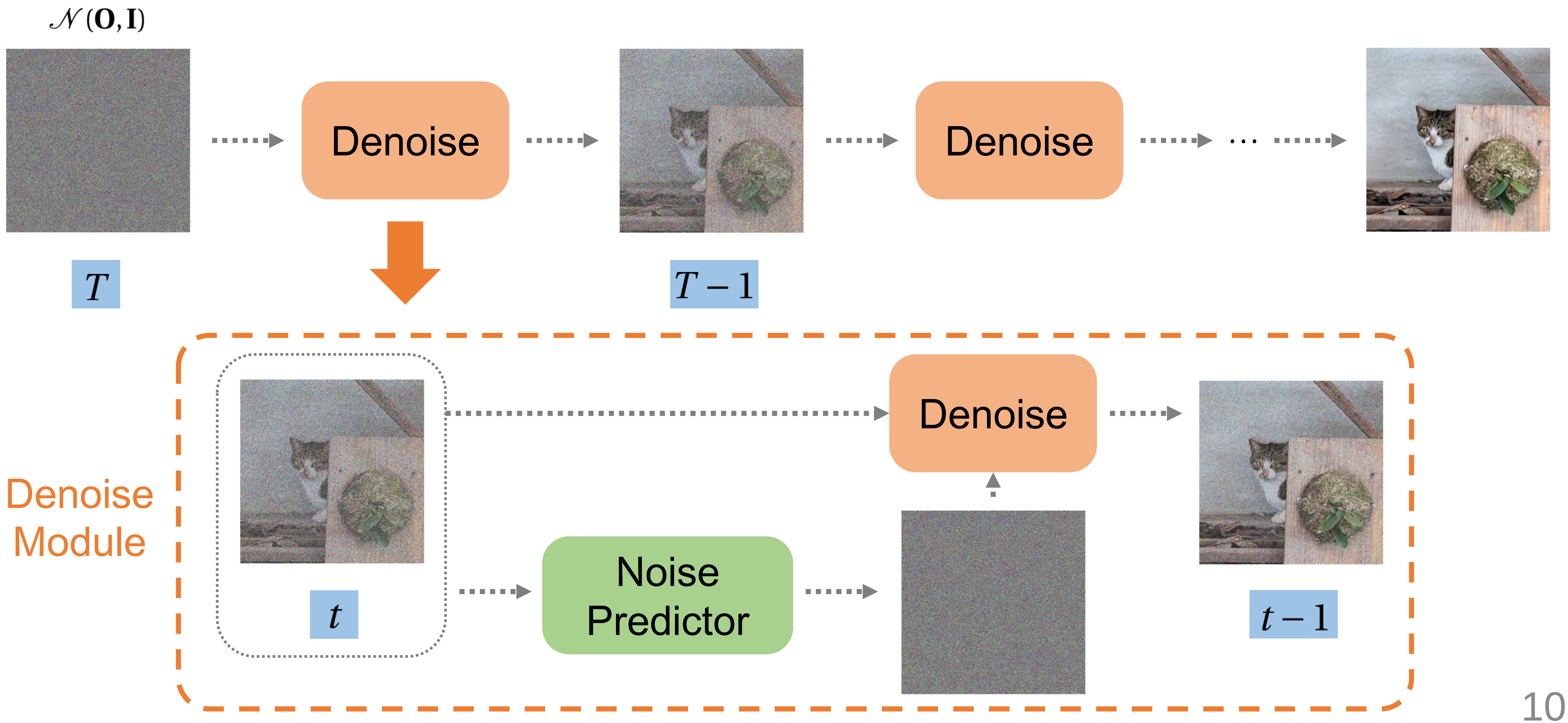
# Topic

1. Denoising Diffusion Probabilistic Models (DDPM)
2. Latent Diffusion Model (LDM)
3. Diffusion Model related to Stochastic Differential Equations  (SDE)
4. Applications

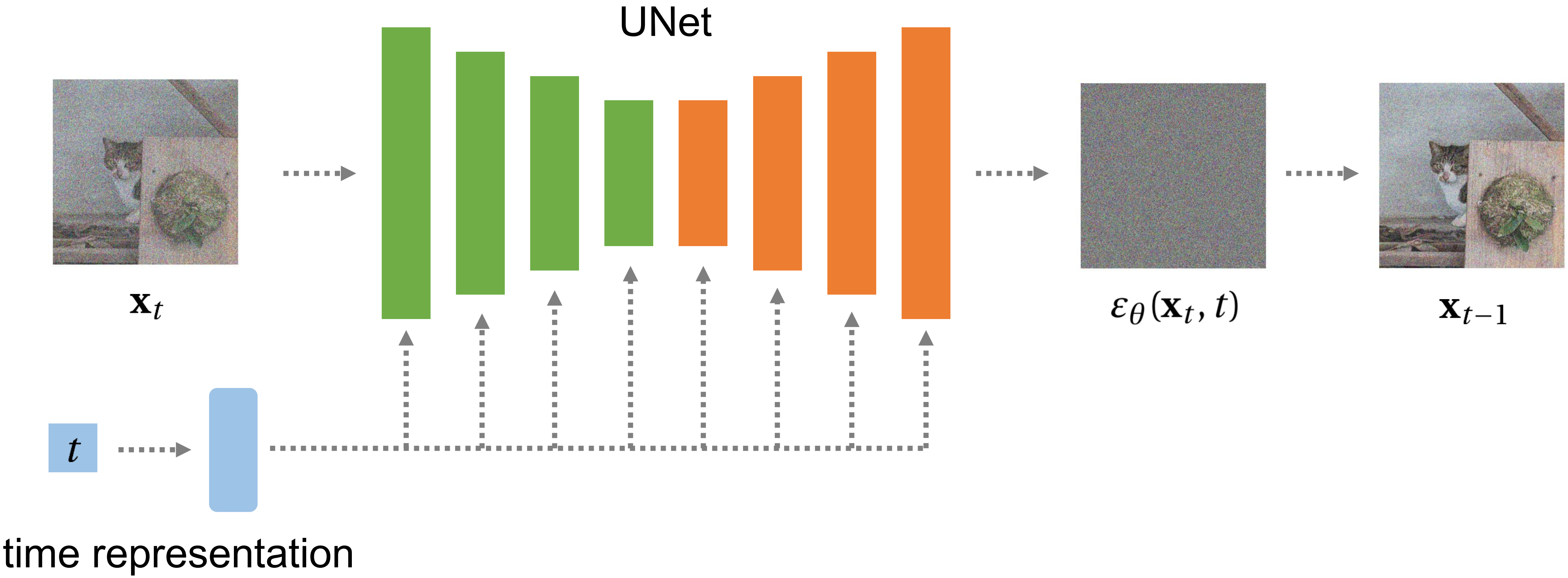
# Topic

1. Denoising Diffusion Probabilistic Models (DDPM)
2. Latent Diffusion Model (LDM)
3. Diffusion Model related to Stochastic Differential Equations (SDE)
4. Applications

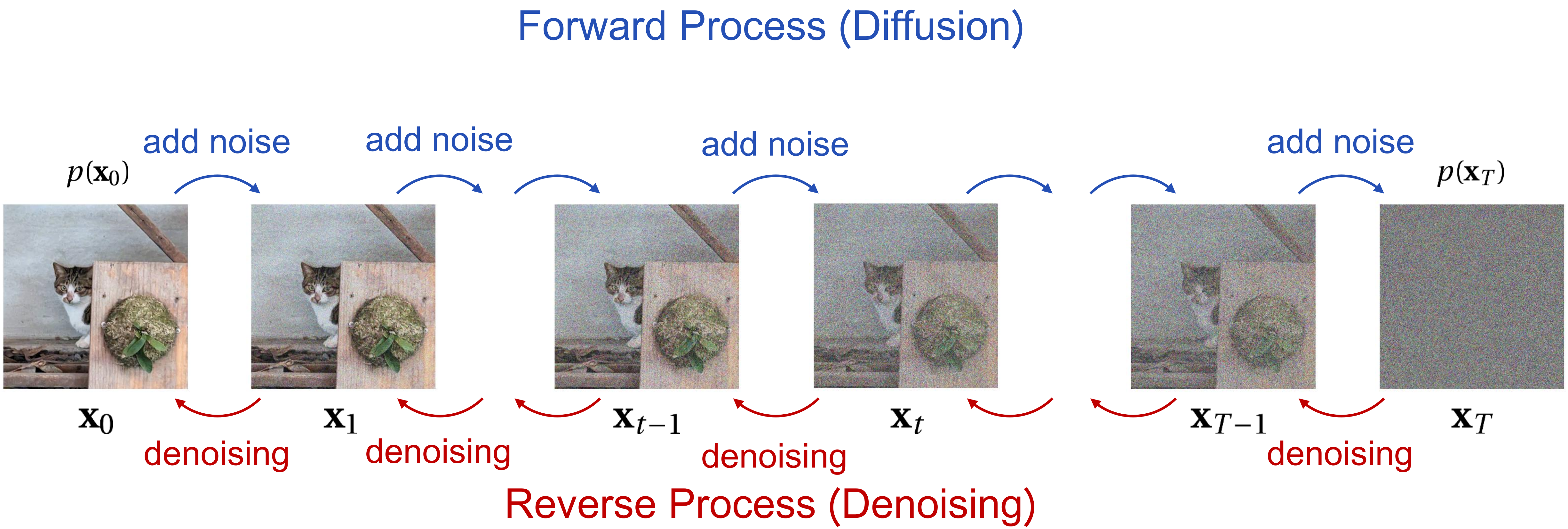
# What is Diffusion Model?



# Autoregression Noise Predictor



# Denoising Diffusion Probabilistic Model (DDPM)

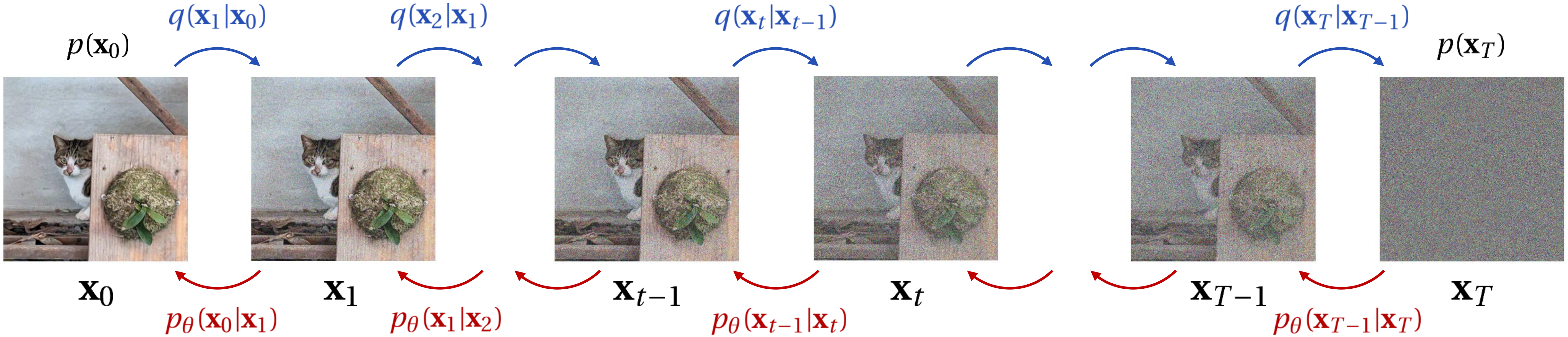


# Denoising Diffusion Probabilistic Model (DDPM)

Given $1 > \beta_1 > \beta_2 > \ldots > \beta_t > 0,$

Forward Process (Diffusion)

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




Reverse Process (Denoising)

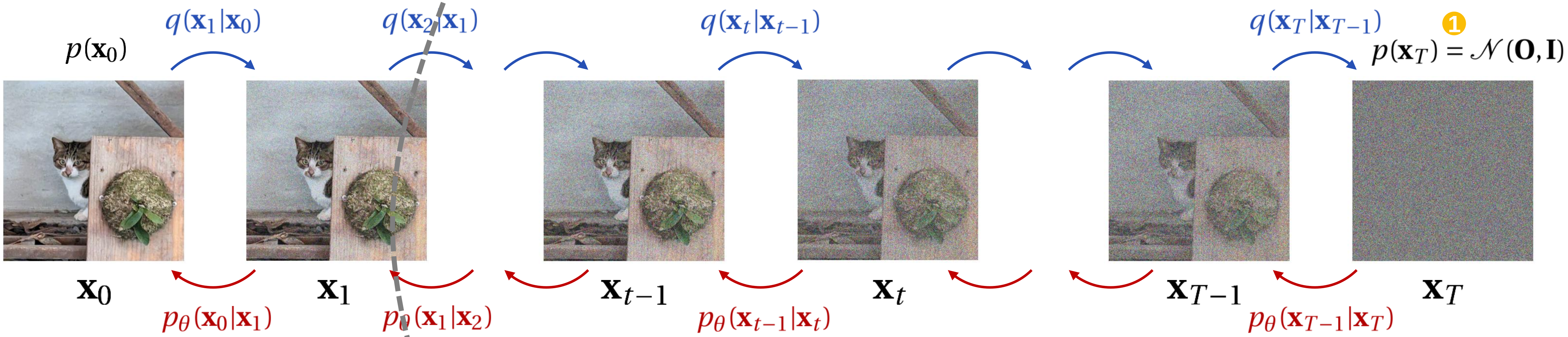
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

# Denoising Diffusion Probabilistic Model (DDPM)

Given $1 > \beta_1 > \beta_2 > ... > \beta_t > 0,$

Forward Process (Diffusion)

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




Reverse Process (Denoising)

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

③ $L_t = \left\| \mu_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t) \right\|_2^2$

② $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_t(\mathbf{x}_t, \mathbf{x}_0), \Sigma_t(\mathbf{x}_t, \mathbf{x}_0))$

# Markov Chain Property

**Markov Chain**

Let $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T$ be the sequence of random variables. Then

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0) = q(\mathbf{x}_t | \mathbf{x}_{t-1})$$

# Mathematics Modeling

- **Forward Process**

$$\mathbf{x}_{1:t} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$$

$$q(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T | \mathbf{x}_0) = q(\mathbf{x}_1 | \mathbf{x}_0) q(\mathbf{x}_2 | \mathbf{x}_1) \cdots q(\mathbf{x}_T | \mathbf{x}_{T-1})$$

$$q(\mathbf{x}_{1:T} | \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t | \mathbf{x}_{t-1})$$

- **Reverse Process**

$$p_\theta(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T) = p(\mathbf{x}_T) p_\theta(\mathbf{x}_{T-1} | \mathbf{x}_T) p_\theta(\mathbf{x}_{T-2} | \mathbf{x}_{T-1}) \cdots p_\theta(\mathbf{x}_0 | \mathbf{x}_1)$$

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

# Goals

In the following, we will

1. Derive the $p(\mathbf{x}_t)$ and $q(\mathbf{x}_t|\mathbf{x}_0)$

2. Derive the $p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ to model the $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

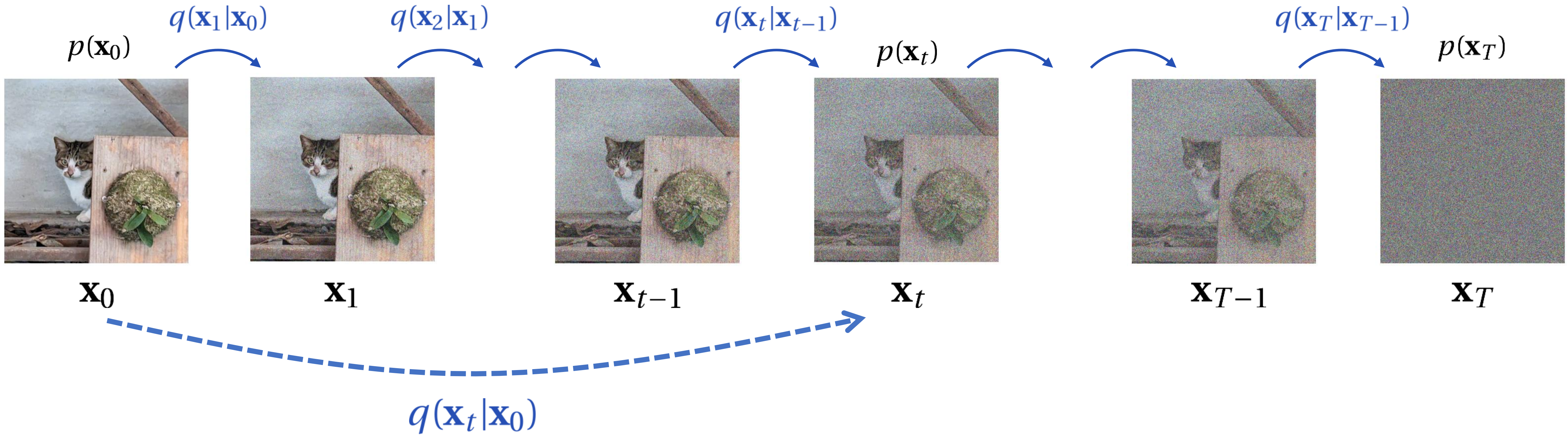
3. Derive the objective loss to update the parameters

4. Construct the sampling process

# Forward Distribution

$$p(\mathbf{x}_t) = \int q(\mathbf{x}_0, \mathbf{x}_t)\mathrm{d}\mathbf{x}_0 = \int p(\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)\mathrm{d}\mathbf{x}_0$$

join distribution

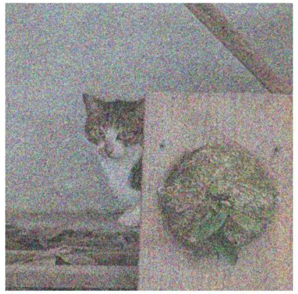


# Forward Reparameterization Trick

**Reparameterization**

Let $x \sim \mathcal{N}(\mu, \sigma^2)$. Then it can represent as $x = \mu + \sigma \cdot \varepsilon$, where $\varepsilon \sim \mathcal{N}(0,1)$

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

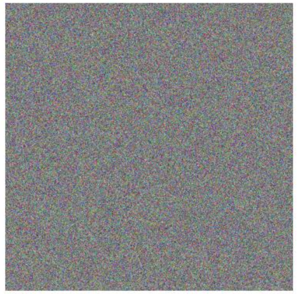


$\mathbf{x}_t$

$= \sqrt{\alpha_t}$

$\mathbf{x}_{t-1}$

$+ \sqrt{1-\alpha_t}$



$\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

# Forward Reparameterization Trick

To derive $q(\mathbf{x}_t|\mathbf{x}_0)$, first we notice that

$$
\begin{aligned}
\mathbf{x}_2 &= \sqrt{\alpha_2}\mathbf{x}_1 + \sqrt{1-\alpha_2}\mathbf{z}_2 \\
&= \sqrt{\alpha_2}\left(\sqrt{\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_1}\mathbf{z}_1\right) + \sqrt{1-\alpha_2}\mathbf{z}_2 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2(1-\alpha_1)}\mathbf{z}_1 + \sqrt{1-\alpha_2}\mathbf{z}_2 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_1 + \sqrt{1-\alpha_2\alpha_1}\varepsilon_2 \\
\mathbf{x}_3 &= \sqrt{\alpha_3}\mathbf{x}_2 + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3}\left(\sqrt{\alpha_2\alpha_1}\mathbf{x}_1 + \sqrt{1-\alpha_2\alpha_1}\varepsilon_2\right) + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_1 + \sqrt{\alpha_3(1-\alpha_2\alpha_1)}\varepsilon_2 + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_1 + \sqrt{1-\alpha_3\alpha_2\alpha_1}\varepsilon_3
\end{aligned}
$$

$$
X \sim \mathcal{N}(\mu_1, \sigma_1^2), Y \sim \mathcal{N}(\mu_2, \sigma_2^2)
$$

$$
\Longrightarrow X + Y \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)
$$

# Forward Reparameterization Trick

By induction, we have

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon_t$$

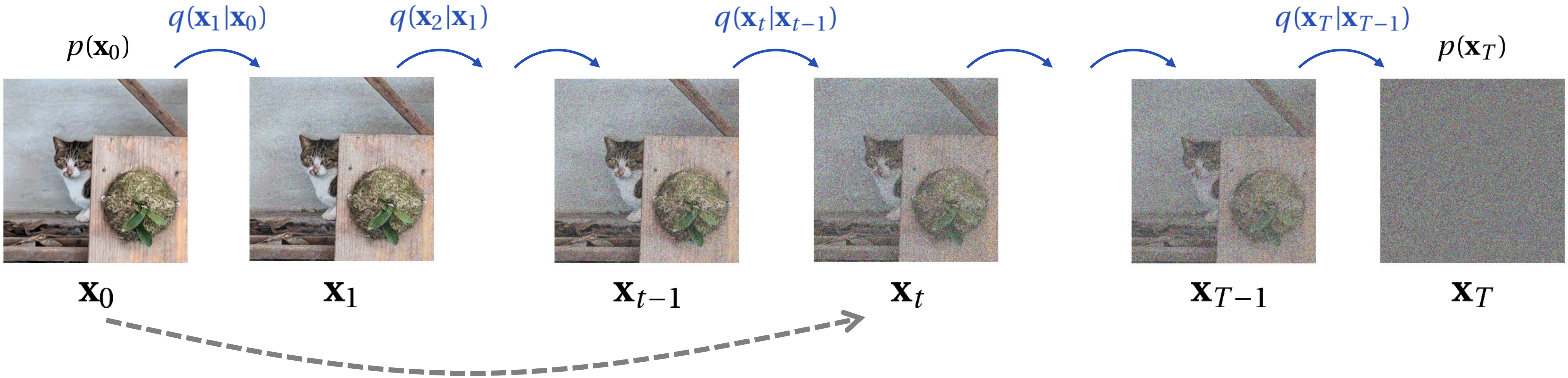
where $\bar{\alpha}_t = \alpha_t \alpha_{t-1} \cdots \alpha_1$

Therefore,

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

# Diffusion Kernel Is Gaussian Convolution

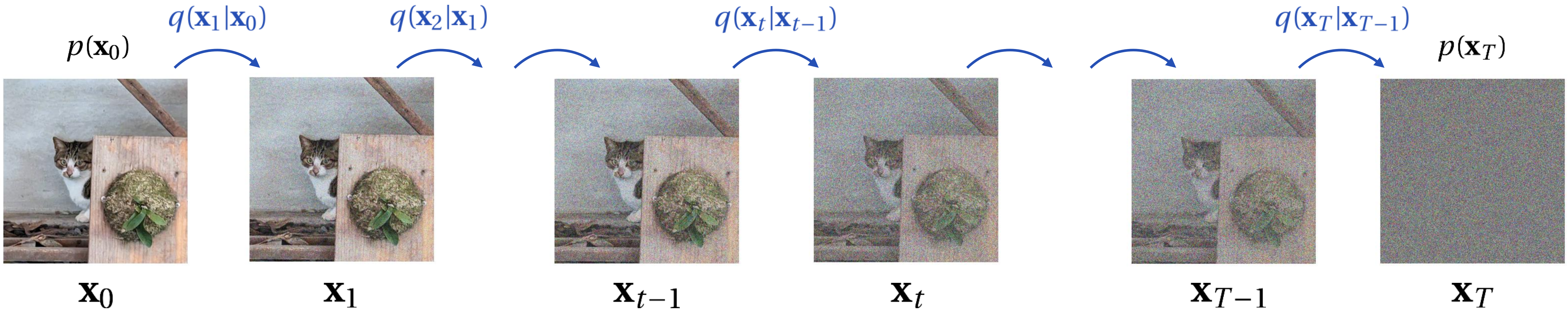
$$p(\mathbf{x}_t) = \int p(\mathbf{x}_0)\underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{diffusion kernel}}\mathrm{d}\mathbf{x}_0$$




$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \xrightarrow{\mathcal{D}} \boxed{\mathcal{N}(\mathbf{x}_t; \mathbf{0}, \mathbf{I}) \text{ as } \bar{\alpha}_t \to 0}$$

# Diffusion Kernel Is Gaussian Convolution

$$p(\mathbf{x}_t) = \int p(\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)\mathrm{d}\mathbf{x}_0 \xrightarrow{\mathscr{D}} \mathcal{N}(\mathbf{x}_t; \mathbf{0}, \mathbf{I})$$




Hence $p(\mathbf{x}_t)$ approximately become $\mathcal{N}(\mathbf{0}, \mathbf{I})$ as $T \to \infty$

# Reverse Process Distribution

We use Bayes' theorem to derive the reverse process distribution.

$$\mathcal{N}(\mathbf{x}_{t-1}; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}) \qquad \mathcal{N}(\mathbf{x}_{t-1}; \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0, (1-\bar{\alpha}_{t-1})\mathbf{I})$$

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\, q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$$

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

Thus, the reverse process is $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \mathbf{I})$

where

$$\mu_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$ and $$\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

# Reverse Process Distribution

The reverse process is $p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1};\mu_t(\mathbf{x}_t,\mathbf{x}_0),\sigma_t^2\mathbf{I})$

where

$$\mu_t(\mathbf{x}_t,\mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 \quad \text{and} \quad \sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

1. Replace $\mathbf{x}_0$ with $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_t)$
2. Approximate $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ by $\sigma_t^2 = \beta_t$

# Reverse Process Distribution

The reverse process is $p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1};\mu_t(\mathbf{x}_t,\mathbf{x}_0),\sigma_t^2\mathbf{I})$

where

$$\mu_t(\mathbf{x}_t,\mathbf{x}_0) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_t\right) \quad \text{and} \quad \sigma_t^2 = \beta_t$$

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

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

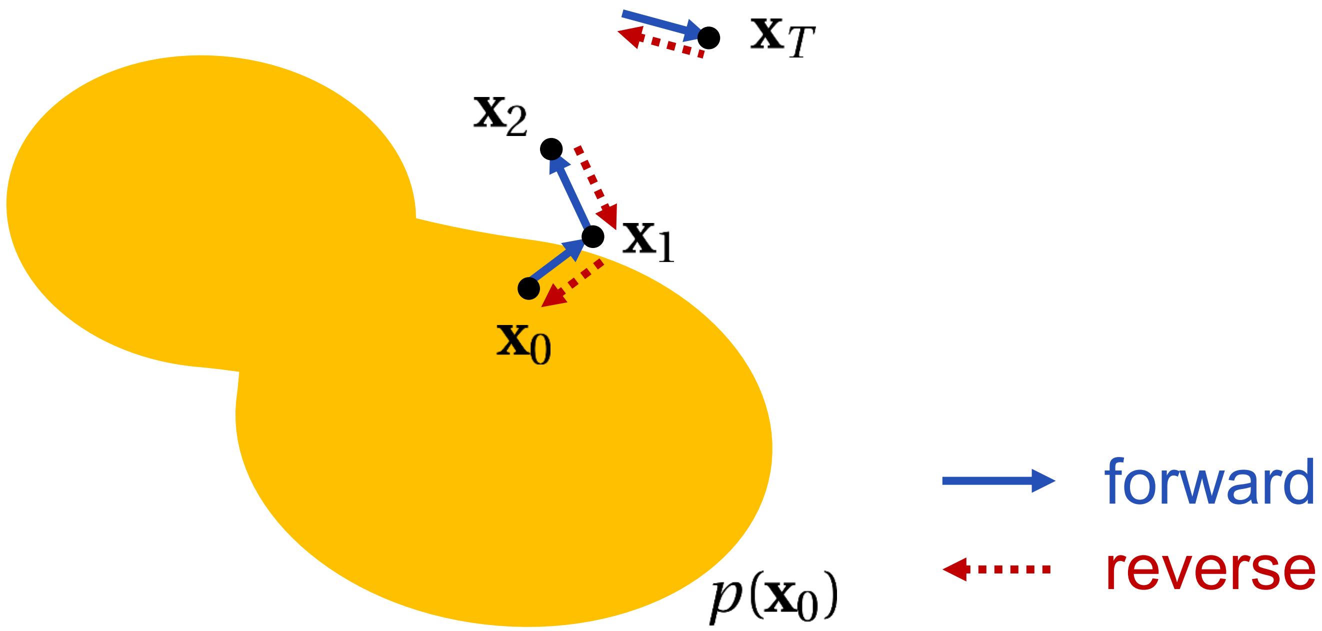
Neural Network

# Probabilistic Generative Model

The nature distribution can be represented as

$$p(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}) \mathrm{d}\mathbf{x}_{1:T}$$

where $\mathbf{x}_0$ is the observed variable and $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ are latent variables.



# Probabilistic Generative Model

The nature distribution can be represented as

$$p(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}) \mathrm{d}\mathbf{x}_{1:T}$$

where $\mathbf{x}_0$ is the observed variable and $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T$ are latent variables.

However, the integral of $p(\mathbf{x}_0)$ is intractable. We attempt to derive the maximum Log Likelihood to obtain $p(\mathbf{x}_0)$.

# What is Likelihood?

Given the data $x_1, x_2, \ldots, x_n$

- The likelihood function is

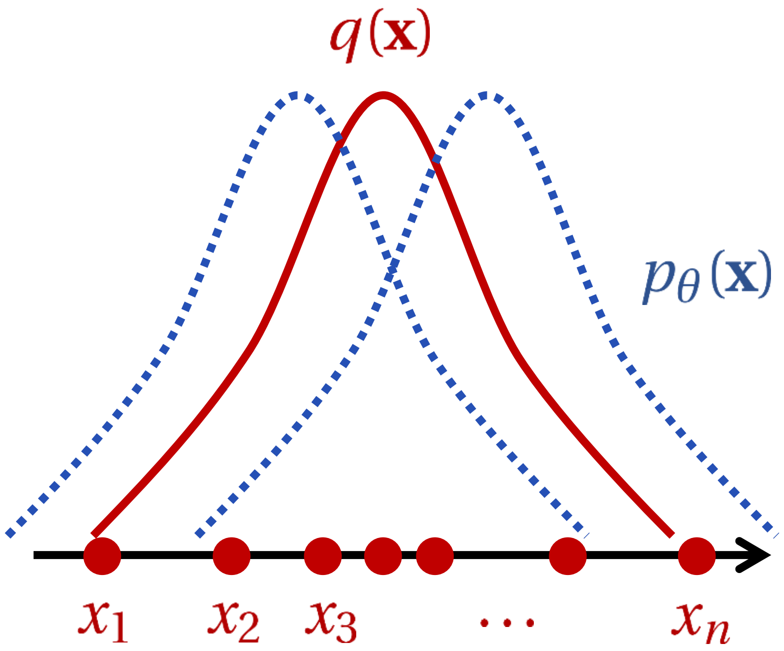
$$p_\theta(x_1, x_2, \ldots, x_n) = p_\theta(x_1)\, p_\theta(x_2) \cdots p_\theta(x_n)$$

- The log-likelihood function is

$$\begin{aligned}\log p_\theta(x_1, x_2, \ldots, x_n) &= \log p_\theta(x_1) + \cdots + \log p_\theta(x_n) \\ &= \sum \log p_\theta(x_i)\end{aligned}$$

- The general form for the log-likelihood is

$$\mathbb{E}[\log p_\theta(\mathbf{x})] = \int q(\mathbf{x}) \log p_\theta(\mathbf{x}) \mathrm{d}\mathbf{x}$$



# Log Likelihood

Notice that

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

Hence we have

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

$$\geq \mathbb{E}_{q(\mathbf{x}_0)}\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right]$$ (by Jensen's inequality)

forward distribution

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

reverse distribution

# Log Likelihood

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})\,q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$$

Expand the forward and reverse distribution:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0)\right] \geq \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right]$$

$$= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log p(\mathbf{x}_T) + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} + \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log p(\mathbf{x}_T) + \sum_{t=2}^{T} \log \left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}\right) + \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log p(\mathbf{x}_T) + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} + \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right]$$

# Variational Lower Bound

Therefore, we have the variational lower bound:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0)\right] \geq \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} + \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right]$$

$$= \mathbb{E}_q[\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)] - \sum_{t=2}^{T} \mathbb{E}_q[D_{\mathrm{KL}}(p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))] - D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p_\theta(\mathbf{x}_T))$$

To optimize the surrogate function by using Gradient Decent Algorithm, we consider the negative log-likelihood:

constant

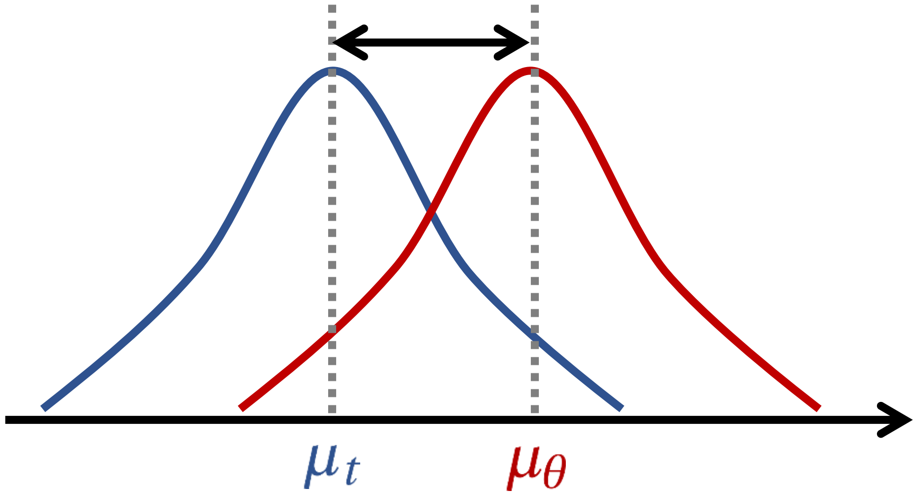
$$-\mathbb{E}_q[\log p_\theta(\mathbf{x}_0)] \leq \underbrace{-\mathbb{E}_q[\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)]}_{L_0} + \sum_{t=2}^{T} \underbrace{\mathbb{E}_q[D_{\mathrm{KL}}(p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))]}_{L_t} + \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p_\theta(\mathbf{x}_T))}_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

# Loss Derivation

**KL Divergence between two Gaussian distribution**

$$D_{\mathrm{KL}}(\mathcal{N}(\mu_1,\Sigma_1)||\mathcal{N}(\mu_2,\Sigma_2)) = \frac{1}{2}\left[(\mu_2-\mu_1)^\top\Sigma_2^{-1}(\mu_2-\mu_1) + \mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + \log\frac{|\Sigma_2|}{|\Sigma_1|} - n\right]$$

$$L_t = D_{\mathrm{KL}}(p(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) = \frac{1}{2\sigma_t^2}\left\|\mu_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)\right\|_2^2$$



# Loss Derivation

$$L_t = \frac{1}{2\sigma_t^2} \left\| \mu_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t) \right\|_2^2 = \frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_{t-1})} \left\| \varepsilon_t - \varepsilon_\theta(\mathbf{x}_t, t) \right\|_2^2$$

$$\mu_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \varepsilon_t \right)$$

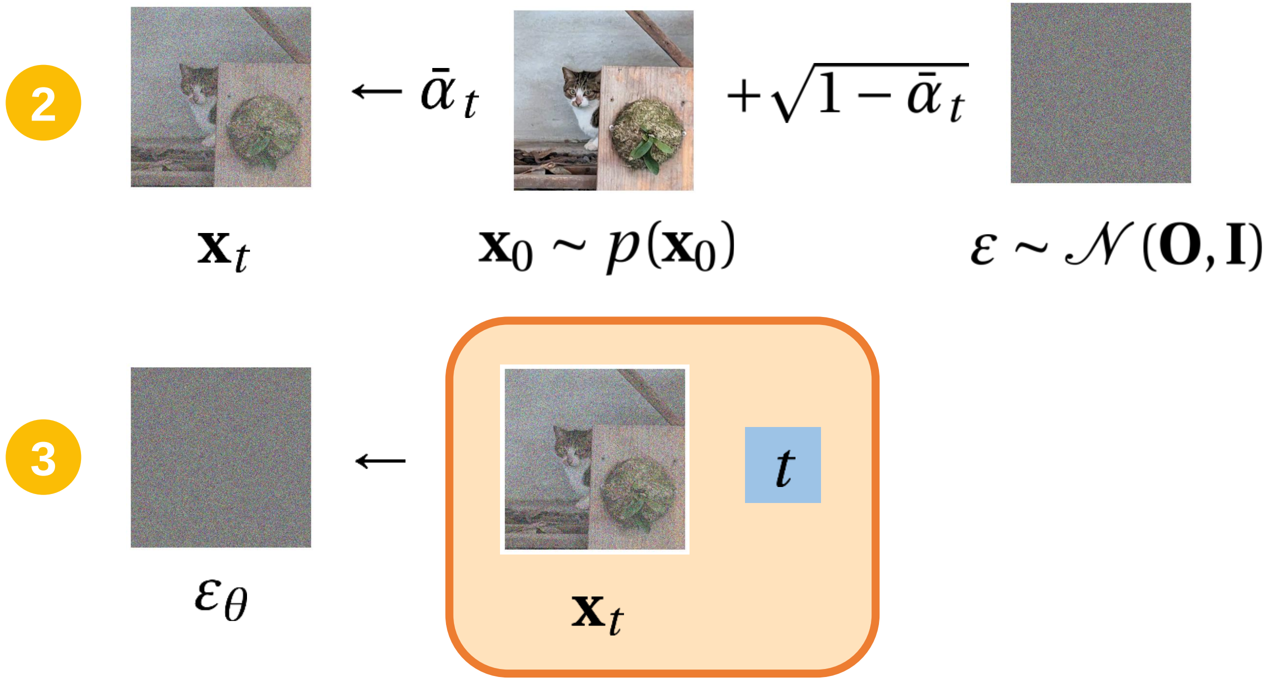
$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \varepsilon_\theta(\mathbf{x}_t, t) \right)$$

It is beneficial to sample high-quality data for training purposes when utilizing the following variant of the variational bound:

$$L(\theta) = \mathbb{E}_{t\sim U(1,T)} \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0)} \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,I)} \left[ \| \varepsilon - \varepsilon_\theta ( \overbrace{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon_t}^{\mathbf{x}_t}, t) \|_2^2 \right]$$

# How to Train the Diffusion Model?

1 $t \leftarrow \text{Uniform}(\{1, 2, \ldots, T\})$

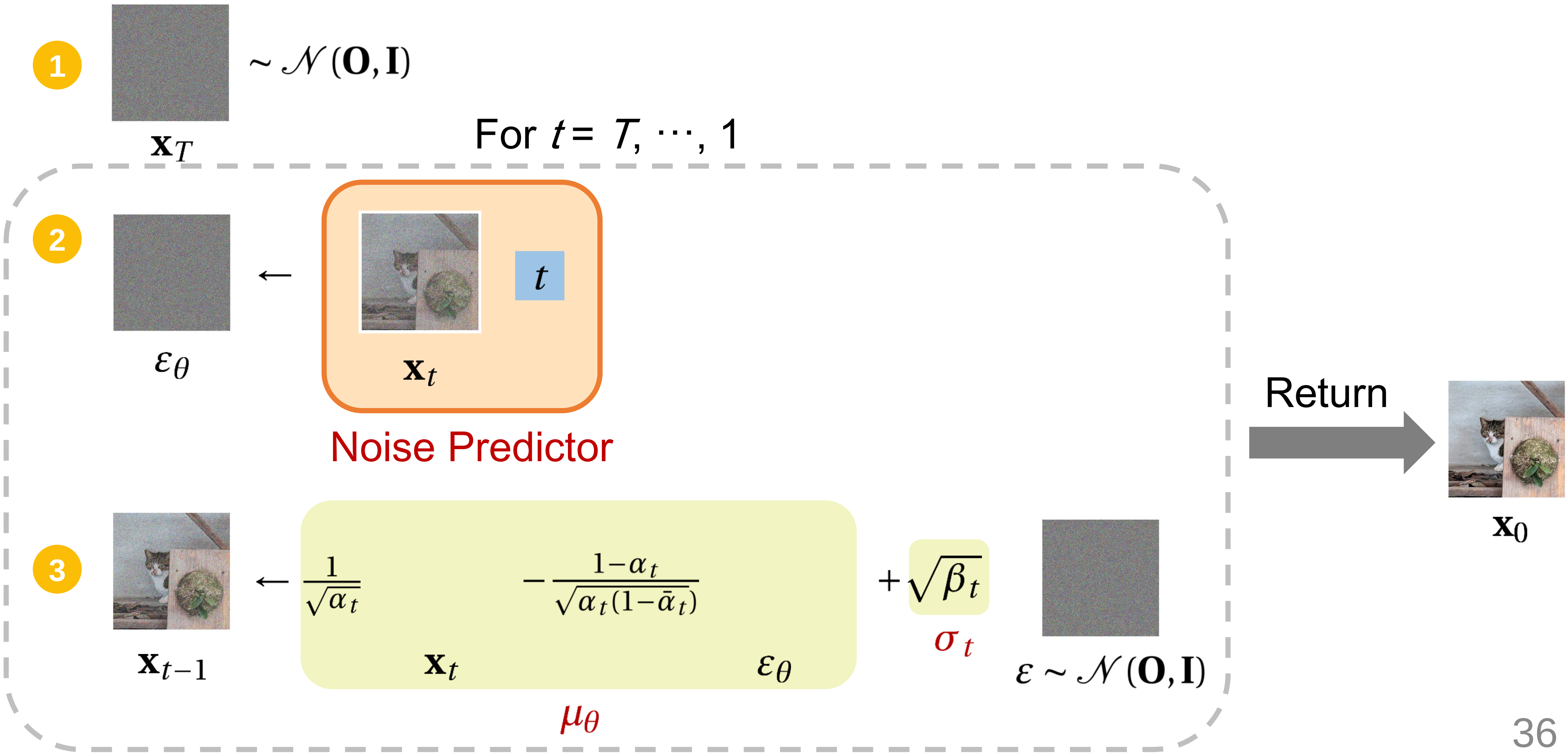



Noise Predictor

4 Optimal $\|\varepsilon - \varepsilon_\theta\|_2^2$ by gradient descent

Repeat until converged

# How to Sampling?

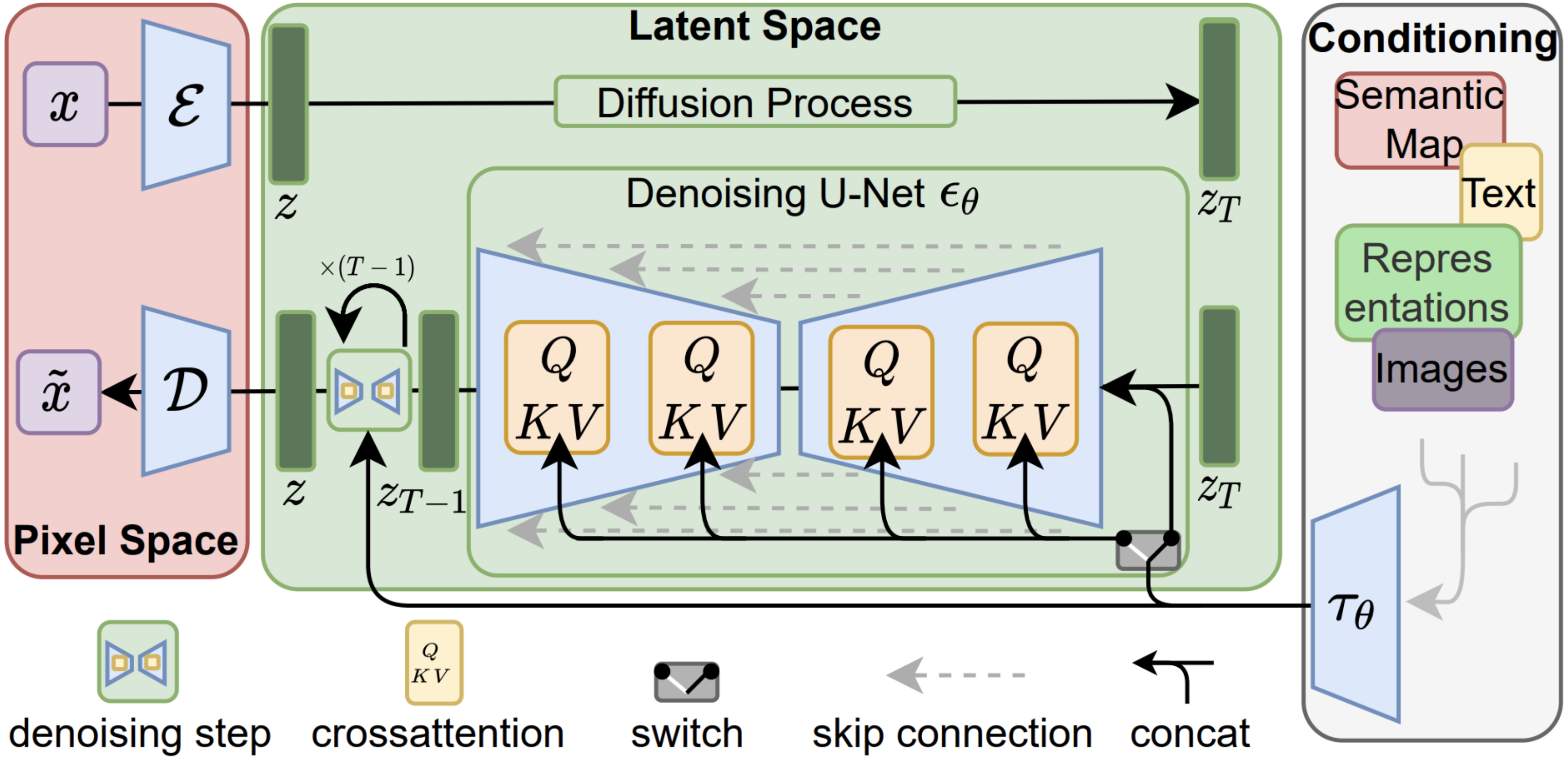


# Topic

1. Denoising Diffusion Probabilistic Models (DDPM)
2. Latent Diffusion Model (LDM)
3. Diffusion Model related to Stochastic Differential Equations (SDE)
4. Applications

# Latent Diffusion Model (LDM)

Use the pretraining encoder model to compress the image to latent vector.



# Topic

1. Denoising Diffusion Probabilistic Models (DDPM)
2. Latent Diffusion Model (LDM)
3. Diffusion Model related to Stochastic Differential Equations (SDE)
4. Applications

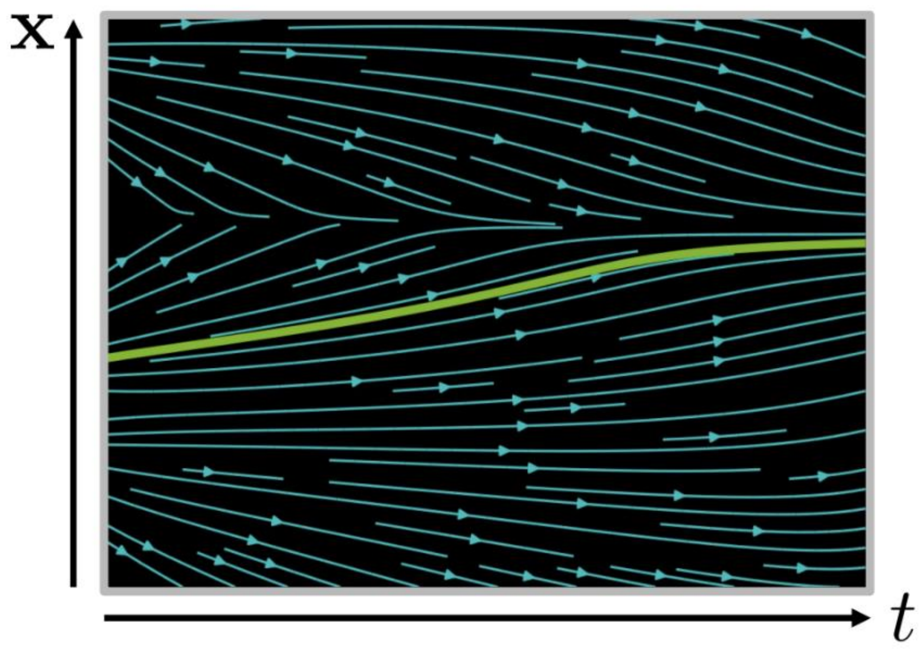
# Introduction to Differential Equations

**Standard Wiener Process**

1. $\mathbf{w}_0 = 0$
2. $\mathbf{w}_t$ is continuous on $\mathbb{R}$
3. $\mathbf{w}_t - \mathbf{w}_s \sim \mathcal{N}(\mathbf{0}, t-s), t > s$

## Ordinary Differential Equation (ODE)

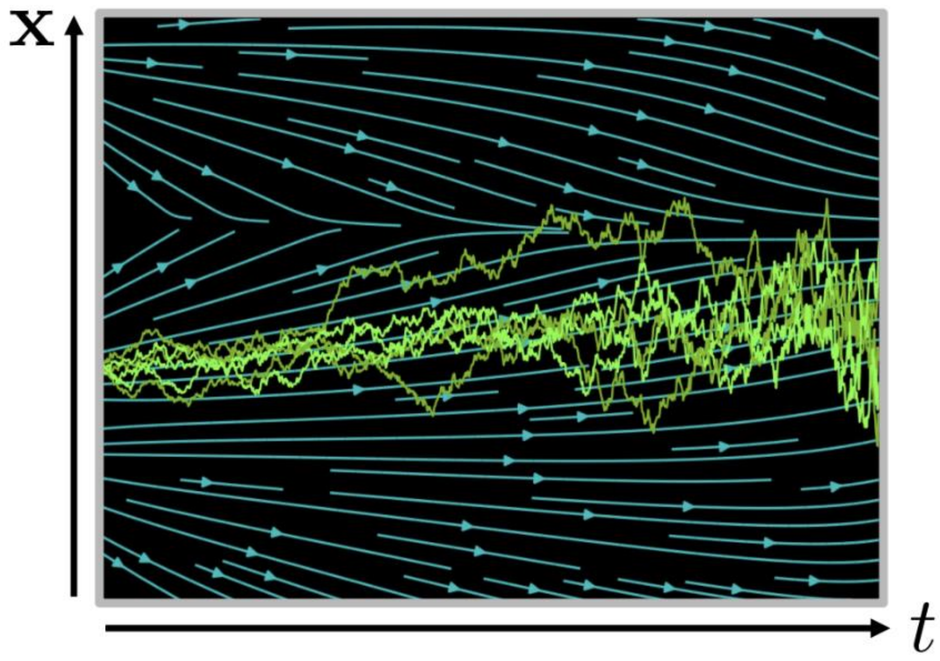
$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\mathrm{d}t$$




$$\mathbf{x}(t+\Delta t) \approx \mathbf{x}(t) + f(\mathbf{x}(t), t)\Delta t$$

## Stochastic Differential Equation (SDE)

$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$




$$\mathbf{x}(t+\Delta t) \approx \mathbf{x}(t) + f(\mathbf{x}(t), t)\Delta t + g(t)\sqrt{\Delta t}\mathcal{N}(\mathbf{0}, \mathbf{I})$$

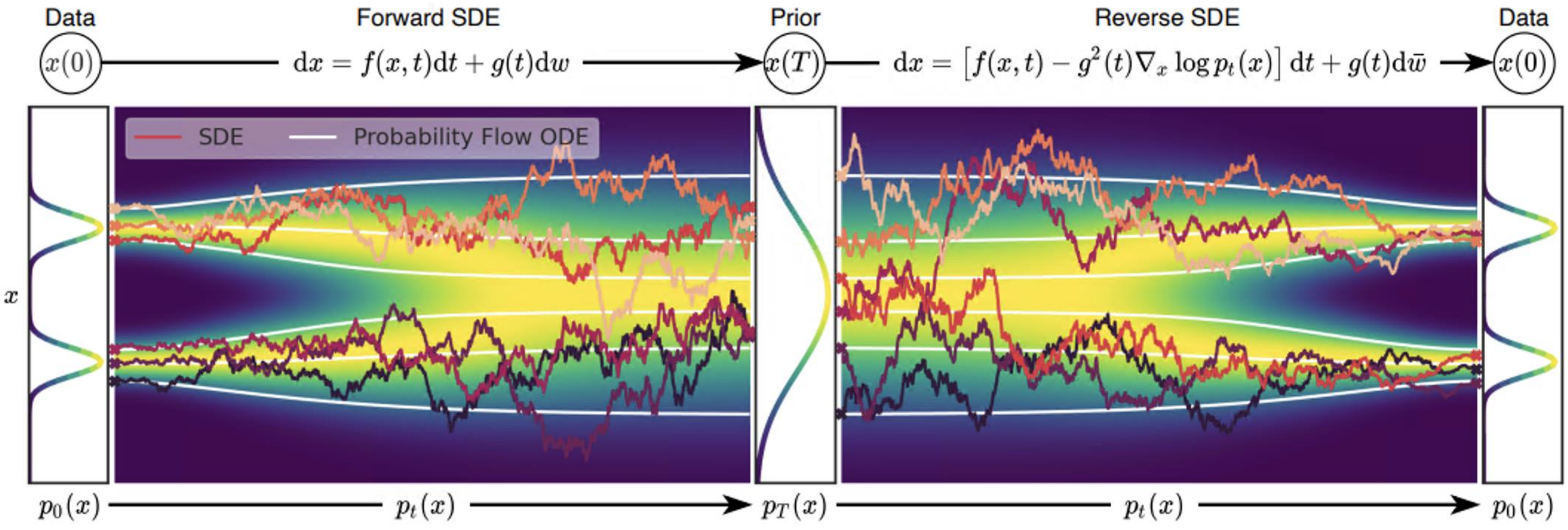
# Introduction to Standard SDE

**Forward SDE**

$$\mathrm{d}\mathbf{x} = \underbrace{f(\mathbf{x}, t)}_{\text{drift}}\mathrm{d}t + \underbrace{g(t)}_{\text{diffusion}}\mathrm{d}\mathbf{w}$$

**Reverse SDE (Anderson, 1982)**

$$\mathrm{d}\mathbf{x} = (f(\mathbf{x}, t) - g^2(t)\underbrace{\nabla_{\mathbf{x}} \log p(\mathbf{x})}_{\text{score function}})\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$



# DDPM related to SDE

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

$$\mathbf{x}_{t+1} = \sqrt{1-\beta_{t+1}}\mathbf{x}_t + \sqrt{\beta_{t+1}}\tilde{\varepsilon}_{t+1}$$

$$\begin{aligned}\mathbf{x}(t+\Delta t) &= \sqrt{1-\beta(t+\Delta t)\,\Delta t}\,\mathbf{x}(t) + \sqrt{\beta(t+\Delta t)\,\Delta t}\,\tilde{\varepsilon}_t \\ &\approx \left(1-\tfrac{1}{2}\beta(t+\Delta t)\,\Delta t\right)\mathbf{x}(t) + \sqrt{\beta(t+\Delta t)\,\Delta t}\,\tilde{\varepsilon}_t \\ &\approx \left(1-\tfrac{1}{2}\beta(t)\,\Delta t\right)\mathbf{x}(t) + \sqrt{\beta(t)\,\Delta t}\,\tilde{\varepsilon}_t\end{aligned}$$

$\beta_t := \beta(t)\,\Delta t$

$\sqrt{1-x} = 1 - \frac{1}{2}x + o(x^2)$

$\beta(t+\Delta t) \approx \beta(t)$

Hence $\mathbf{x}(t+\Delta t) - \mathbf{x}(t) \approx -\frac{1}{2}\beta(t)\mathbf{x}(t)\,\Delta t + \sqrt{\beta(t)\,\Delta t}\,\tilde{\varepsilon}_t$

Take the limit by letting $\Delta t \to 0$ , Then we have

$$\mathrm{d}\mathbf{x}(t) = -\frac{1}{2}\beta(t)\mathbf{x}(t)\,\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}$$

where $\mathbf{w}_t$ is a standard Wiener process

# DDPM related to SDE

**Forward SDE**

$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

$$\mathrm{d}\mathbf{x}(t) = -\frac{1}{2}\beta(t)\mathbf{x}(t)\,\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}$$

**Reverse SDE**

$$\mathrm{d}\mathbf{x} = (f(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}))\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$

$$\mathrm{d}\mathbf{x}(t) = \left[-\frac{1}{2}\beta(t)\mathbf{x}(t) - \beta(t)\nabla_{\mathbf{x}(t)}\log p(\mathbf{x}(t))\right]\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\bar{\mathbf{w}}$$

# Score Function

score of diffused data

$$\mathrm{d}\mathbf{x}(t) = \left[-\tfrac{1}{2}\beta(t)\,\mathbf{x}(t) - \beta(t)\nabla_{\mathbf{x}(t)}\log p(\mathbf{x}(t))\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\mathbf{w}}$$

$$L(\theta) = \mathbb{E}_{t\sim U(1,T)}\mathbb{E}_{\mathbf{x}(t)\sim p(\mathbf{x}(t))}\left[\|\mathbf{s}_\theta(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p(\mathbf{x}(t))\|_2^2\right]$$

$$\mathrm{d}\mathbf{x}(t) = \left[-\tfrac{1}{2}\beta(t)\,\mathbf{x}(t) - \beta(t)\,\mathbf{s}_\theta(\mathbf{x}(t),t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\mathbf{w}}$$

neural network

# Score Matching

$$L(\theta) = \mathbb{E}_{t\sim U(1,T)}\mathbb{E}_{\mathbf{x}(t)\sim p(\mathbf{x}(t))}\left[\|\mathbf{s}_\theta(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p(\mathbf{x}(t))\|_2^2\right]$$

Denoising Score Matching (Pascal Vincent, 2010)

$$\sim \mathbb{E}_{t\sim U(1,T)}\mathbb{E}_{\mathbf{x}(0)\sim p(\mathbf{x}(0))}\mathbb{E}_{\mathbf{x}(t)\sim p(\mathbf{x}(t)|\mathbf{x}(0))}\left[\|\mathbf{s}_\theta(\mathbf{x}(t),t) - \nabla_{\mathbf{x}_t}\log p(\mathbf{x}(t)|\mathbf{x}(0))\|_2^2\right]$$

Sliced Score Matching (Yang Song et al., 2019)

$$\sim \mathbb{E}_{t\sim U(1,T)}\mathbb{E}_{\mathbf{x}(0)\sim p(\mathbf{x}(0))}\mathbb{E}_{\mathbf{v}\sim \mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{1}{2}\|\mathbf{s}_\theta(\mathbf{x}(t),t)\|^2 + \mathbf{v}^\top \mathbf{s}_\theta(\mathbf{x}(t),t)\mathbf{v}\right]$$

# How to solve the SDE?

Use the numerical method to solve the following SDE

$$\mathrm{d}\mathbf{x}(t) = \left[-\frac{1}{2}\beta(t)\,\mathbf{x}(t) - \beta(t)\,\mathbf{s}_\theta(\mathbf{x}(t), t)\right] \mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\bar{\mathbf{w}}$$

1. Euler Maruyama
2. Stochastic Runge-Kutta
3. Milstein

# Topic

1. Denoising Diffusion Probabilistic Models (DDPM)
2. Latent Diffusion Model (LDM)
3. Diffusion Model related to Stochastic Differential Equations (SDE)
4. Applications

# Diffusion Models Beat GANs on Image Synthesis

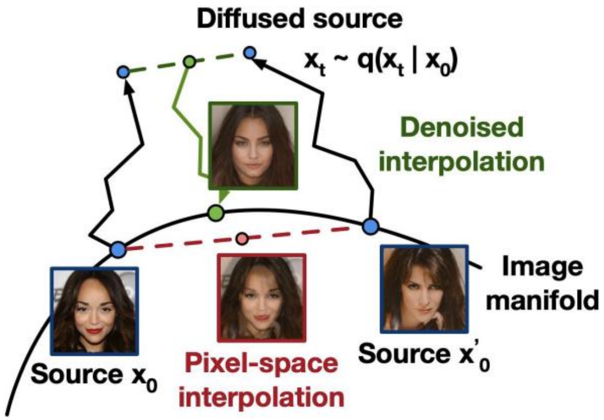
Diffusion model achieve the state-of-the-art performance on image synthesis.

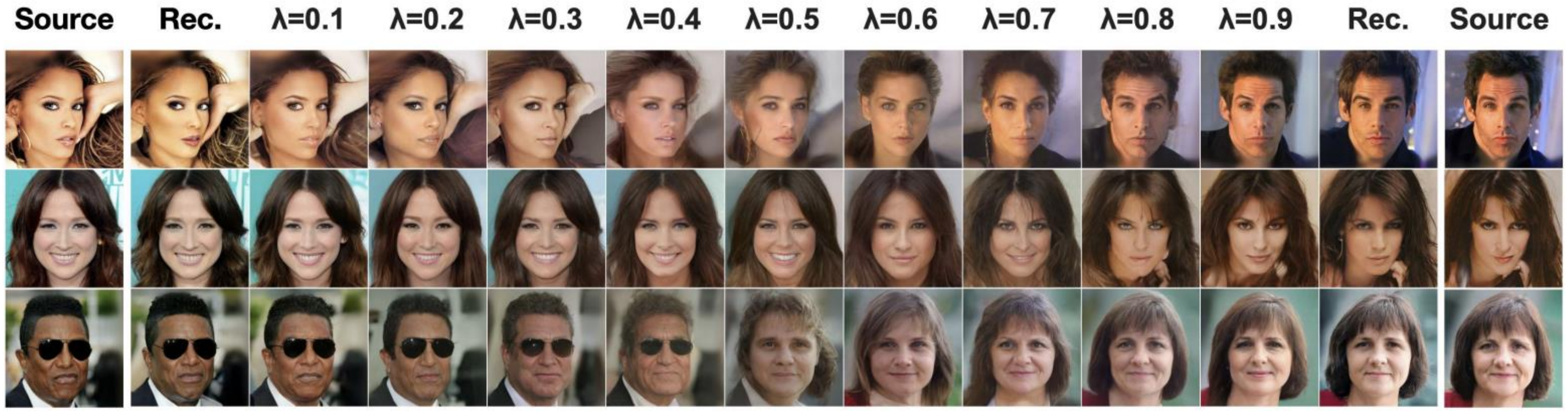
| Model | FID | sFID | Prec | Rec |
|---|---|---|---|---|
| **LSUN Bedrooms 256×256** | | | | |
| DCTransformer† [42] | 6.40 | 6.66 | 0.44 | **0.56** |
| DDPM [25] | 4.89 | 9.07 | 0.60 | 0.45 |
| IDDPM [43] | 4.24 | 8.21 | 0.62 | 0.46 |
| StyleGAN [27] | 2.35 | 6.62 | 0.59 | 0.48 |
| **ADM (dropout)** | **1.90** | **5.59** | **0.66** | 0.51 |
| **LSUN Horses 256×256** | | | | |
| StyleGAN2 [28] | 3.84 | 6.46 | 0.63 | 0.48 |
| **ADM** | 2.95 | **5.94** | 0.69 | **0.55** |
| **ADM (dropout)** | **2.57** | 6.81 | **0.71** | **0.55** |
| **LSUN Cats 256×256** | | | | |
| DDPM [25] | 17.1 | 12.4 | 0.53 | 0.48 |
| StyleGAN2 [28] | 7.25 | **6.33** | 0.58 | 0.43 |
| **ADM (dropout)** | **5.57** | 6.69 | **0.63** | **0.52** |
| **ImageNet 64×64** | | | | |
| BigGAN-deep* [5] | 4.06 | 3.96 | **0.79** | 0.48 |
| IDDPM [43] | 2.92 | **3.79** | 0.74 | 0.62 |
| **ADM** | 2.61 | **3.77** | 0.73 | 0.63 |
| **ADM (dropout)** | **2.07** | 4.29 | 0.74 | **0.63** |

| Model | FID | sFID | Prec | Rec |
|---|---|---|---|---|
| **ImageNet 128×128** | | | | |
| BigGAN-deep [5] | 6.02 | 7.18 | **0.86** | 0.35 |
| LOGAN† [68] | 3.36 | | | |
| **ADM** | 5.91 | **5.09** | 0.70 | **0.65** |
| **ADM-G (25 steps)** | 5.98 | 7.04 | 0.78 | 0.51 |
| **ADM-G** | **2.97** | **5.09** | 0.78 | 0.59 |
| **ImageNet 256×256** | | | | |
| DCTransformer† [42] | 36.51 | 8.24 | 0.36 | **0.67** |
| VQ-VAE-2†‡ [51] | 31.11 | 17.38 | 0.36 | 0.57 |
| IDDPM‡ [43] | 12.26 | 5.42 | 0.70 | 0.62 |
| SR3†‡ [53] | 11.30 | | | |
| BigGAN-deep [5] | 6.95 | 7.36 | **0.87** | 0.28 |
| **ADM** | 10.94 | 6.02 | 0.69 | 0.63 |
| **ADM-G (25 steps)** | 5.44 | 5.32 | 0.81 | 0.49 |
| **ADM-G** | **4.59** | **5.25** | 0.82 | 0.52 |
| **ImageNet 512×512** | | | | |
| BigGAN-deep [5] | 8.43 | 8.13 | **0.88** | 0.29 |
| **ADM** | 23.24 | 10.19 | 0.73 | **0.60** |
| **ADM-G (25 steps)** | 8.41 | 9.67 | 0.83 | 0.47 |
| **ADM-G** | **7.72** | **6.57** | 0.87 | 0.42 |

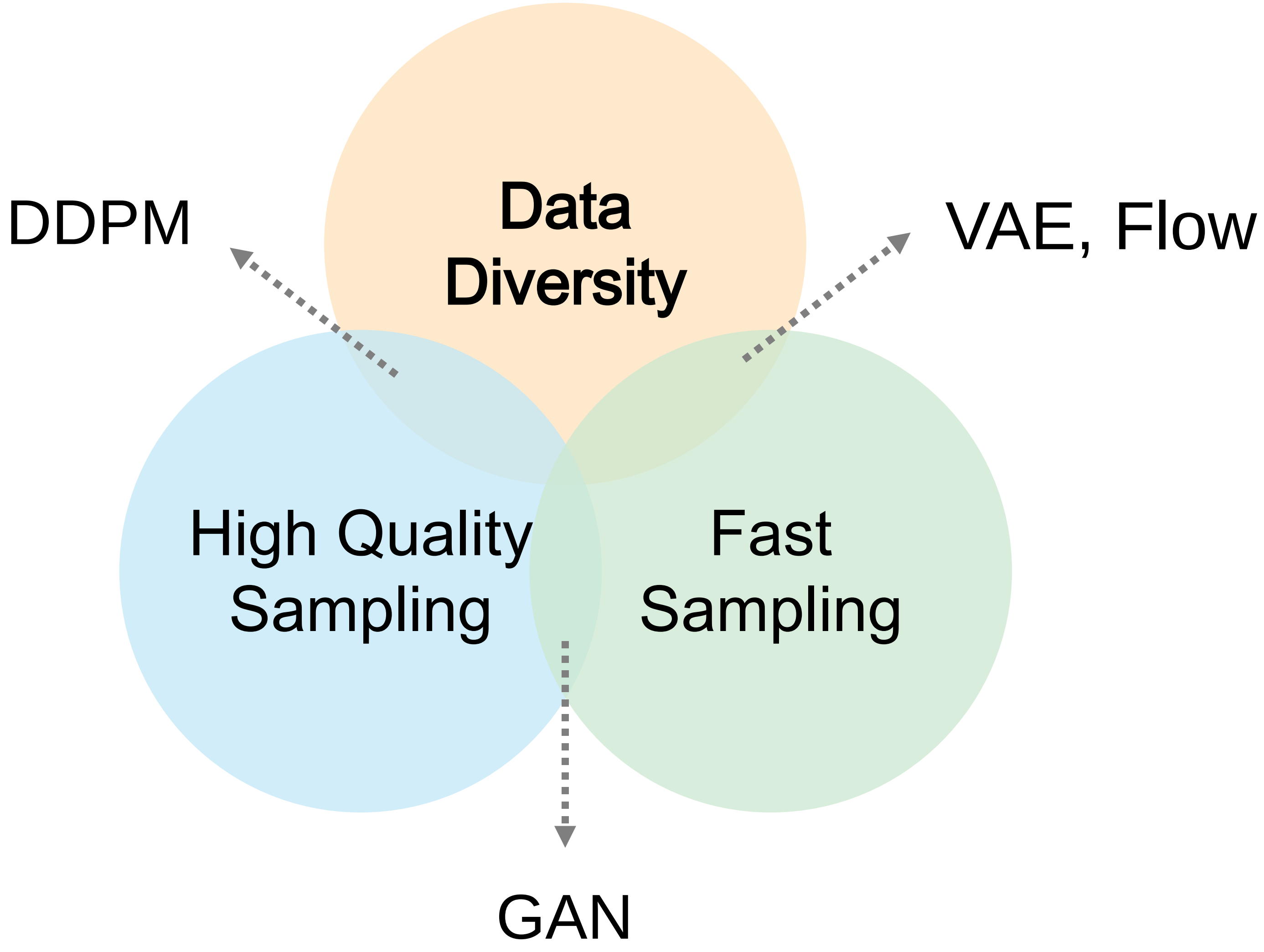
# Interpolation on Image Manifold

Interpolations of CelebA-HQ 256x256 images with 500 timesteps of diffusion.



# Reflection


DDPM
Data Diversity
VAE, Flow
High Quality Sampling
Fast Sampling
GAN

# Summary

1. Diffusion model has strong mathematical theoretical support including Markov process, variational inference, SDE, score matching, etc.
2. DDPM can generate high quality and diverse image samples but the sampling time is lengthy.
3. LDM use the pretrained encoder to reduce the computational complexity.
4. DDPM can extend to continuous SDE process which can be solved quickly by stochastic numerical methods.

# Reference


- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. NeurIPS 2020.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022.
- Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. ICLR 2021.

# Thanks for listening!