SC395:

Image Generative Models in Computer Vision

Viraj Shah

Lecture 4

Jan 12th, 2026

sc395.virajshah.com

ILLINOIS

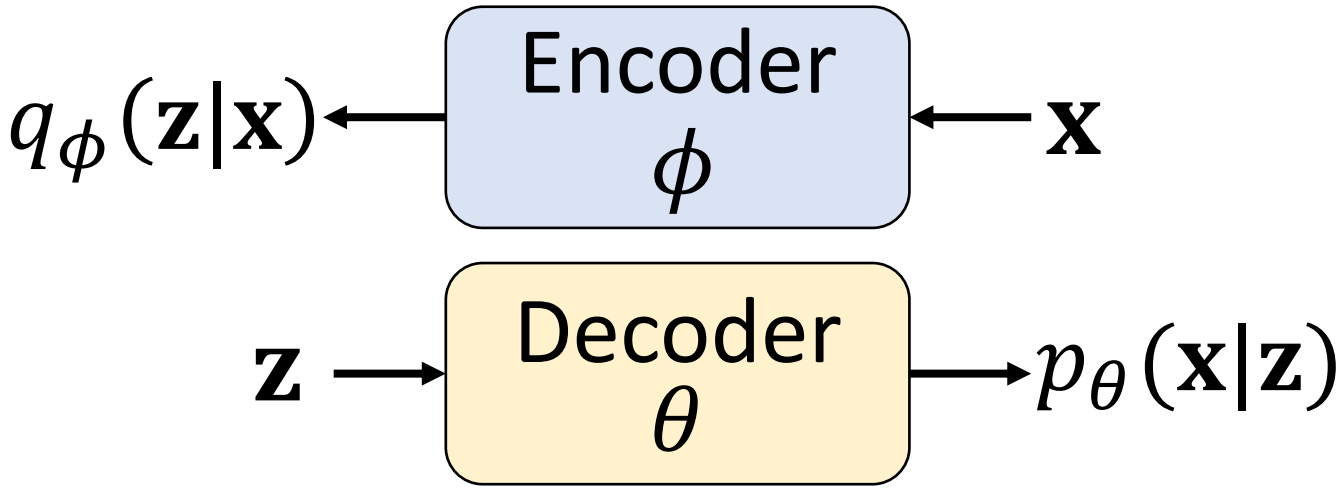
Google

Diffusion Models

Guidance, LDMs, Efficiency, Adaptation

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\right]$$

Evidence **Evidence Lower Bound (ELBO)**



Slide from Jia-Bin Huang

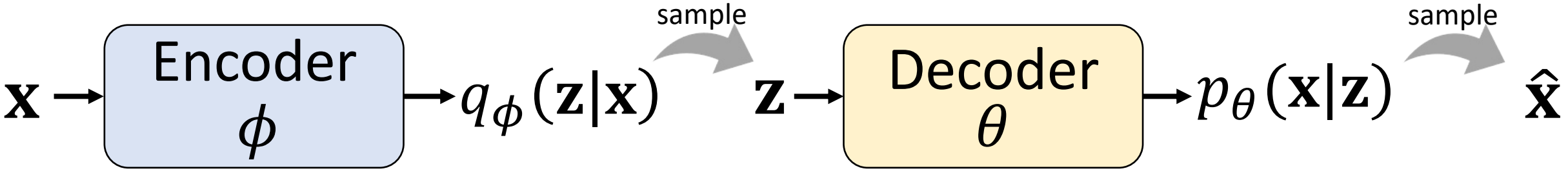
$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\right] \quad \textbf{(ELBO)}$$

$$\begin{aligned}
\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}\mid\boldsymbol{x})}\right] &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}\mid\boldsymbol{z})p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}\mid\boldsymbol{x})}\right] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}\mid\boldsymbol{z})] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}\mid\boldsymbol{x})}\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}\mid\boldsymbol{z})]}_{\text{Reconstruction term}} - \underbrace{D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{z}\mid\boldsymbol{x}) \,\|\, p(\boldsymbol{z})\big)}_{\text{Regularization term}}
\end{aligned}$$

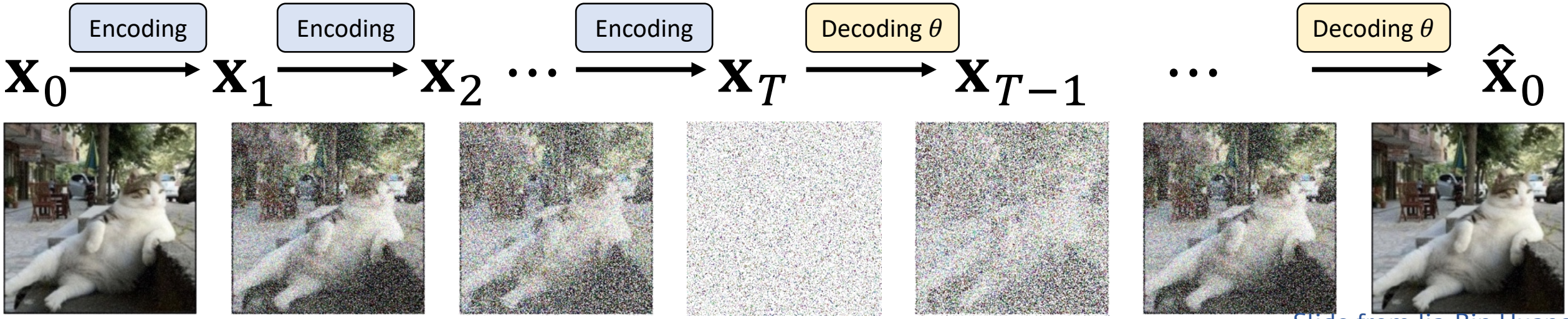
$$\|x - \hat{x}\|_2 = \|x - DE(EN(z))\|_2 \quad -\frac{1}{2}\sum_{j=0}^{k-1}\left(\sigma_j + \mu_j^2 - 1 - \log \sigma_j\right)$$

Slide from Jia-Bin Huang

Variational Autoencoder (VAE)

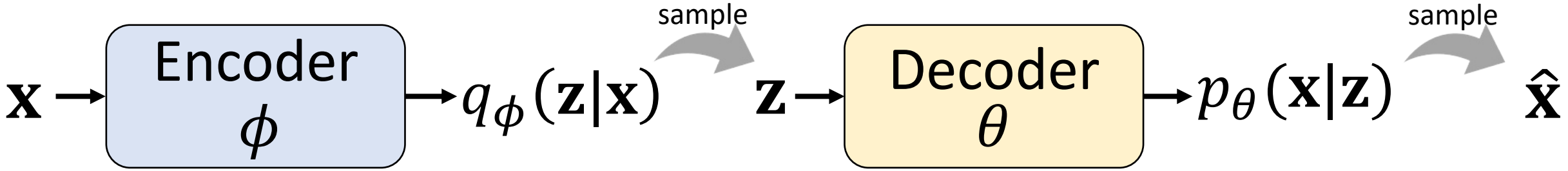


Diffusion models



Slide from Jia-Bin Huang

Variational Autoencoder (VAE)

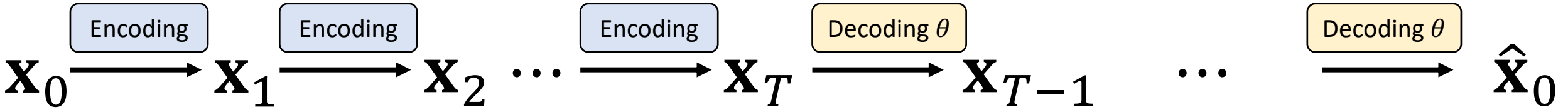


Observed $\mathbf{x}$

Latent $\mathbf{z}$

Maximize $\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\right]$

Diffusion models



Observed $\mathbf{x}_0$

Latent $\mathbf{x}_1, \ldots, \mathbf{x}_T$

Maximize $\mathbb{E}_{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_0:\mathbf{x}_T)}{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\right]$

Slide from Jia-Bin Huang

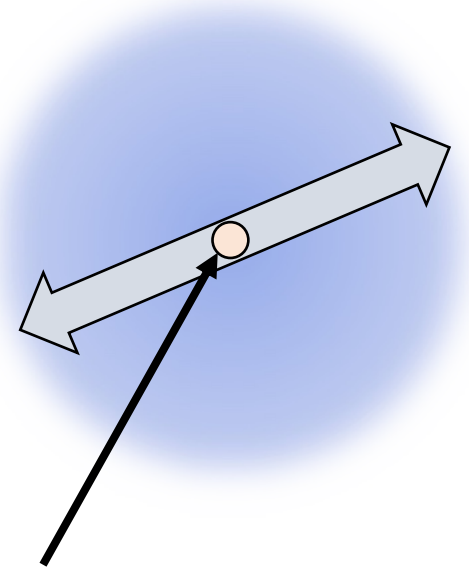
Encoding Encoding Encoding Decoding θ Decoding θ

$$\mathbf{x}_0 \longrightarrow \mathbf{x}_1 \longrightarrow \mathbf{x}_2 \cdots \longrightarrow \mathbf{x}_T \longrightarrow \mathbf{x}_{T-1} \cdots \longrightarrow \hat{\mathbf{x}}_0$$

Maximize $\mathbb{E}_{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_0:\mathbf{x}_T)}{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\right]$

$$q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

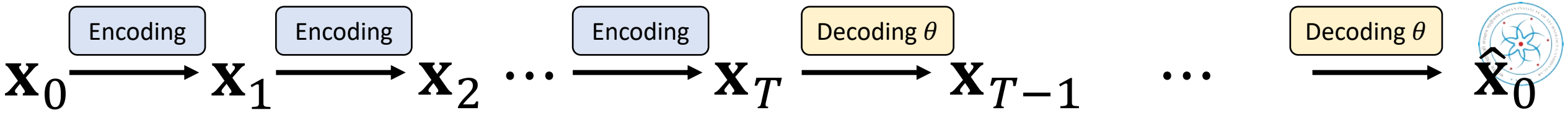
Encoding



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

Encoding Mean Variance

Slide from Jia-Bin Huang



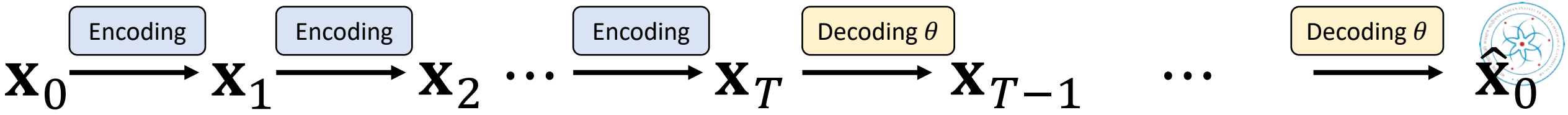
$$\mathbb{E}_{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_0:\mathbf{x}_T)}{q(\mathbf{x}_1:\mathbf{x}_T|\mathbf{x}_0)}\right]$$

1 $-\mathrm{D}_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))$ **Prior matching**

2 $\mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)]$ **Reconstruction**

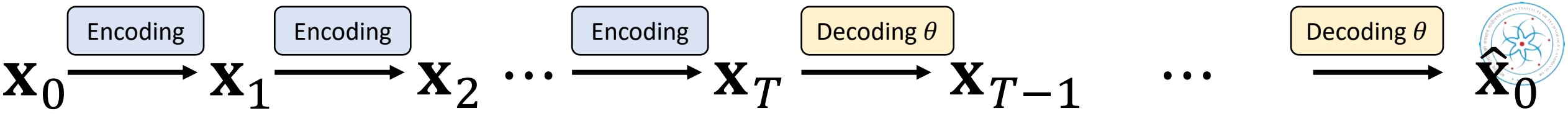
3 $-\sum_{t=2}^{\mathrm{T}} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))]$ **Denoising matching**

Slide from Jia-Bin Huang



$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\frac{1}{2\boldsymbol{\sigma}_q^2(t)} \left[\| \mu_\theta(\mathbf{x}_t, t) - \mu_q(\mathbf{x}_t, \mathbf{x}_0) \|_2^2 \right] \right]$$

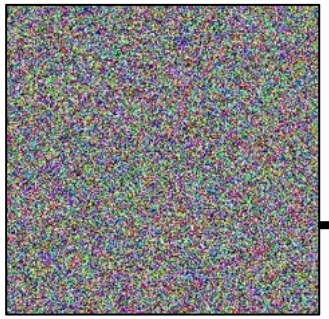
Slide from Jia-Bin Huang



$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\| \; \hat{\mathbf{x}}_\theta(\mathbf{x}_t, t) \; - \; \mathbf{x}_0 \; \|_2^2 \right] \right]$$

$$\mu_q(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\overline{\alpha_{t-1}})}{1-\overline{\alpha_t}}\mathbf{x}_t + \frac{\sqrt{\overline{\alpha_{t-1}}}(1-\alpha_t)}{1-\overline{\alpha_t}}\mathbf{x}_0$$

$$\mu_\theta(\mathbf{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\overline{\alpha_{t-1}})}{1-\overline{\alpha_t}}\mathbf{x}_t + \frac{\sqrt{\overline{\alpha_{t-1}}}(1-\alpha_t)}{1-\overline{\alpha_t}}\hat{\mathbf{x}}_\theta(\mathbf{x}_t, t)$$



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

$\mathbf{x}_0$

Encoding

$q(\mathbf{x}_t|\mathbf{x}_0)$

$\mathbf{x}_t = \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}$

$\mathbf{x}_t$

t

θ

U-Net

$\mu_\theta(\mathbf{x}_t, t)$ → L_2 ← $\mu_q(\mathbf{x}_t, \mathbf{x}_0)$

Slide from Jia-Bin Huang

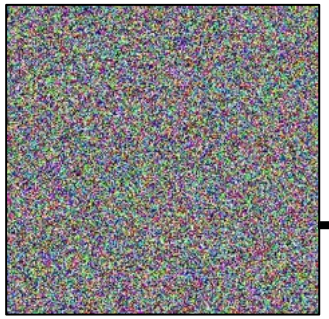
Encoding Encoding Encoding Decoding θ Decoding θ

$$\mathbf{x}_0 \rightarrow \mathbf{x}_1 \rightarrow \mathbf{x}_2 \cdots \rightarrow \mathbf{x}_T \rightarrow \mathbf{x}_{T-1} \cdots \rightarrow \hat{\mathbf{x}}_0$$

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[w(t) \left[\left\| \hat{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2 \right] \right]$$

$$\mu_q(\mathbf{x}_t, \mathbf{x}_0) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)\alpha_t}}\boldsymbol{\epsilon}$$

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



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

$\mathbf{x}_0$

Encoding

$q(\mathbf{x}_t|\mathbf{x}_0)$

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

$\mathbf{x}_t$

t

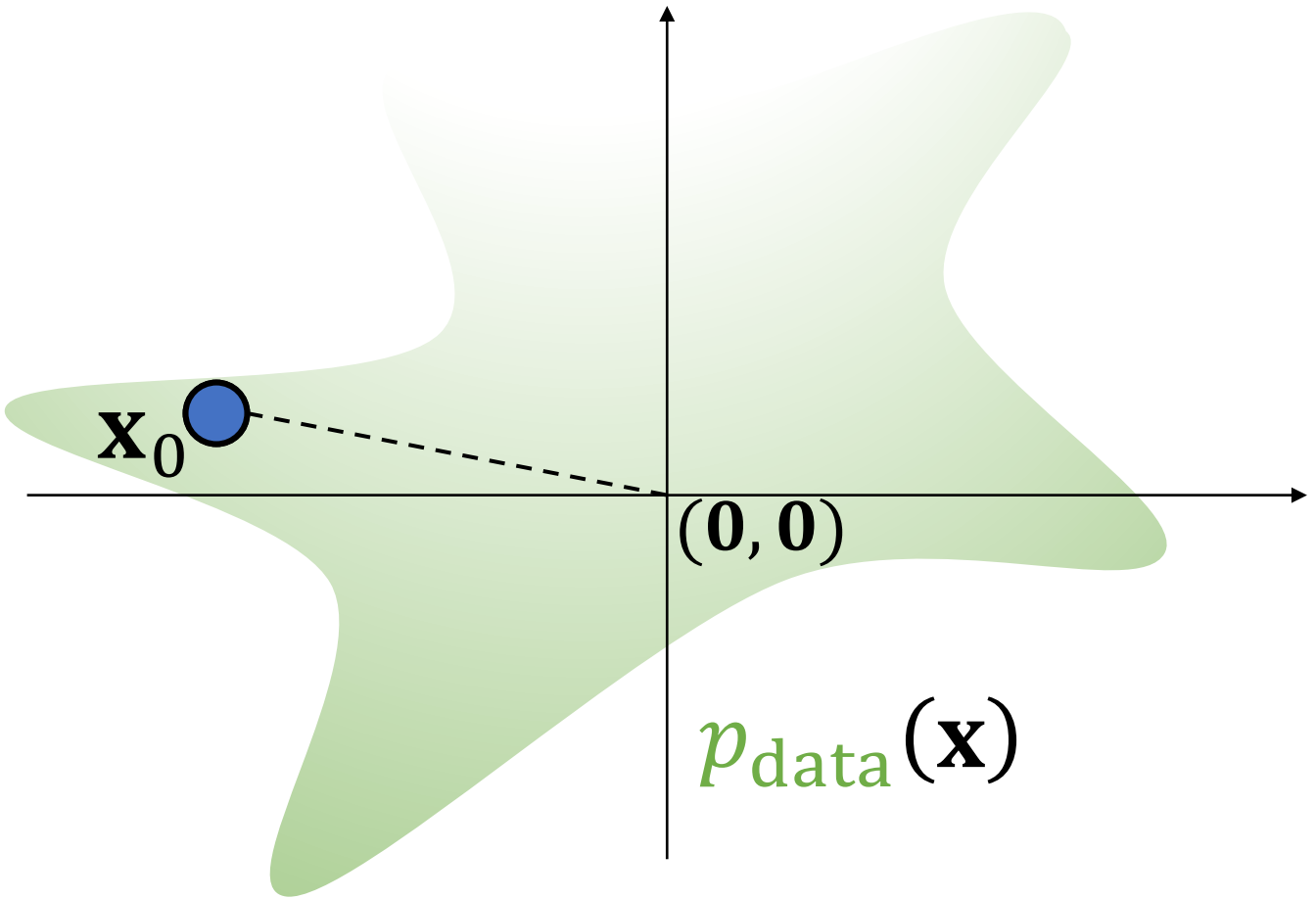
θ

U-Net

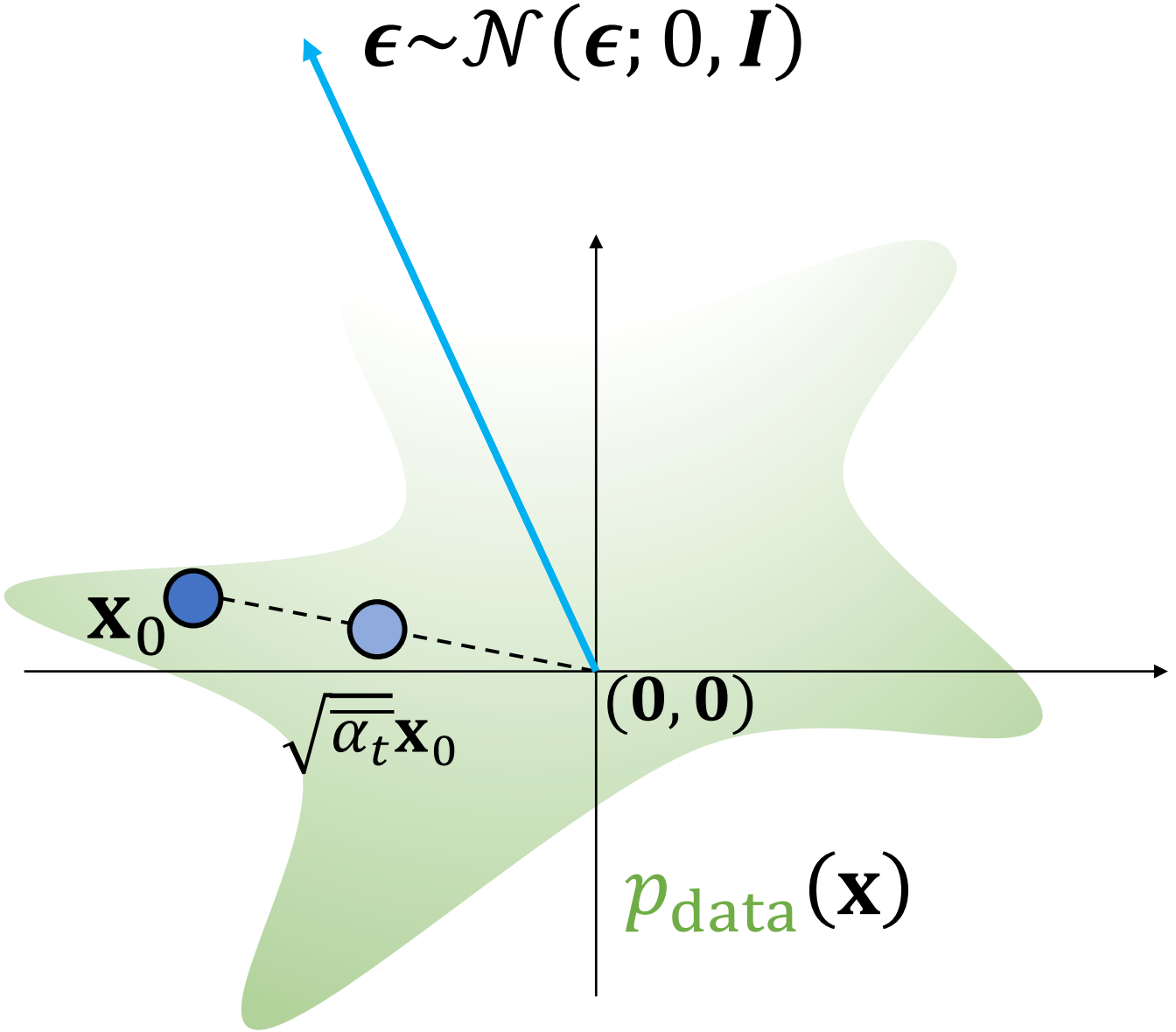
$\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t)$

L_2

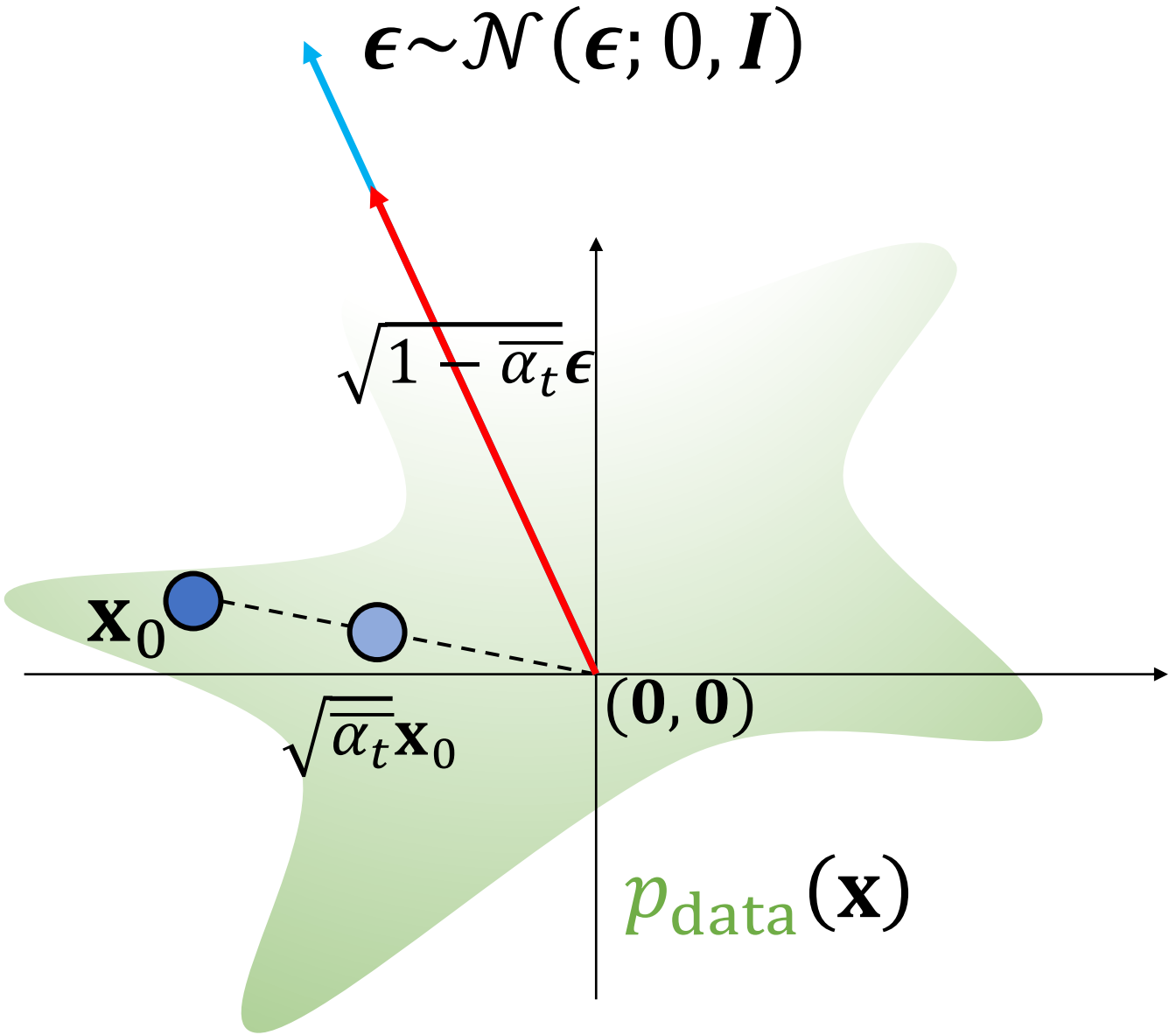
Slide from Jia-Bin Huang



Slide from Jia-Bin Huang

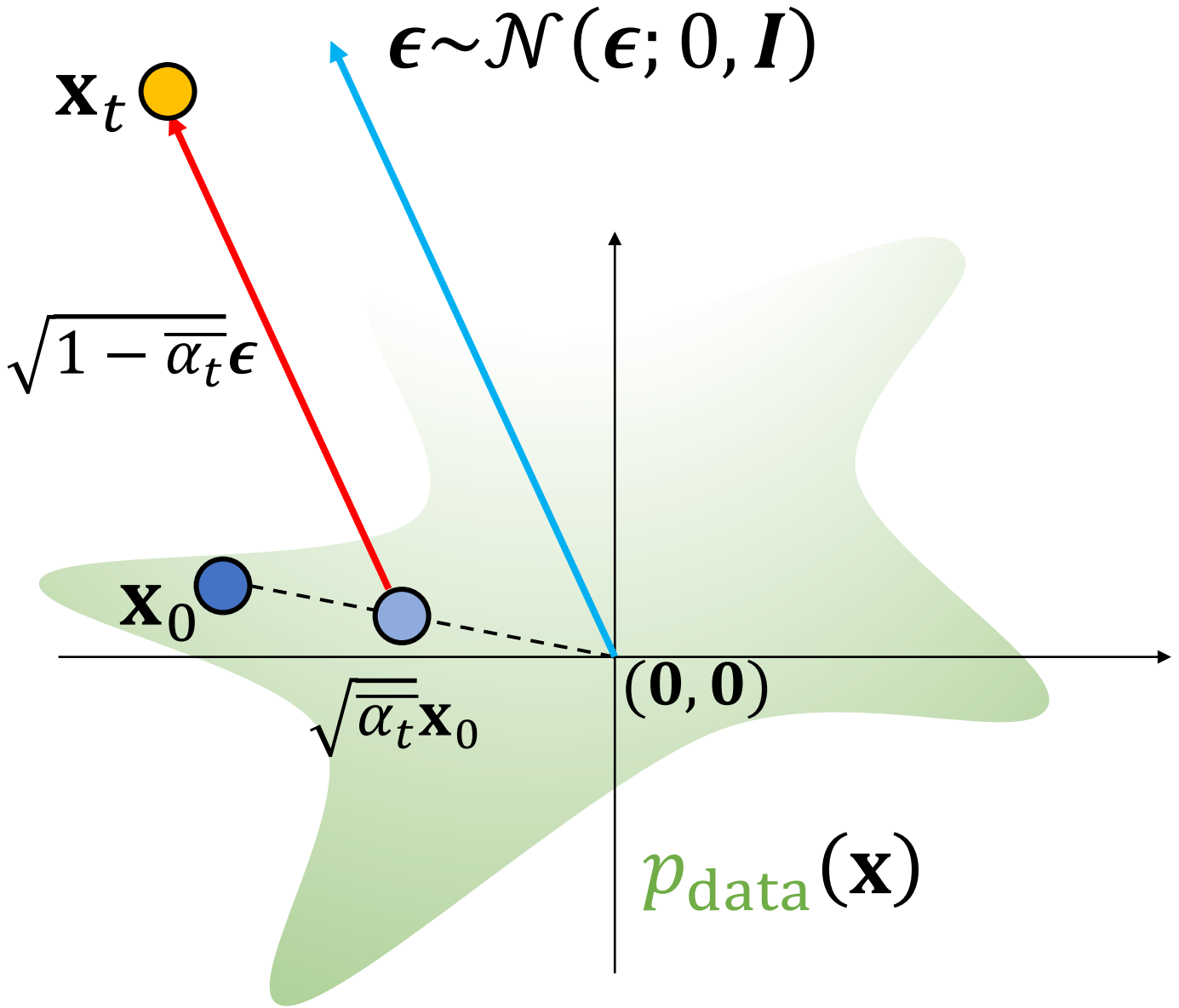


Slide from Jia-Bin Huang



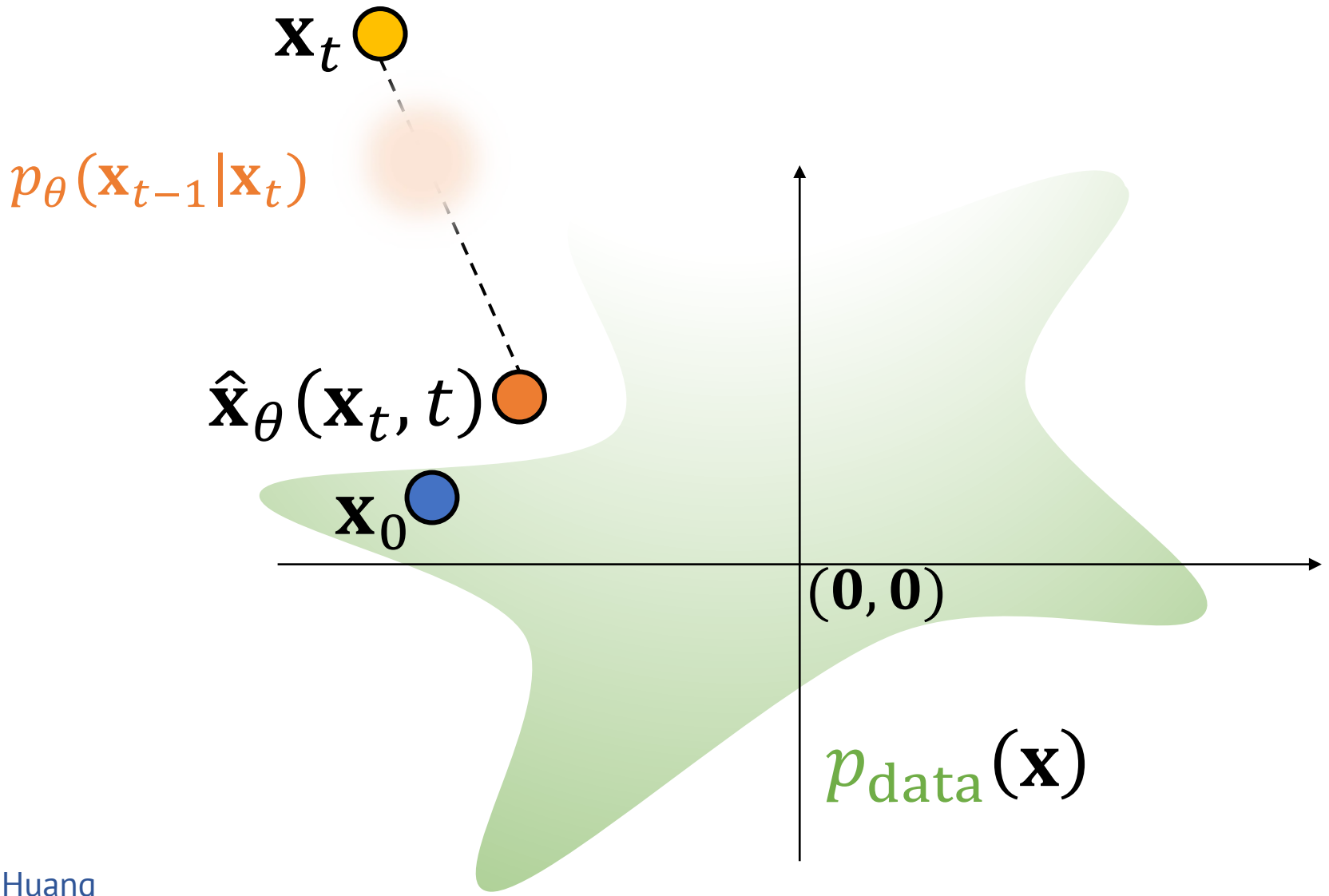
Slide from Jia-Bin Huang

$$\mathbf{x}_t = \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}$$



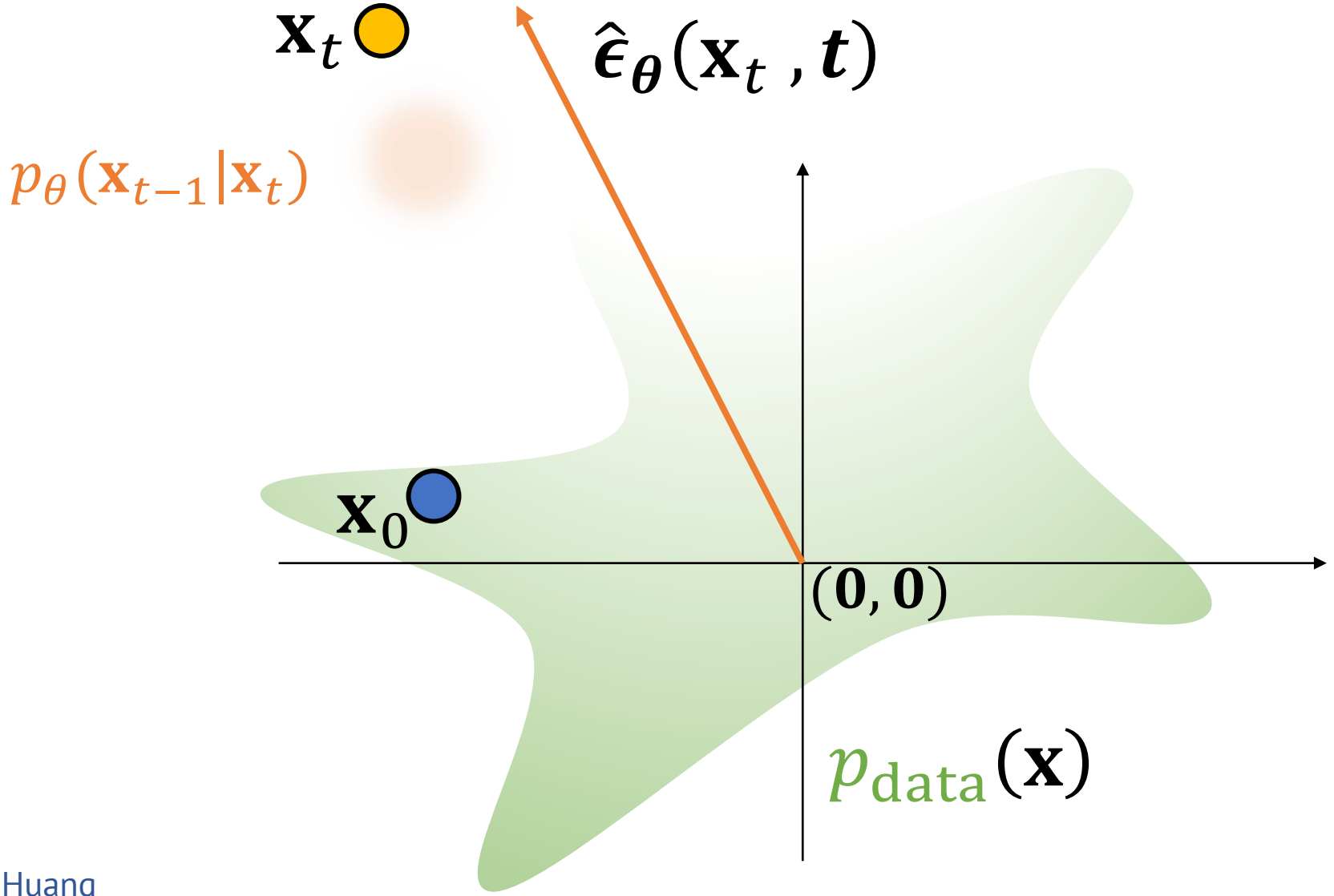
Slide from Jia-Bin Huang

$$\mu_\theta(\mathbf{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\overline{\alpha_{t-1}})}{1-\overline{\alpha_t}}\mathbf{x}_t + \frac{\sqrt{\overline{\alpha_{t-1}}}(1-\alpha_t)}{1-\overline{\alpha_t}}\hat{\mathbf{x}}_\theta(\mathbf{x}_t, t)$$



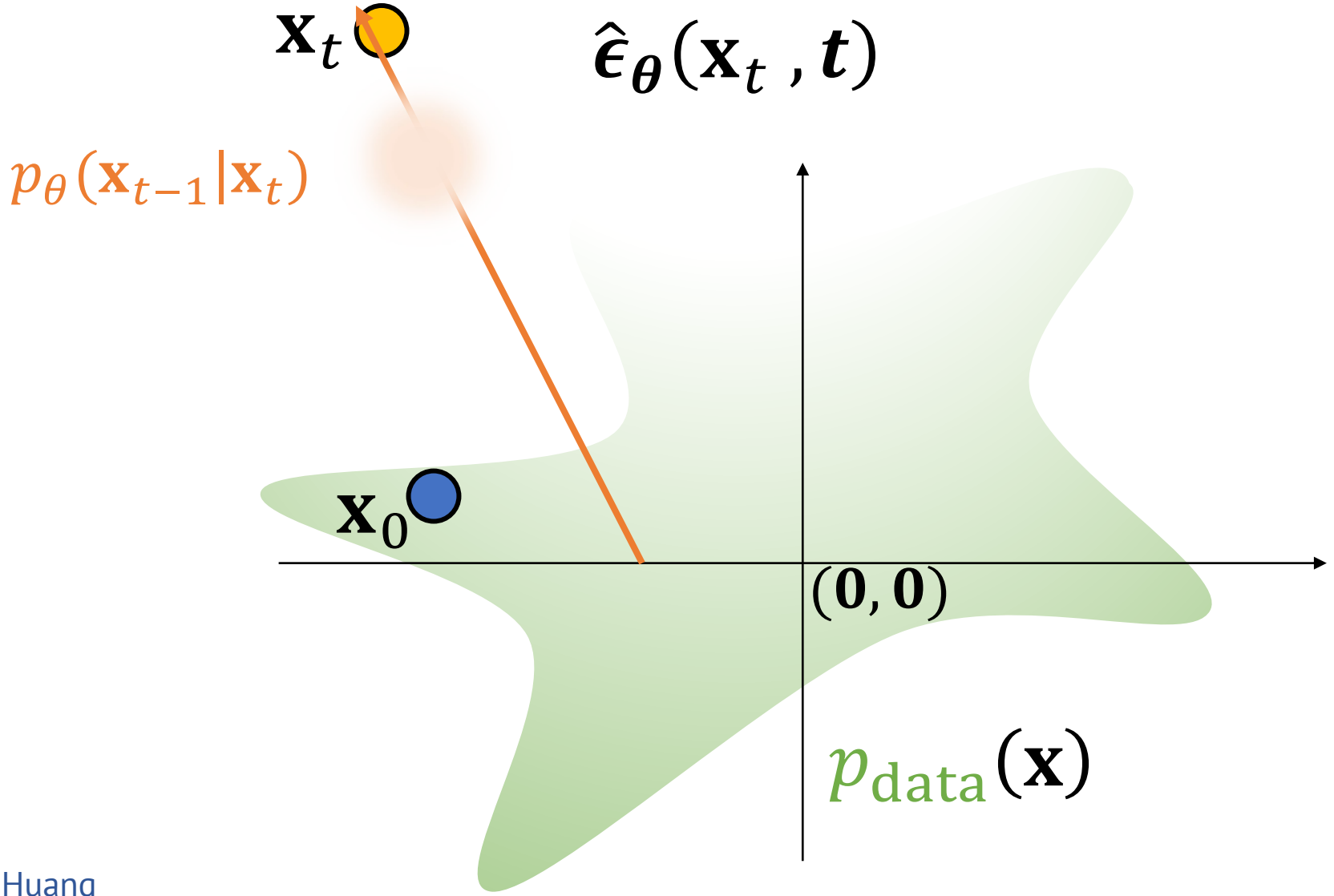
Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)\alpha_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$



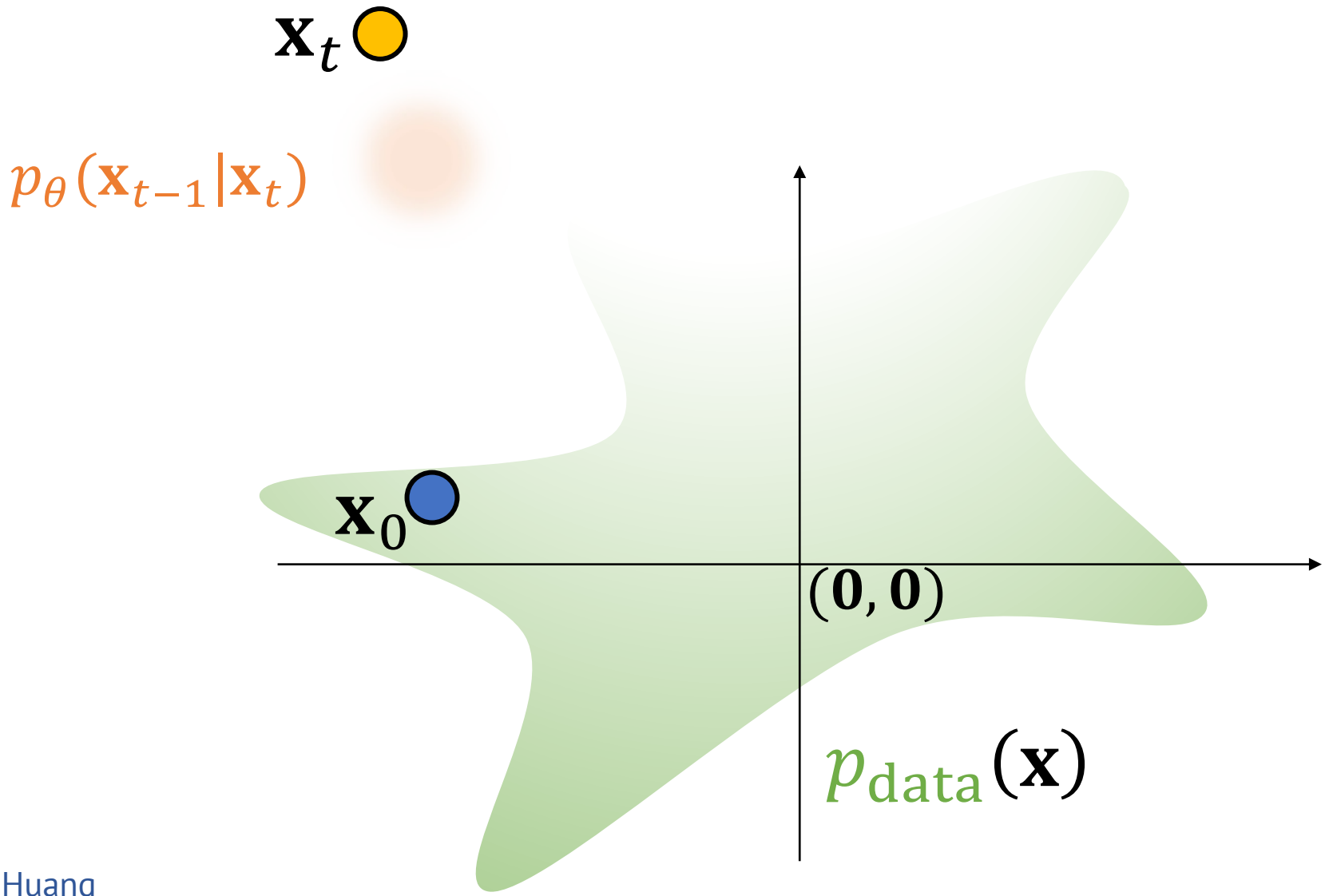
Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)\alpha_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$



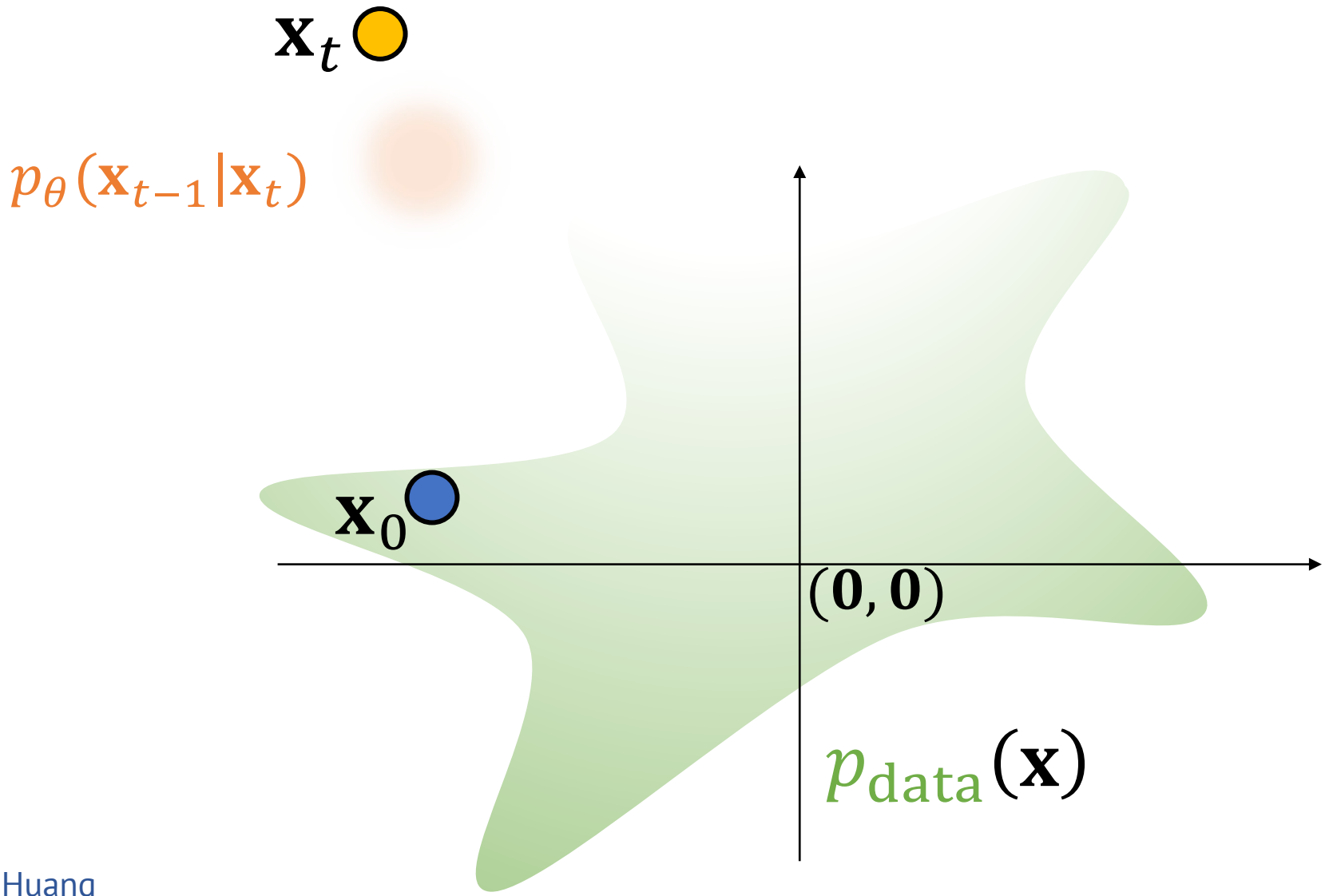
Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) =$$



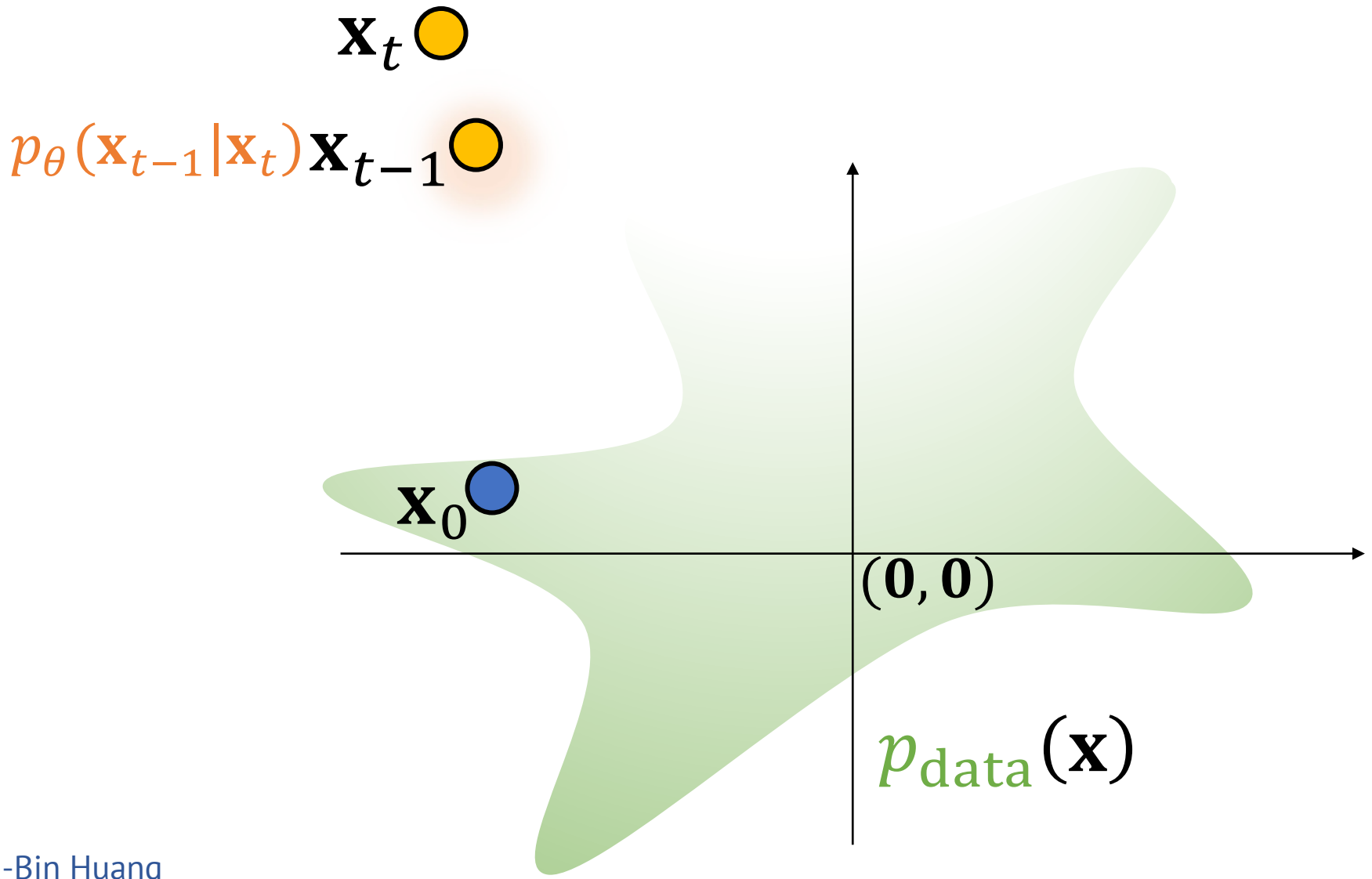
Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s_\theta}(\mathbf{x}_t, t)$$



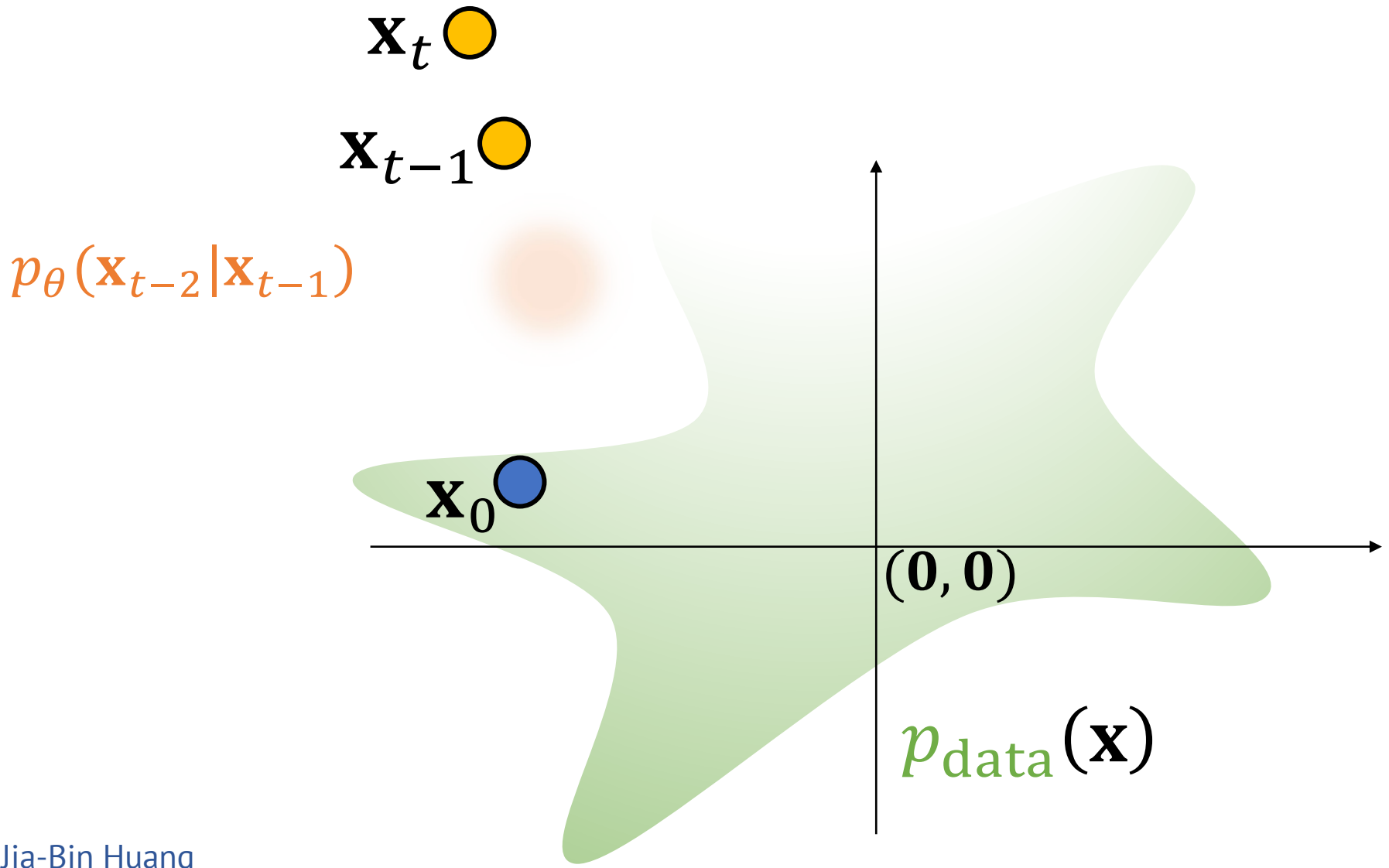
Slide from Jia-Bin Huang

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$



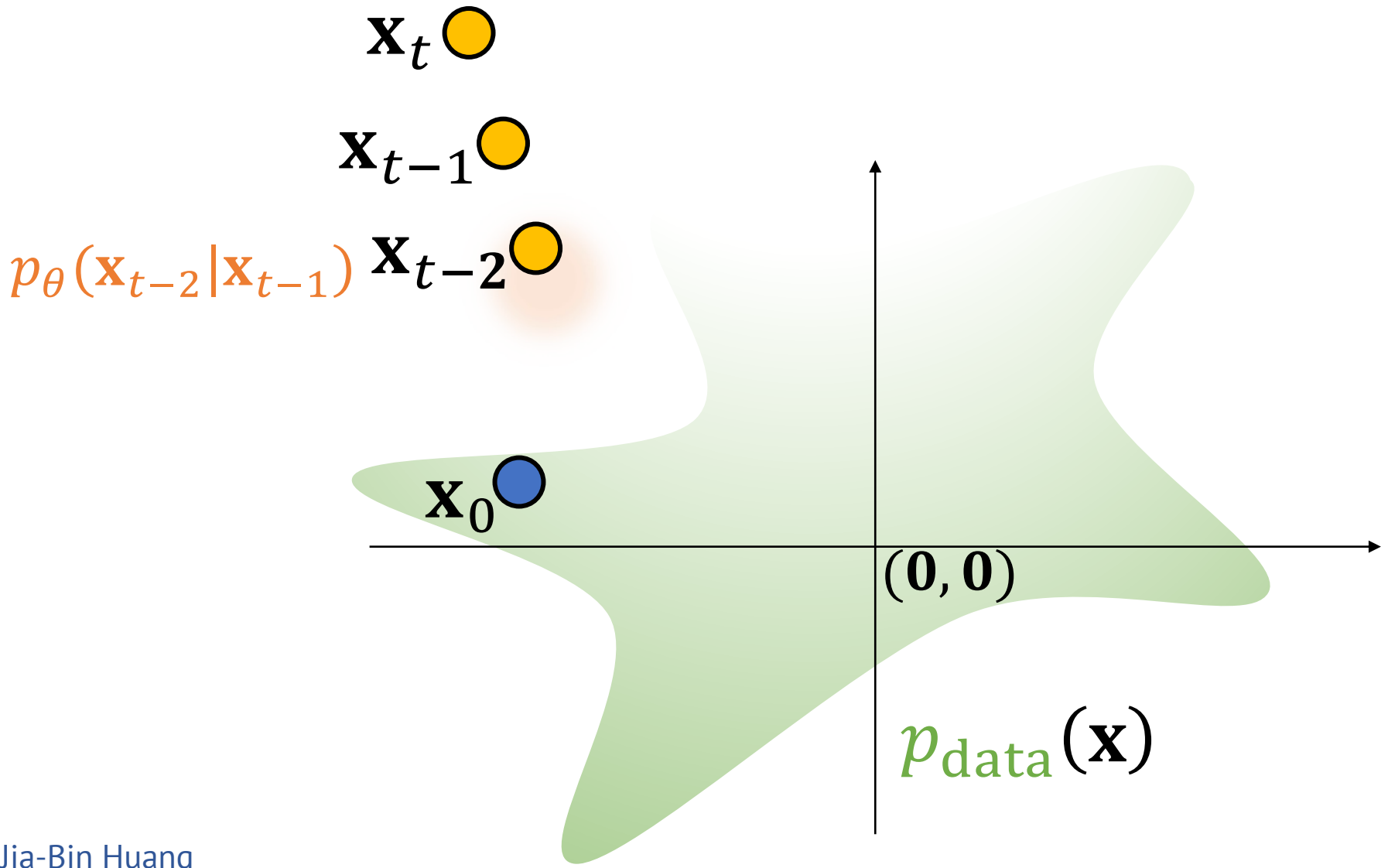
Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s_\theta}(\mathbf{x}_t, t)$$



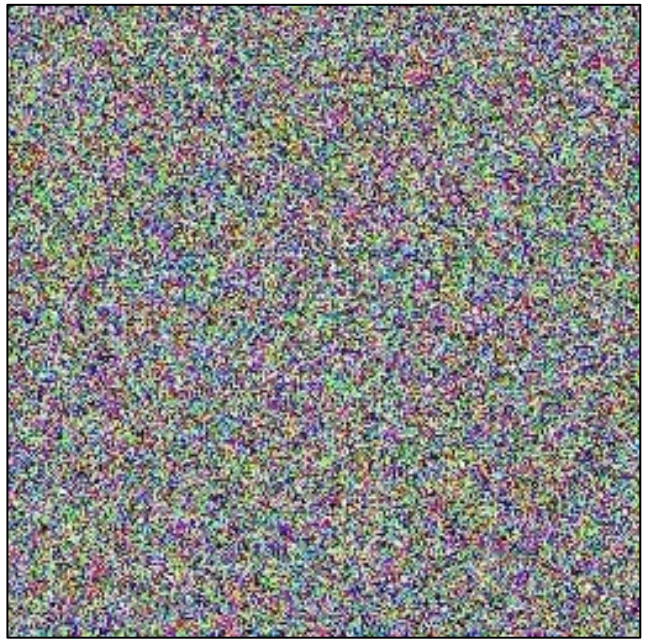
Slide from Jia-Bin Huang

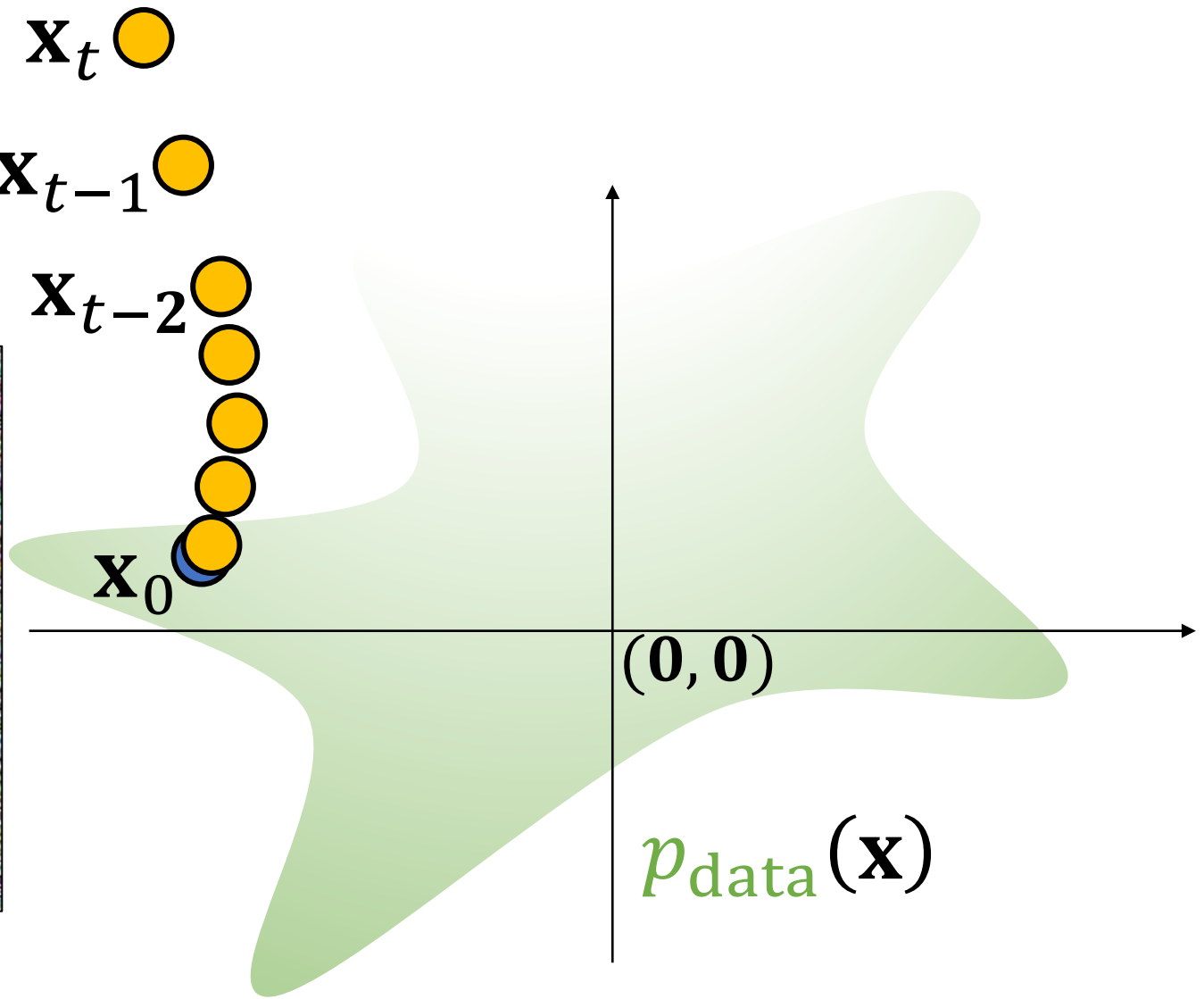
$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s_\theta}(\mathbf{x}_t, t)$$



Slide from Jia-Bin Huang

$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s_\theta}(\mathbf{x}_t, t)$$





Slide from Jia-Bin Huang

Denoising diffusion probabilistic models (DDPMs)

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

Abstract

We present high quality image synthesis results using diffusion probabilistic models, a class of latent variable models inspired by considerations from nonequilibrium thermodynamics. Our best results are obtained by training on a weighted variational bound designed according to a novel connection between diffusion probabilistic models and denoising score matching with Langevin dynamics, and our models naturally admit a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding. On the unconditional CIFAR10 dataset, we obtain an Inception score of 9.46 and a state-of-the-art FID score of 3.17. On 256x256 LSUN, we obtain sample quality similar to ProgressiveGAN. Our implementation is available at https://github.com/hojonathanho/diffusion.

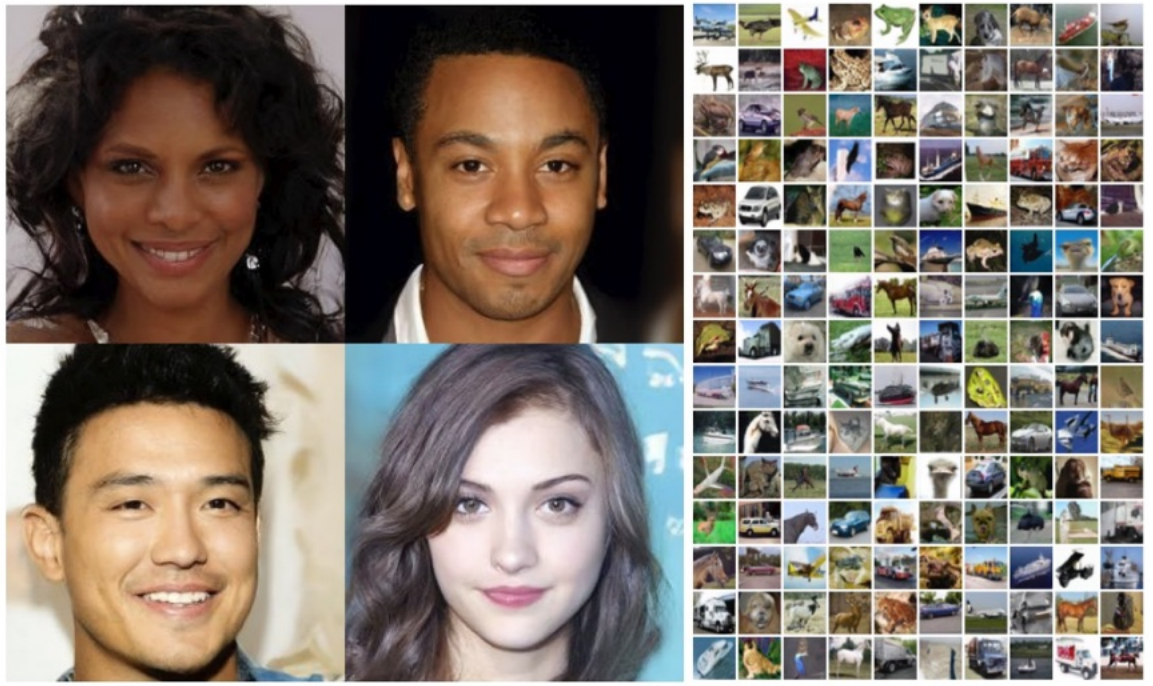
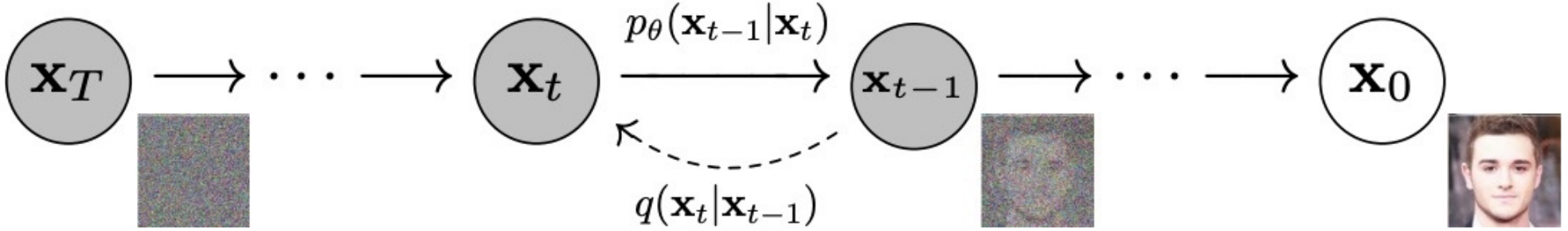


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

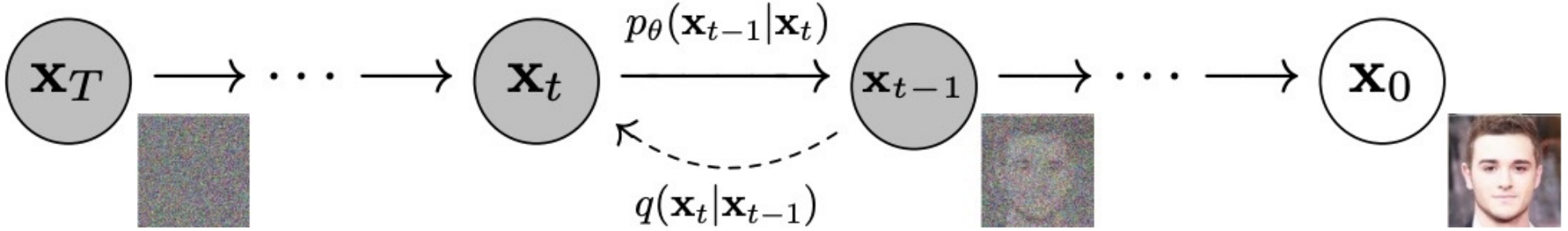
DDPMs: Basic idea



Unconditional CIFAR10 sample generation

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020
Blog introduction: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
CVPR 2022 tutorial

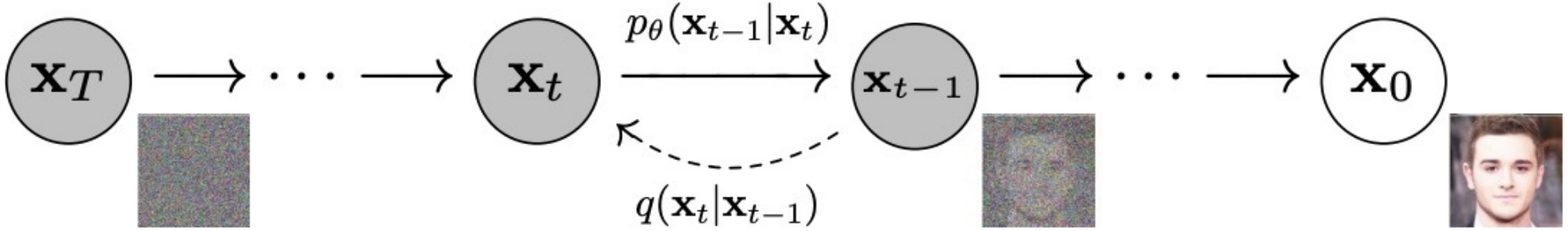
DDPMs: Basic idea



- *Forward process* q turns images into Gaussian noise
- *Reverse process* p turns noise into images
- Provided the increments of t are small enough, $p_\theta(x_{t-1}|x_t)$ is Gaussian and we can train a neural network to estimate the mean of x_{t-1} given x_t

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

DDPMs: Basic idea



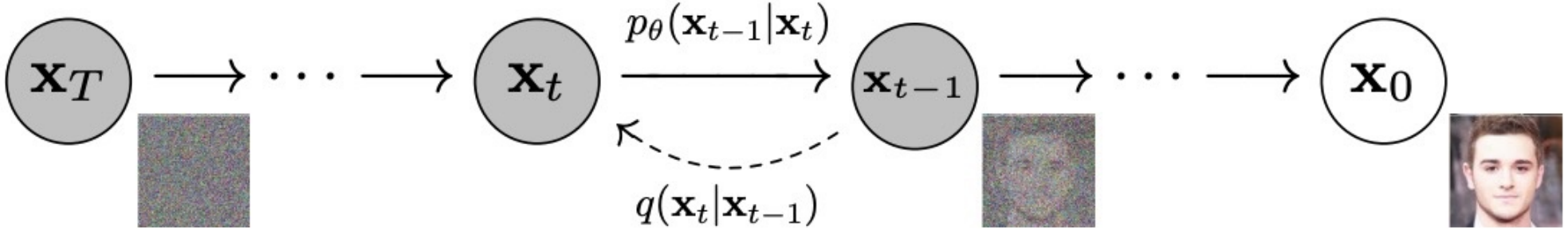
Algorithm 1 Training

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
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\quad x_t \quad , t) \right\|^2$
6: **until** converged

- $\epsilon_\theta(x_t, t)$ is the predicted noise component of image x_t given noise level t
- Network parameters θ are updated to reduce L2 error between actual noise ϵ and predicted noise $\epsilon_\theta(x_t, t)$

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

DDPMs: Basic idea



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

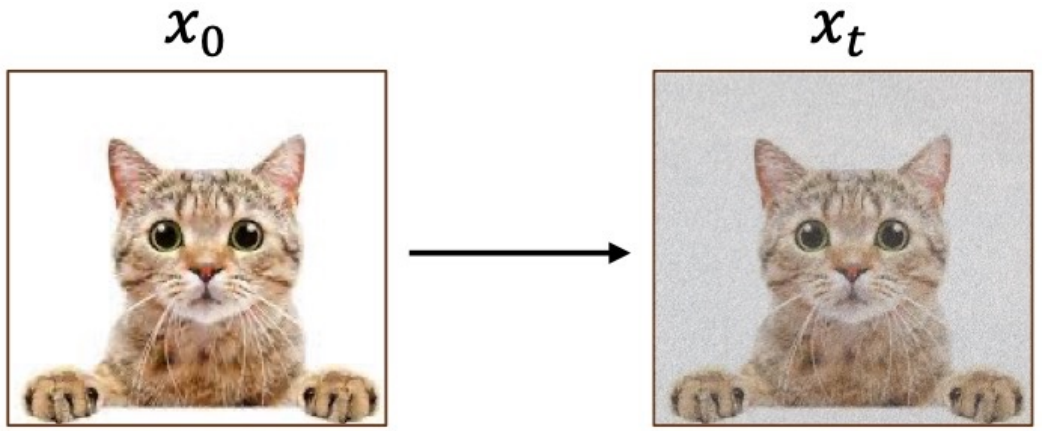
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}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

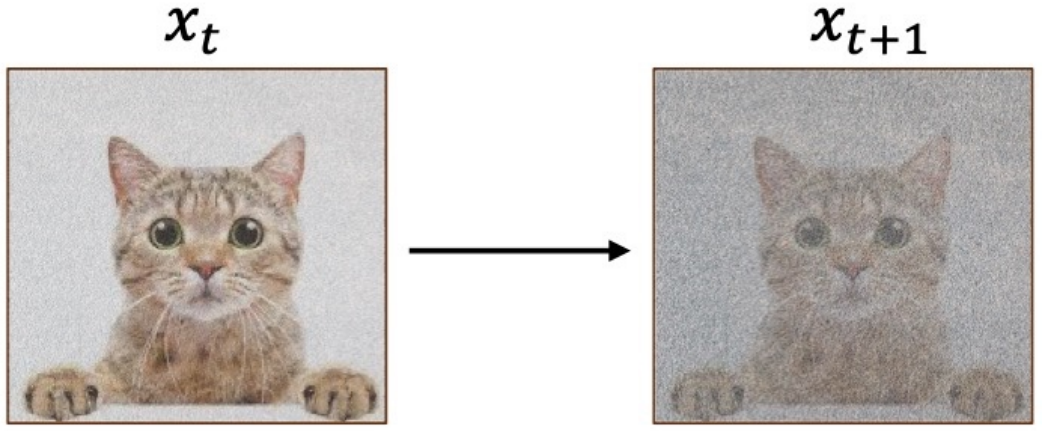
J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

How do we do this in practice?

- Step 1: Sample image from the dataset, generate noisy image using forward process

- Step 2: Given noisy image, generate slightly noisier image

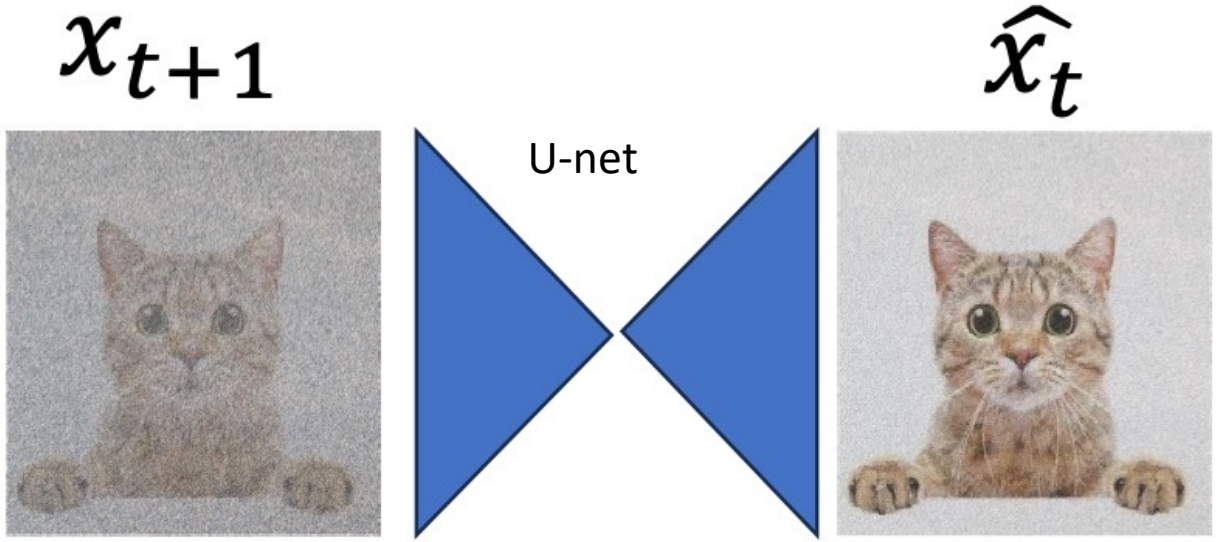




https://www.eecs.umich.edu/courses/eecs442-ahowens/fa23/slides/lec11-diffusion.pdf

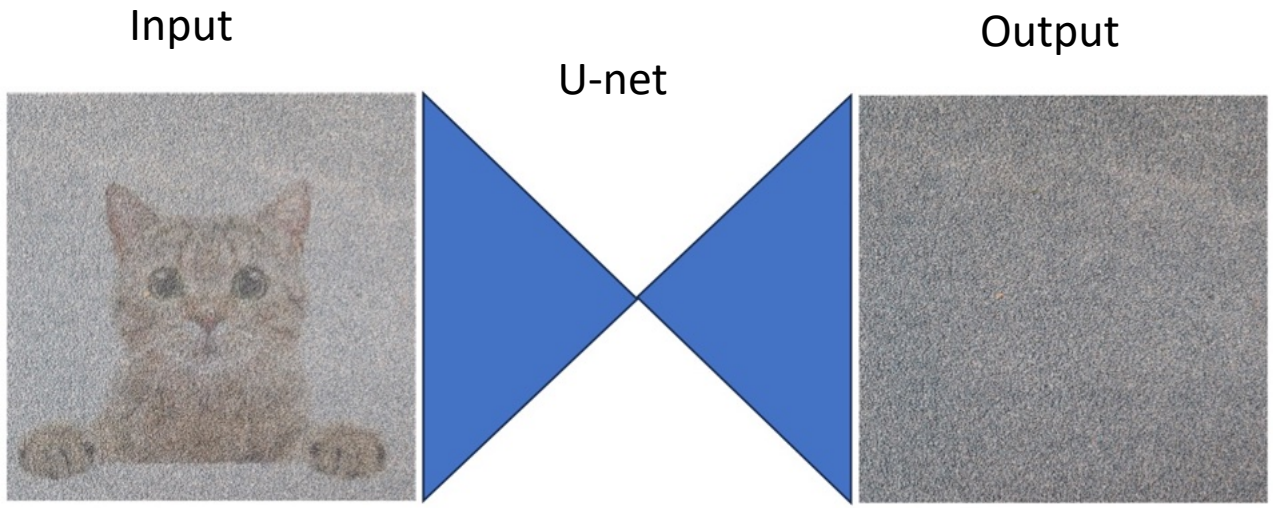
How do we do this in practice?

During Training



Loss: MSE$(x_t, \hat{x}_t)$

During Inference



U-net predicts the noise

https://www.eecs.umich.edu/courses/eecs442-ahowens/fa23/slides/lec11-diffusion.pdf

Class-conditioned DDPMs

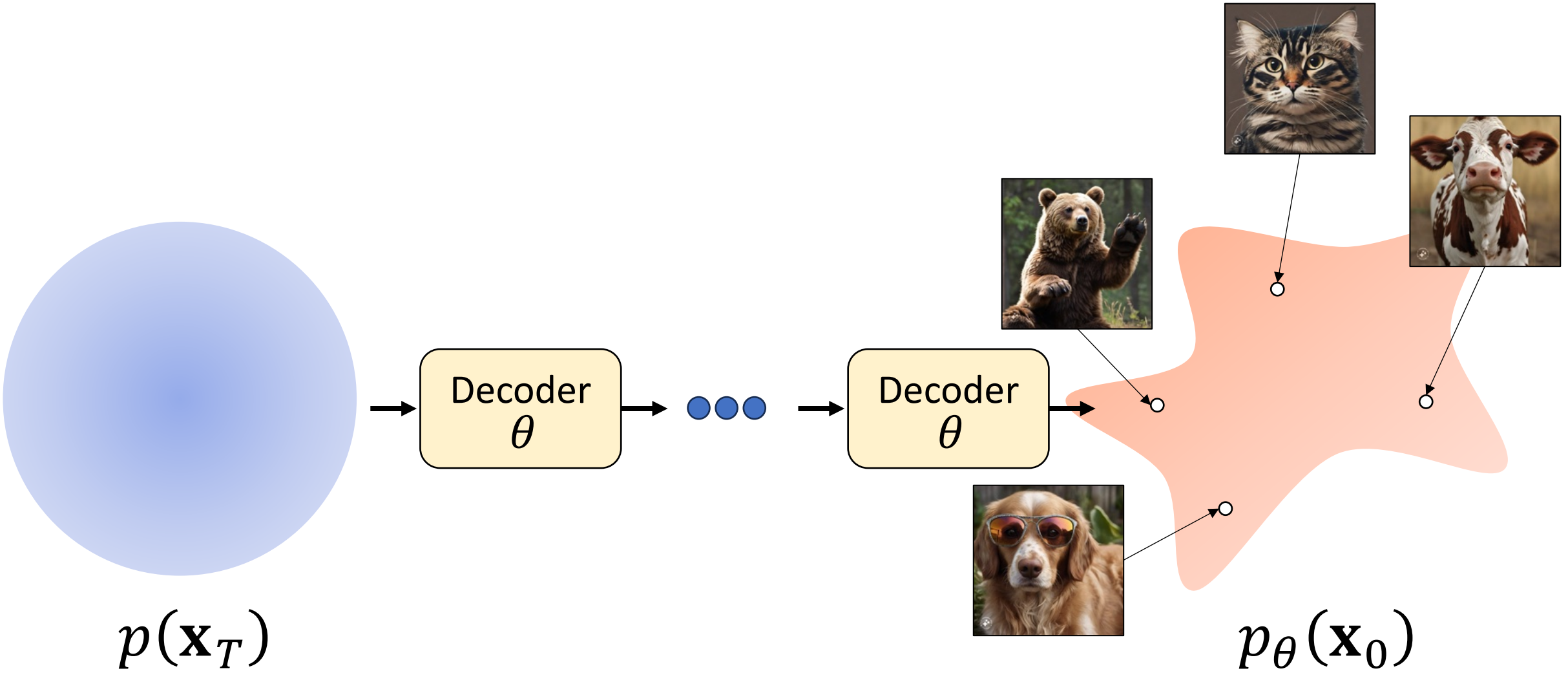
- “We can sample with as few as 25 forward passes while maintaining FIDs comparable to BigGAN”

Figure 1: Selected samples from our best ImageNet 512×512 model (FID 3.85)

Abstract

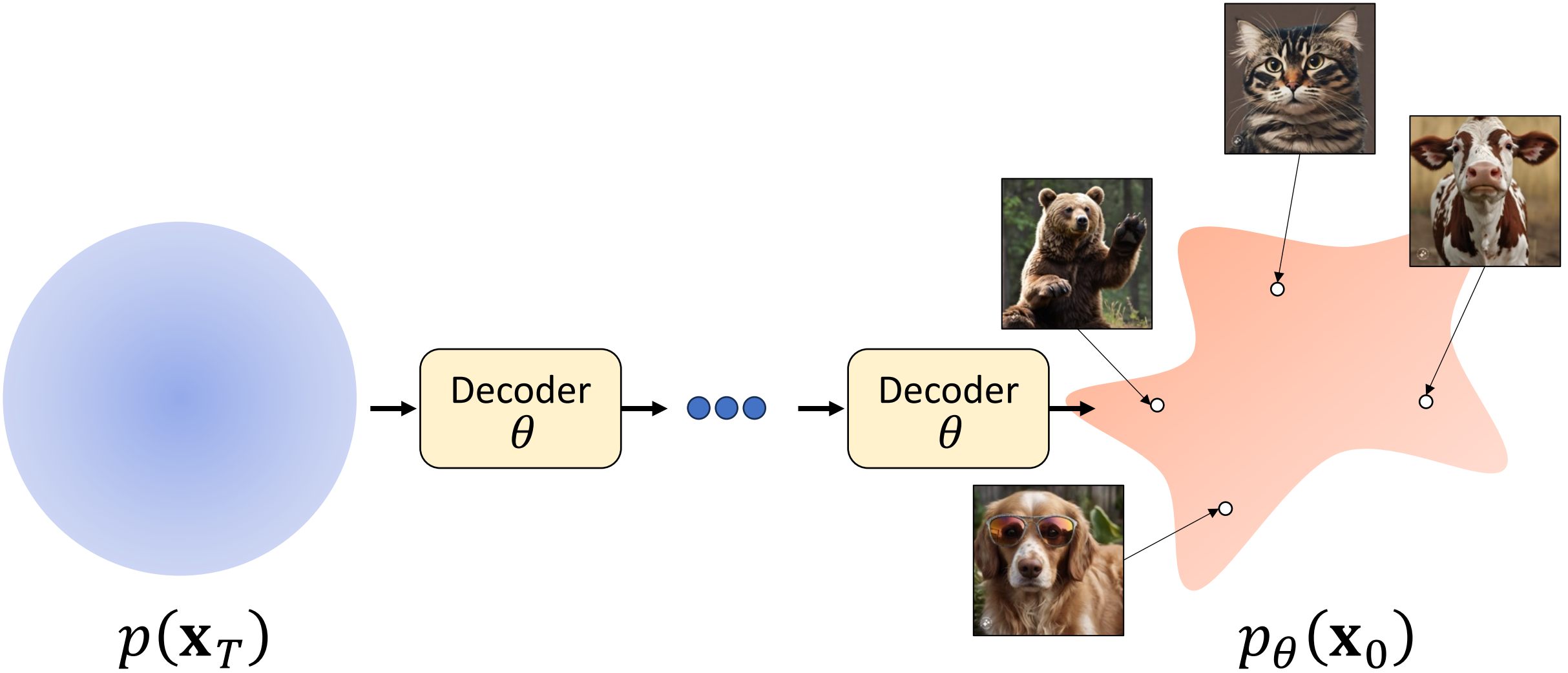
We show that diffusion models can achieve image sample quality superior to the current state-of-the-art generative models. We achieve this on unconditional image synthesis by finding a better architecture through a series of ablations. For conditional image synthesis, we further improve sample quality with classifier guidance: a simple, compute-efficient method for trading off diversity for fidelity using gradients from a classifier. We achieve an FID of 2.97 on ImageNet 128×128, 4.59 on ImageNet 256×256, and 7.72 on ImageNet 512×512, and we match BigGAN-deep even with as few as 25 forward passes per sample, all while maintaining better coverage of the distribution. Finally, we find that classifier guidance combines well with upsampling diffusion models, further improving FID to 3.94 on ImageNet 256×256 and 3.85 on ImageNet 512×512. We release our code at `https://github.com/openai/guided-diffusion`.

P. Dhariwal and A. Nichol. Diffusion Models Beat GANs on Image Synthesis. NeurIPS 2021



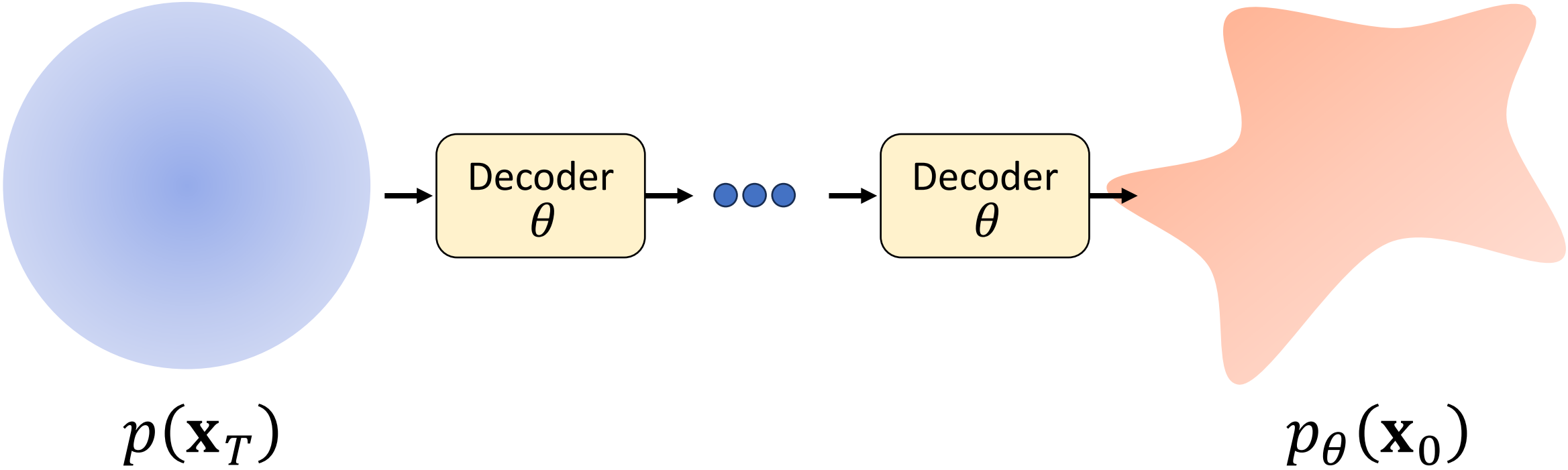
Slide from Jia-Bin Huang

Unconditional generation



Slide from Jia-Bin Huang

Conditional generation

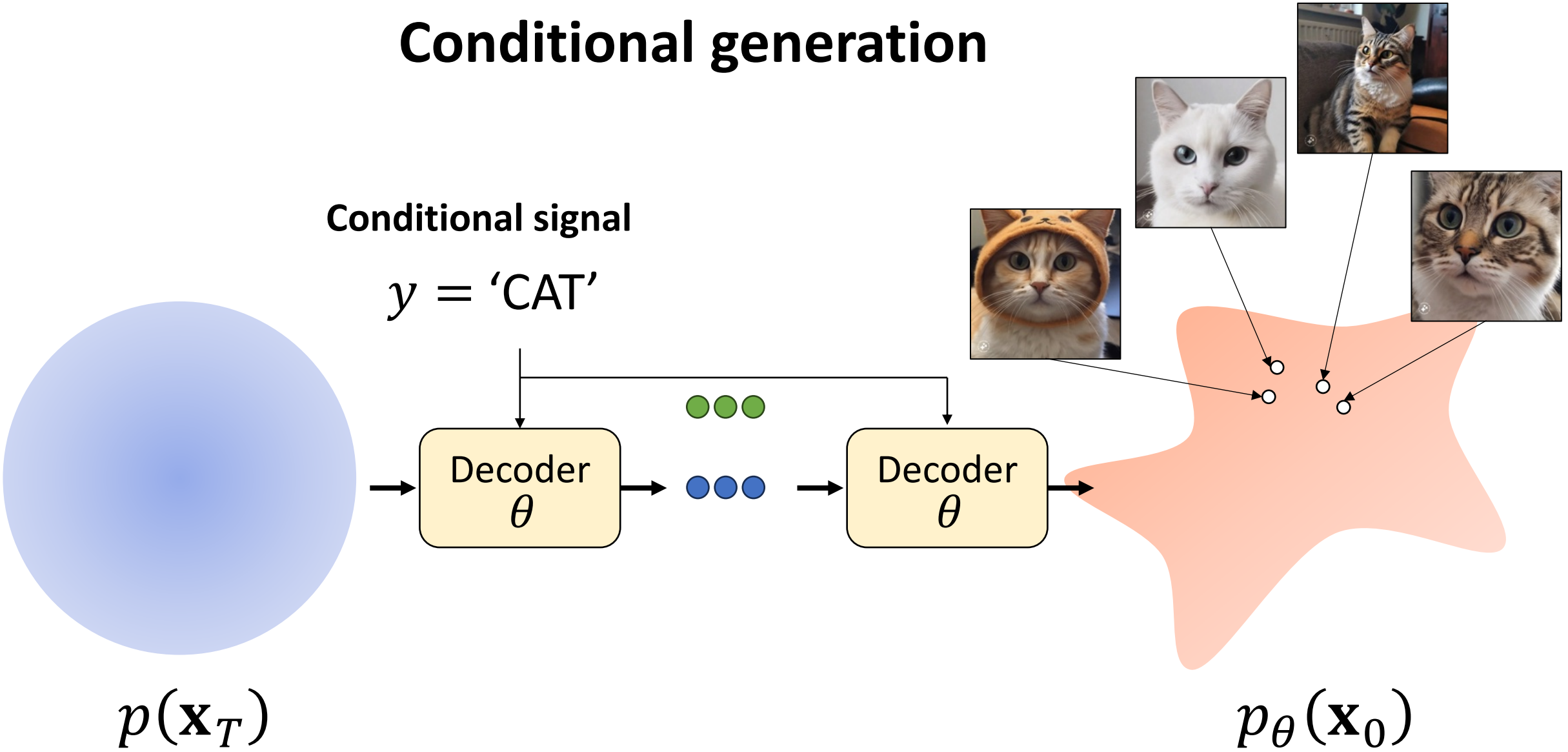


Slide from Jia-Bin Huang

Conditional generation

Conditional signal

$y =$ ‘CAT’



$p(\mathbf{x}_T)$

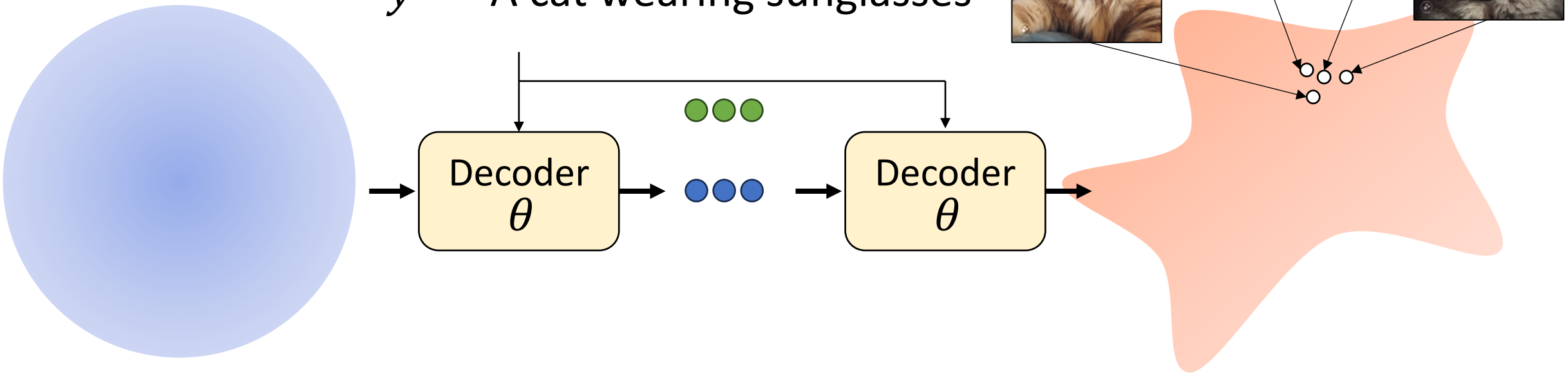
$p_\theta(\mathbf{x}_0)$

Slide from Jia-Bin Huang

Conditional generation

Conditional signal

$y =$ "A cat wearing sunglasses"

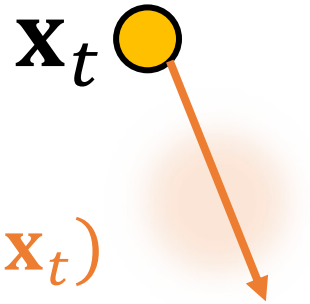


$p(\mathbf{x}_T)$

$p_\theta(\mathbf{x}_0)$

Slide from Jia-Bin Huang

Unconditional generation

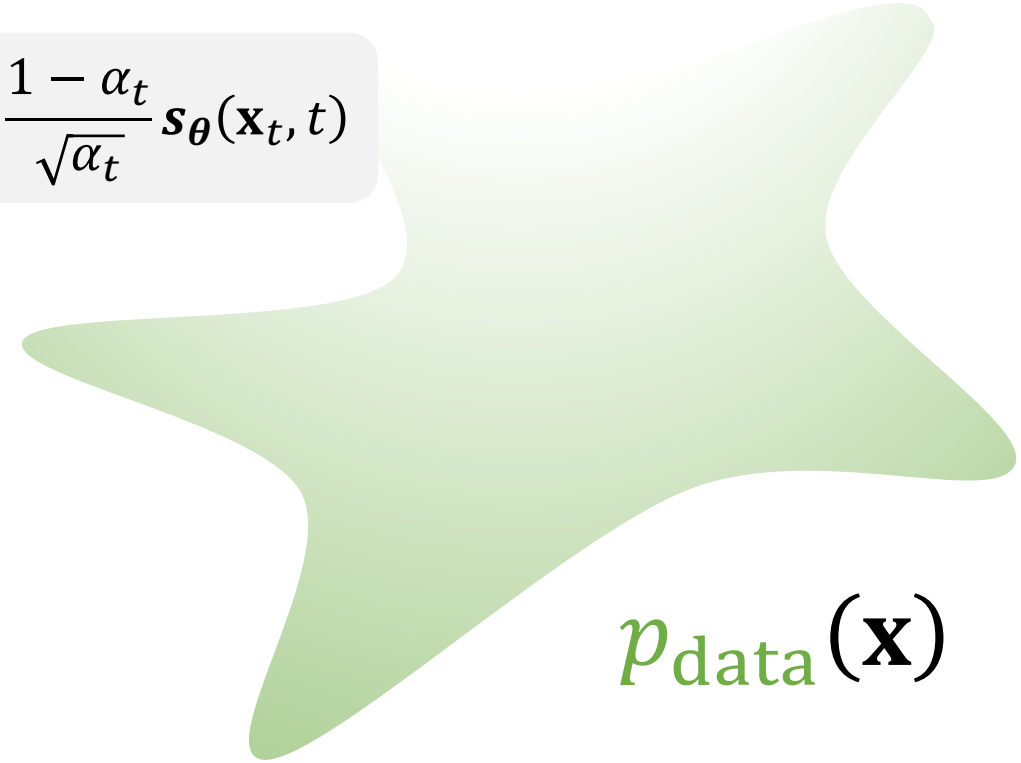


$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}) \approx \nabla \log p(\mathbf{x}_t)$$

Unconditional score

$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

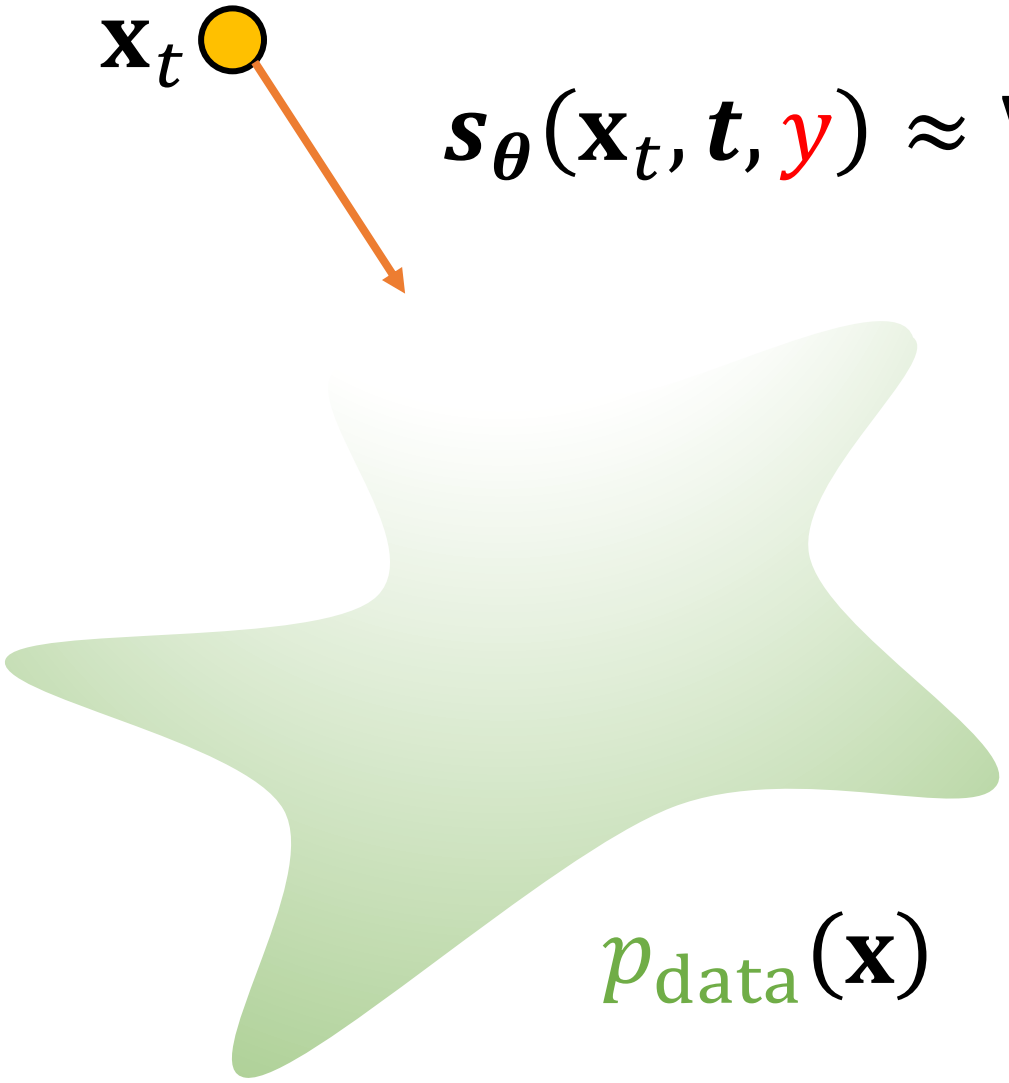
$$\mu_\theta(\mathrm{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$



$p_{\text{data}}(\mathbf{x})$

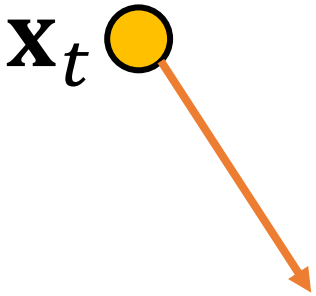
Slide from Jia-Bin Huang

Conditional generation



Slide from Jia-Bin Huang

Conditional generation

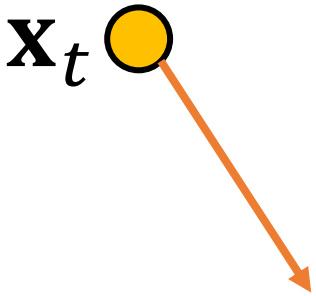


$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t | y)$$

Conditional score

Slide from Jia-Bin Huang

Conditional generation



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

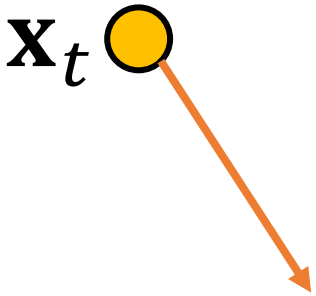
$$\nabla \log p(\mathbf{x}_t|y) = \nabla \log \left(\frac{p(\mathbf{x}_t)p(y|\mathbf{x}_t)}{p(y)} \right)$$

Conditional score

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

$$\nabla \log p(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \nabla \log p(y|\mathbf{x}_t) - \nabla \log p(y)$$

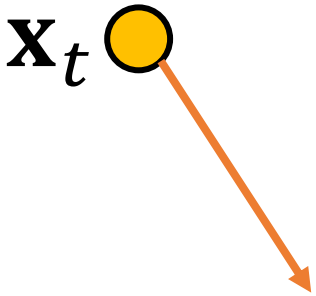
Conditional score

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

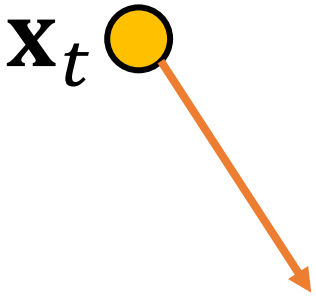
$$\nabla \log p(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

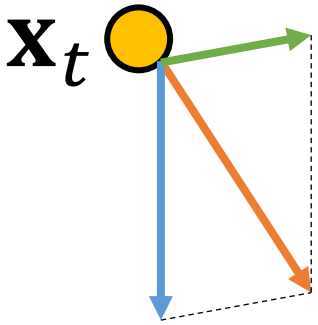
Conditional score | Unconditional score | Adversarial gradient

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

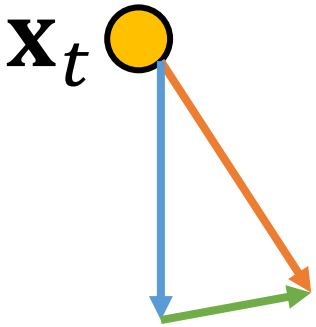
Conditional score | Unconditional score | Adversarial gradient

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance

Classifier guidance scale = 1

Classifier guidance scale = 10

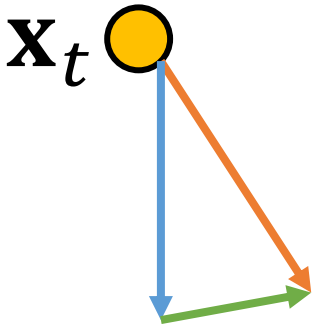
y = “Pembroke Welsh corgi”

Slide from Jia-Bin Huang

[Dhariwal & Nichol 2021]

Conditional generation

Classifier Guidance



$$\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \boldsymbol{t}, y) \approx \nabla \log p(\mathbf{x}_t|y)$$

Conditional score

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

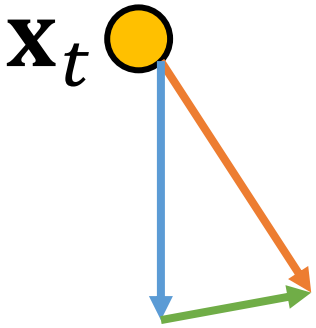
Conditional score | Unconditional score | Adversarial gradient

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

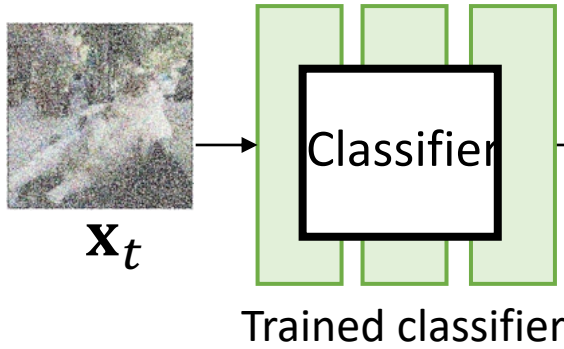
Conditional generation

Classifier Guidance



$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

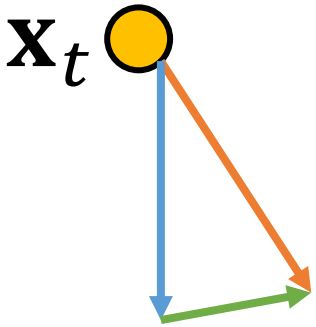
Conditional score | Unconditional score | Adversarial gradient



Slide from Jia-Bin Huang

Conditional generation

Classifier Guidance



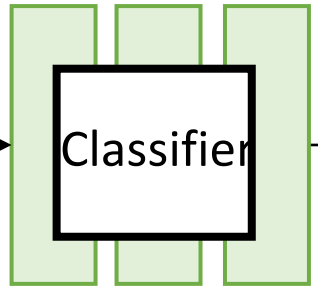
$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\hat{\mathbf{x}}_0)$$

Conditional score | Unconditional score | Adversarial gradient

$$\mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1-\overline{\alpha_t}}\,\hat{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{\overline{\alpha_t}}}$$

$\hat{\mathbf{x}}_0$

Estimated clean image



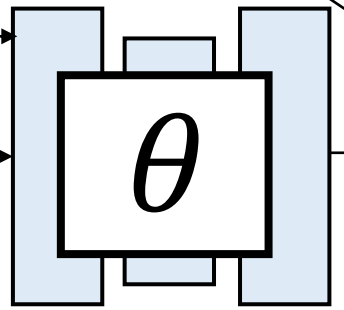
Off-the-shelf classifier

$p(\hat{y}|\hat{\mathbf{x}}_0)$

$p_{\text{data}}(\mathbf{x})$

$\mathbf{x}_t$

t



U-Net

$\hat{\epsilon}_\theta(\mathbf{x}_t, t)$

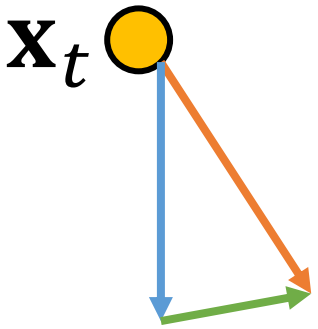
Estimated noise

Slide from Jia-Bin Huang

Universal Guidance [Bansal et al. 2023]

Conditional generation

Classifier Guidance



$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

Slide from Jia-Bin Huang

Conditional generation

$\mathbf{x}_t$

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

$$\nabla \log p(y|\mathbf{x}_t) = \nabla \log \left(\frac{p(\mathbf{x}_t|y)\, p(y)}{p(\mathbf{x}_t)} \right)$$

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

$\mathbf{x}_t$

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

$$\nabla \log p(y|\mathbf{x}_t) = \nabla \log p(\mathbf{x}_t|y) - \nabla \log p(\mathbf{x}_t)$$

$$+ \nabla \log p(y)$$

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

$\mathbf{x}_t$

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(y|\mathbf{x}_t)$$

Conditional score | Unconditional score | Adversarial gradient

$$\nabla \log p(y|\mathbf{x}_t) = \nabla \log p(\mathbf{x}_t|y) - \nabla \log p(\mathbf{x}_t)$$

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

$\mathbf{x}_t$

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma$$

Conditional score | Unconditional score | Adversarial gradient

$$\nabla \log p(y|\mathbf{x}_t) = \nabla \log p(\mathbf{x}_t|y) - \nabla \log p(\mathbf{x}_t)$$

$p_{\text{data}}(\mathbf{x})$

Slide from Jia-Bin Huang

Conditional generation

$$\nabla \log p_\gamma(\mathbf{x}_t|y) = \nabla \log p(\mathbf{x}_t) + \gamma(\nabla \log p(\mathbf{x}_t|y) - \nabla \log p(\mathbf{x}_t))$$

Conditional score

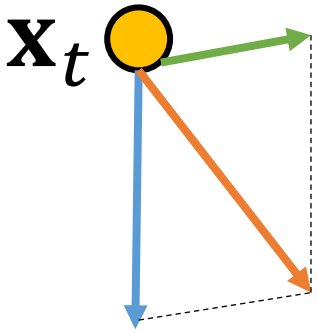
Unconditional score

Adversarial gradient

Slide from Jia-Bin Huang

Conditional generation

Classifier-free Guidance

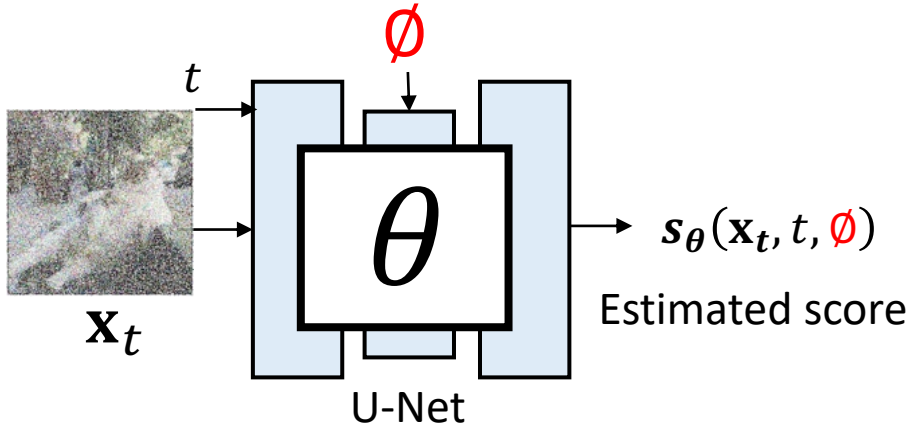


$$\nabla \log p_\gamma(\mathbf{x}_t|y) = (1-\gamma)\nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(\mathbf{x}_t|y)$$

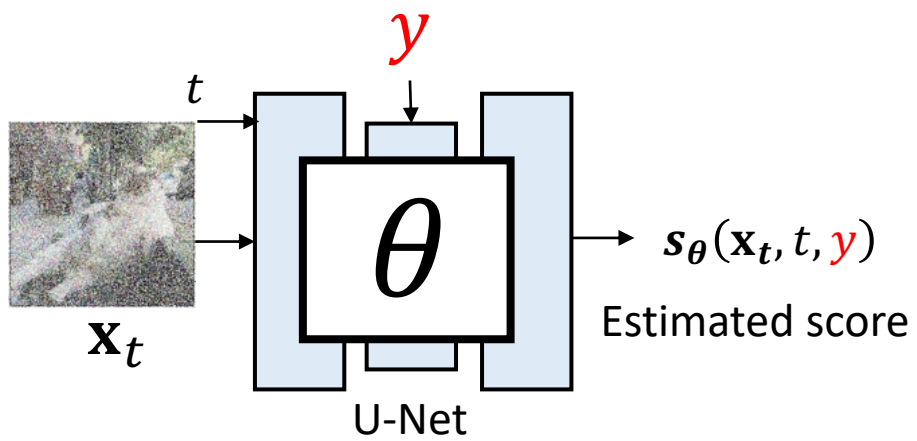
Conditional score

Unconditional score

Conditional score



$p_{\text{data}}(\mathbf{x})$



Slide from Jia-Bin Huang

Conditional generation

Classifier-free Guidance

Unguided samples

Guided samples

Slide from Jia-Bin Huang

[Ho and Salimans 2021]

Classifier guidance

- We can sample from the class-conditional density $q(x_t|c)$ with the help of a pre-trained classifier $P(c|x_t)$
- Bayes rule:

$$\bullet q(x_t|c) \propto P(c|x_t)q(x_t)$$

$$\bullet \log q(x_t|c) = \log P(c|x_t) + \log q(x_t) + \text{const.}$$

$$\bullet \nabla_{x_t} \log q(x_t|c) = \nabla_{x_t} \log P(c|x_t) + \nabla_{x_t} \log q(x_t)$$

conditional score function — obtained from classifier output — unconditional *score function* (pre-trained)

- To sample from class c, steer sample in the modified direction $\nabla_{x_t}[\log q(x_t) + w \log P(c|x_t)]$

Classifier-free guidance

- Instead of training an additional classifier, get an "implicit classifier" by jointly training a conditional and unconditional diffusion model: $P(c|x_t) \propto q(x_t|c)/q(x_t)$
- Both $q(x_t|c)$ and $q(x_t)$ are represented using the same network, trained by dropping out c with some probability
  (corresponding to the unconditional case)
- The modified score function corresponding to this implicit classifier is
- $\nabla_{x_t}[\log q(x_t) + w \log P(c|x_t)] = \quad \nabla_{x_t}[\log q(x_t) + w(\log q(x_t|c) - \log q(x_t))]$

Sample is steered away from the unconditional distribution in the direction of the conditional one

J. Ho and T. Salimans. Classifier-Free Diffusion Guidance. arXIv 2021

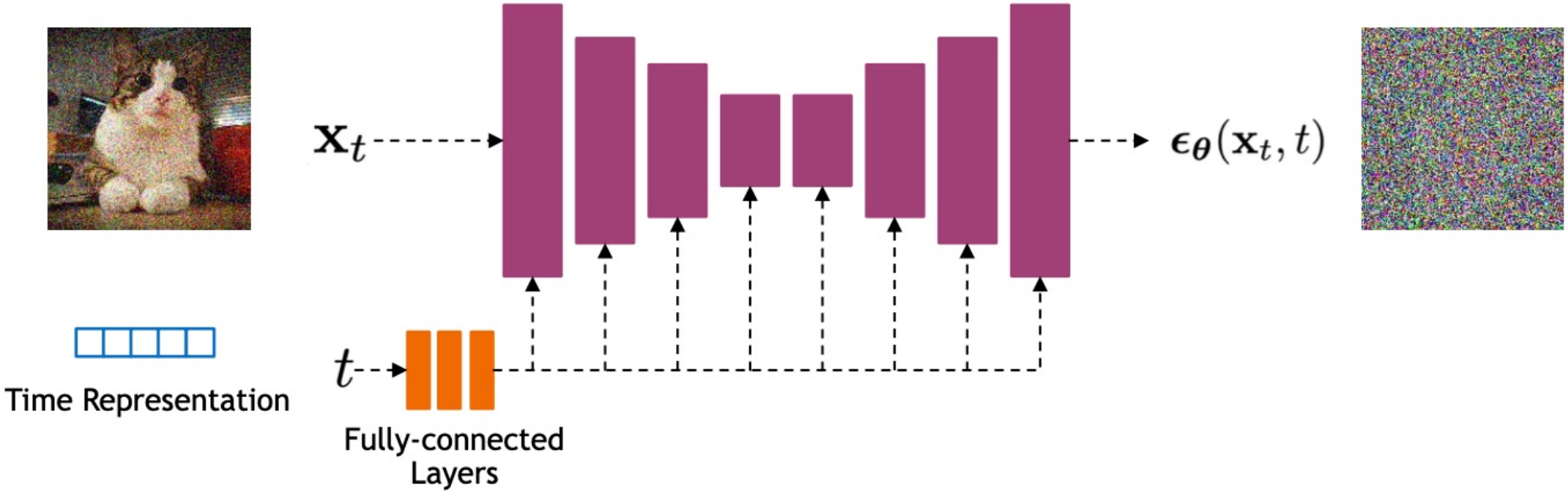
Classifier-free guidance

Figure 1: Classifier-free guidance on the malamute class for a 64x64 ImageNet diffusion model. Left to right: increasing amounts of classifier-free guidance, starting from non-guided samples on the left.

J. Ho and T. Salimans. Classifier-Free Diffusion Guidance. arXIv 2021

DDPMs: Implementation

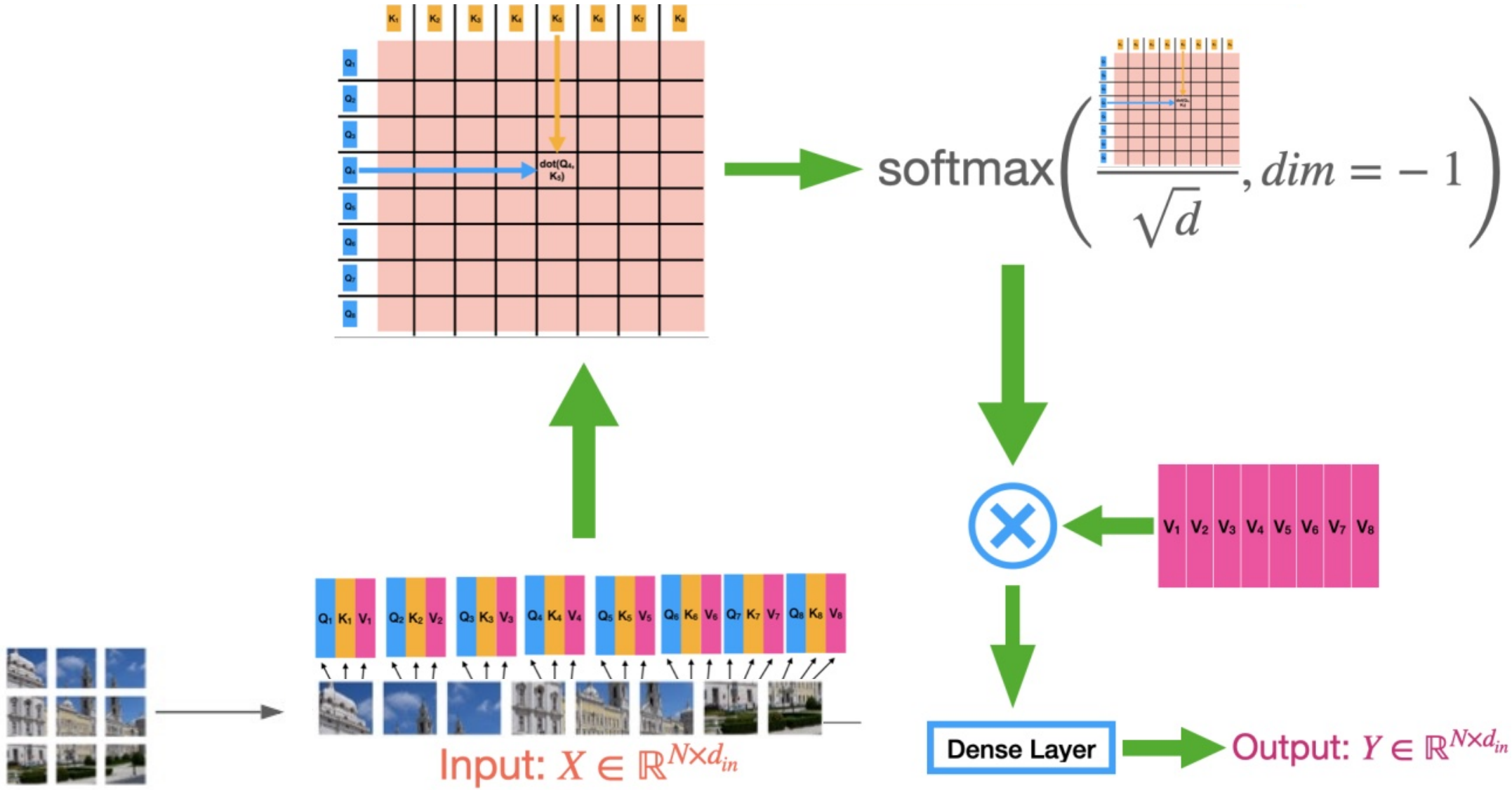
- U-Net architectures are typically used to represent $\epsilon_\theta(x_t, t)$
  - Bells and whistles: residual blocks, self-attention



- Time is encoded using sinusoidal positional embeddings or random Fourier features, fed into the U-Net using addition or adaptive normalization

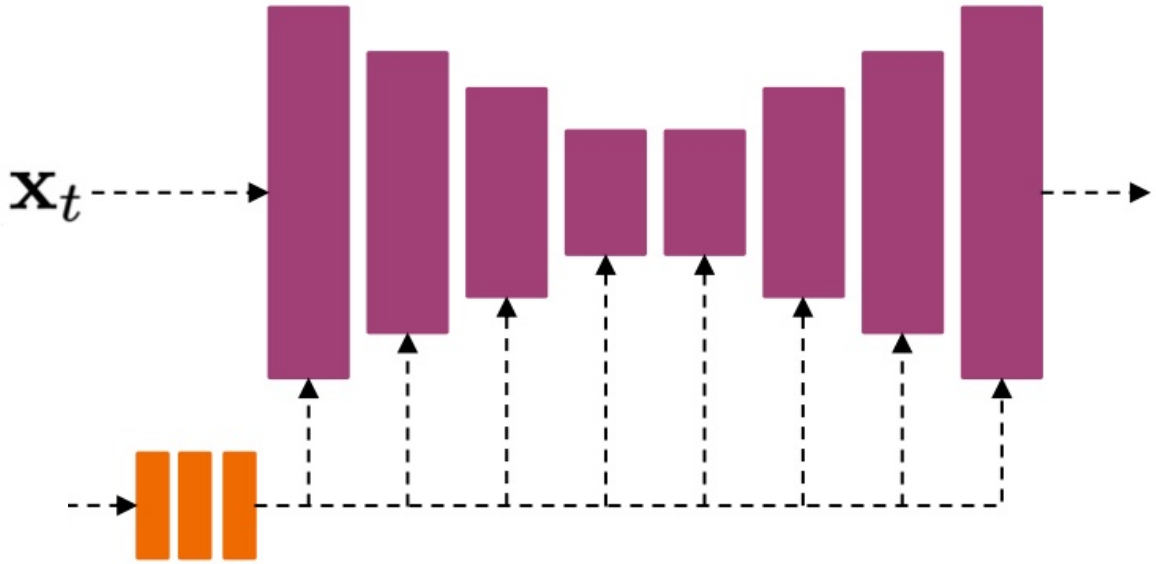
Source: CVPR 2002 DM tutorial

Attention Module

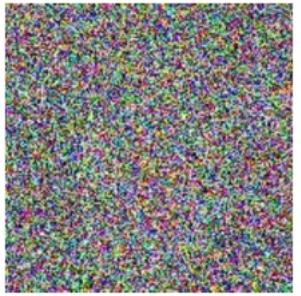


Text-guided diffusion

- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*



$\epsilon_\theta(\mathbf{x}_t, \mathbf{c}, t)$



Conditioning information:
text c, time *t*

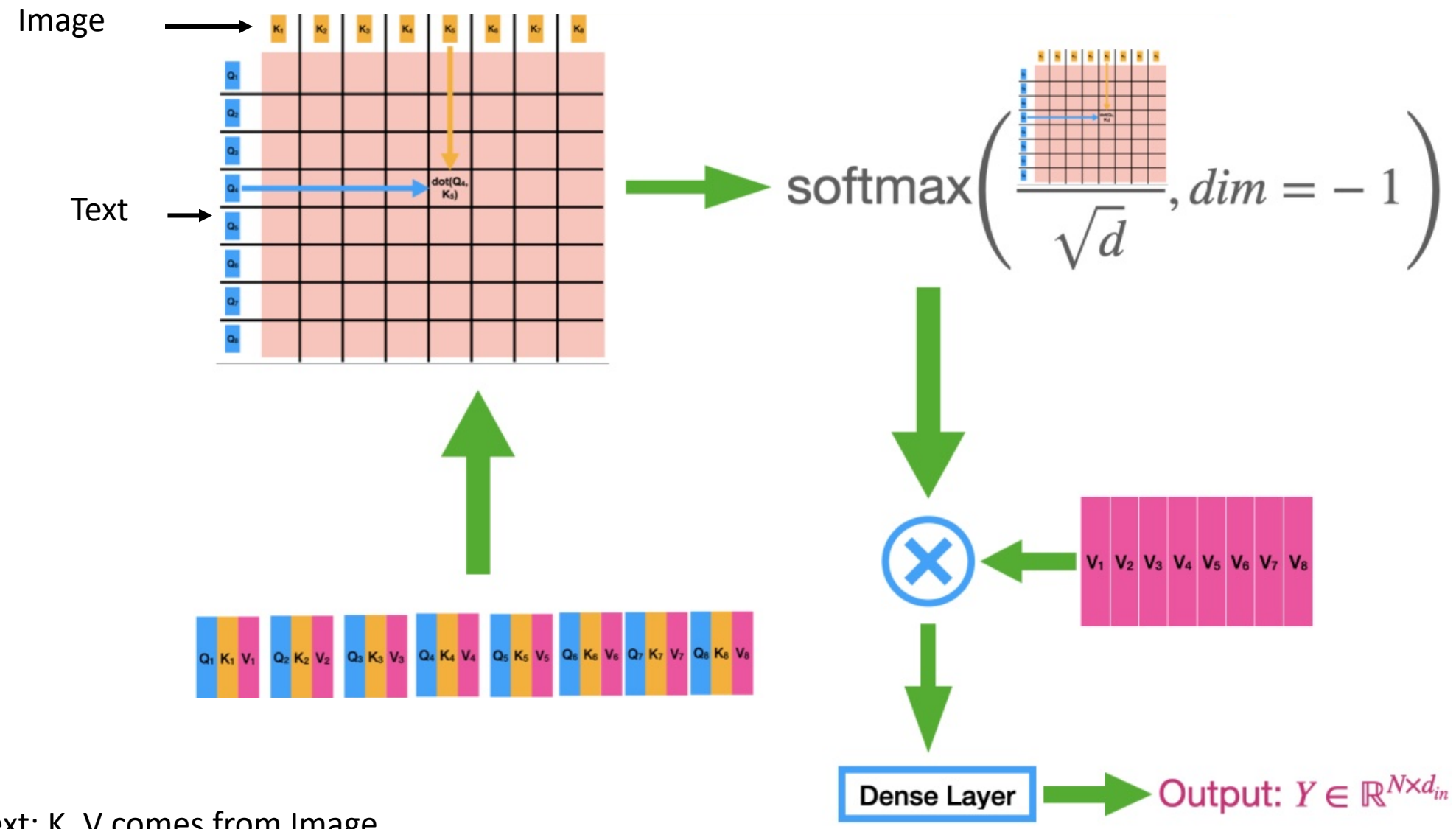
Image source

Slide from Jia-Bin Huang

Text-guided diffusion

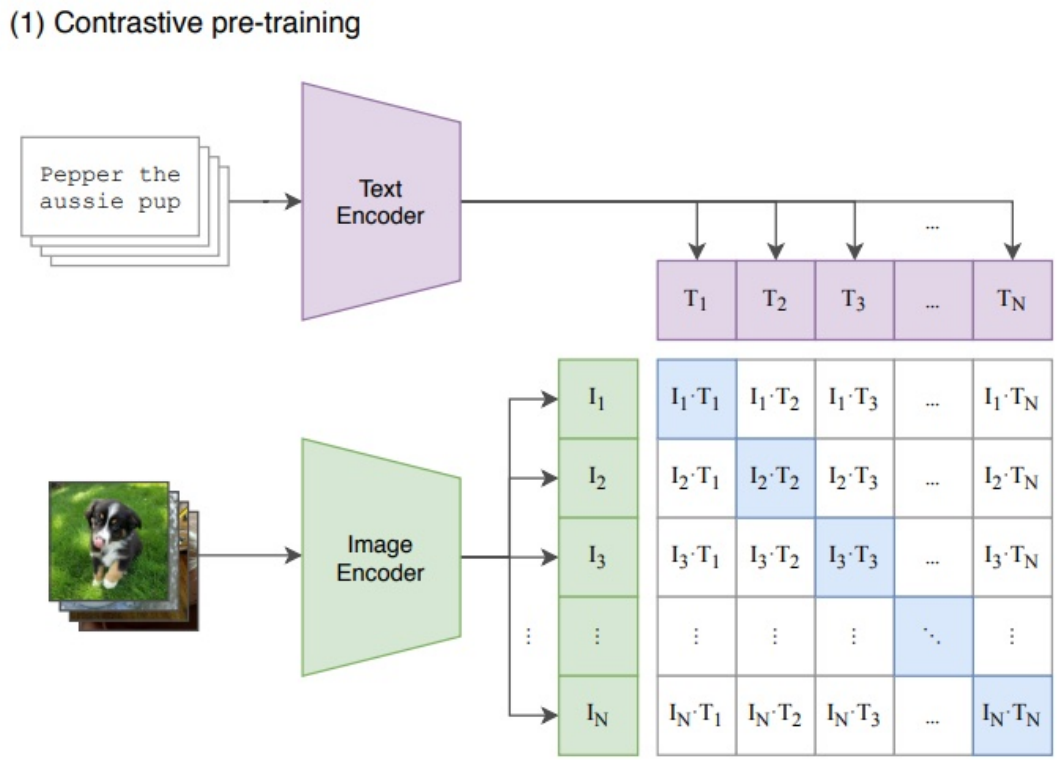
- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*
- Classifier-free guidance works the same way as before, by training both conditional and unconditional models using text dropout
- CLIP guidance: steer samples in the direction of $\nabla_{x_t} \text{CLIP}(x_t, c)$
- Note: both classifier and CLIP must be *noise-aware* (trained on noised images)

Cross Attention Module



Q comes from Text; K, V comes from Image

CLIP



Contrastive objective: in a batch of N image-text pairs, classify each text string to the correct image and vice versa

A. Radford et al., Learning Transferable Visual Models From Natural Language Supervision, ICML 2021
https://openai.com/blog/clip/

Text-guided diffusion

- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*

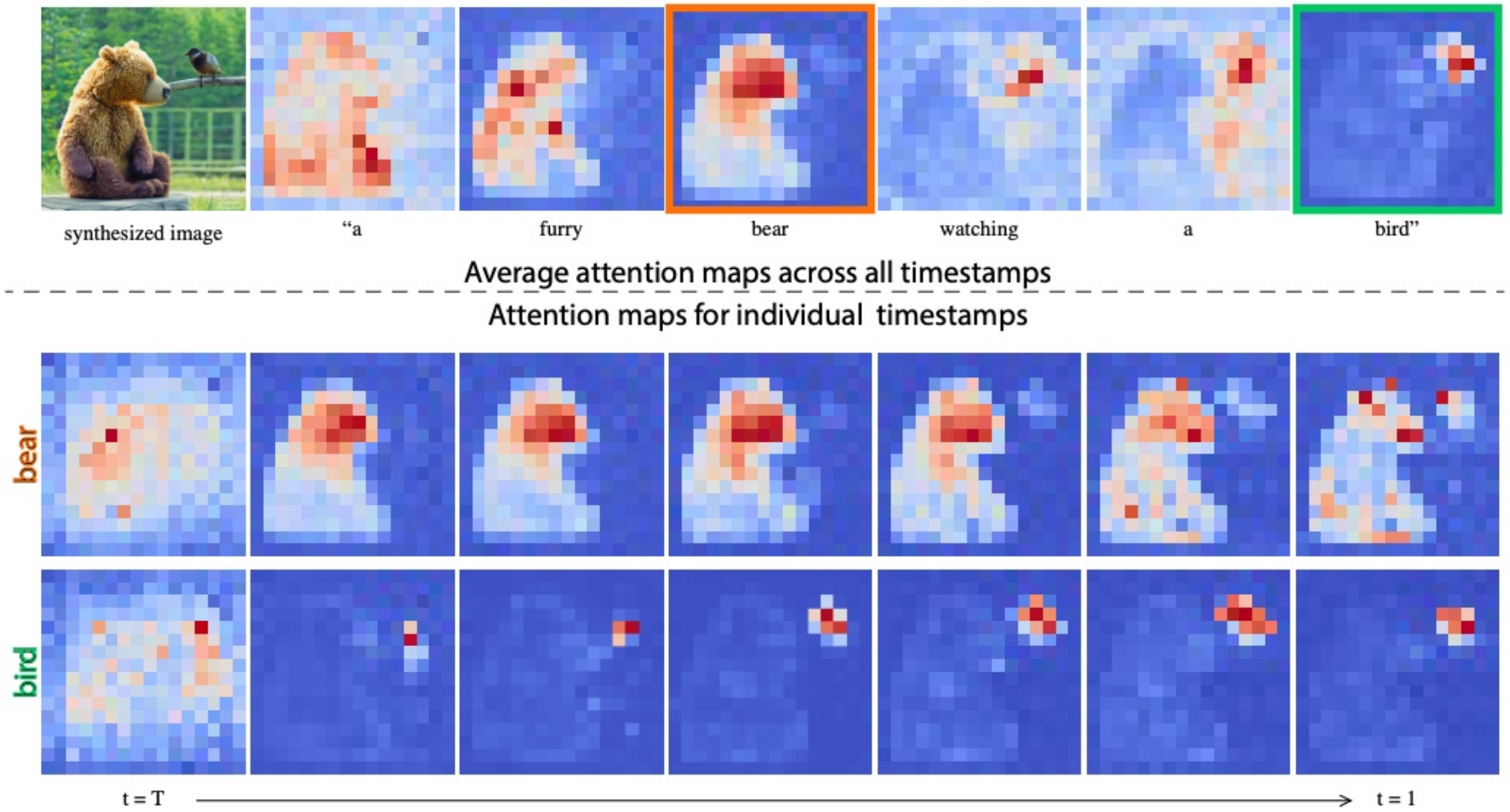
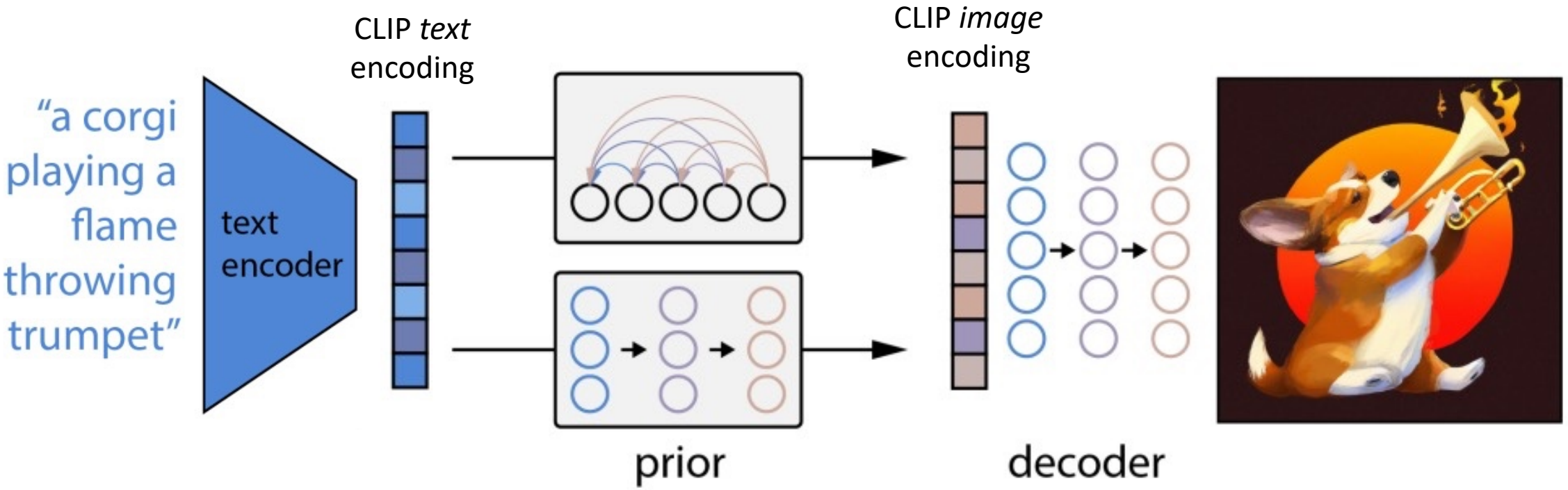


Image source

DALL-E 2

A. Ramesh et al. Hierarchical text-conditional image generation with CLIP latents. Preprint 2022

DALL-E 2



Generative model to produce CLIP *image* encoding given CLIP *text* encoding

Diffusion model (GLIDE) conditioned on CLIP image embedding and text prompt

Generate at 64x64, upsample to 256x256, then upsample to 1024x1024

DALL-E 2: Results

Figure 19: Random samples from unCLIP for prompt "A close up of a handpalm with leaves growing from it."

Figure 18: Random samples from unCLIP for prompt "Vibrant portrait painting of Salvador Dali with a robotic half face"

DALL-E 2: Results

Figure 3: Variations of an input image by encoding with CLIP and then decoding with a diffusion model. The variations preserve both semantic information like presence of a clock in the painting and the overlapping strokes in the logo, as well as stylistic elements like the surrealism in the painting and the color gradients in the logo, while varying the non-essential details.

DALL-E 2: Results

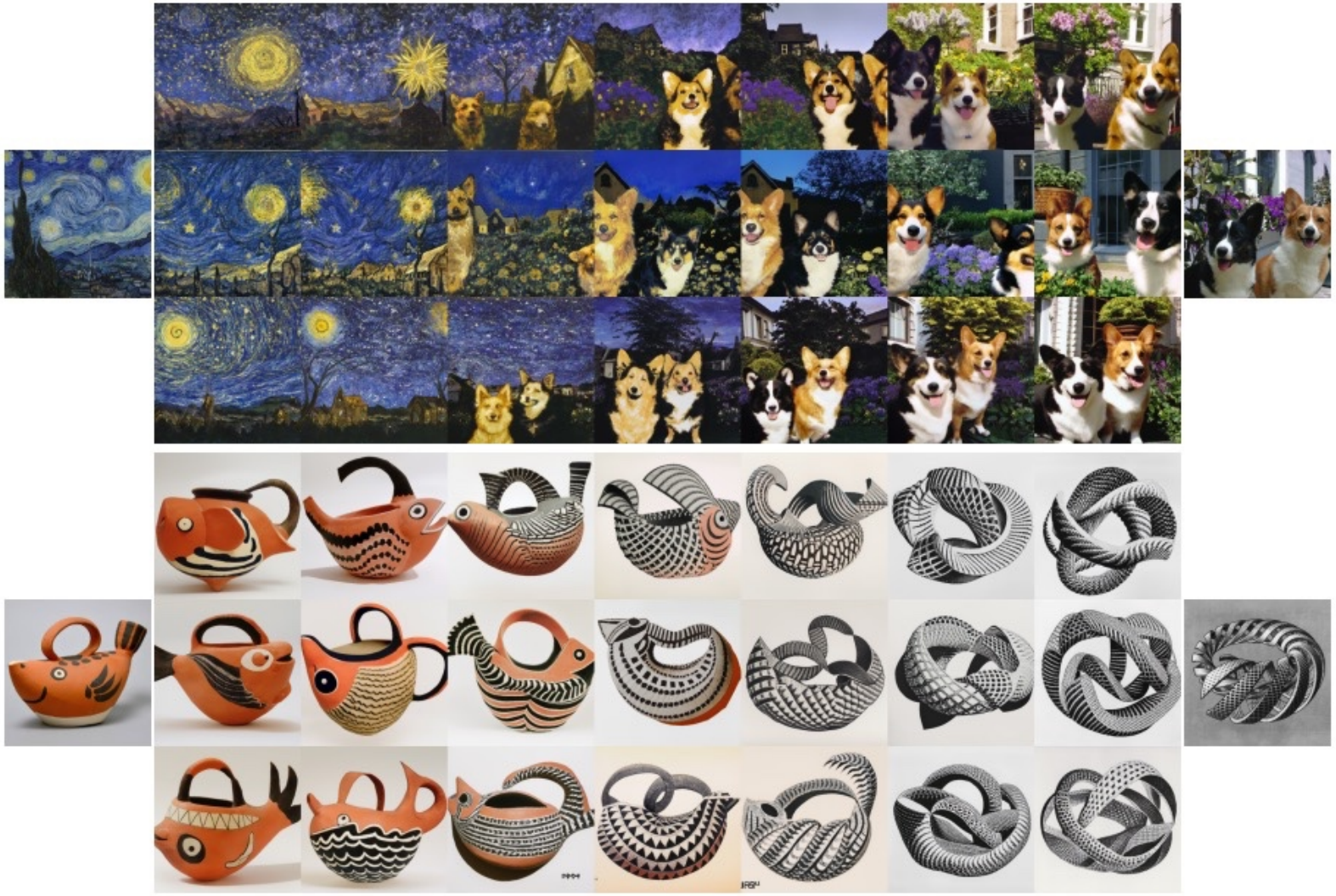


Figure 4: Variations between two images by interpolating their CLIP image embedding and then decoding with a diffusion model. We fix the decoder seed across each row. The intermediate variations naturally blend the content and style from both input images.

DALL-E 2: Limitations

Figure 15: Reconstructions from the decoder for difficult binding problems. We find that the reconstructions mix up objects and attributes. In the first two examples, the model mixes up the color of two objects. In the rightmost example, the model does not reliably reconstruct the relative size of two objects.

Google Imagen (not public)

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.

A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.

Teddy bears swimming at the Olympics 400m Butterfly event.

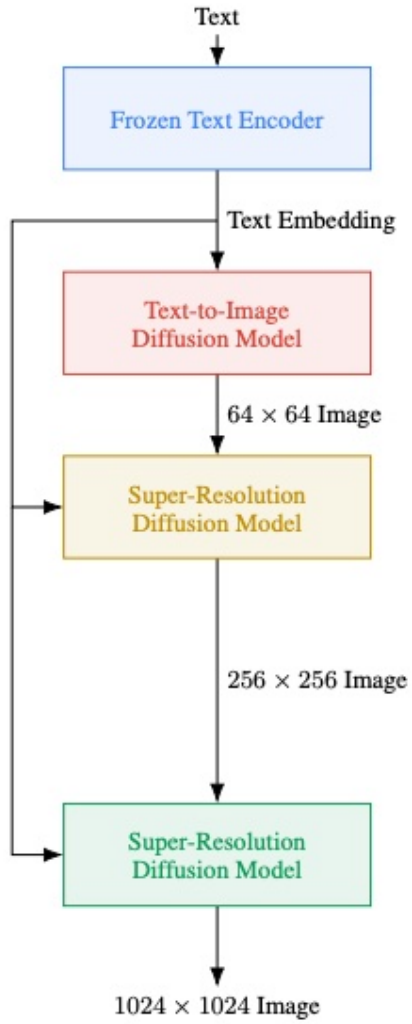
A cute corgi lives in a house made out of sushi.

A cute sloth holding a small treasure chest. A bright golden glow is coming from the chest.

C. Saharia et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. NeurIPS 2022

Google Imagen: Details

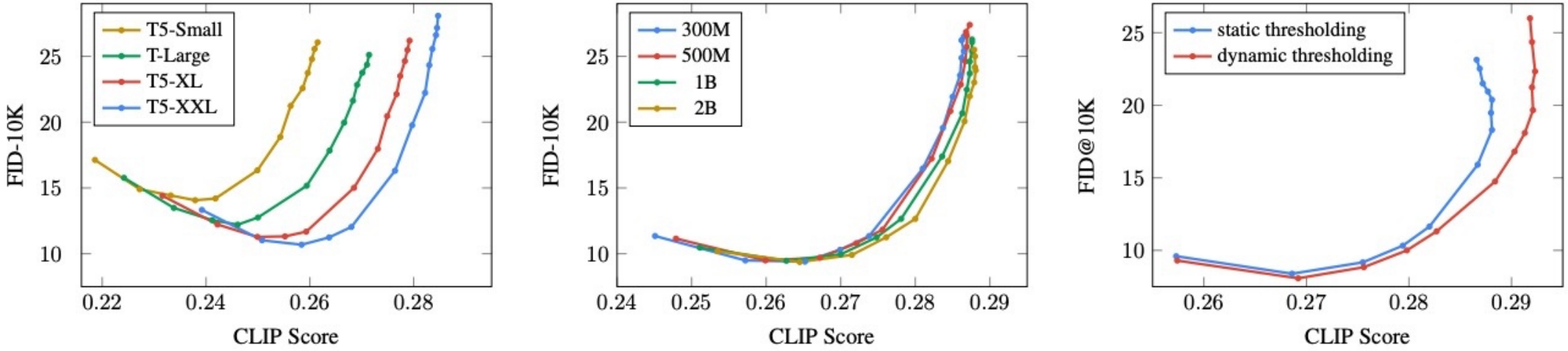
- Text encoder is a large language model (4.6B parameters) trained on text only
- Diffusion model to generate at 64x64, upsample to 256x256, then 1024x1024
  - Architecture: *efficient U-Net* (2B parameters): more parameters at lower resolutions, convolutions *after* downsampling and *before* upsampling
  - Classifier-free guidance with a *dynamic thresholding* technique, enabling good generation quality with high guidance weights
  - Training dataset: 460M image-text pairs (internally collected), 400M pairs from the LAION dataset



C. Saharia et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. NeurIPS 2022

Google Imagen: Evaluation

- Impact of model size, implementation choices



(a) Impact of encoder size. (b) Impact of U-Net size. (c) Impact of thresholding.

Curves are obtained by varying guidance weight

FID evaluated on COCO dataset by sampling prompts and generating images using the same prompts

Google Imagen: Evaluation

- Human evaluation on DrawBench (set of 200 prompts)

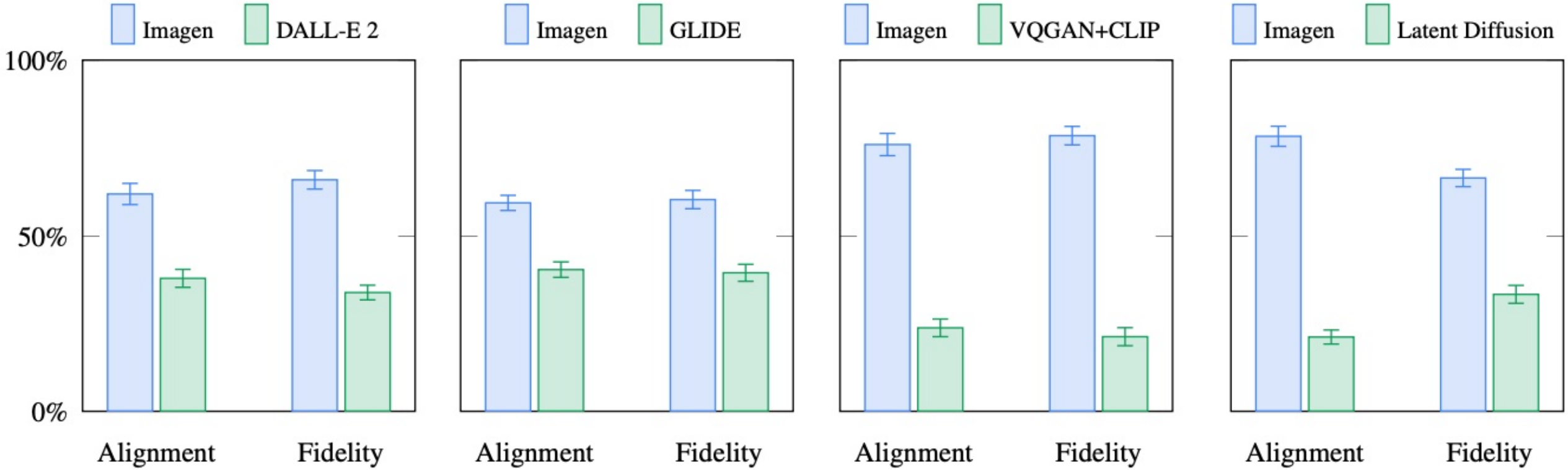


Imagen vs. DALL-E 2

“A yellow book and a red vase”

Imagen vs. DALL-E 2

“A black apple and a green backpack”

Imagen is better than DALL-E 2 in assigning the colors to the objects

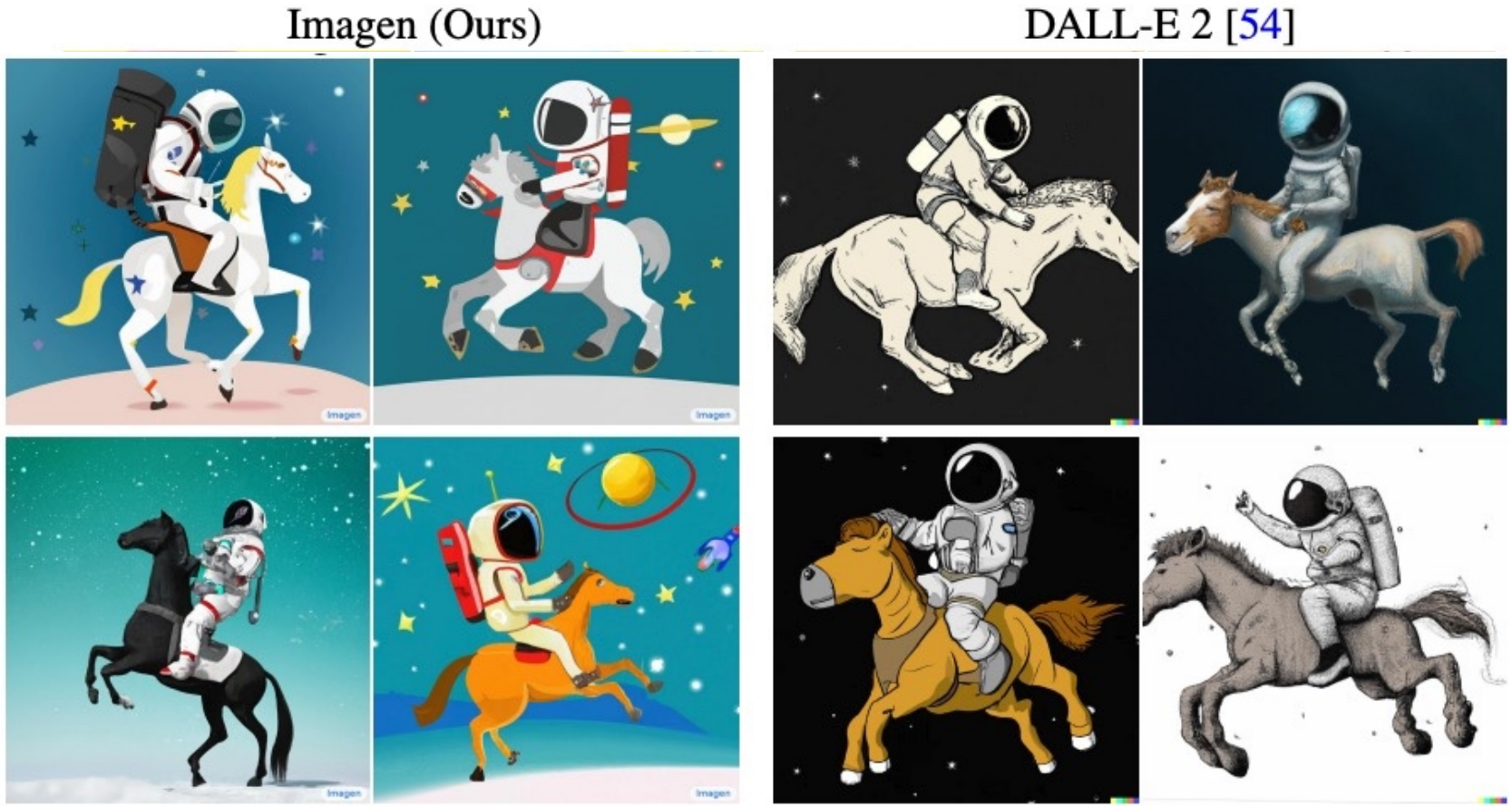
Imagen vs. DALL-E 2

“A storefront with Text to Image written on it”

Imagen vs. DALL-E 2

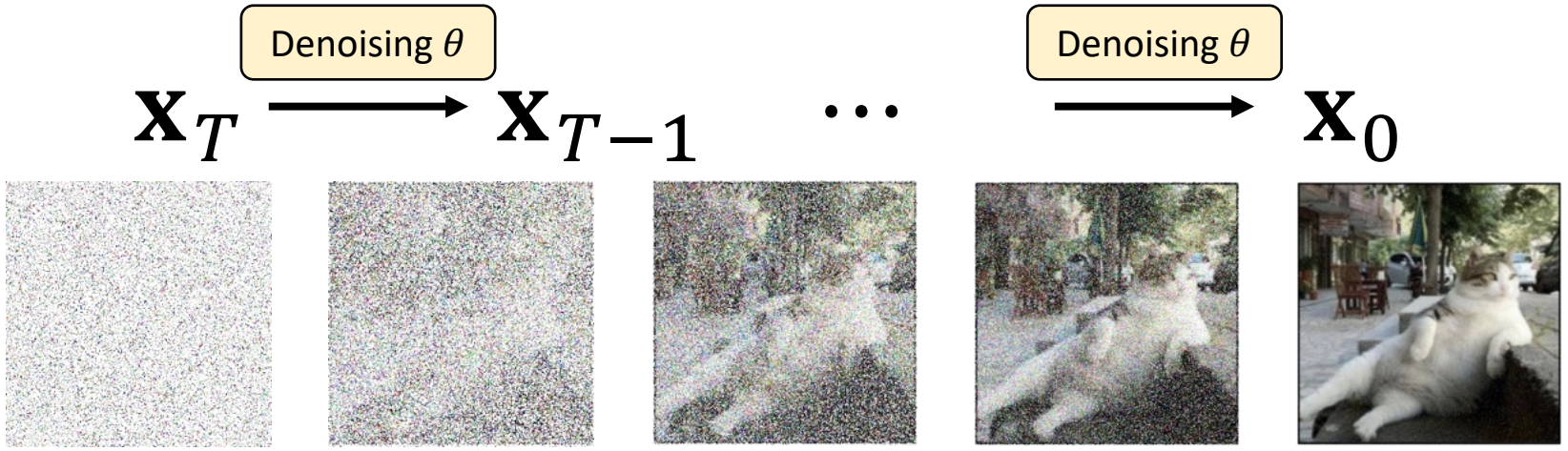
“A panda making latte art”

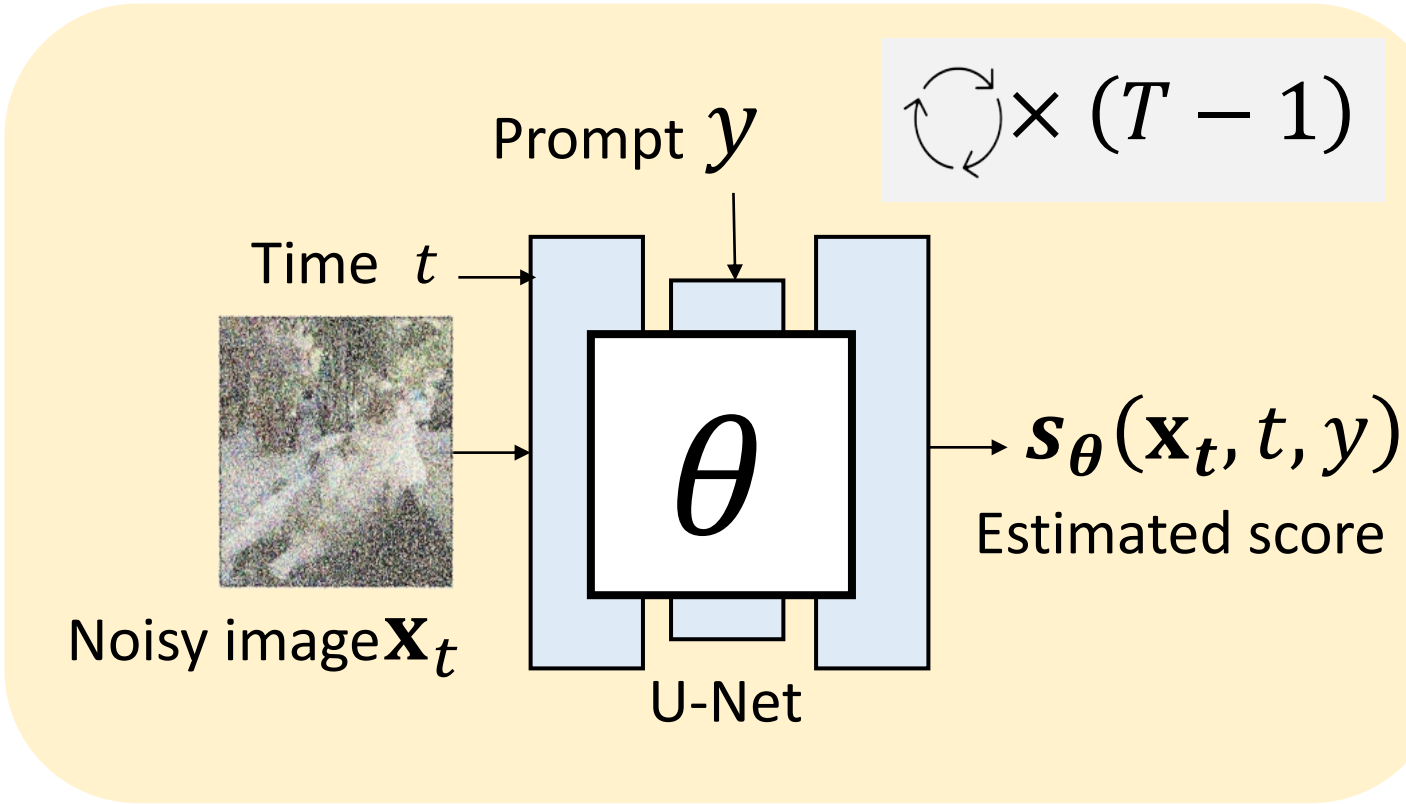
Imagen vs. DALL-E 2



“A horse riding an astronaut”

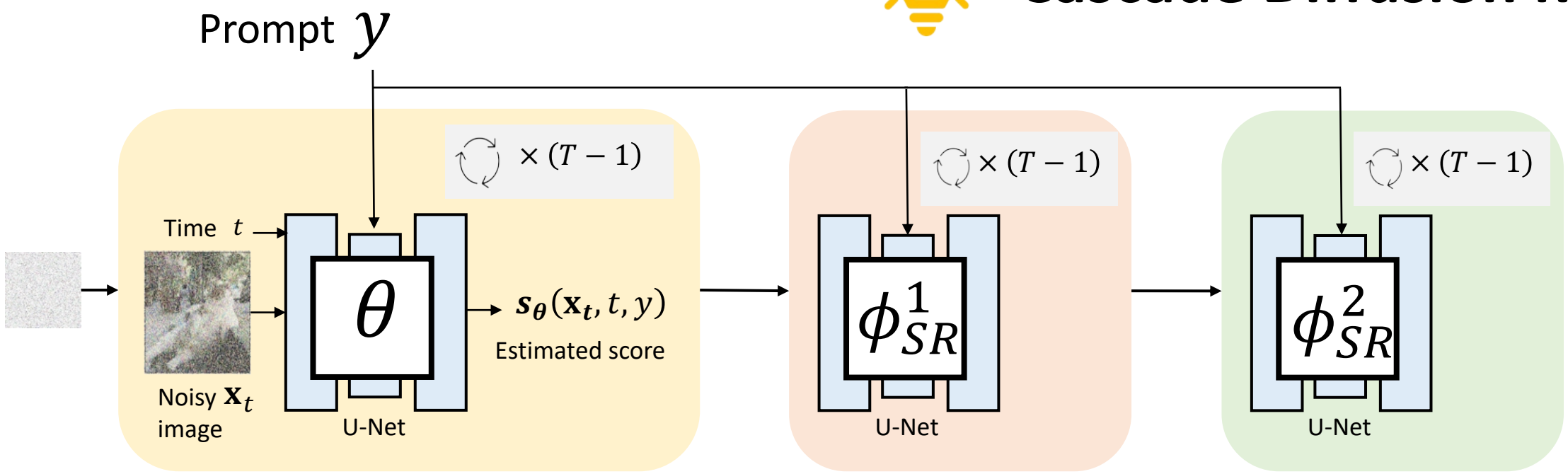
Challenge: Resolution





Slide from Jia-Bin Huang

Cascade Diffusion Models

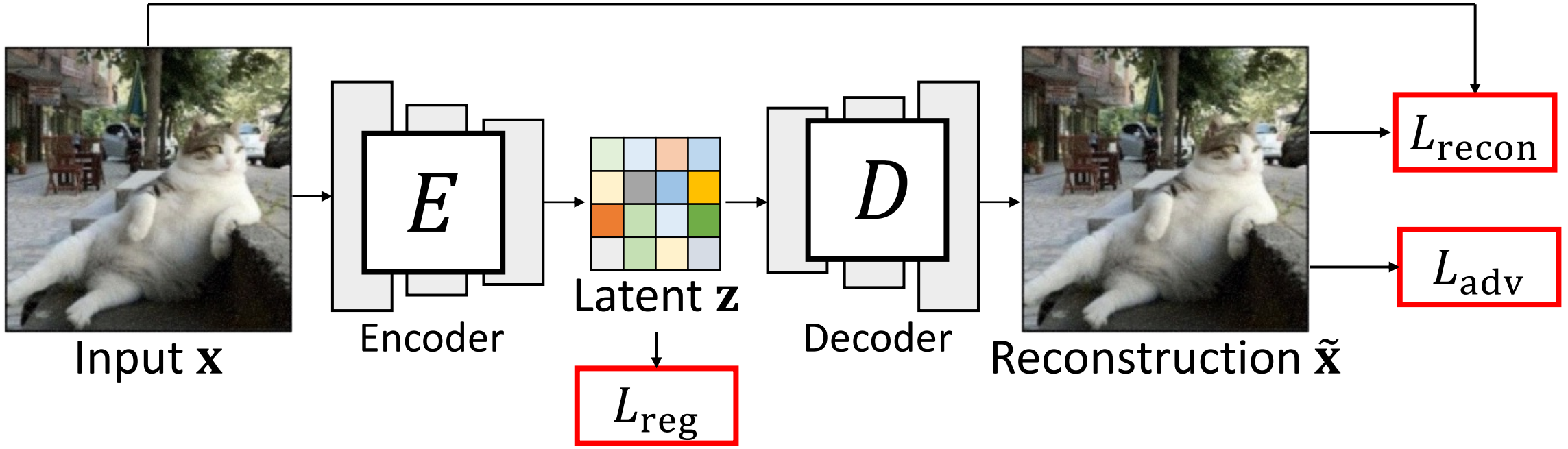


Text-to-image Diffusion Model **SR Diffusion Model** **SR Diffusion Model**

[Ho et al. 2021] [Imagen: Saharia et al. 2022]

Slide from Jia-Bin Huang

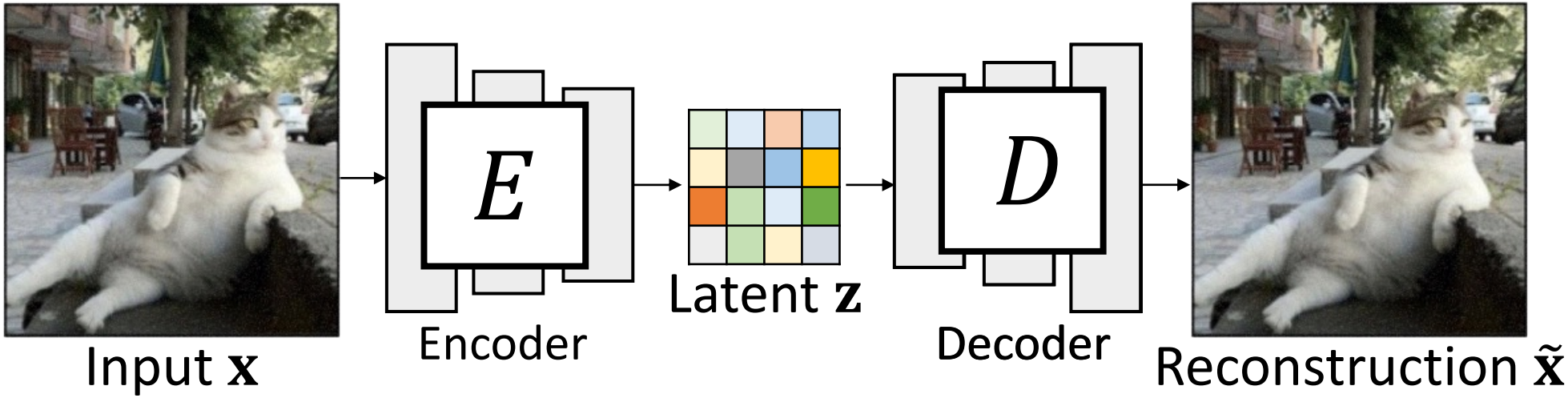
Latent Diffusion Models

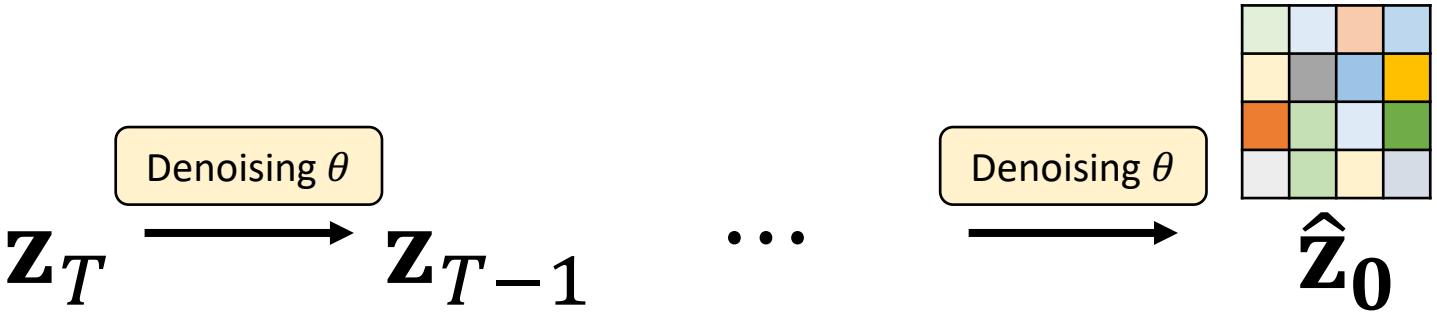


[Rombach et al. 2022] [Vahdat et al. 2021]

Slide from Jia-Bin Huang

Latent Diffusion Models





Slide from Jia-Bin Huang

Latent Diffusion Models

Input $\mathbf{x}$

E

Encoder

Latent $\mathbf{z}$

D

Decoder

Reconstruction $\tilde{\mathbf{x}}$

Denoising θ

$\mathbf{z}_T$ → $\mathbf{z}_{T-1}$ ⋯

Denoising θ

$\hat{\mathbf{z}}_0$

D

Decoder

Sample

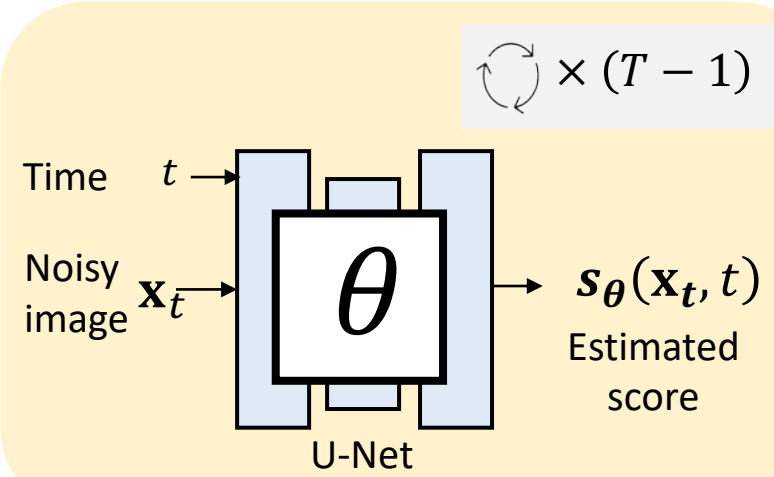
Slide from Jia-Bin Huang

End-to-end Diffusion Models

$$\mathbf{x}_T \xrightarrow{\text{Denoising } \theta} \mathbf{x}_{T-1} \cdots \xrightarrow{\text{Denoising } \theta} \mathbf{x}_0$$

Adjusted noise schedules

[Chen et al. 2023] [Hoogeboom et al. 2023]



Multi-scale loss

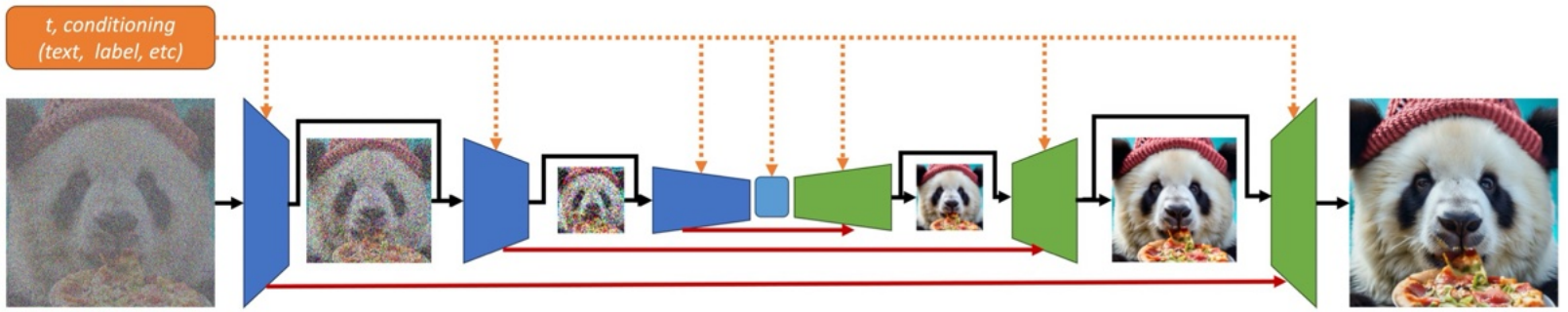
simple diffusion [Hoogeboom et al. 2023]

$$\boldsymbol{L_\theta}(\mathbf{x}) \longrightarrow \sum_s \frac{1}{s} \boldsymbol{L_\theta^{s\times s}}(\mathbf{x})$$

Single-scale → Multi-scale

Progressive training

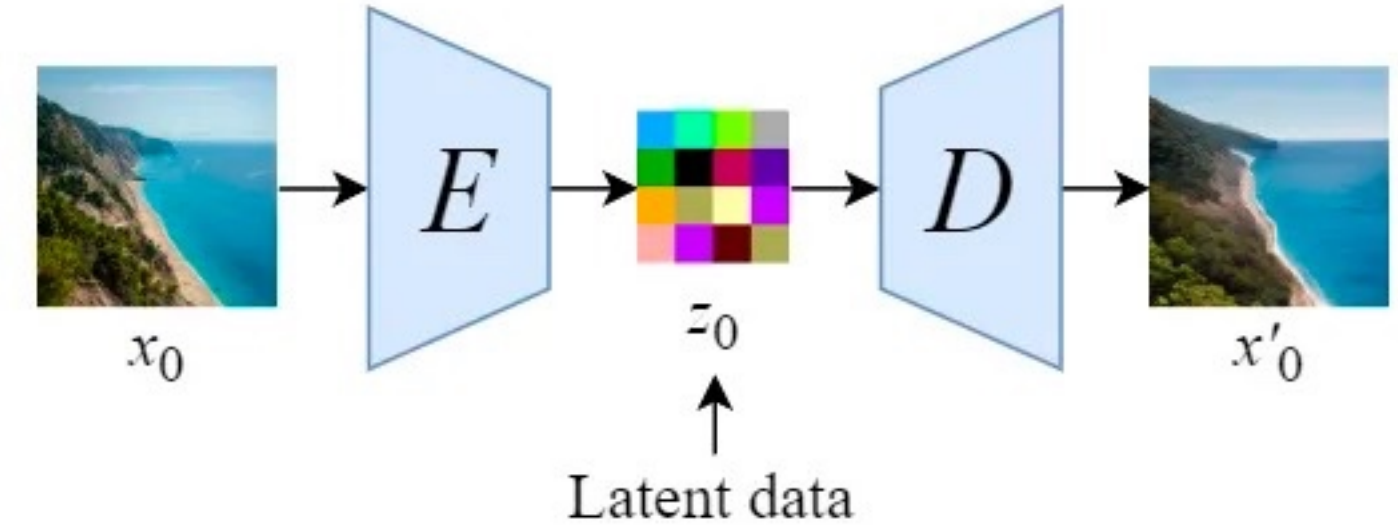
Matryoshka Diffusion Models [Gu et al. 2023]



Slide from Jia-Bin Huang

Latent diffusion model (basis of Stable Diffusion)

- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional diffusion model in latent space

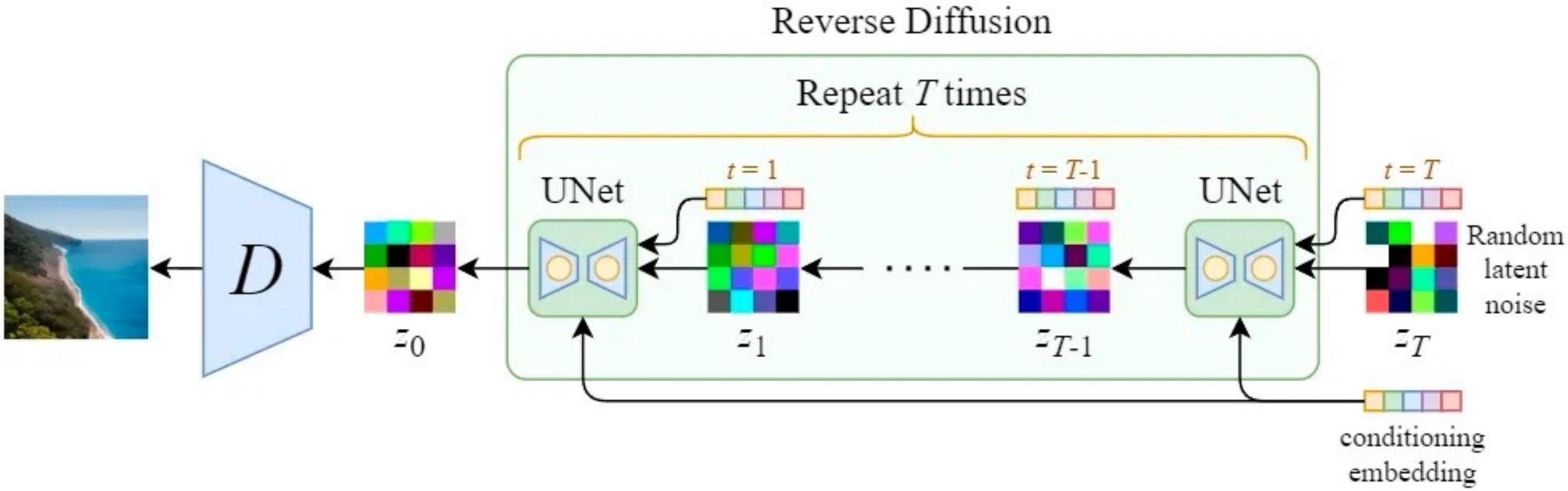


R. Rombach et al. High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022

https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)

- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional

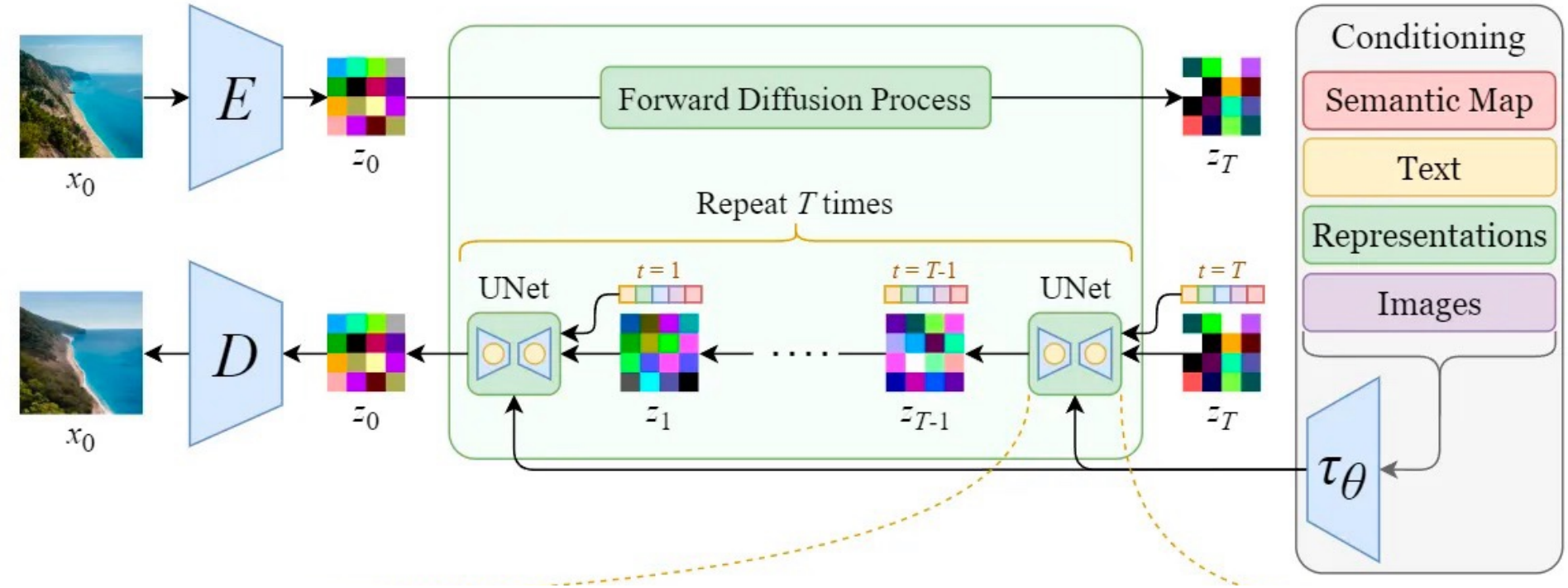


https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)

- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional diffusion model in latent space

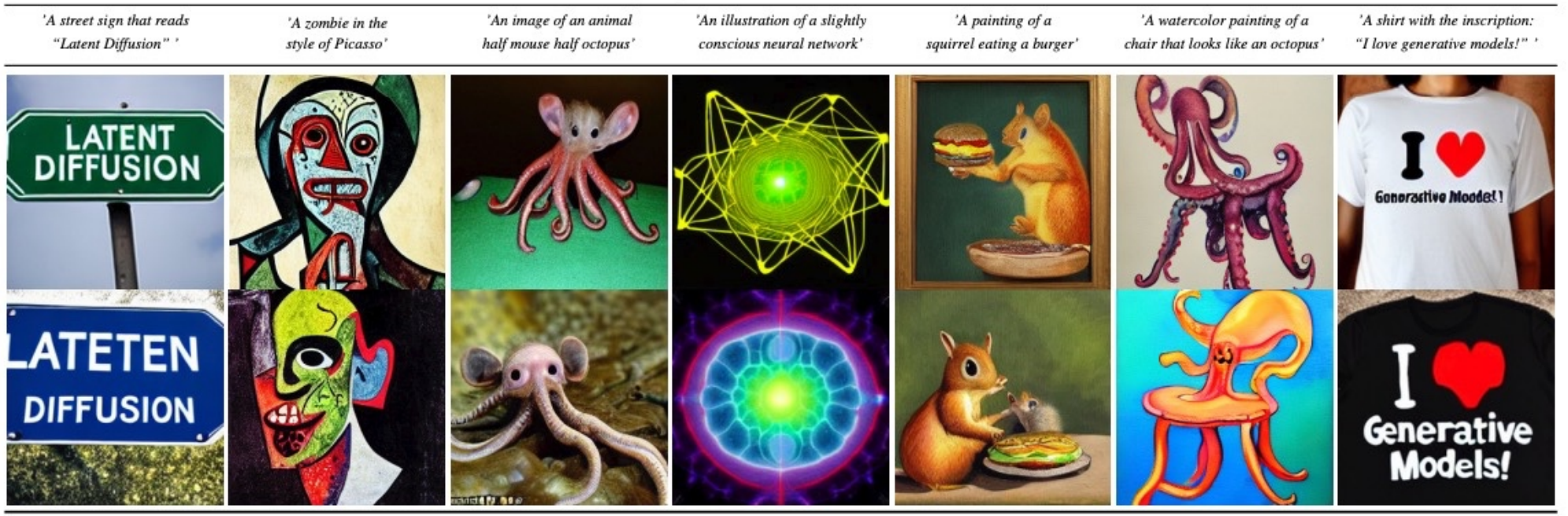
Close-up of U-Net: Conditioning information incorporated using cross-attention



https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)

Text-to-Image Synthesis on LAION. 1.45B Model.



Challenge: Inference Speed

Denoising Diffusion Implicit Models (DDIMs)

- **DDIM roughly sketches the final sample, then refine it with the reverse process**
  - **Key idea:**
    - Given $\mathbf{x}_t$, generate the rough sketch $\mathbf{x}_0$ and refine $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$[1]
    - Unlike original diffusion model, it is not a Markovian structure

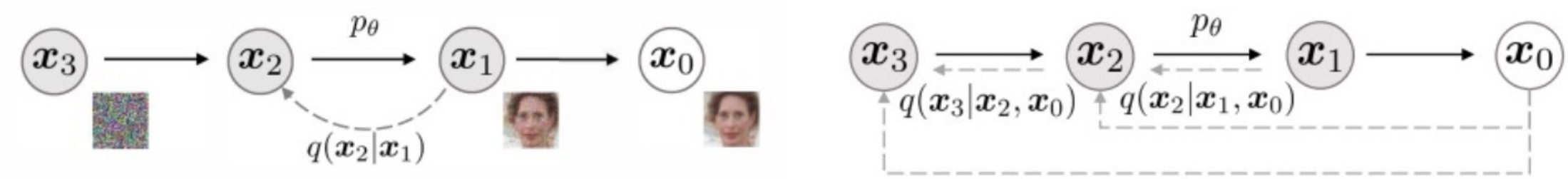


Figure 1: Graphical models for diffusion (left) and non-Markovian (right) inference models.

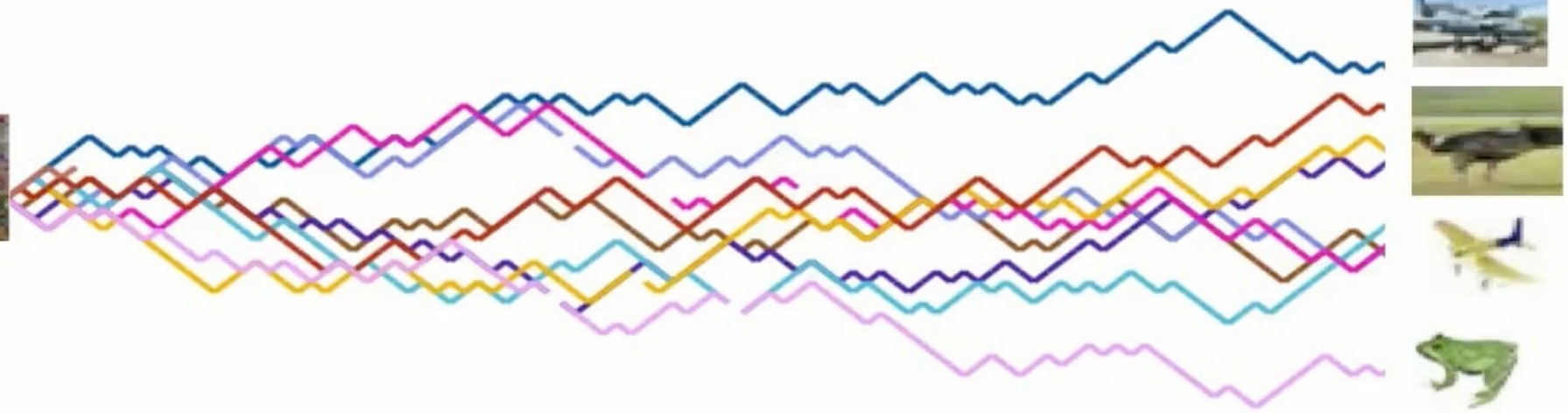
Sangwoo Mo, Introduction to Diffusion Models

DDIM follows a deterministic process

DDPM follows stochastic process

DDPM

x_T



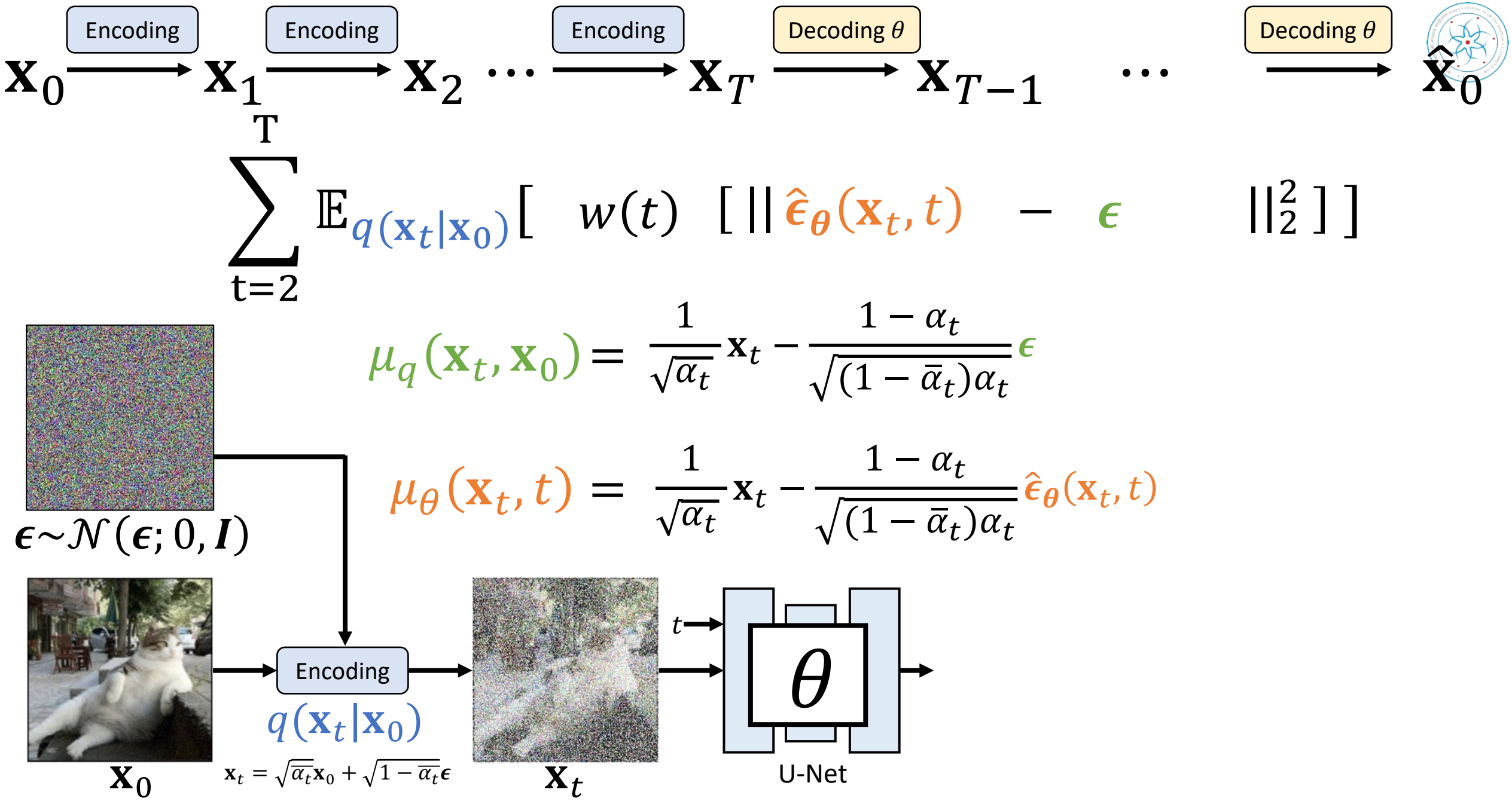
DDIM follows deterministic process

DDIM

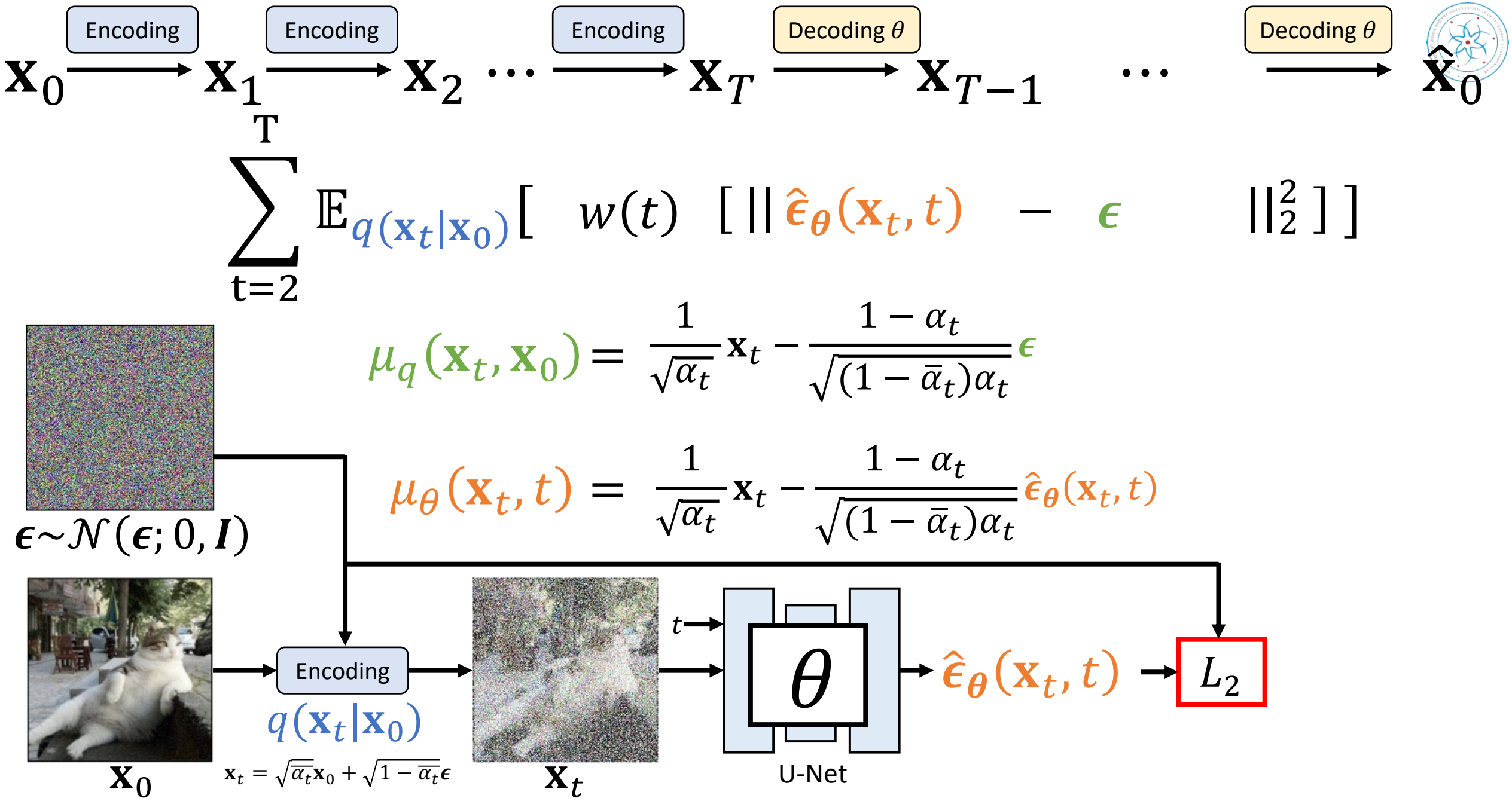
x_T

The same original noise $\boldsymbol{x_T}$ to same image $\boldsymbol{x_0}$

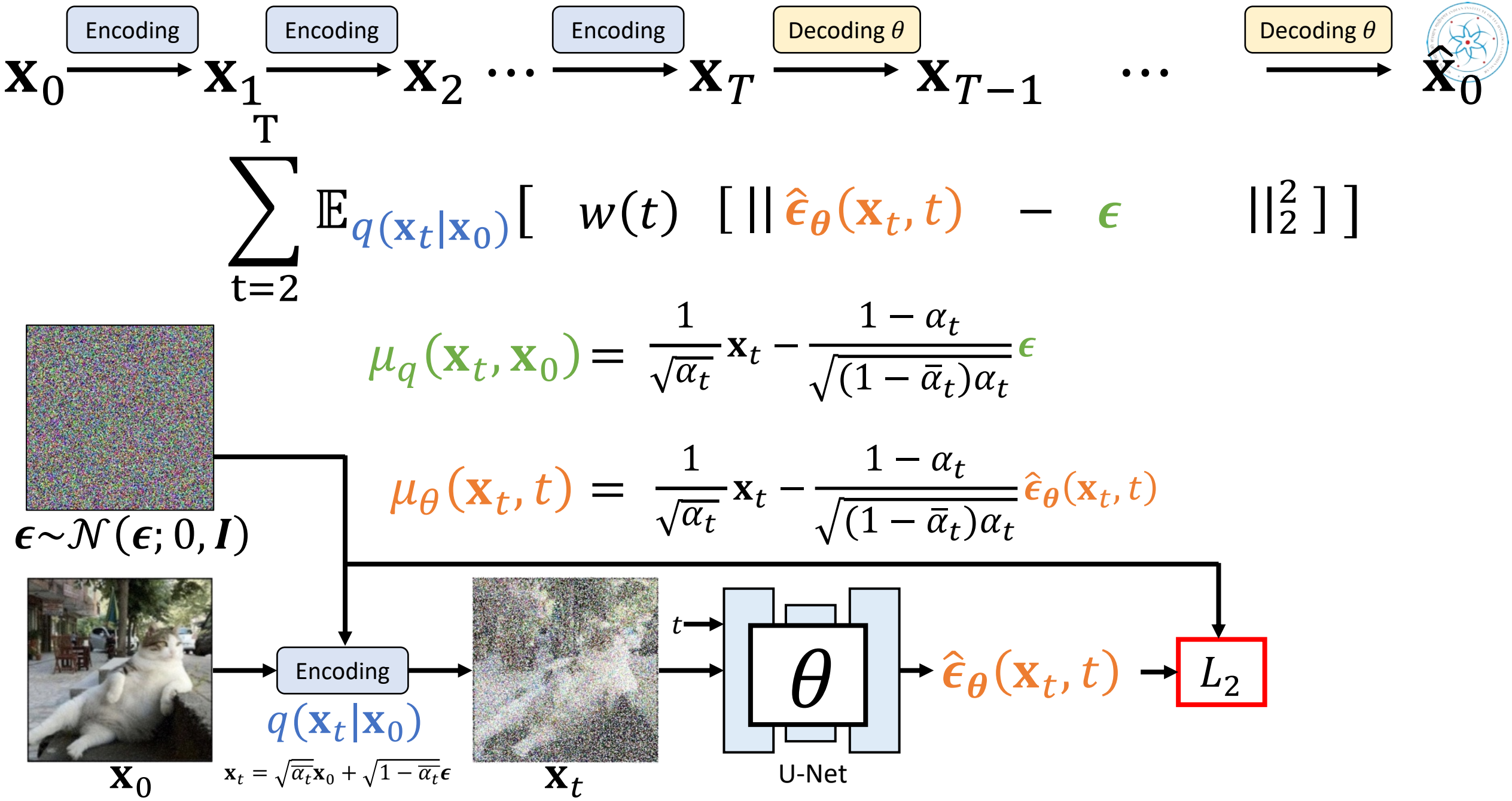
Tidiane Ndir, presentation on Faster diffusion, Uni of Freiburg



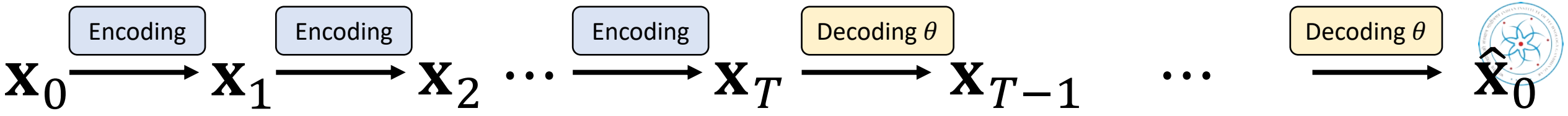
Slide from Jia-Bin Huang



Slide from Jia-Bin Huang



Slide from Jia-Bin Huang



$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2 \right] \right]$$

① $\mathbf{x}_t = \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}$

② $\mu_q(\mathbf{x}_t, \mathbf{x}_0) = a\mathbf{x}_t + b\boldsymbol{\epsilon}$

③ $\mu_\theta(\mathbf{x}_t, t) = a\mathbf{x}_t + b\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\epsilon}_\theta(\mathbf{x}_t, t) - \epsilon \right\|_2^2 \right] \right]$$

① $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\epsilon$

② $\mu_q(\mathbf{x}_t, \mathbf{x}_0) = a\mathbf{x}_t + b\epsilon$

③ $\mu_\theta(\mathbf{x}_t, t) = a\mathbf{x}_t + b\hat{\epsilon}_\theta(\mathbf{x}_t, t)$

$\mathbf{x}_0 \xrightarrow{\text{Encoding}} \mathbf{x}_1 \xrightarrow{\text{Encoding}} \mathbf{x}_2 \cdots \xrightarrow{\text{Encoding}} \mathbf{x}_T$

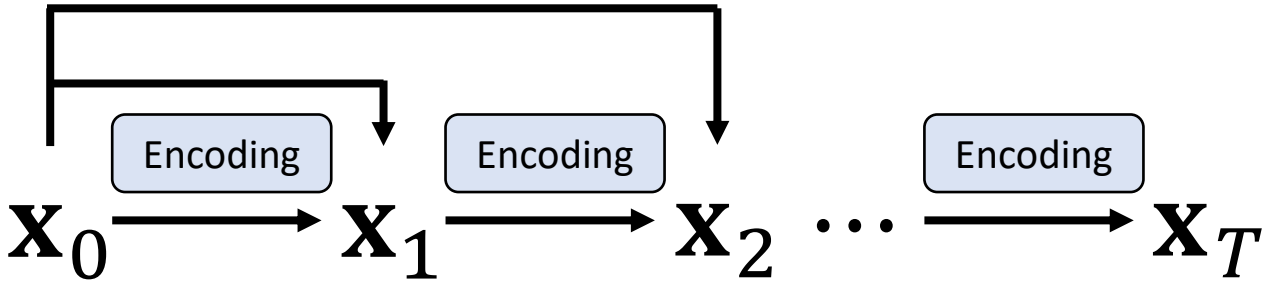
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$

No need to be a Markovian process!

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2 \right] \right]$$



① $\mathbf{x}_t = \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}$

② $\mu_q(\mathbf{x}_t, \mathbf{x}_0) = a\mathbf{x}_t + b\boldsymbol{\epsilon}$

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

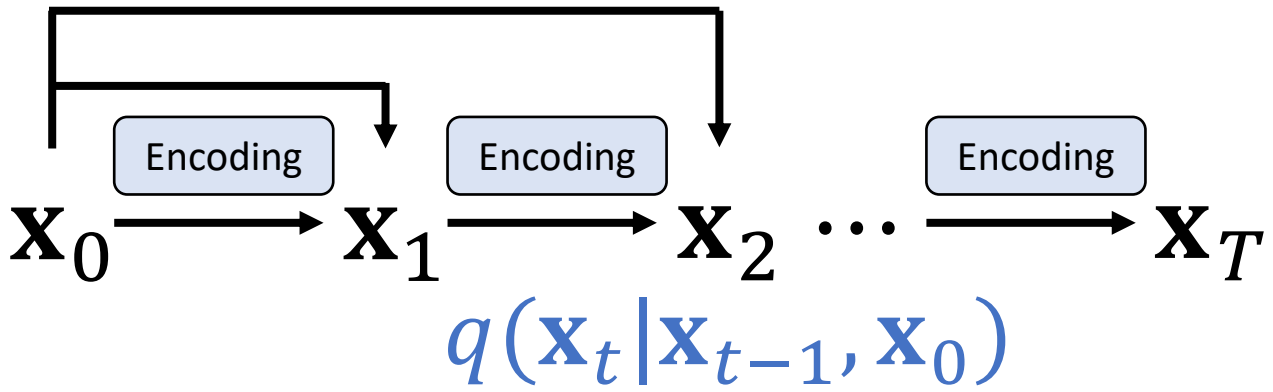
③ $\mu_\theta(\mathbf{x}_t, t) = a\mathbf{x}_t + b\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$

Find a and b such that $\mathbf{x}_t = \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}$!

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

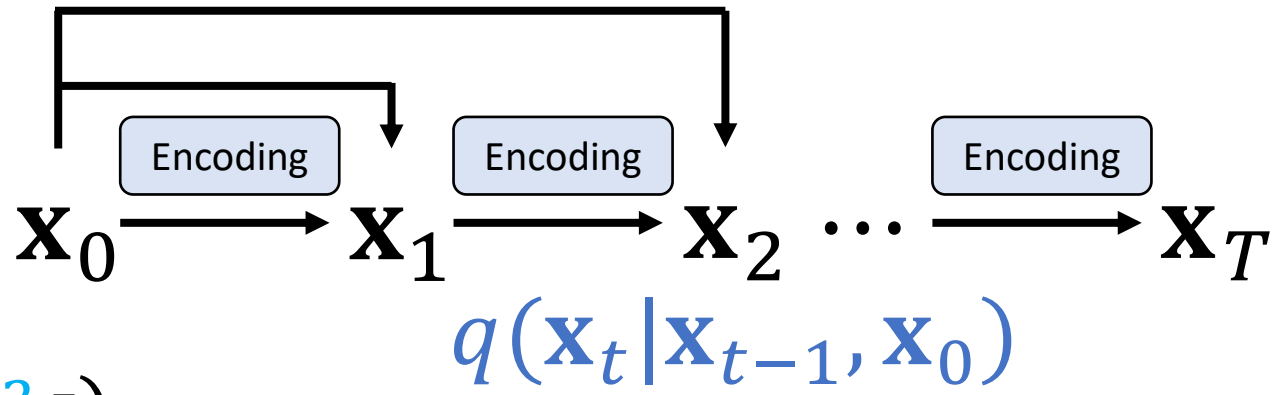
$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\quad w(t) \quad \left[\left\| \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \quad - \quad \boldsymbol{\epsilon} \quad \right\|_2^2 \right] \right]$$



Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2 \right] \right]$$

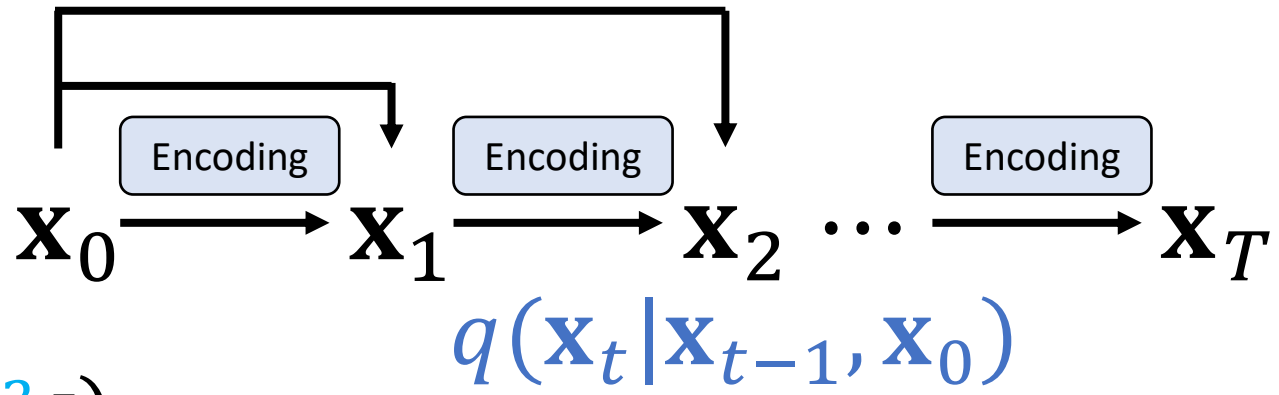


$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mu_q(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \boldsymbol{I}\right)$$

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mu_{\theta}(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \boldsymbol{I}\right)$$

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|_2^2 \right] \right]$$



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

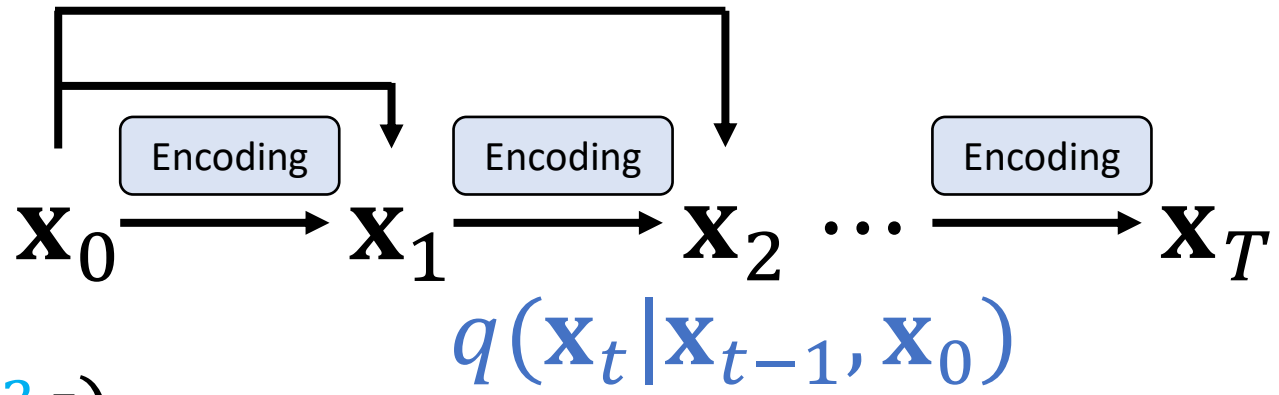
$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mu_\theta(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \boldsymbol{I}\right)$$

$$\mu_q(\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\overline{\alpha_{t-1}}}\mathbf{x}_0 + \sqrt{1 - \overline{\alpha_{t-1}} - \sigma_t}\boldsymbol{\epsilon}$$

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\epsilon}_\theta(\mathbf{x}_t, t) - \epsilon \right\|_2^2 \right] \right]$$



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

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mu_\theta(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \boldsymbol{I}\right)$$

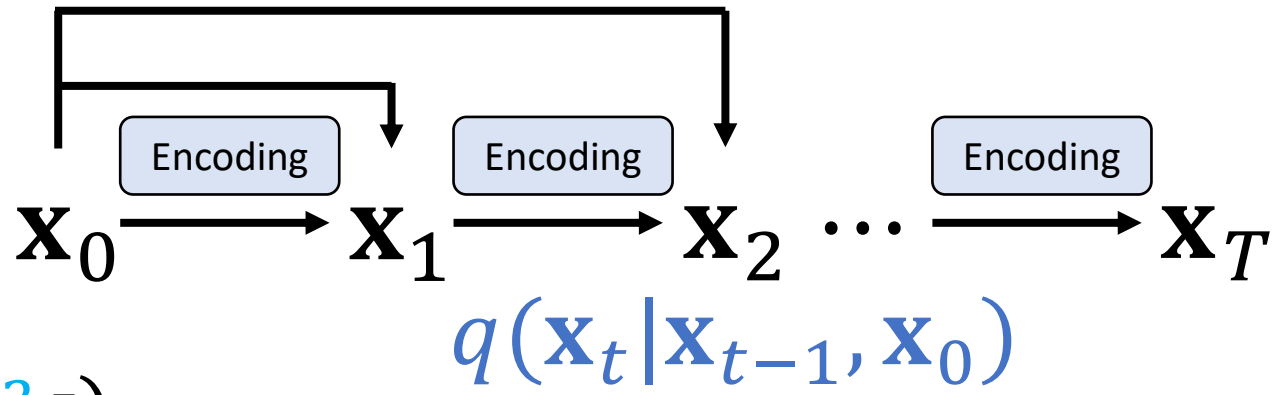
$$\mu_q(\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\overline{\alpha_{t-1}}}\mathbf{x}_0 + \sqrt{1 - \overline{\alpha_{t-1}} - \sigma_t}\boldsymbol{\epsilon}$$

$$\mu_\theta(\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\overline{\alpha_{t-1}}}\hat{\mathbf{x}}_0 + \sqrt{1 - \overline{\alpha_{t-1}} - \sigma_t}\hat{\boldsymbol{\epsilon}}$$

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

$$\sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\; w(t) \; \left[\left\| \hat{\epsilon}_\theta(\mathbf{x}_t, t) - \epsilon \right\|_2^2 \right] \right]$$



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

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mu_\theta(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \boldsymbol{I}\right)$$

DDIM sampler: Set $\sigma_t^2 = 0$

$$\mu_q(\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\overline{\alpha_{t-1}}}\mathbf{x}_0 + \sqrt{1 - \overline{\alpha_{t-1}}}\boldsymbol{\epsilon}$$

$$\mu_\theta(\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\overline{\alpha_{t-1}}}\hat{\mathbf{x}}_0 + \sqrt{1 - \overline{\alpha_{t-1}}}\hat{\boldsymbol{\epsilon}}$$

Deterministic generative process!

Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

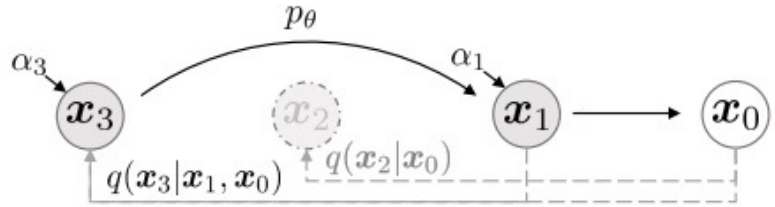
	CIFAR10 (32 × 32)					CelebA (64 × 64)				
	10	20	50	100	1000	10	20	50	100	1000
DDIM	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
DDPM	367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**

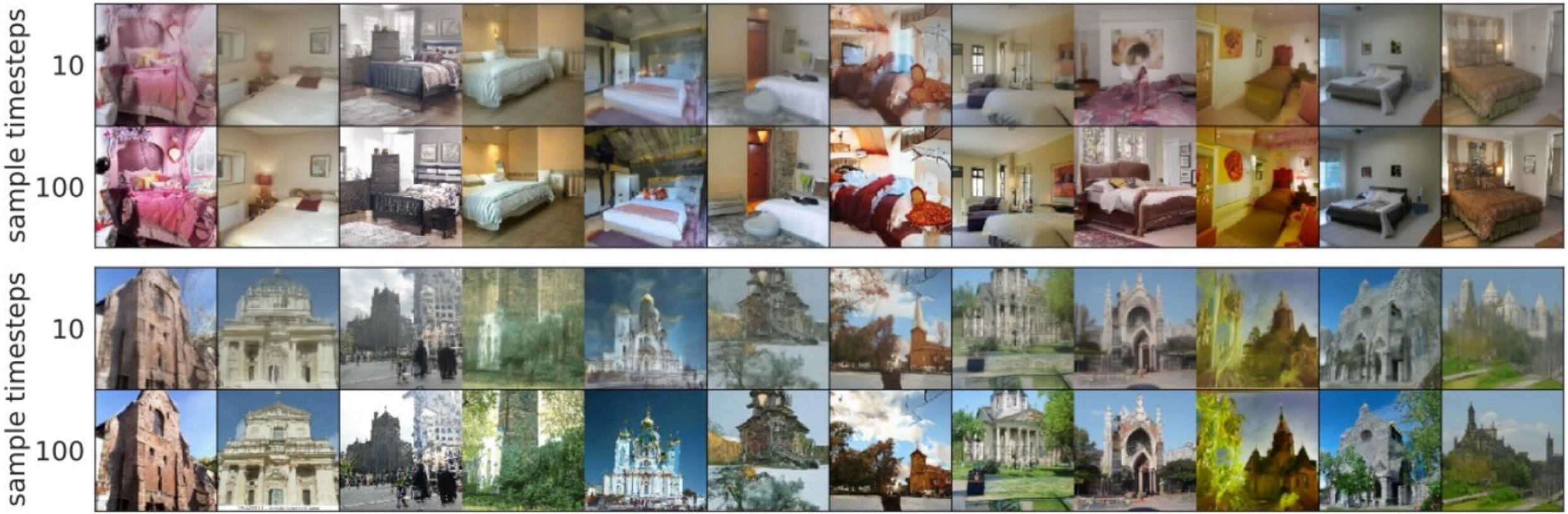
Slide from Jia-Bin Huang

DDIM [Song et al. 2021]

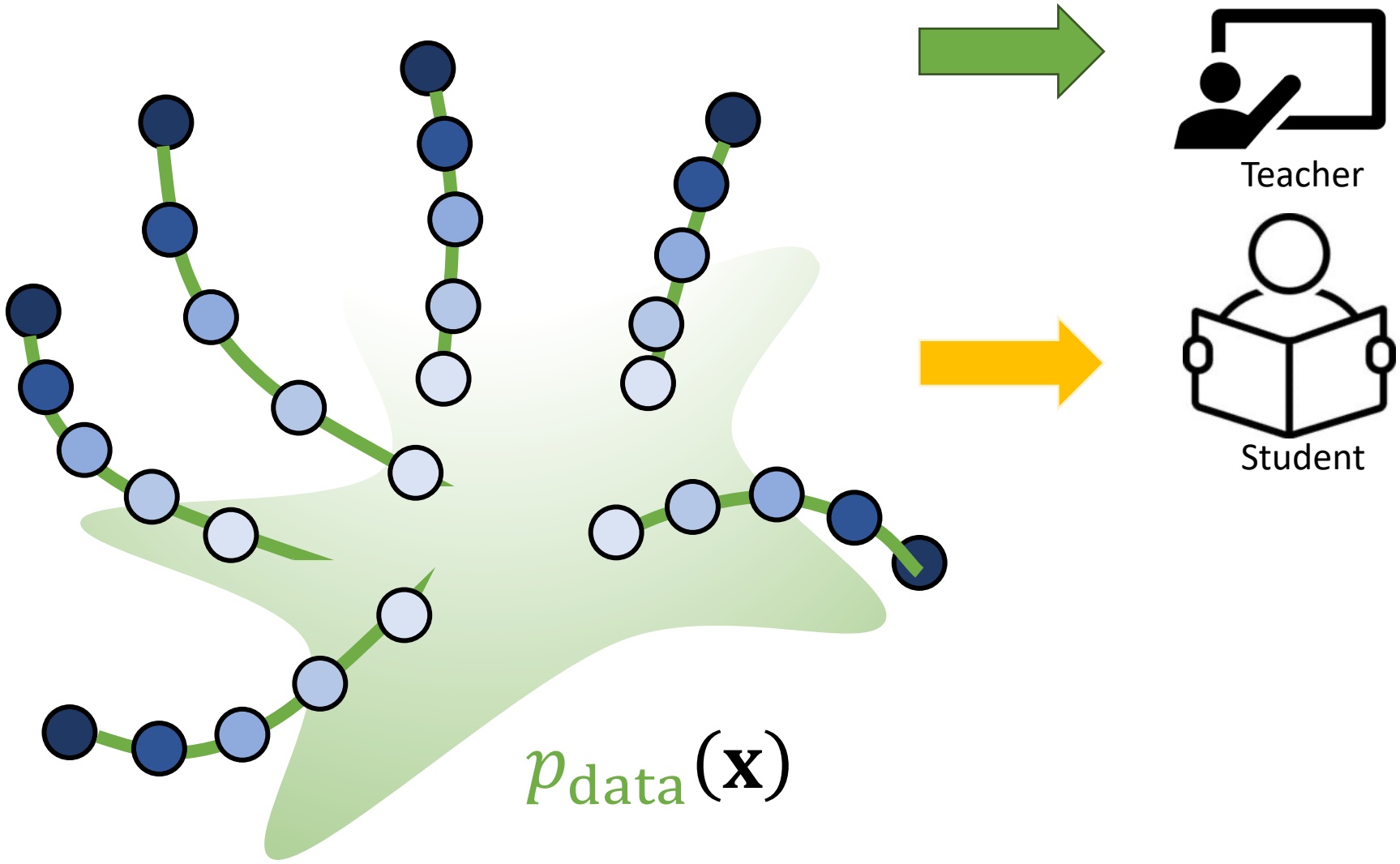
DDIM reduces the sampling steps significantly

- Creates the outline of the sample after only 10 steps (DDPM needs hundreds)



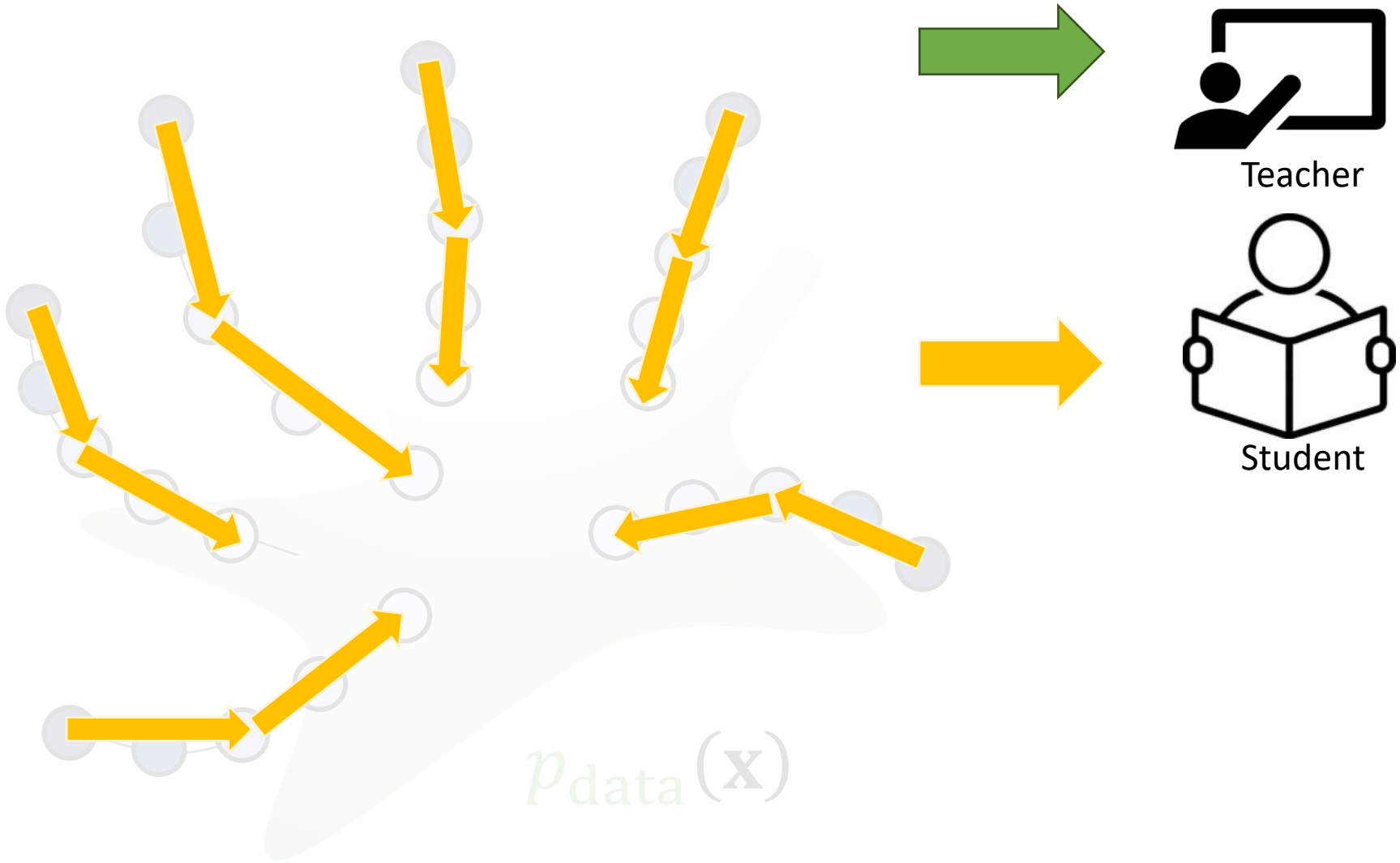


Progressive Distillation



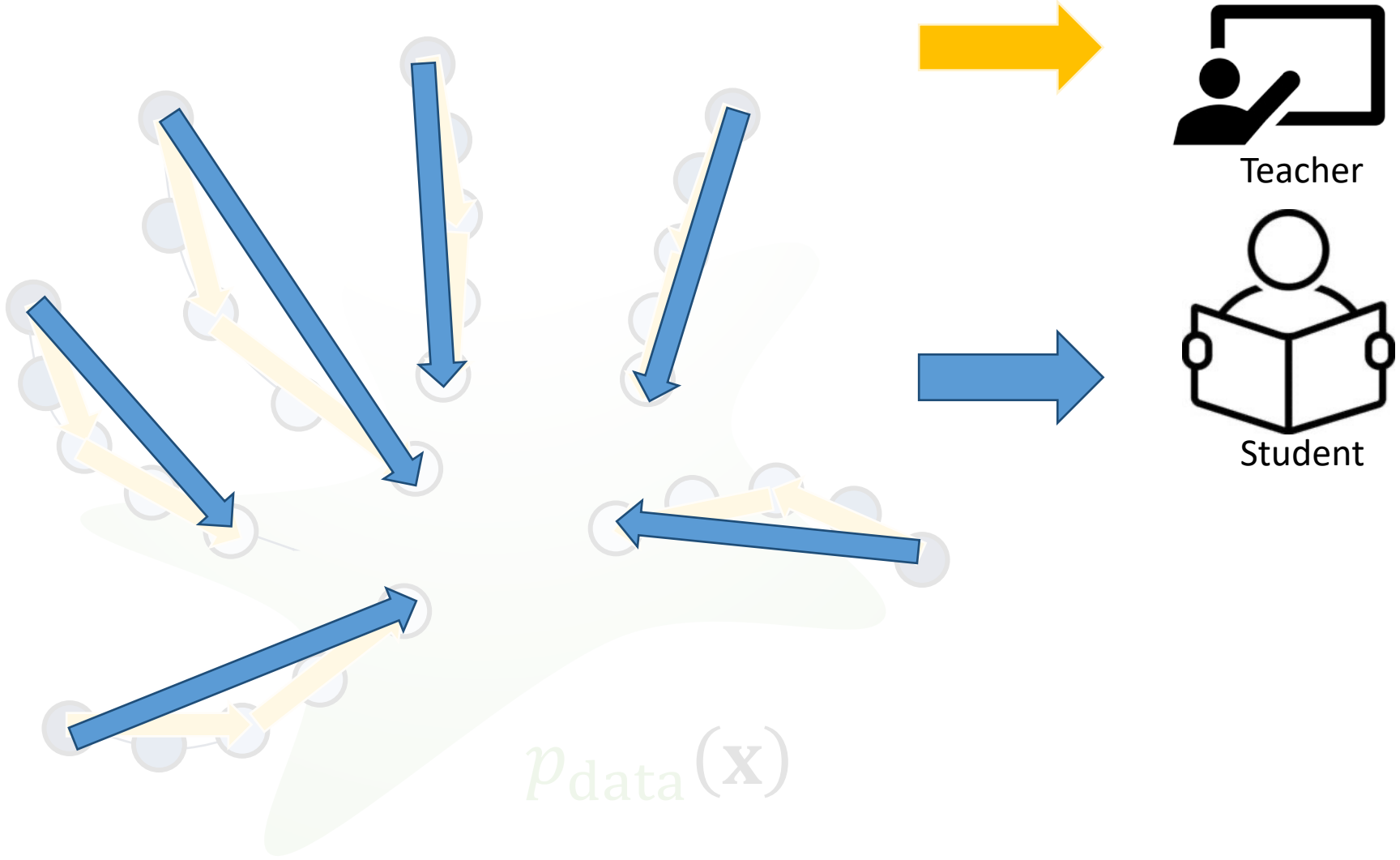
Slide from Jia-Bin Huang

Progressive Distillation



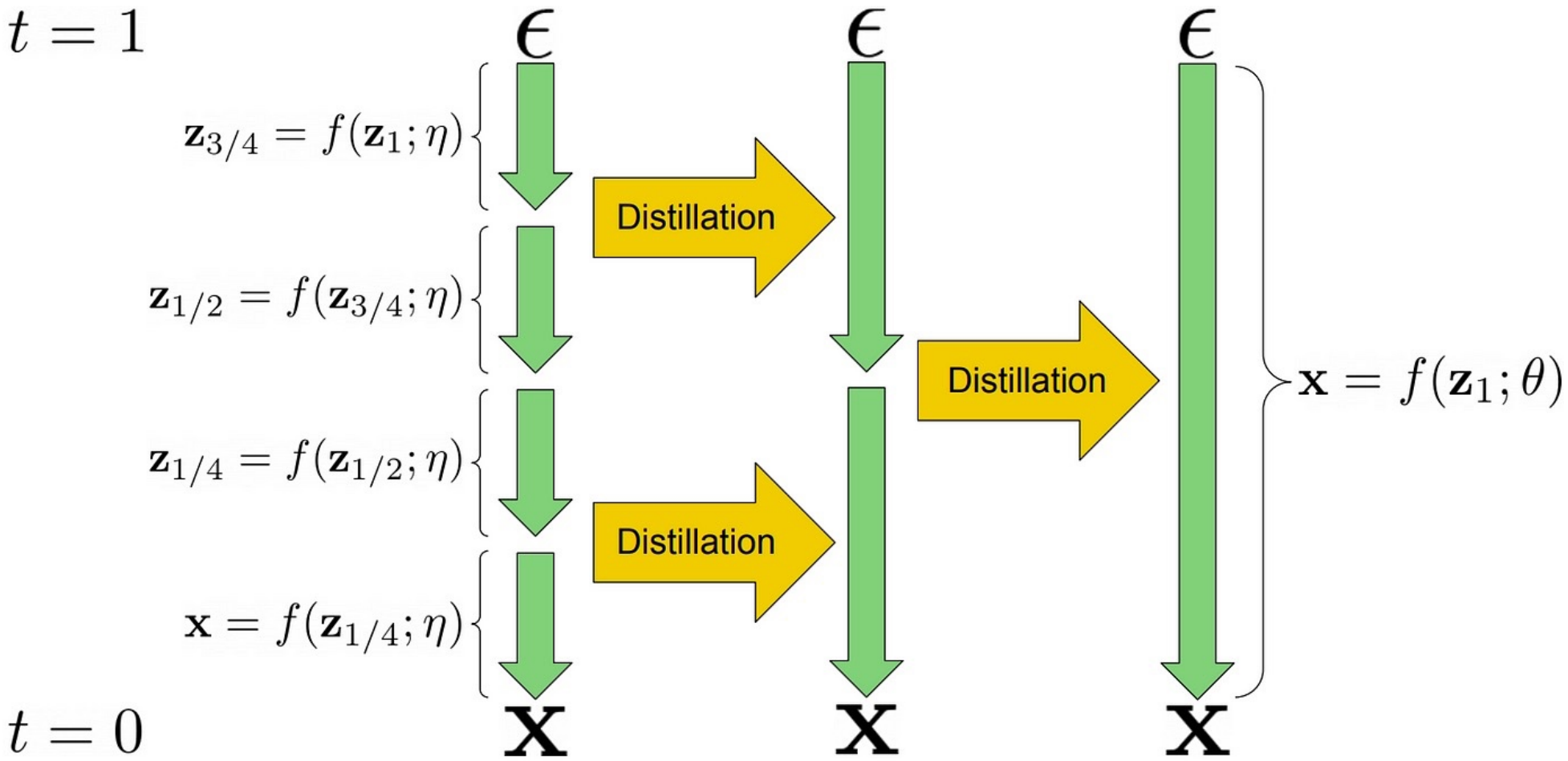
Slide from Jia-Bin Huang

Progressive Distillation



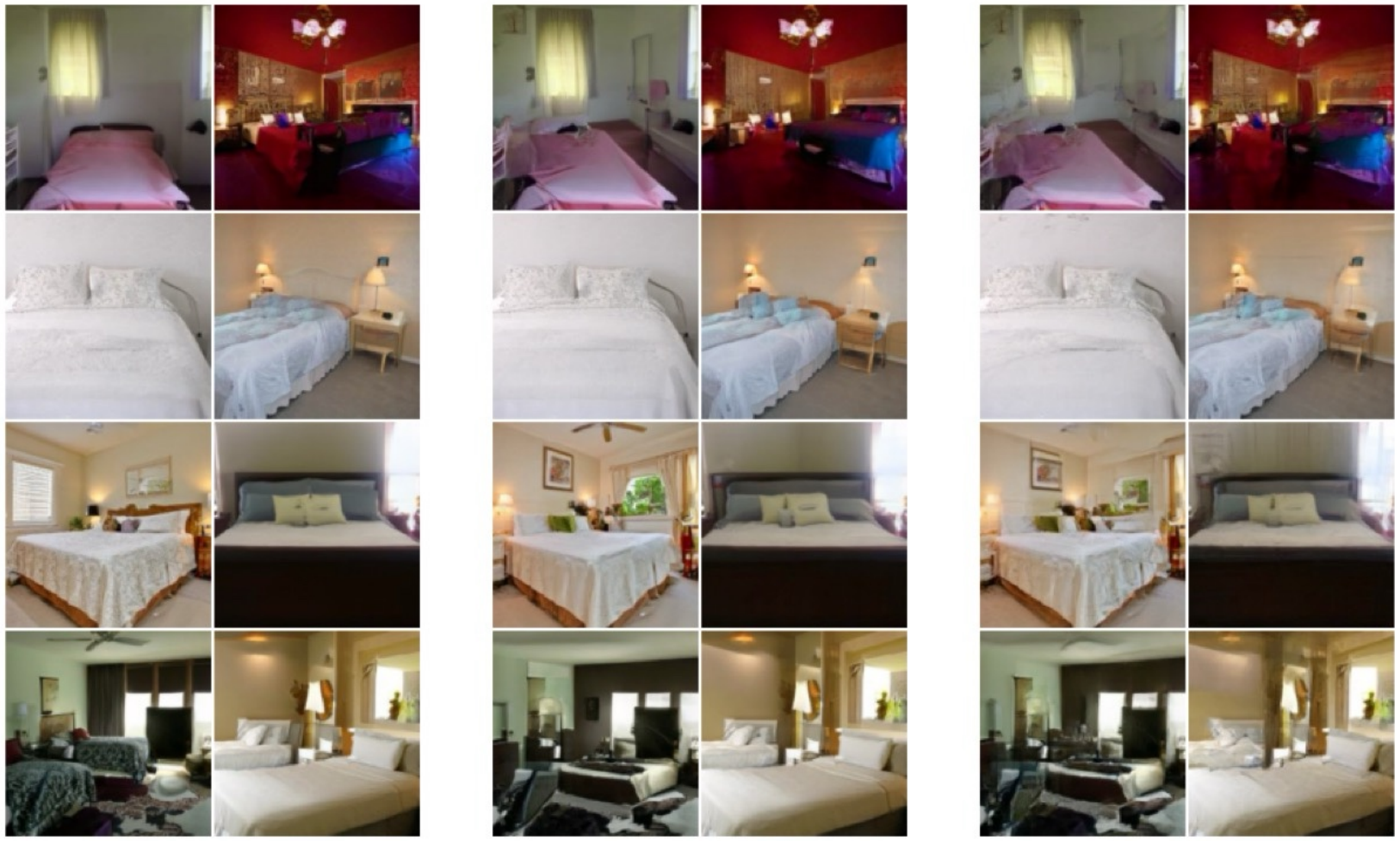
Slide from Jia-Bin Huang

Progressive Distillation



Salimans et al, Progressive distillation for fast sampling of diffusion models

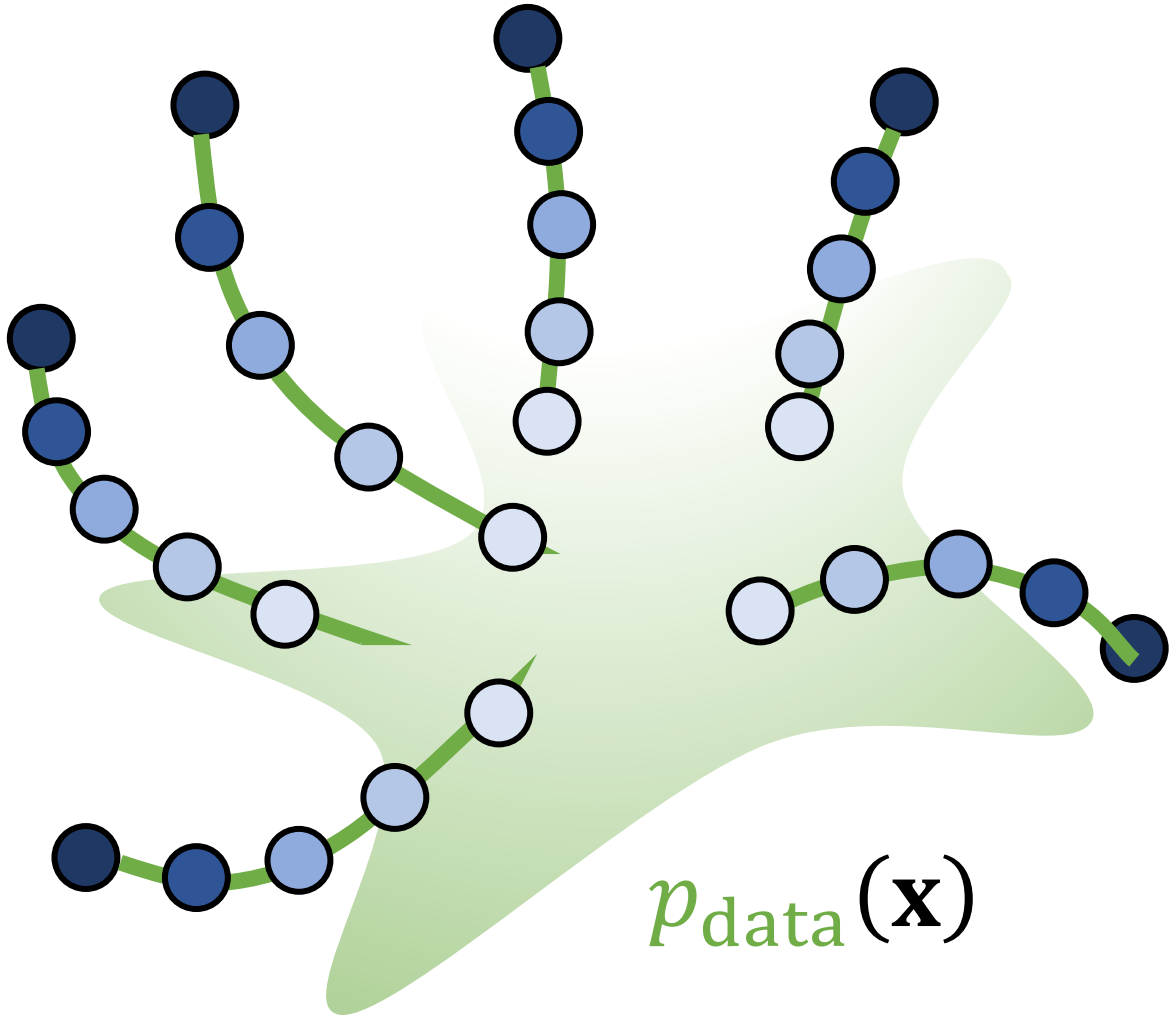
Progressive Distillation: Results



(a) 256 sampling steps (b) 4 sampling steps (c) 1 sampling step

Figure 10: Random samples from our distilled LSUN bedrooms models, for fixed random seed and for varying number of sampling steps.

Guided Distillation



Slide from Jia-Bin Huang

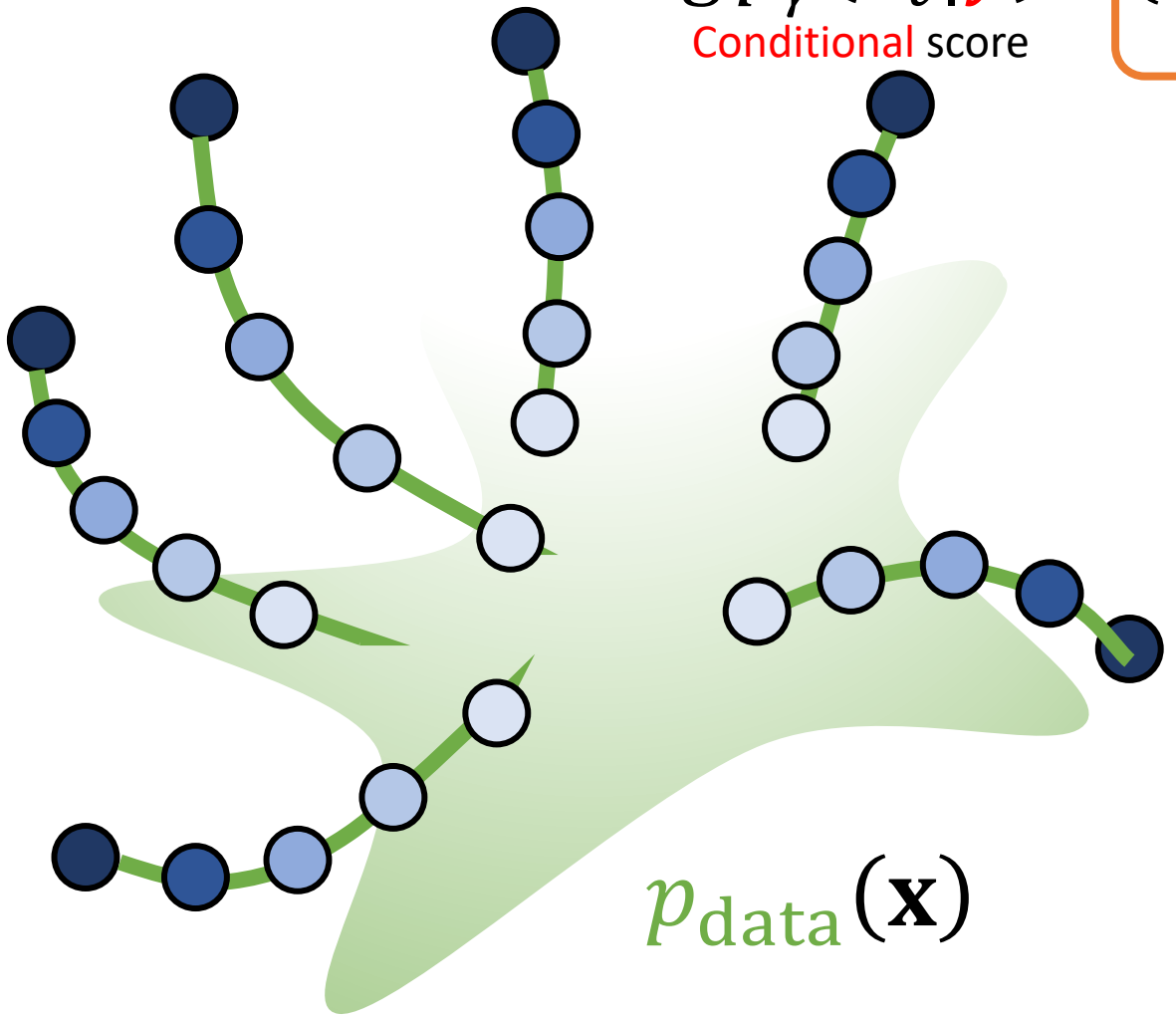
Guided Distillation

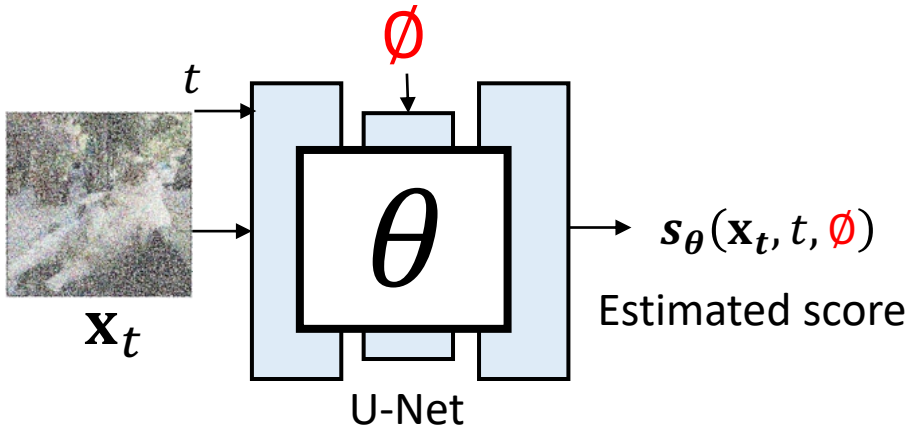
$$\nabla \log p_\gamma(\mathbf{x}_t|y) = (1-\gamma)\nabla \log p(\mathbf{x}_t) + \gamma \nabla \log p(\mathbf{x}_t|y)$$

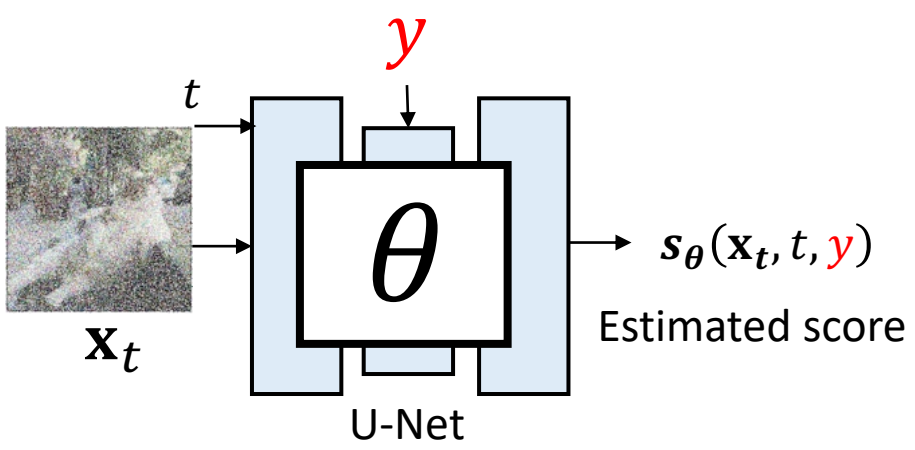
Conditional score

Unconditional score

Conditional score

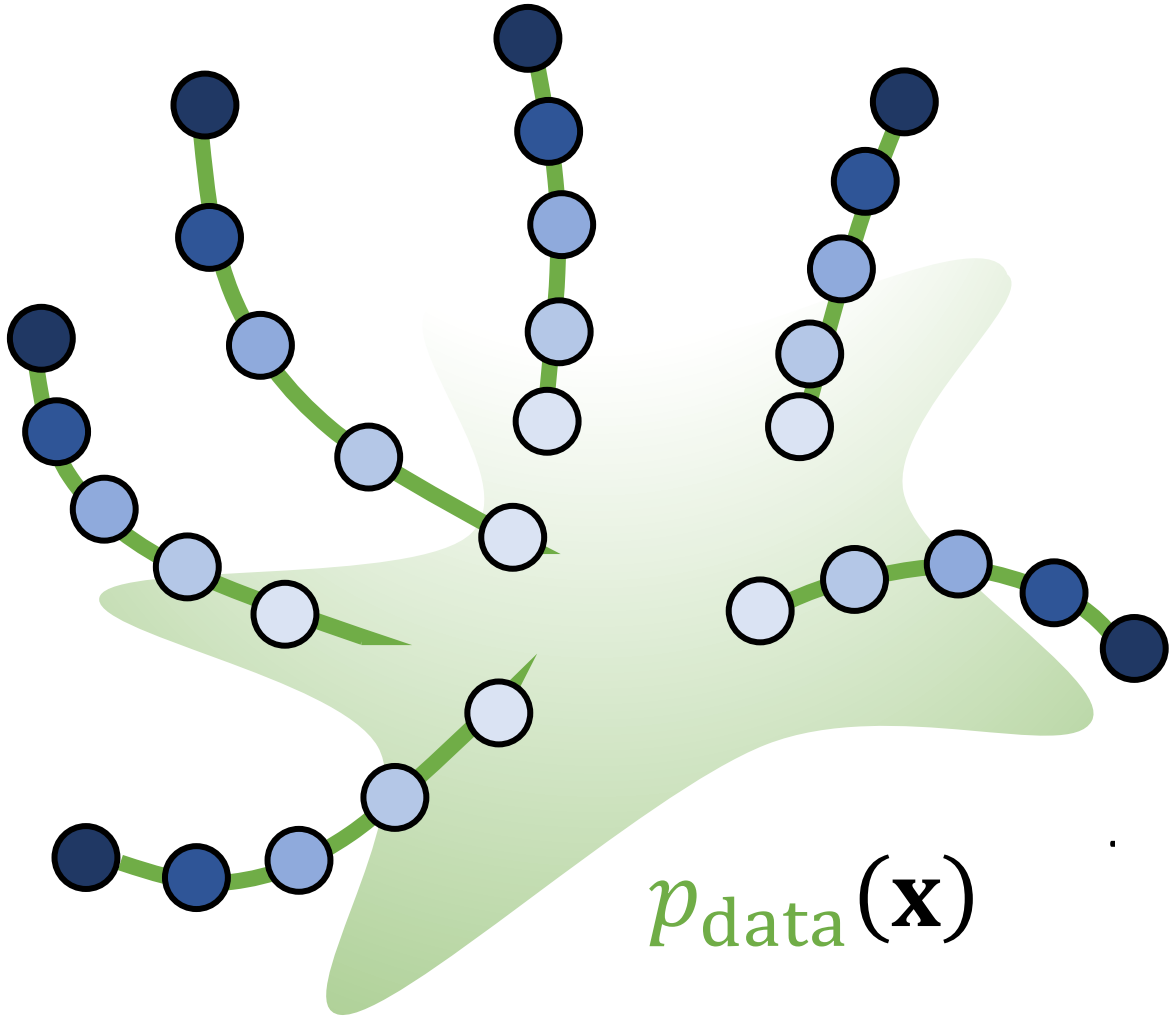






Slide from Jia-Bin Huang

Guided Distillation



Classifier-free guidance distillation

Slide from Jia-Bin Huang

[Meng et al. 2023]

Guided Distillation

(a) 2 denoising steps

(b) 4 denoising steps

(c) 8 denoising steps

Slide from Jia-Bin Huang

[Meng et al. 2023]

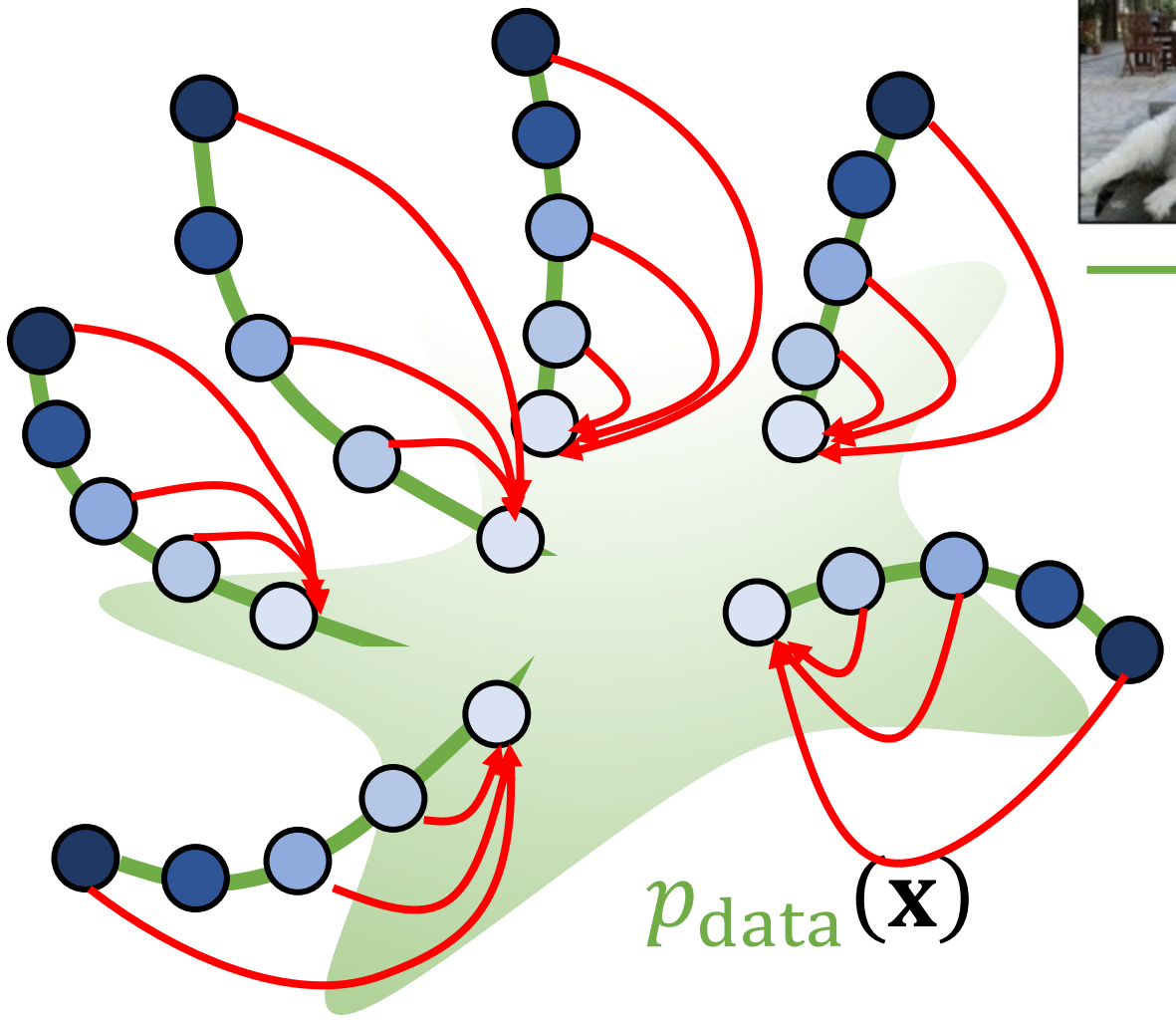
Consistency Models

$(\mathbf{x}_0, 0)$ $(\mathbf{x}_t, t)$ $(\mathbf{x}_{t'}, t')$ $(\mathbf{x}_T, T)$

$f_\theta(\mathbf{x}_t, t)$

$f_\theta(\mathbf{x}_{t'}, t')$

$f_\theta(\mathbf{x}_T, T)$



$$\min_\theta d\Big(f_{EMA}(\mathbf{x}_t, t), f_\theta(\mathbf{x}_{t'}, t')\Big)$$

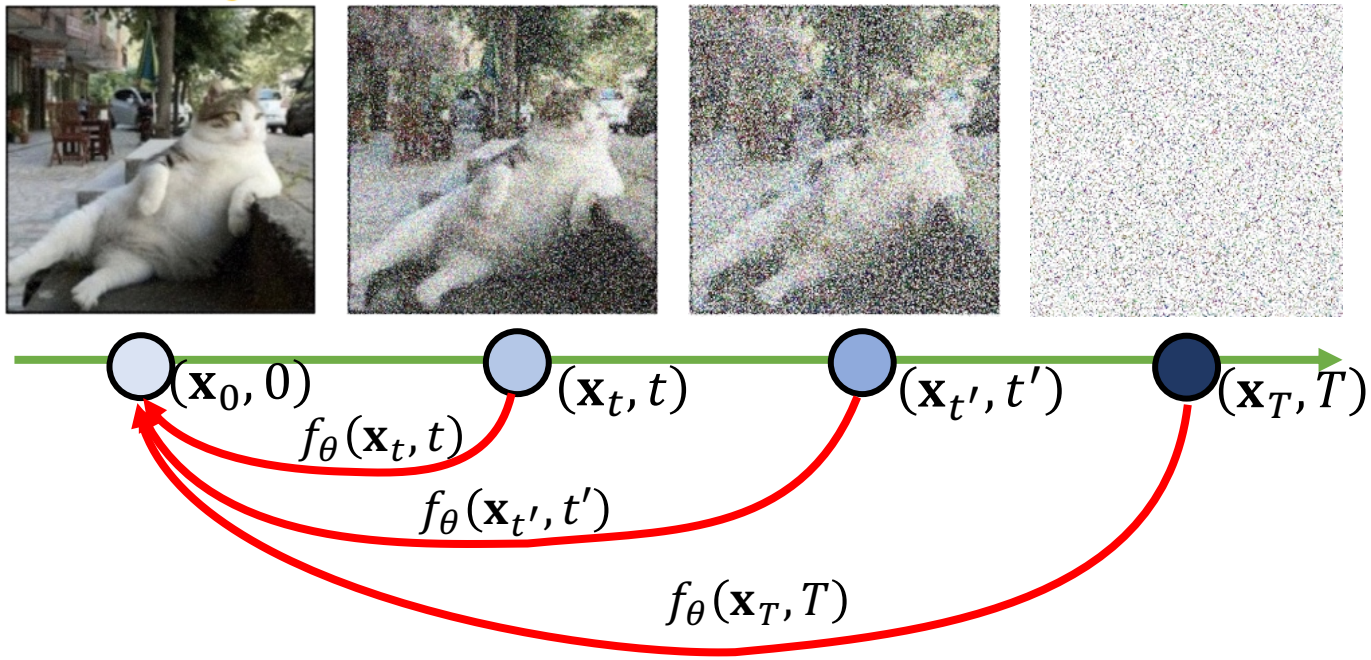
target network (teacher)

online network (student)

Slide from Jia-Bin Huang

[Song et al. 2023]

Consistency Models



$$\min_\theta d\Big(f_{EMA}(\mathbf{x}_t, t), f_\theta(\mathbf{x}_{t'}, t')\Big)$$

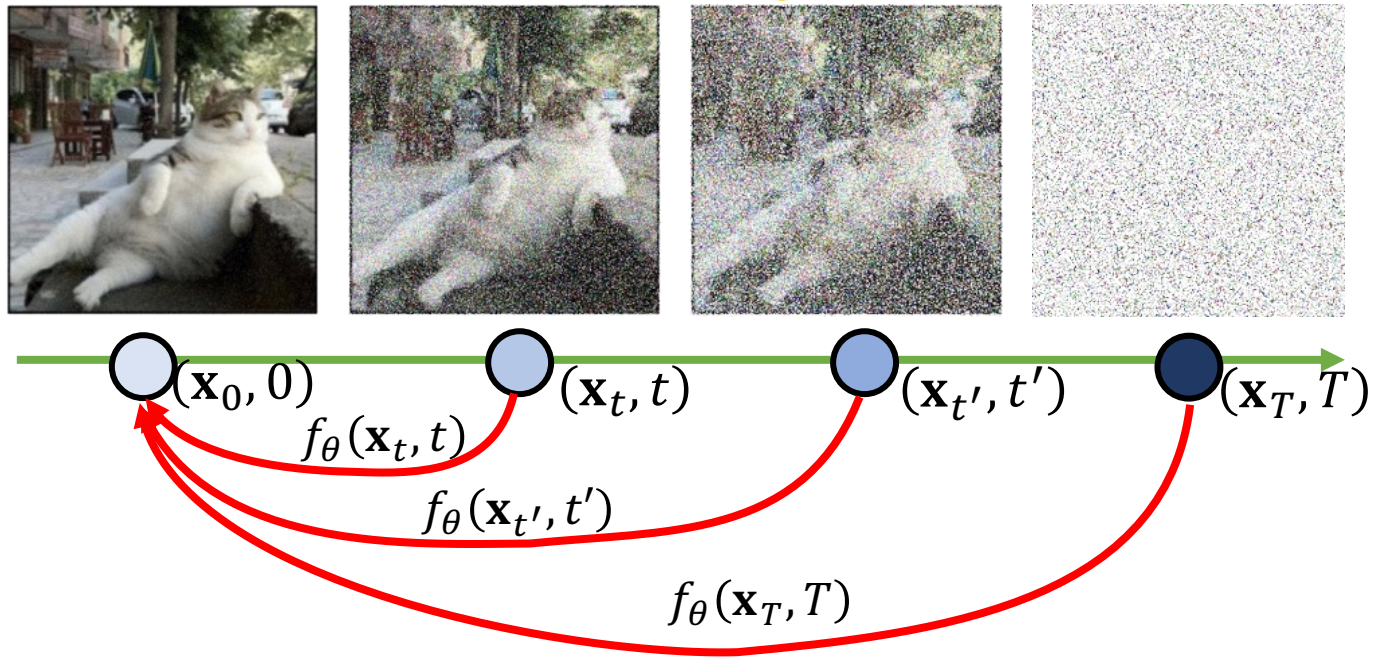
target network (teacher)

online network (student)

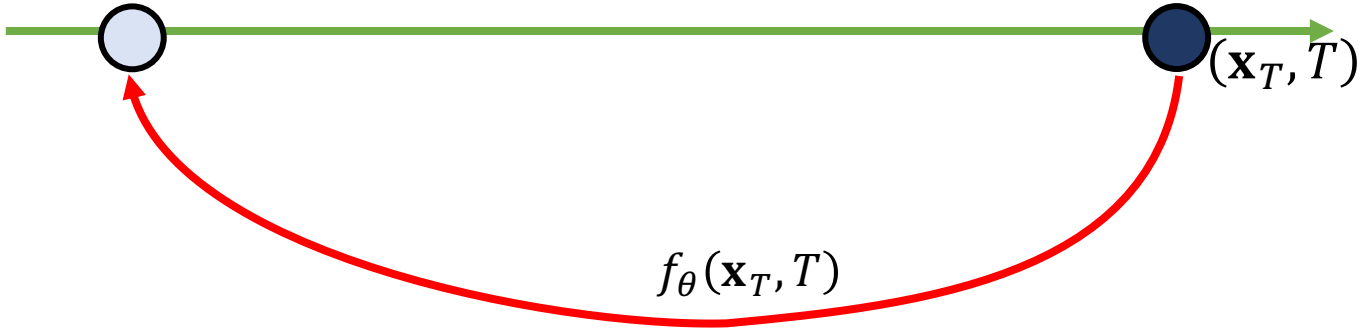
Slide from Jia-Bin Huang

[Song et al. 2023]

Consistency Models



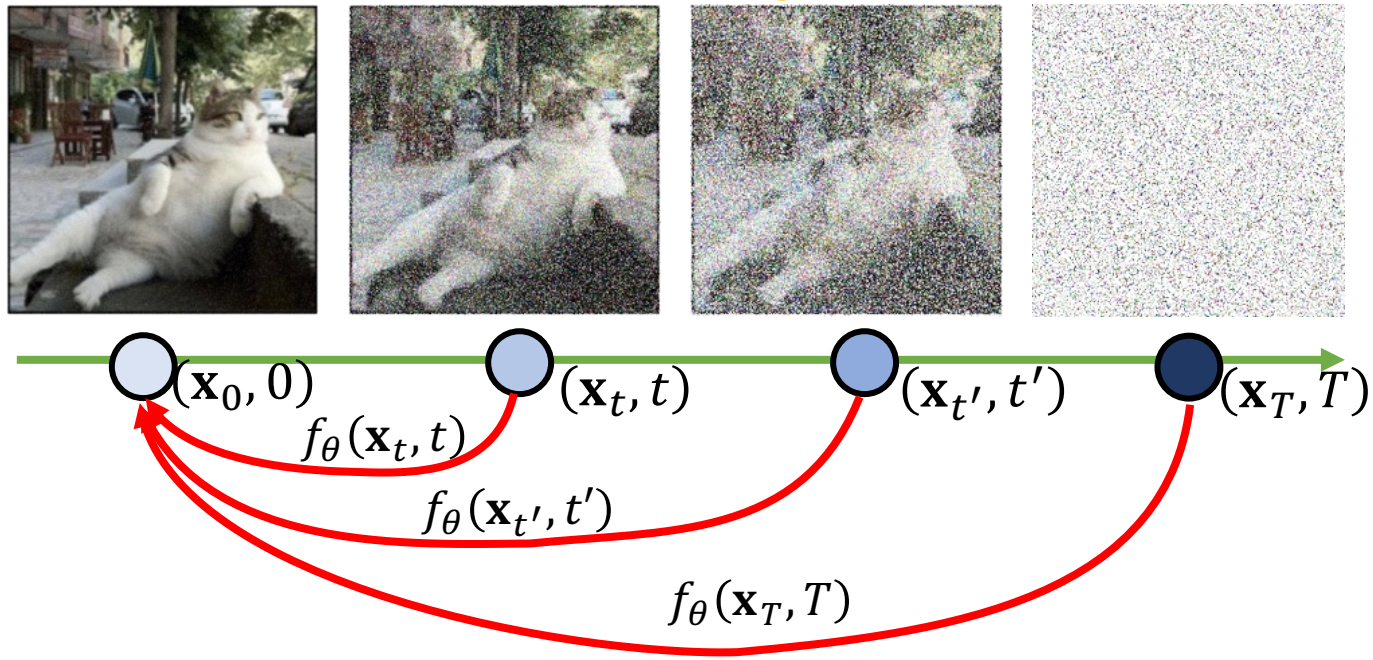
Single-step generation:



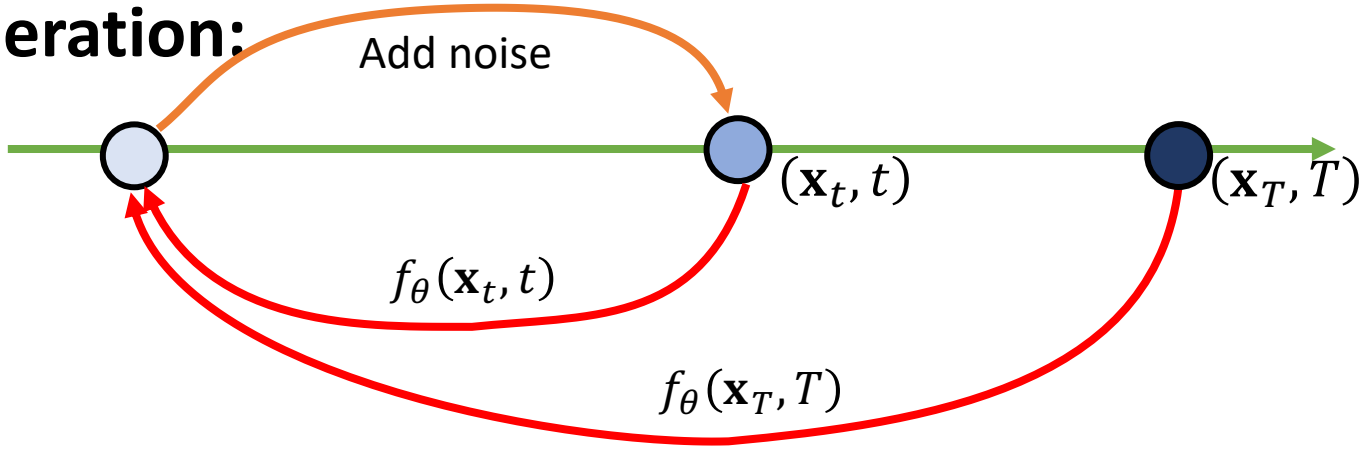
Slide from Jia-Bin Huang

[Song et al. 2023]

Consistency Models



Multi-step generation:

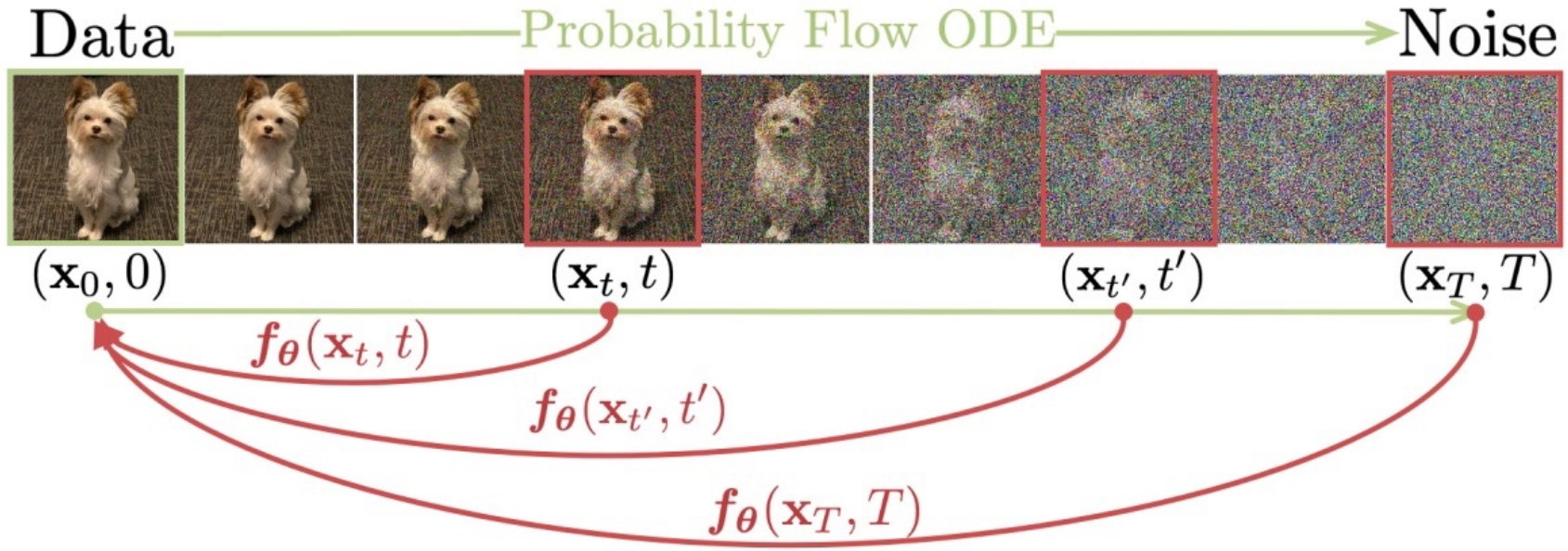


Slide from Jia-Bin Huang

[Song et al. 2023]

Latent Consistency Models

Consistency Models

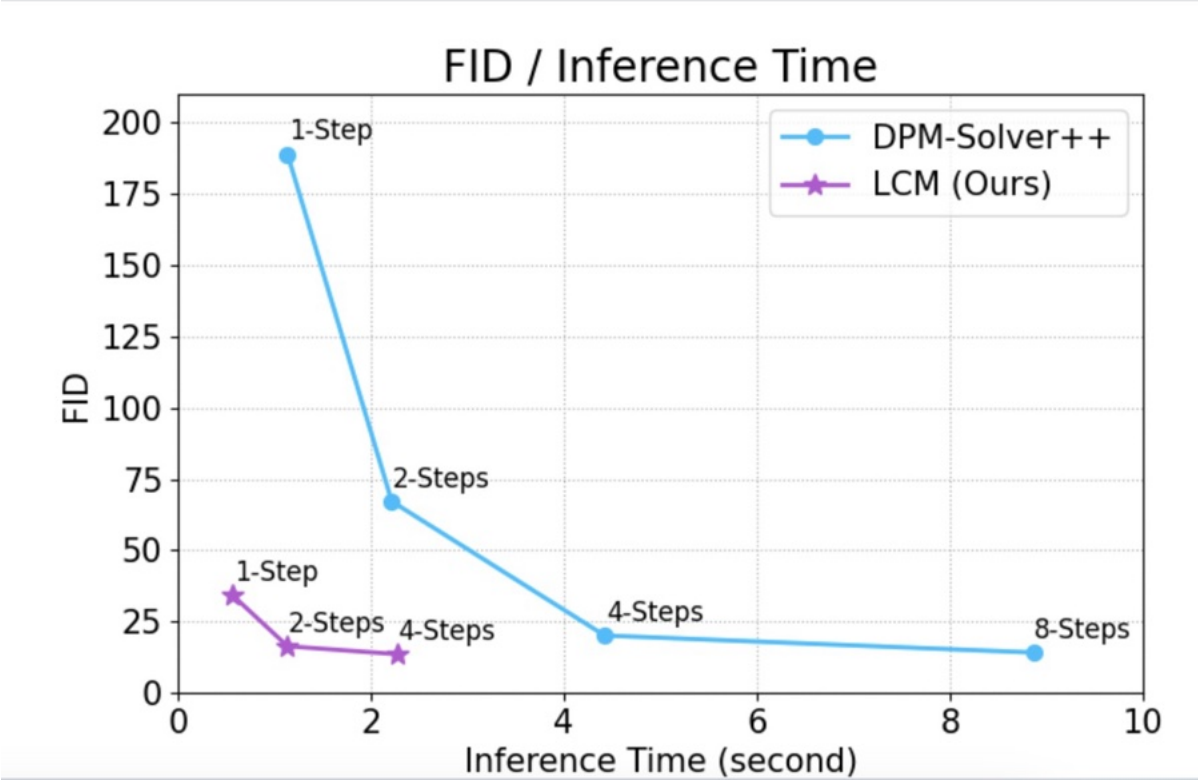


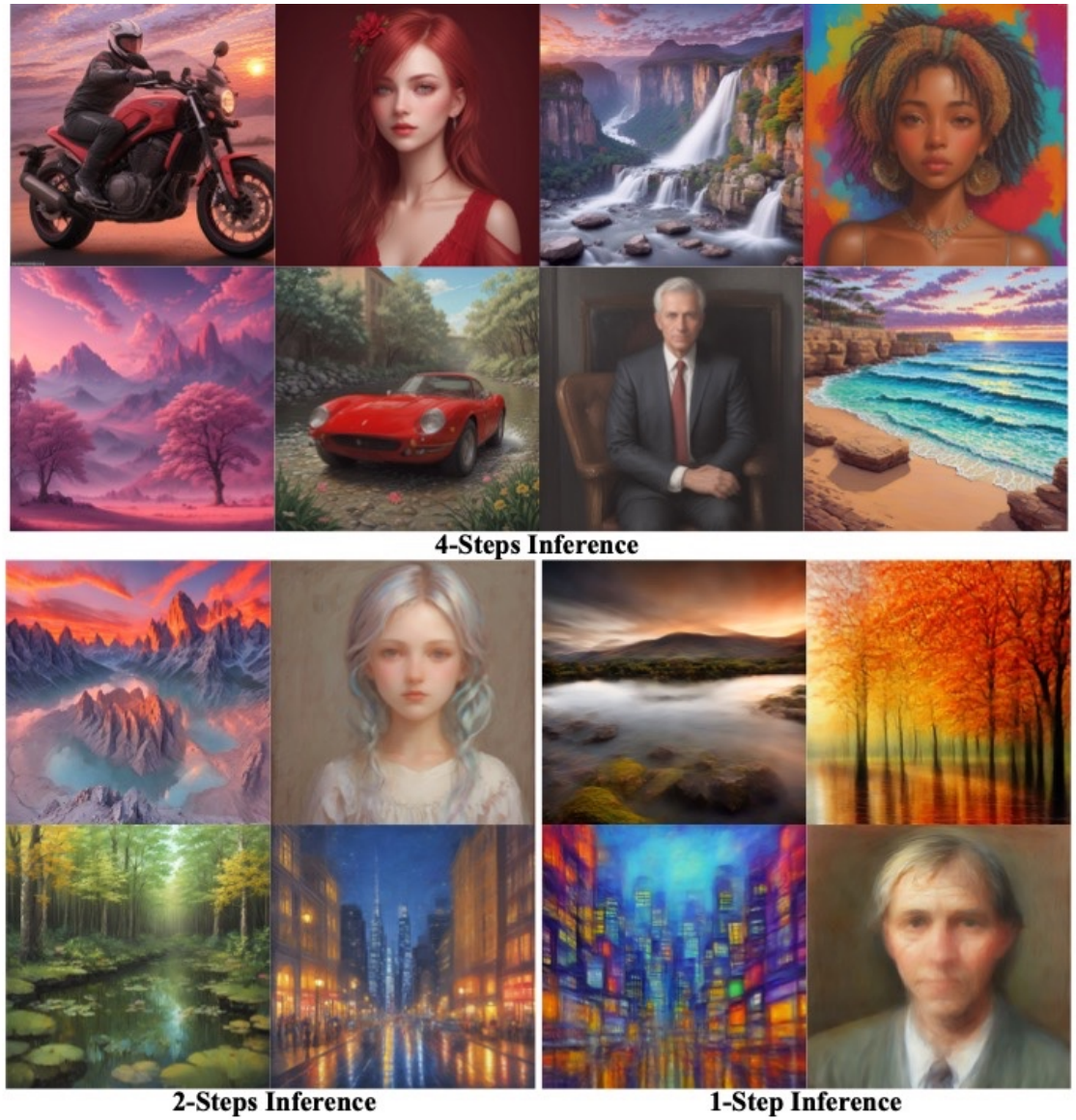
Latent Consistency Models: combine the above idea with Latent Diffusion Models

Song et al., Consistency Models

Luo et al, Latent Consistency Models: Synthesizing High-Resolution Images with Few-step Inference

Latent Consistency Models: Results





Latent Consistency Models

Slide from Jia-Bin Huang

4-step inference

LCM [Luo et al. 2023]

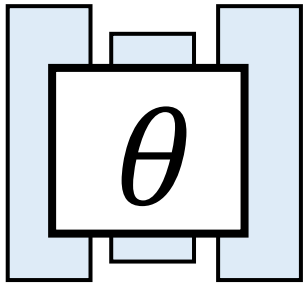
Latent Consistency Models + LoRA

Slide from Jia-Bin Huang

LCM-LoRA [Luo et al. 2023]

Low-rank Adaptation: LoRA

[Hu et al. 2021]

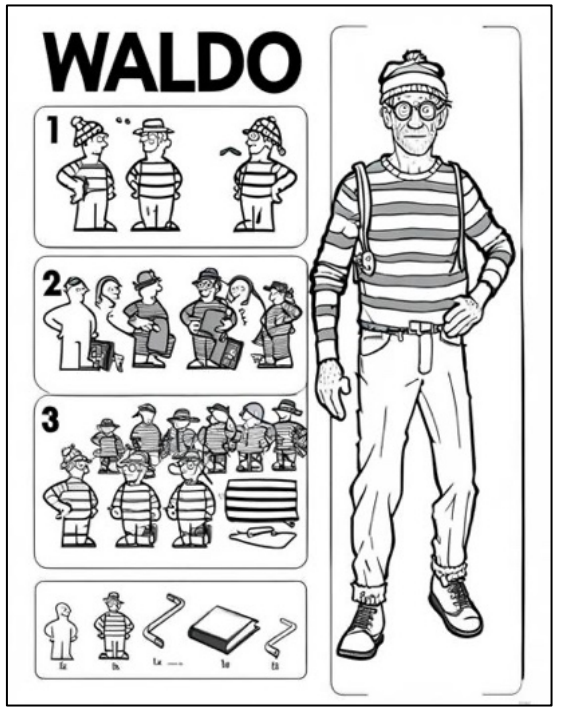


U-Net

Pretrained diffusion model

PixelArt

Lego



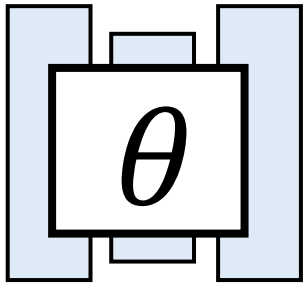
IKEA instructions

Anime

Slide from Jia-Bin Huang

Low-rank Adaptation: LoRA

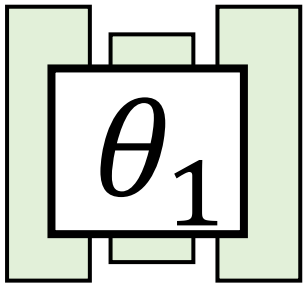
[Hu et al. 2021]



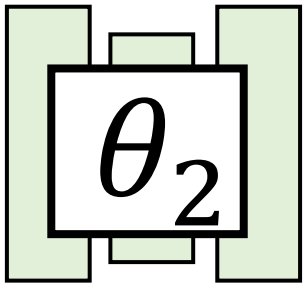
Pretrained diffusion model

Computationally expensive

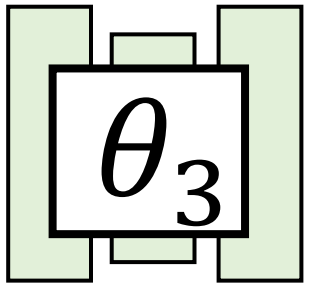
High storage requirement



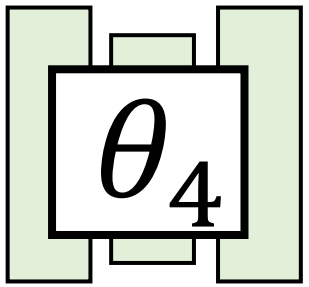
PixelArt



Lego



IKEA instructions

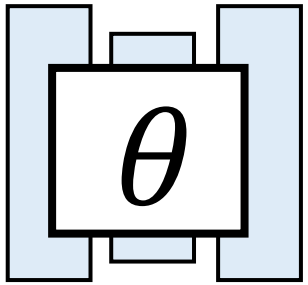


Anime

Slide from Jia-Bin Huang

Low-rank Adaptation: LoRA

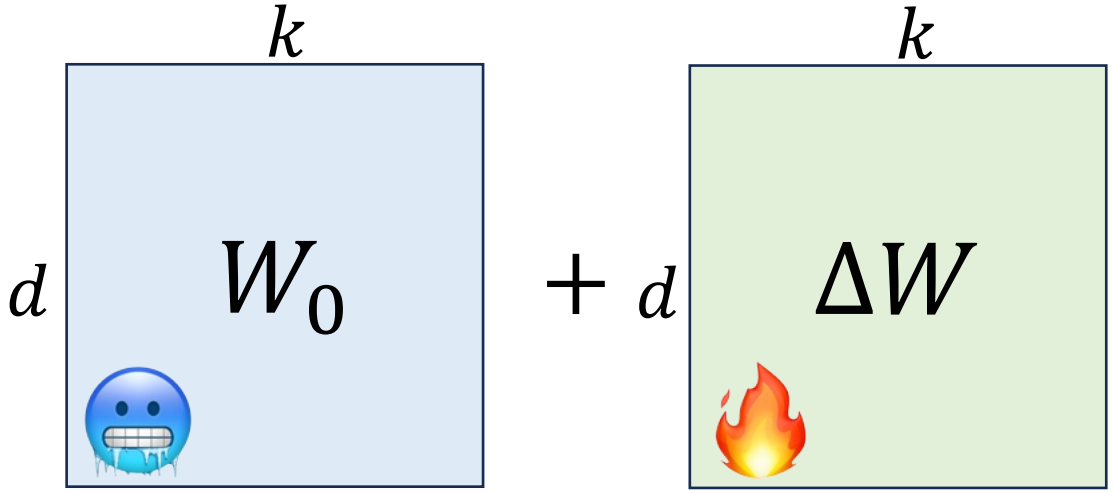
[Hu et al. 2021]

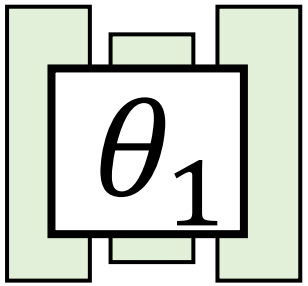


U-Net

Pretrained diffusion model

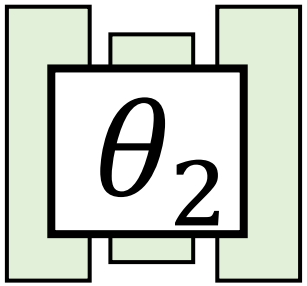
Finetuning cross-attention layers





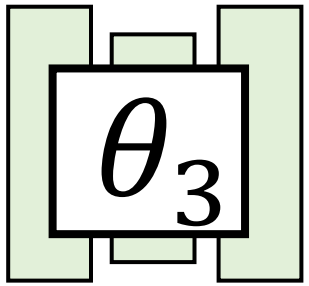
U-Net

PixelArt



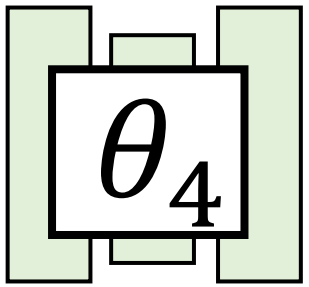
U-Net

Lego



U-Net

IKEA instructions



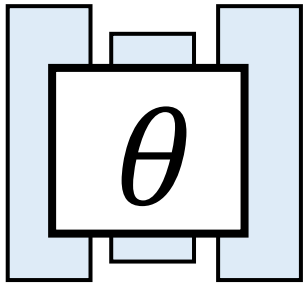
U-Net

Anime

Slide from Jia-Bin Huang

Low-rank Adaptation: LoRA

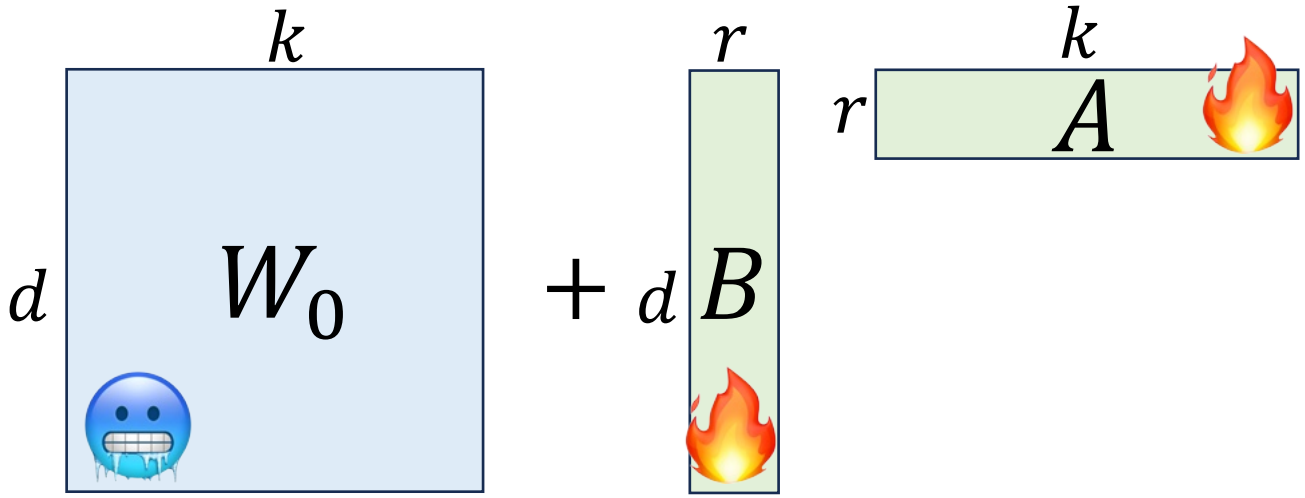
[Hu et al. 2021]

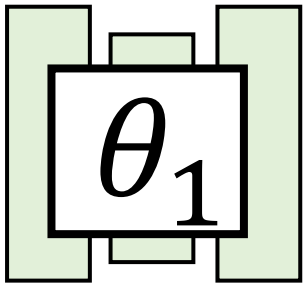


U-Net

Pretrained diffusion model

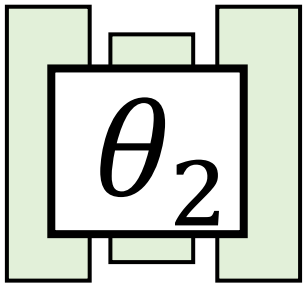
Finetuning cross-attention layers





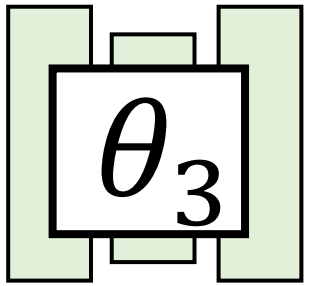
U-Net

PixelArt



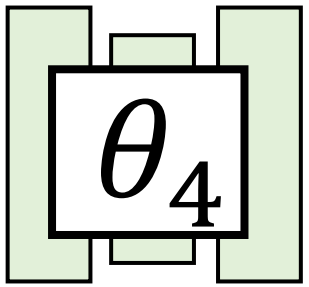
U-Net

Lego



U-Net

IKEA instructions

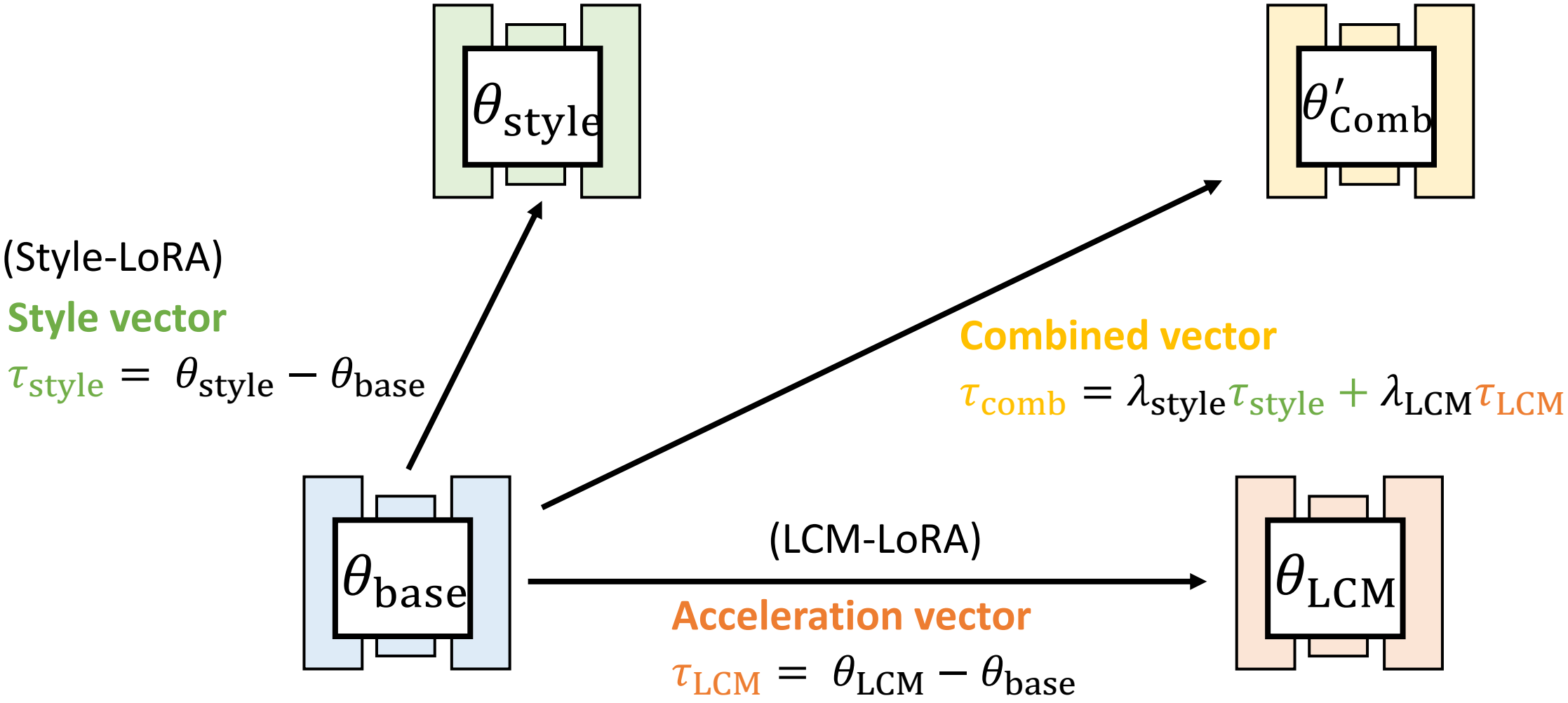


U-Net

Anime

Slide from Jia-Bin Huang

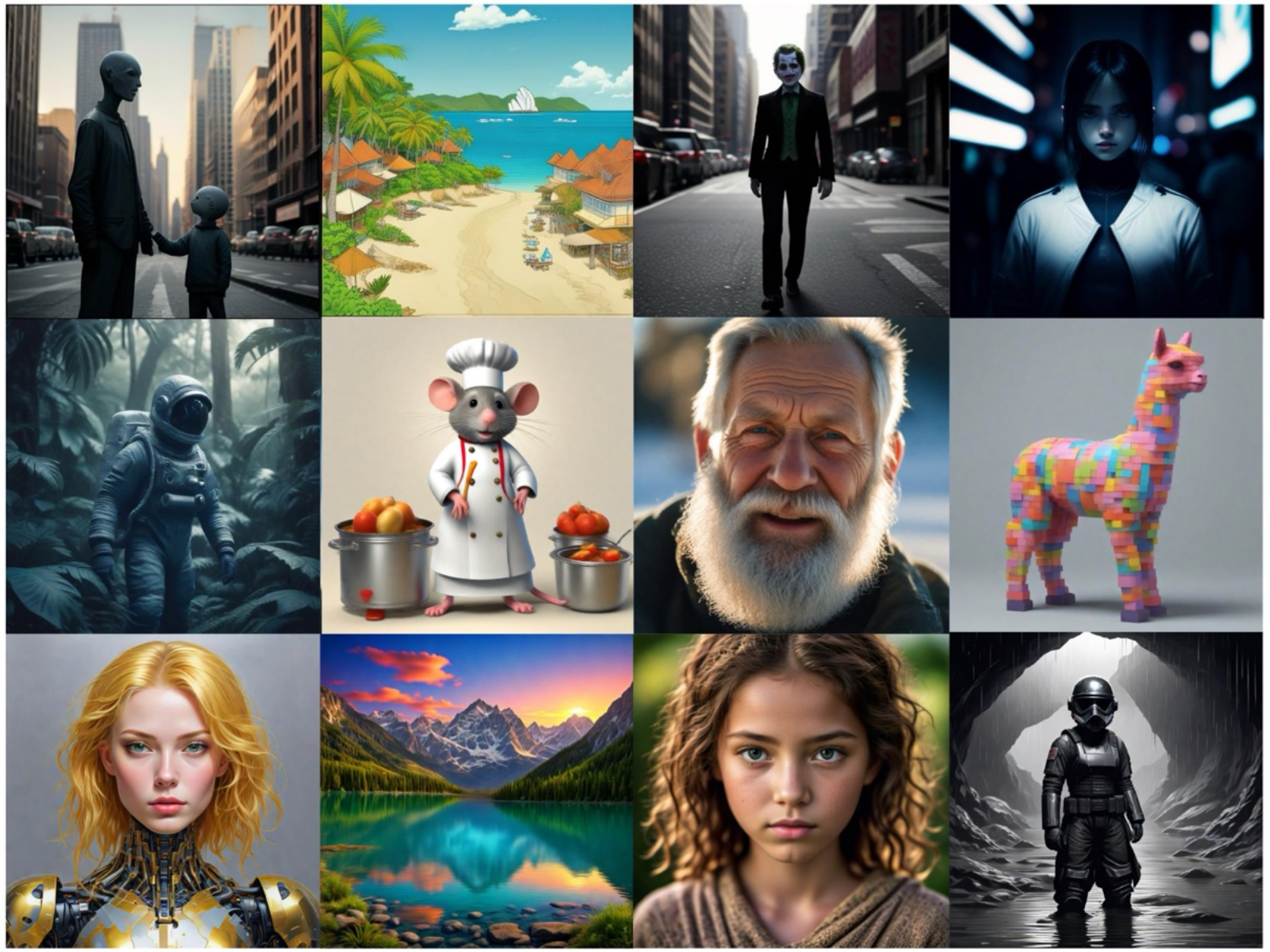
Latent Consistency Models + LoRA



Slide from Jia-Bin Huang

LCM-LoRA [Luo et al. 2023]

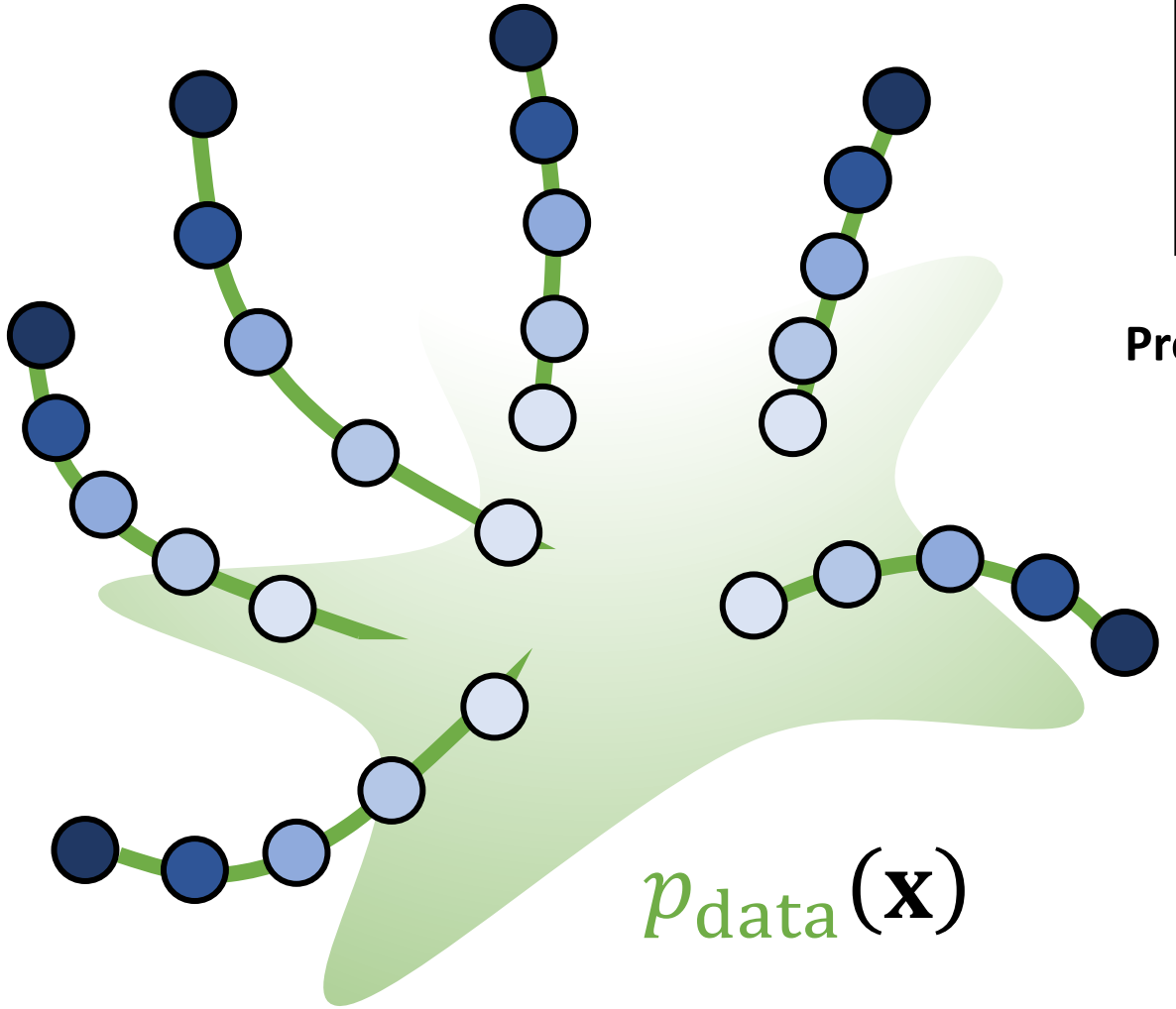
Latent Consistency Models + LoRA

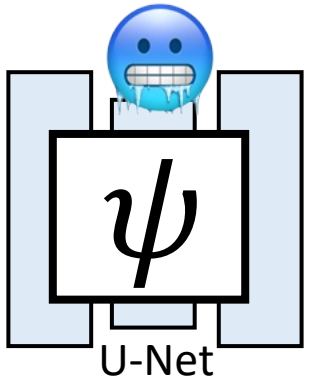


Slide from Jia-Bin Huang

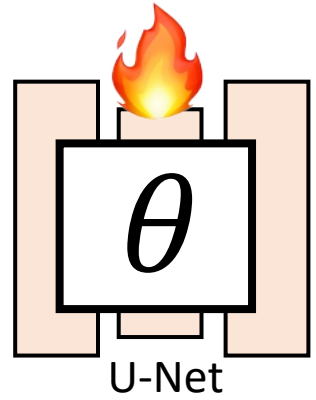
LCM-LoRA [Luo et al. 2023]

Adversarial Diffusion Distillation





Pretrained model

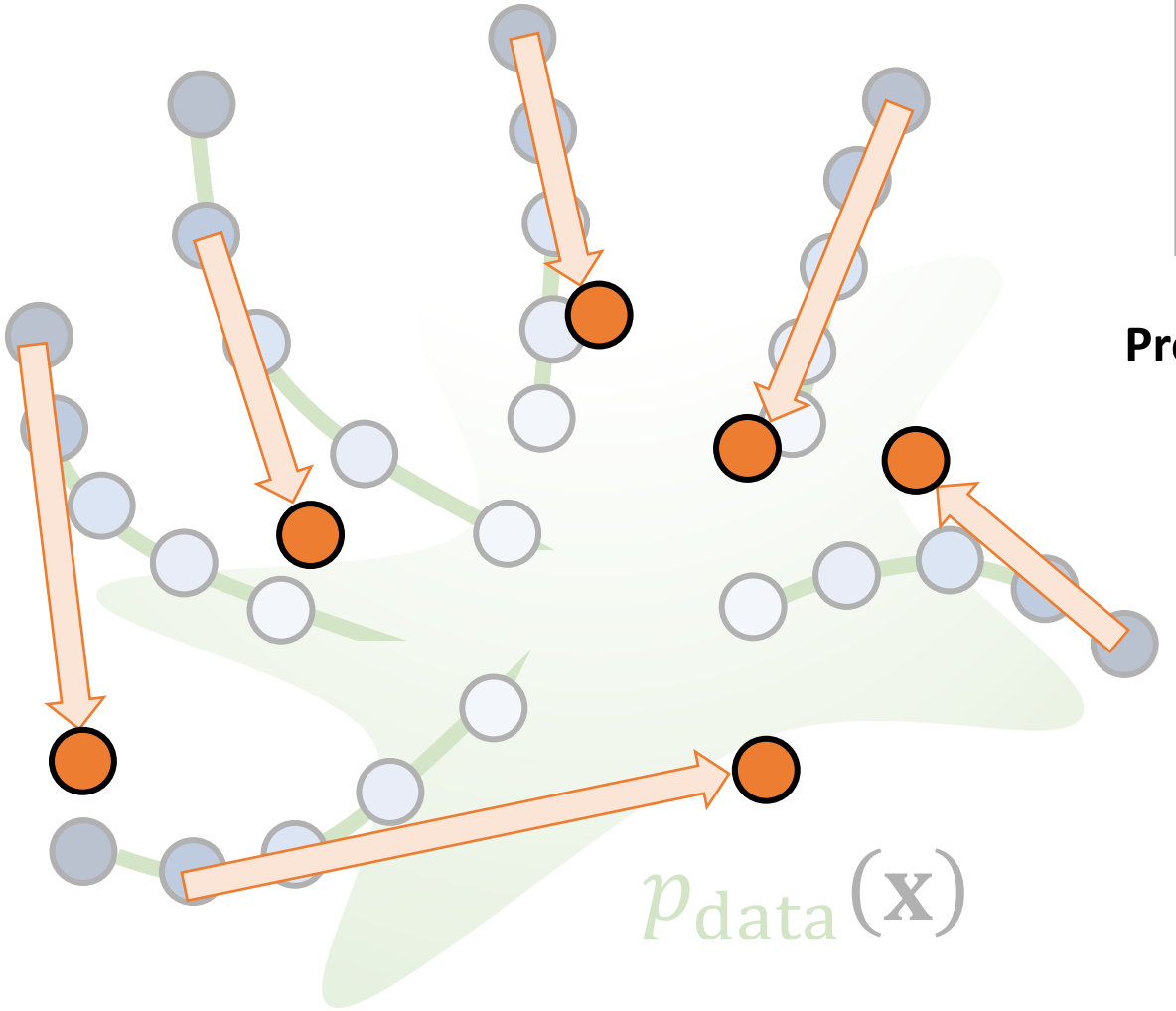


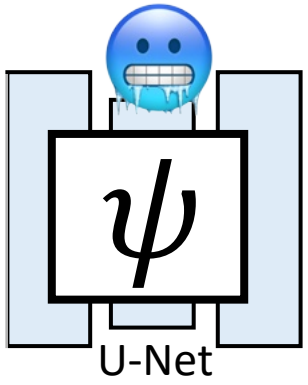
Student model

Slide from Jia-Bin Huang

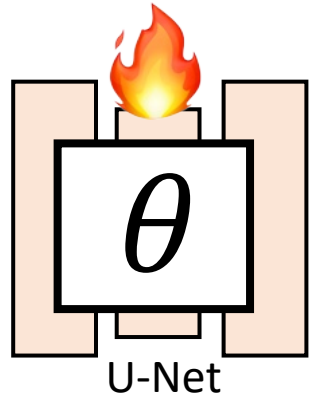
[Sauer et al. 2023]

Adversarial Diffusion Distillation





Pretrained model

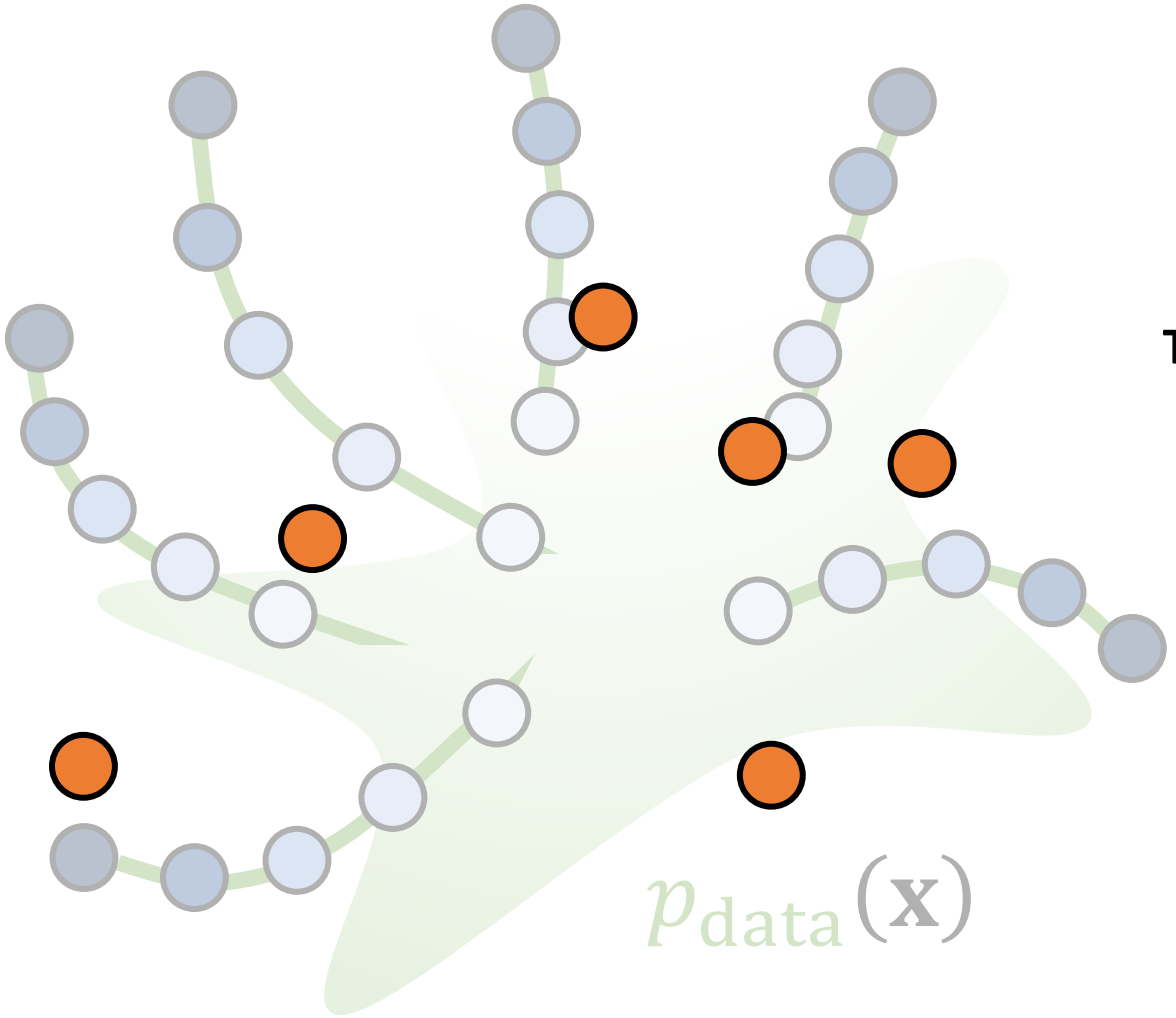


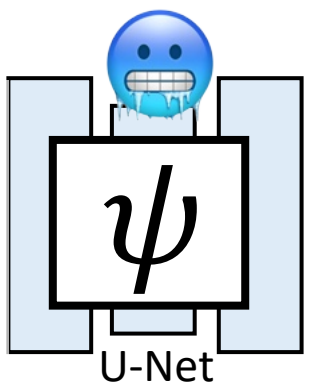
Student model

Slide from Jia-Bin Huang

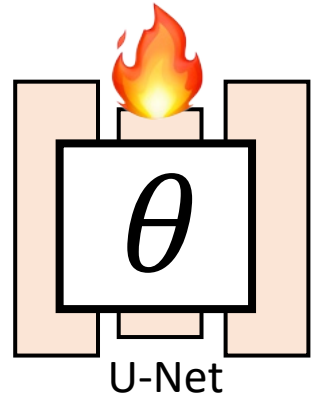
[Sauer et al. 2023]

Adversarial Diffusion Distillation

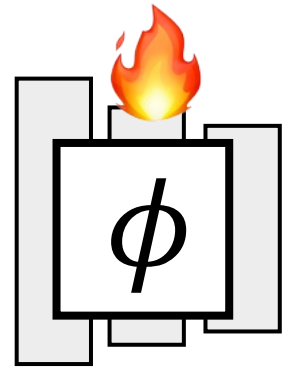




Teacher model



Student model



Discriminator

Problem:

- Denoised image predicted by the teacher: blurry

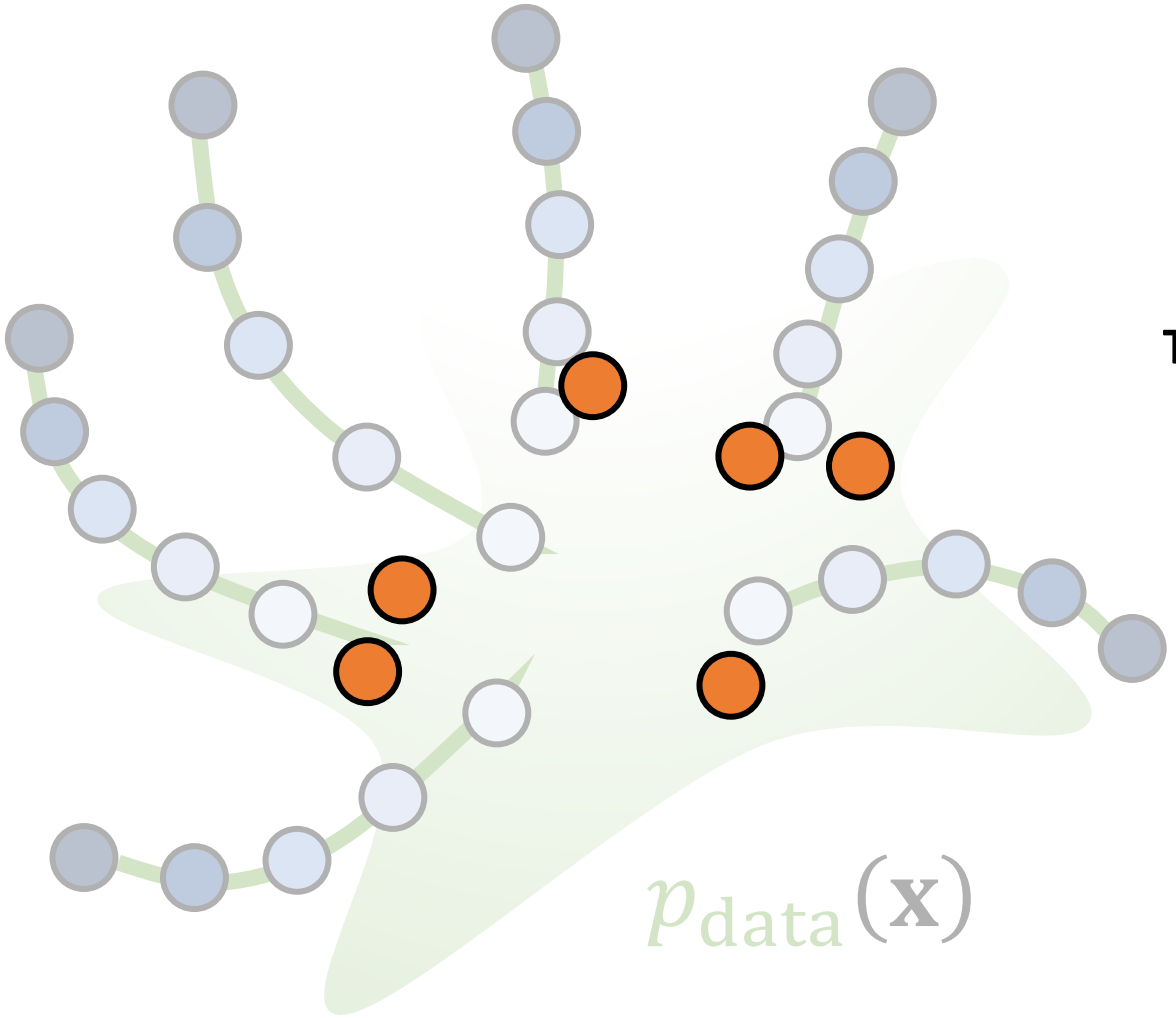
Idea:

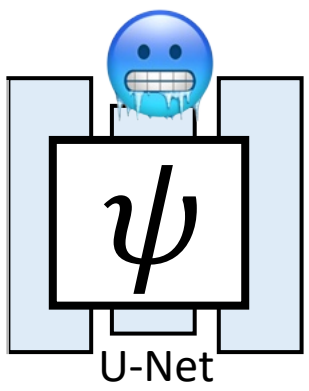
Use both adversarial loss and score distillation loss

Slide from Jia-Bin Huang

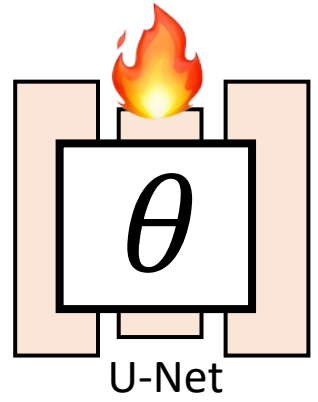
[Sauer et al. 2023]

Adversarial Diffusion Distillation

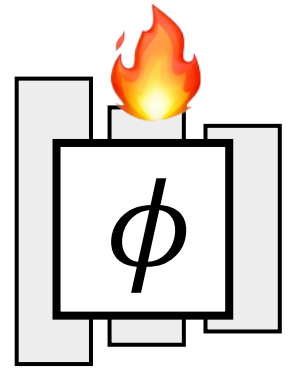




Teacher model



Student model



Discriminator

Problem:

- Denoised image predicted by the teacher: blurry

Idea:

Use both adversarial loss and score distillation loss

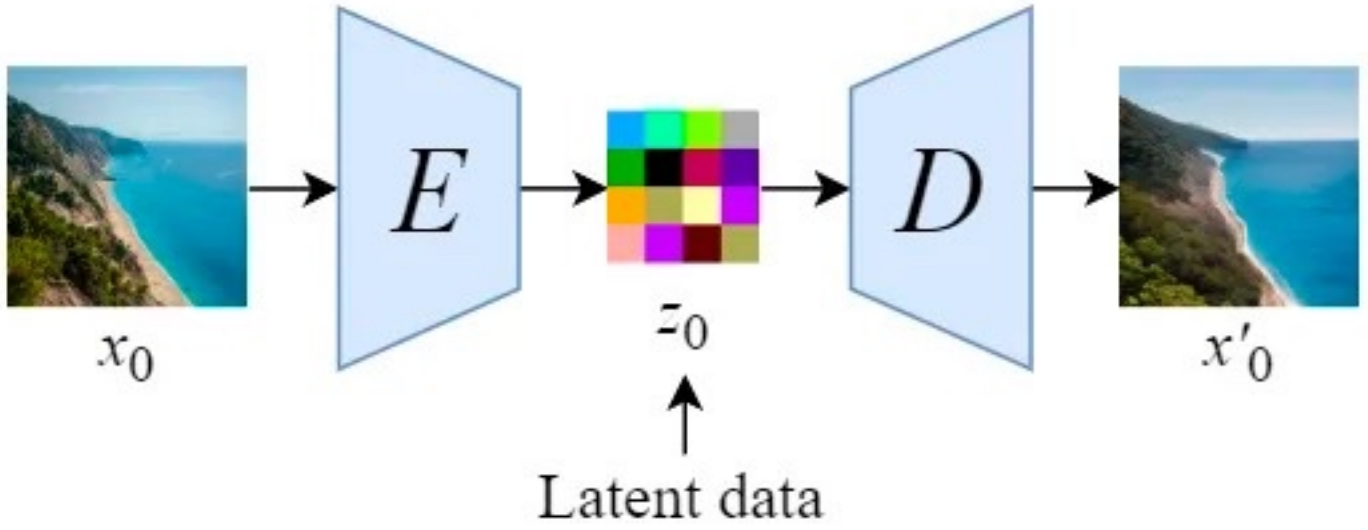
Slide from Jia-Bin Huang

[Sauer et al. 2023]

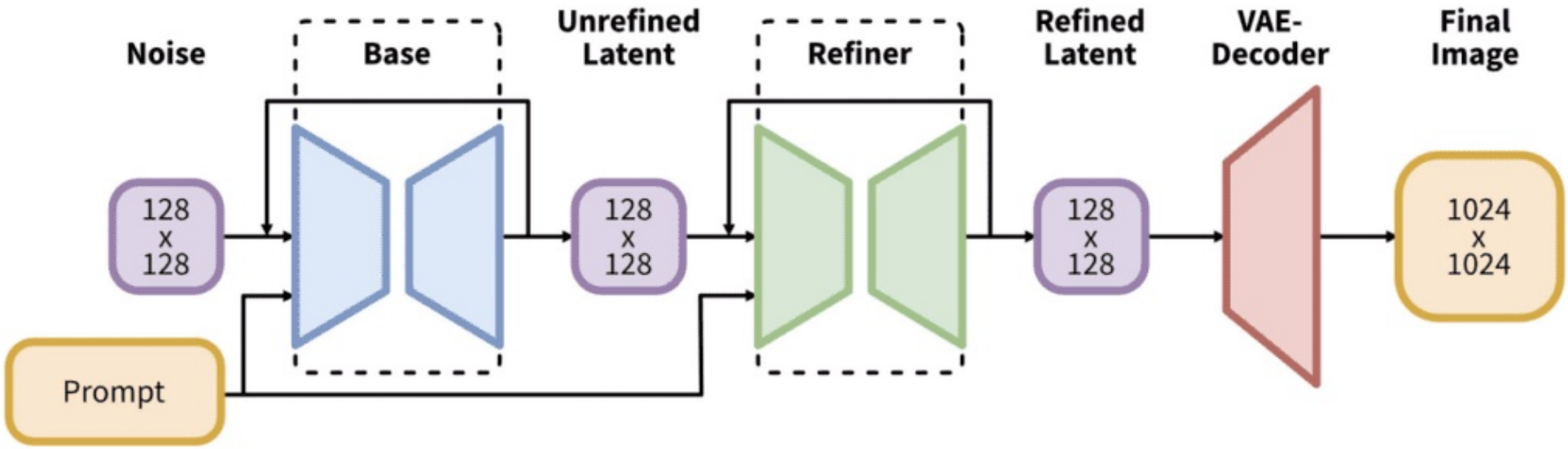
[Sauer et al. 2023]

Recent Improvements

Recall Stable Diffusion (SD v1.4, SD v1.5, SD v2.1)



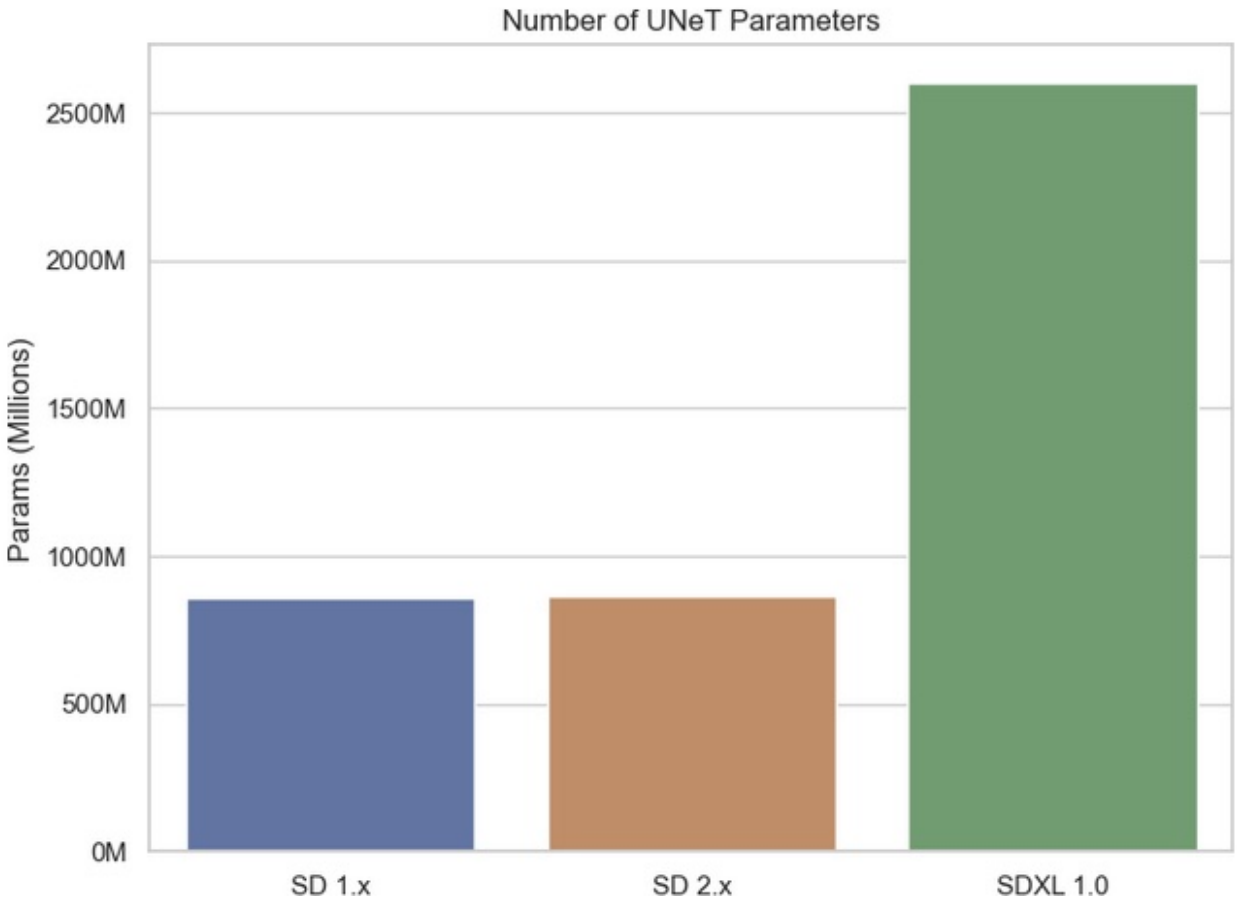
Improved SDXL pipeline

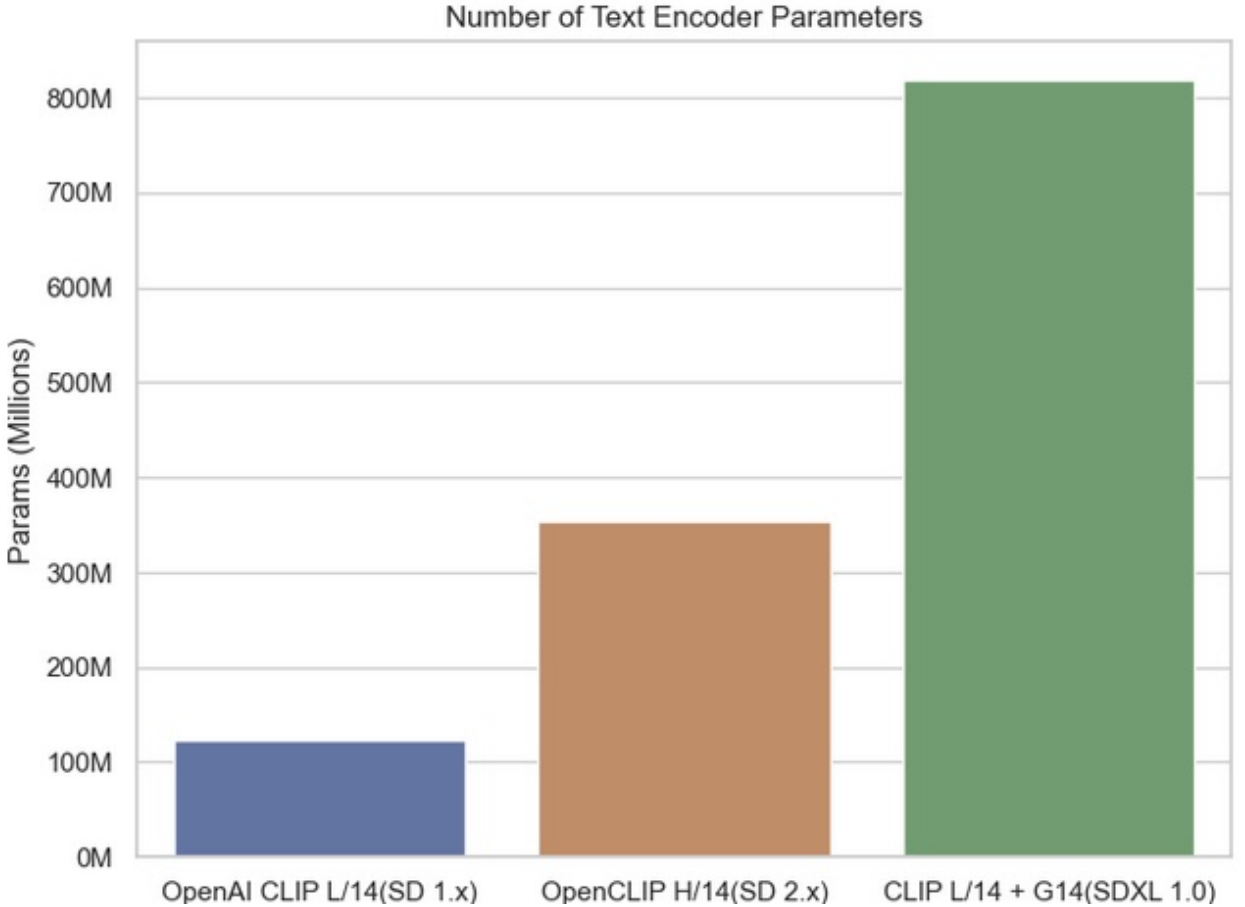


- Separate refiner model
- Two text-encoders
- Bigger U-net with more attention blocks and higher number of parameters

https://towardsdatascience.com/the-arrival-of-sdxl-1-0-4e739d5cc6c7

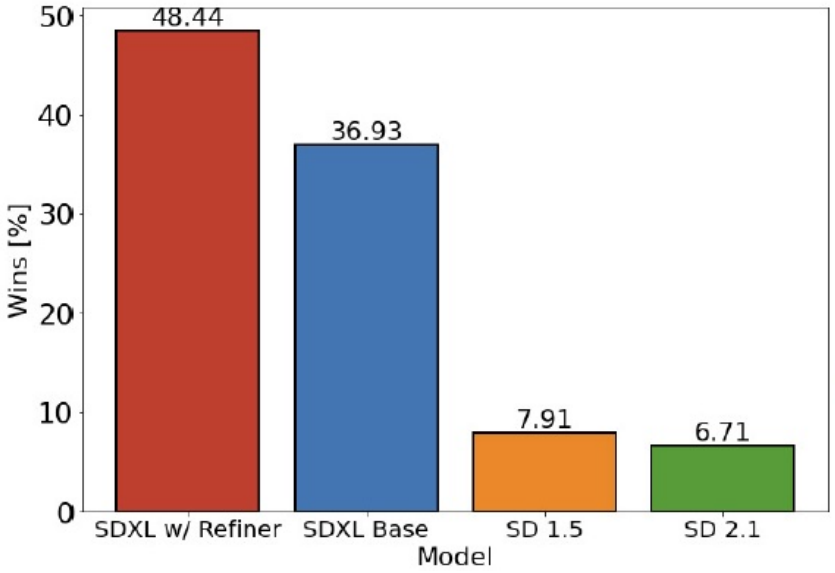
SDXL





https://towardsdatascience.com/the-arrival-of-sdxl-1-0-4e739d5cc6c7

SDXL: Improvements in generation quality



Comparing user preferences between SDXL and previous models.

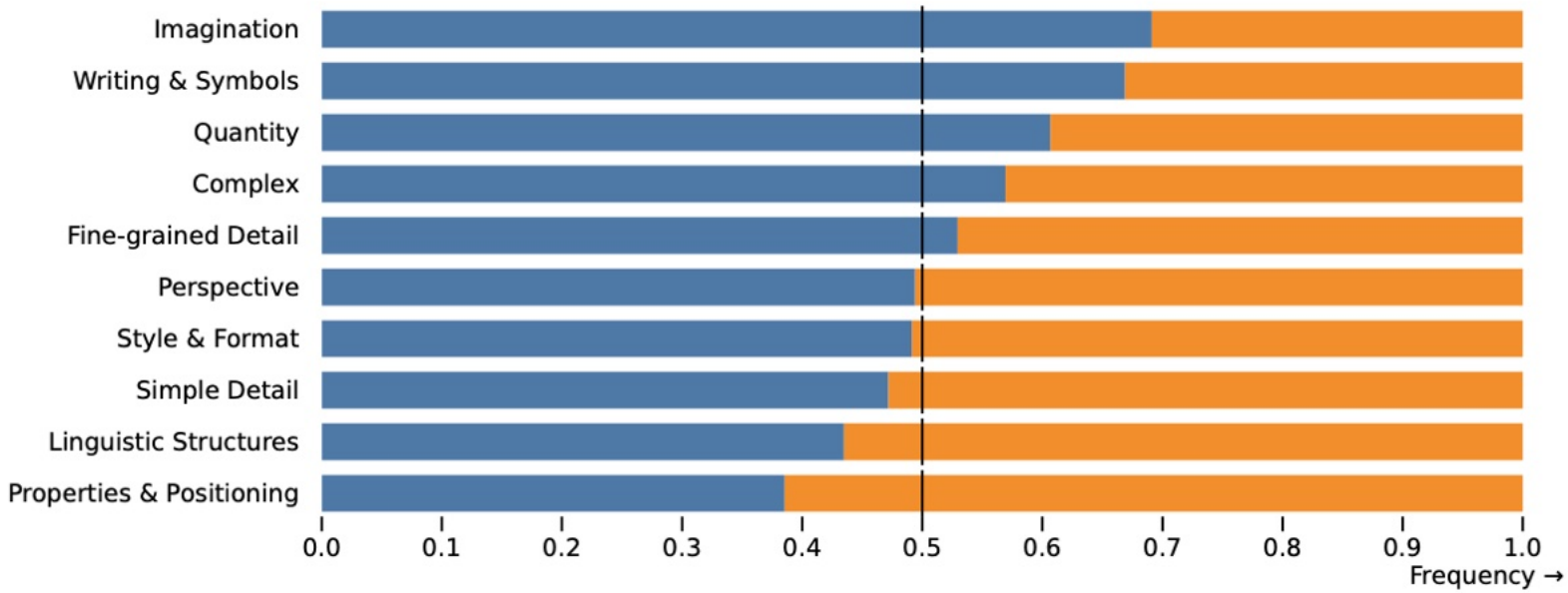


Figure 11: Preference comparisons of *SDXL* (with refinement model) to Midjourney V5.1 on complex prompts. *SDXL* either outperforms or is statistically equal to Midjourney V5.1 in 7 out of 10 categories.

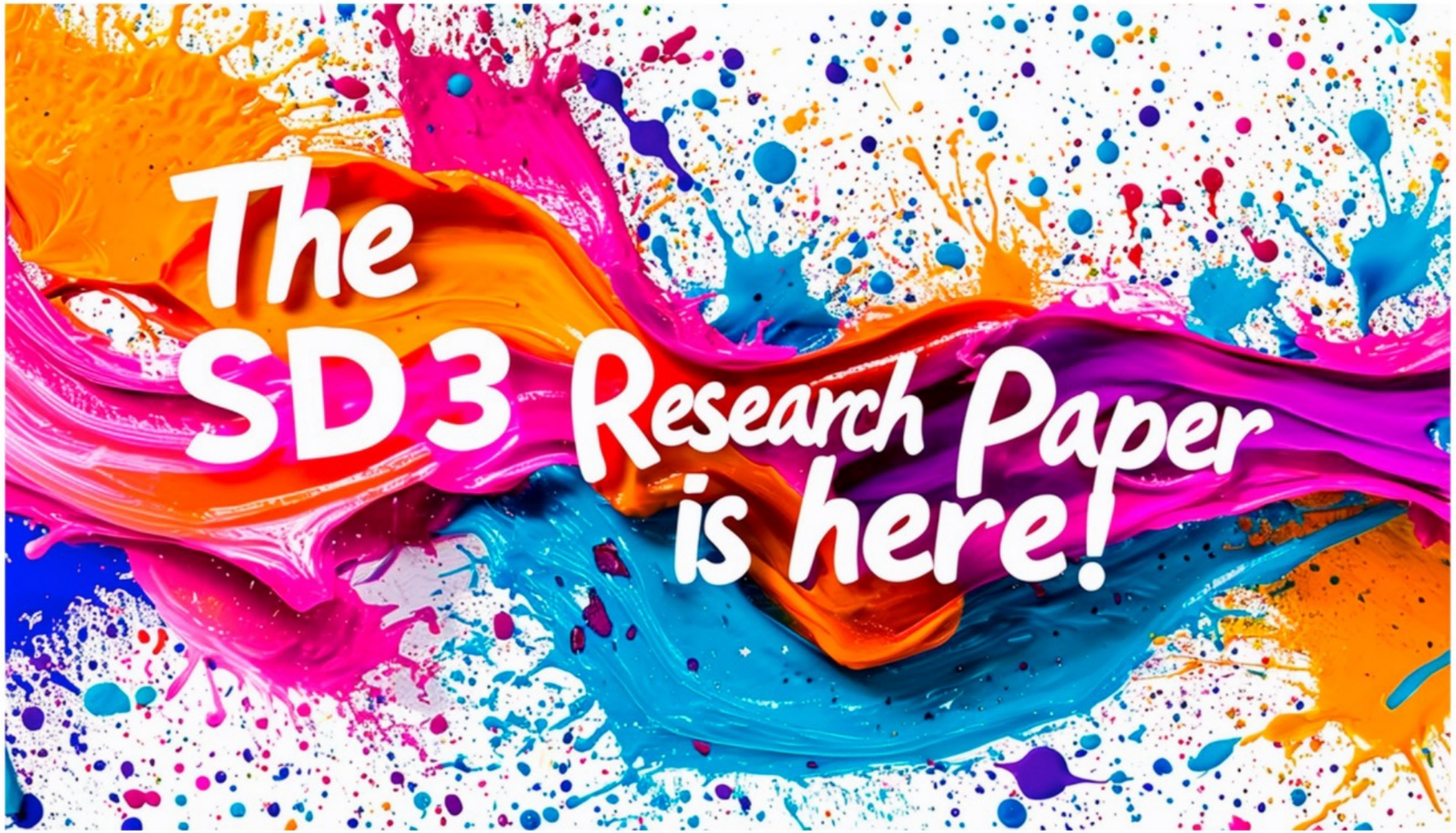
Improving Latent Diffusion Models for High-Resolution Image Synthesis

SDXL: Results

Figure 4: Comparison of the output of *SDXL* with previous versions of *Stable Diffusion*. For each prompt, we show 3 random samples of the respective model for 50 steps of the DDIM sampler [46] and cfg-scale 8.0 [13]. Additional samples in Fig. 14.

SDXL: Results

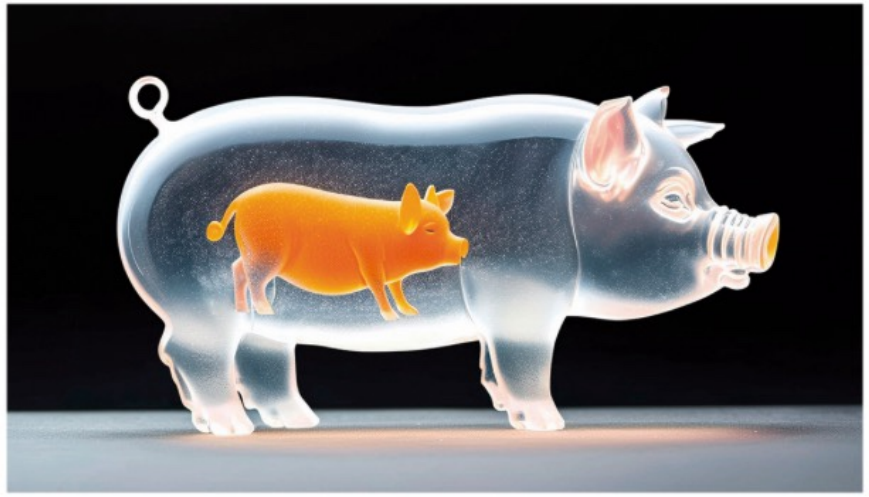
SD3



Prompt: A beautiful painting of flowing colors and styles forming the words "The SD3 research paper is here!", the background is speckled with drops and splashes of paint

https://stability.ai/news/stable-diffusion-3-research-paper

SD3: Results



Prompt: Translucent pig, inside is a smaller pig.

Prompt: A massive alien space ship that is shaped like a pretzel.

Prompt: A kangaroo holding a beer, wearing ski goggles and passionately singing silly songs.

Prompt: An entire universe inside a bottle sitting on the shelf at walmart on sale.

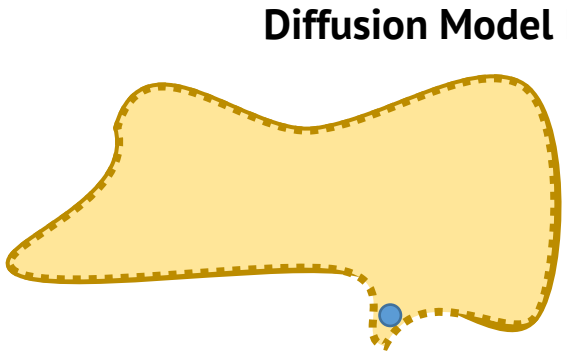
Prompt: A cheeseburger with juicy beef patties and melted cheese sits on top of a toilet that looks like a

Prompt: This dreamlike digital art captures a vibrant, kaleidoscopic bird in a lush rainforest

Prompt: A car made out of vegetables.

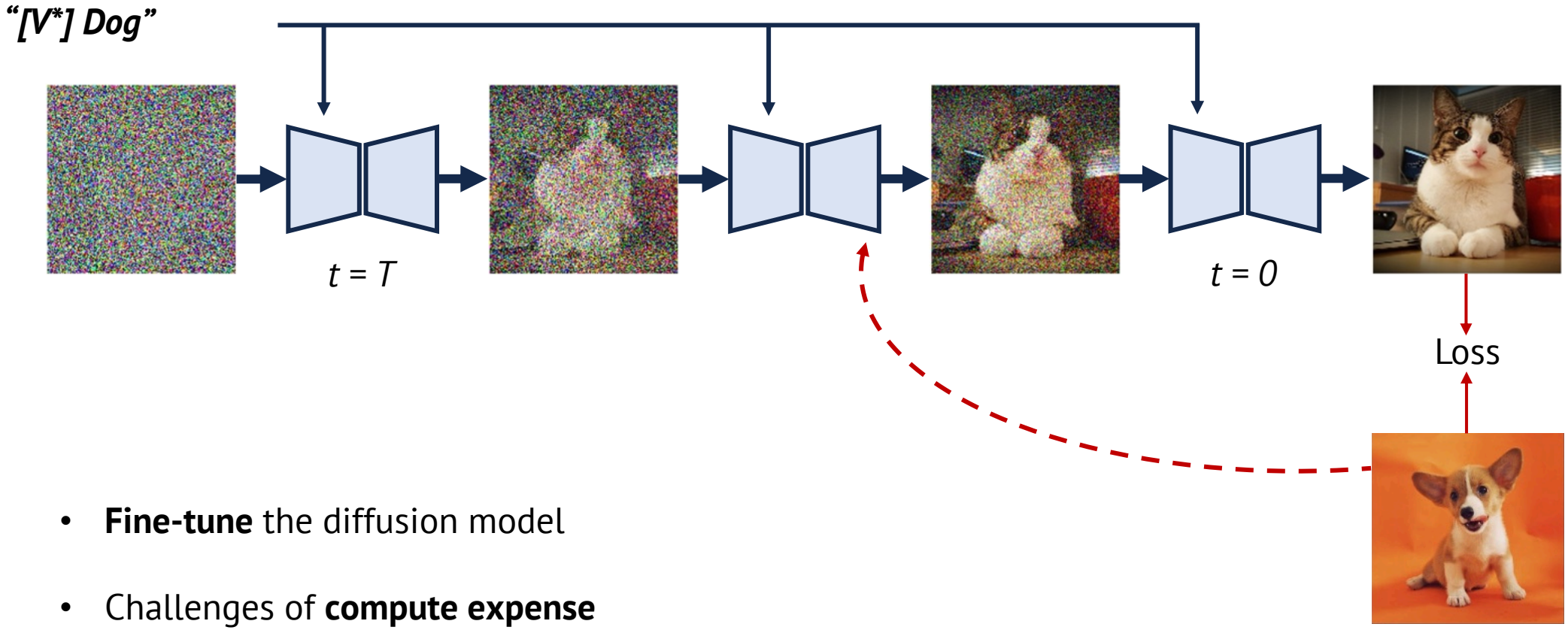
Prompt: Heat death of the universe line art

Local Customization of Diffusion Model



Ref: DreamBooth, Ruiz et al., CVPR '23

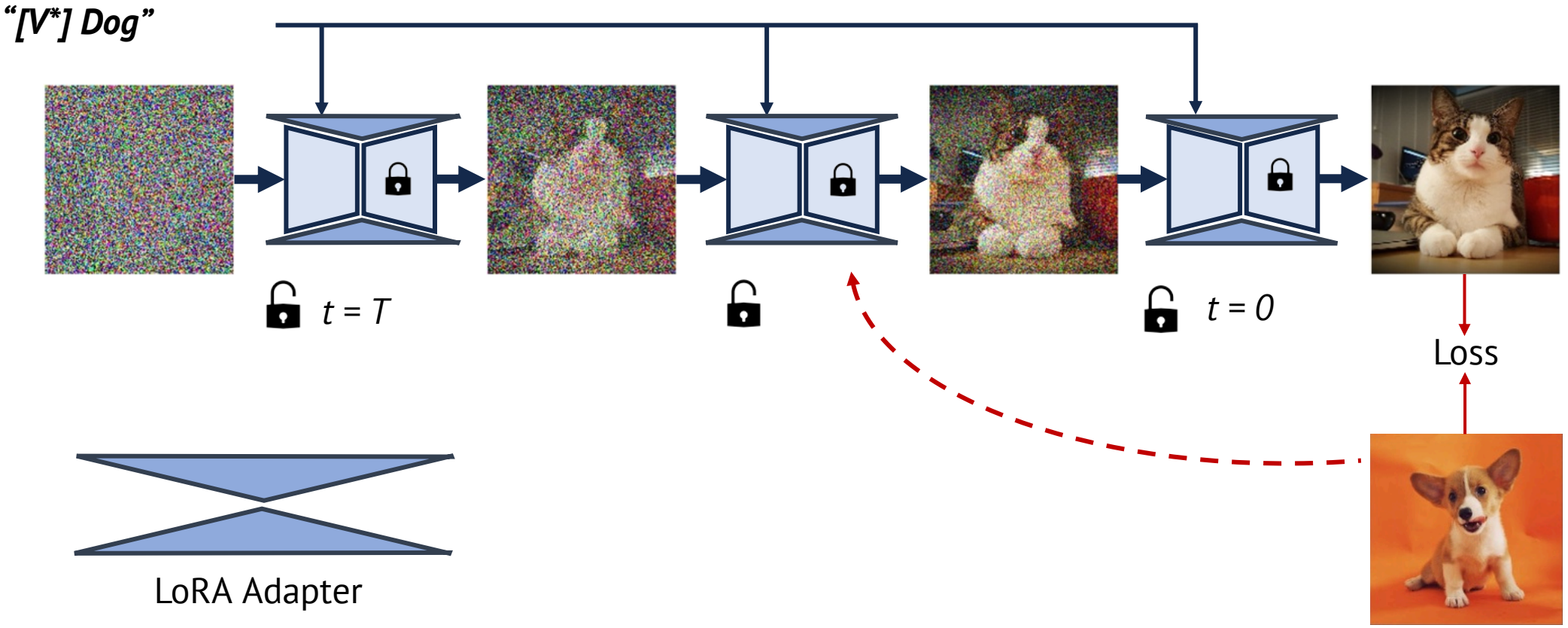
DreamBooth: Local Customization of Diffusion Model



- **Fine-tune** the diffusion model
- Challenges of **compute expense**

Ref: DreamBooth, Ruiz et al., CVPR '23

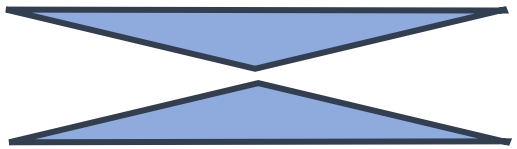
DreamBooth with Low Rank Approximation (LoRA)



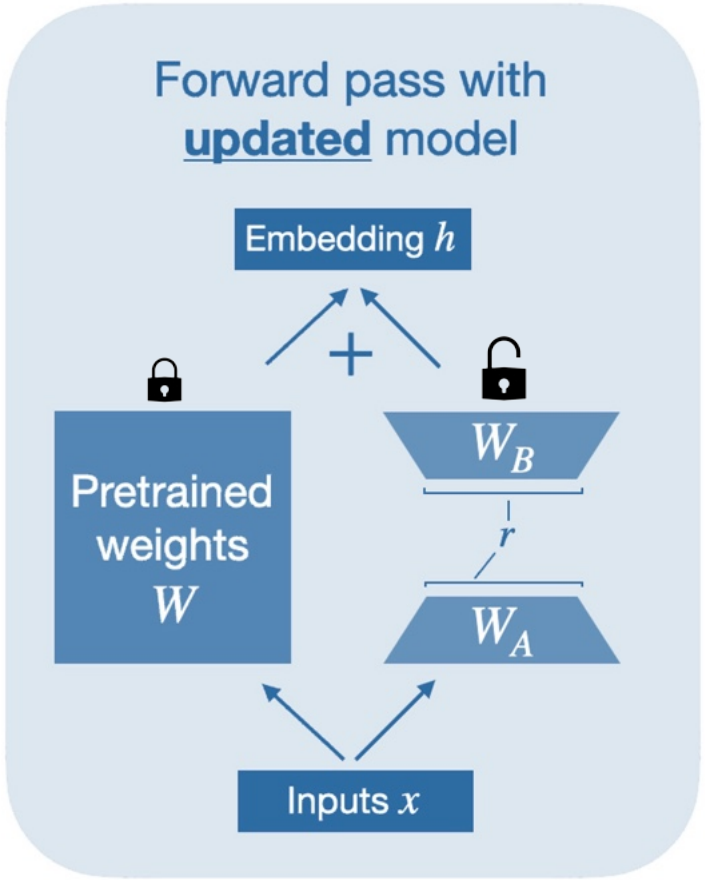
Ref: DreamBooth, Ruiz et al., CVPR '23
LoRA, Hu et al., '21

Low Rank Approximation (LoRA)

LoRA weights, W_A and W_B, represent ΔW



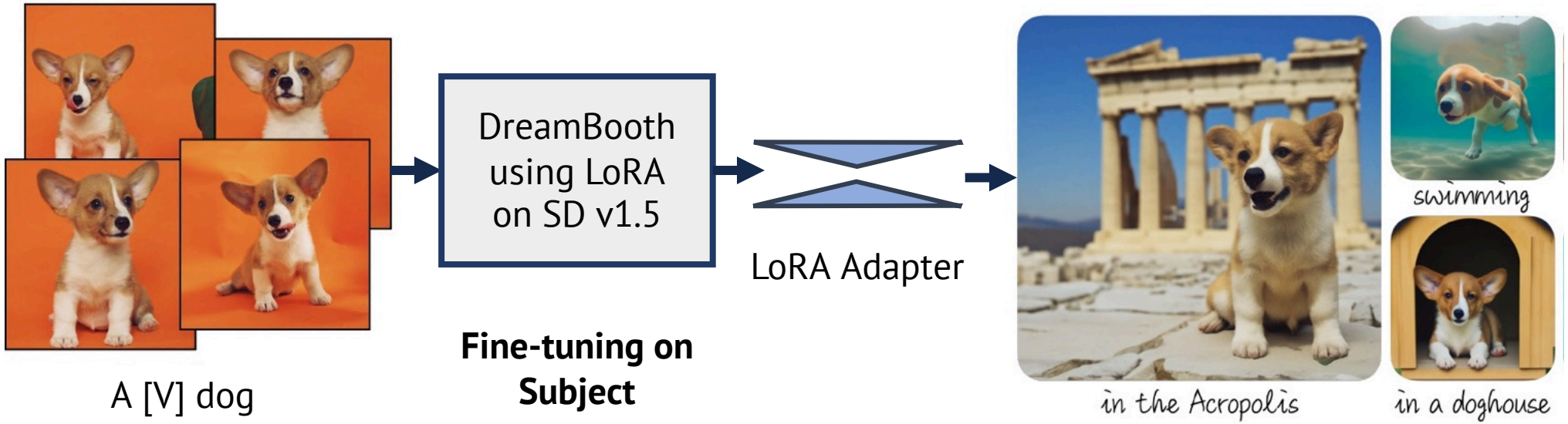
LoRA Adapter



How to use LoRA Fine-tuning for Stylization?

Ref: LoRA, Hu et al. 2021

DreamBooth with Low Rank Approximation (LoRA)



DreamBooth with Low Rank Approximation (LoRA)

A [V] dog

DreamBooth using LoRA on SD v1.5

Fine-tuning on Subject

LoRA Adapter

A statue in [S] style
[S]: “matte black sculpture”

DreamBooth using LoRA on SD v1.5

Fine-tuning on Style

LoRA Adapter

A cat in [S] style; A men in [S] style; Old lady in [S] style

Doesn’t work!

Slide from Jia-Bin Huang

DreamBooth with Low Rank Approximation (LoRA)

A [V] dog

DreamBooth using LoRA on SD v1.5

Fine-tuning on Subject

LoRA Adapter

A statue in [S] style
[S]: “matte black sculpture”

DreamBooth using LoRA on SD v1.5

Fine-tuning on Style

LoRA Adapter

A cat in [S] style; A men in [S] style; Old lady in [S] style

Doesn’t work!

Slide from Jia-Bin Huang

LoRA on SDXL

A statue in [S] style
[S]: “matte black sculpture”

DreamBooth using LoRA on **SDXL**

LoRA Adapter

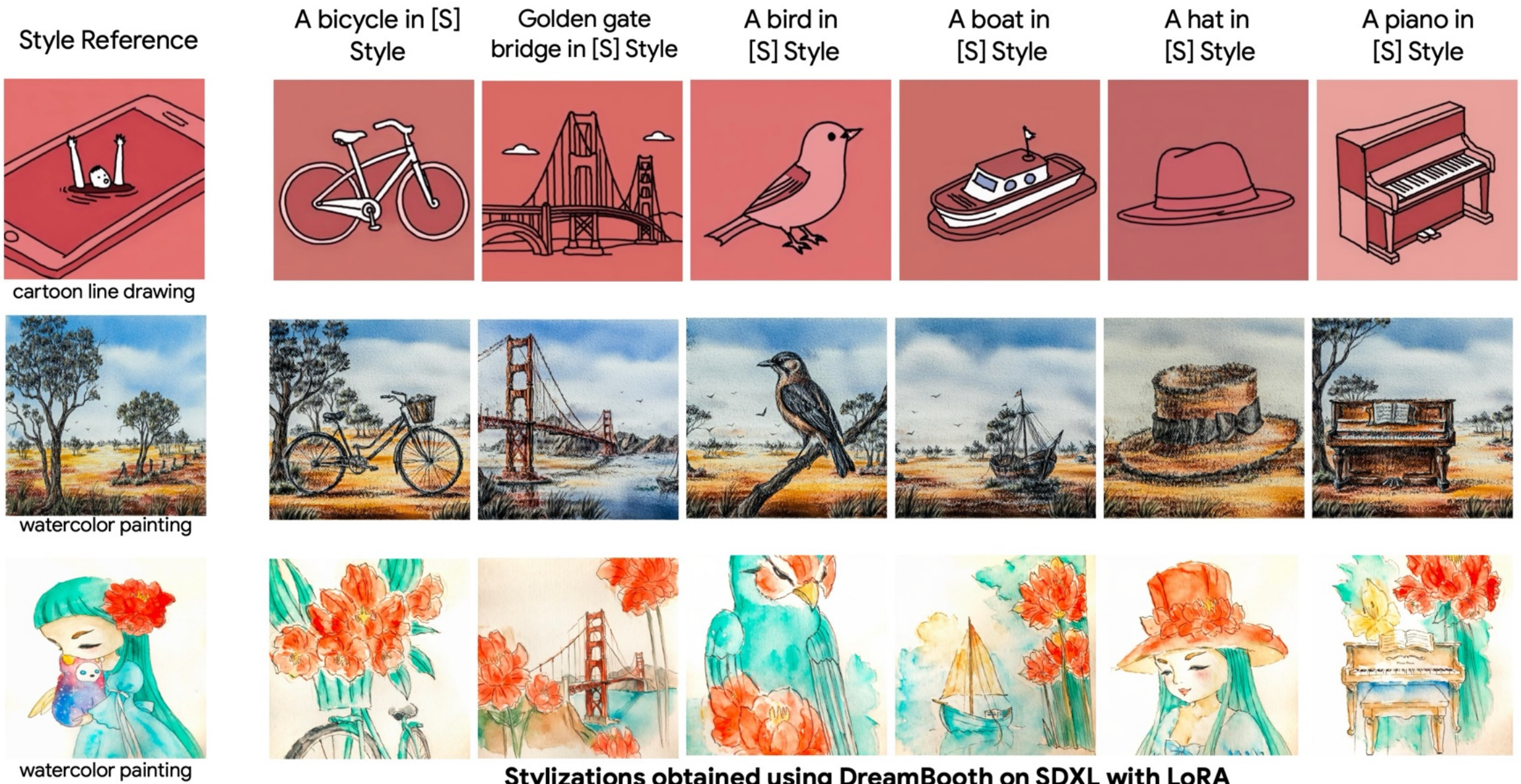
A cat in [S] style; A men in [S] style; Old lady in [S] style

Works with SDXL!

Ref: SDXL, Podell et al.‘23

Slide from Jia-Bin Huang

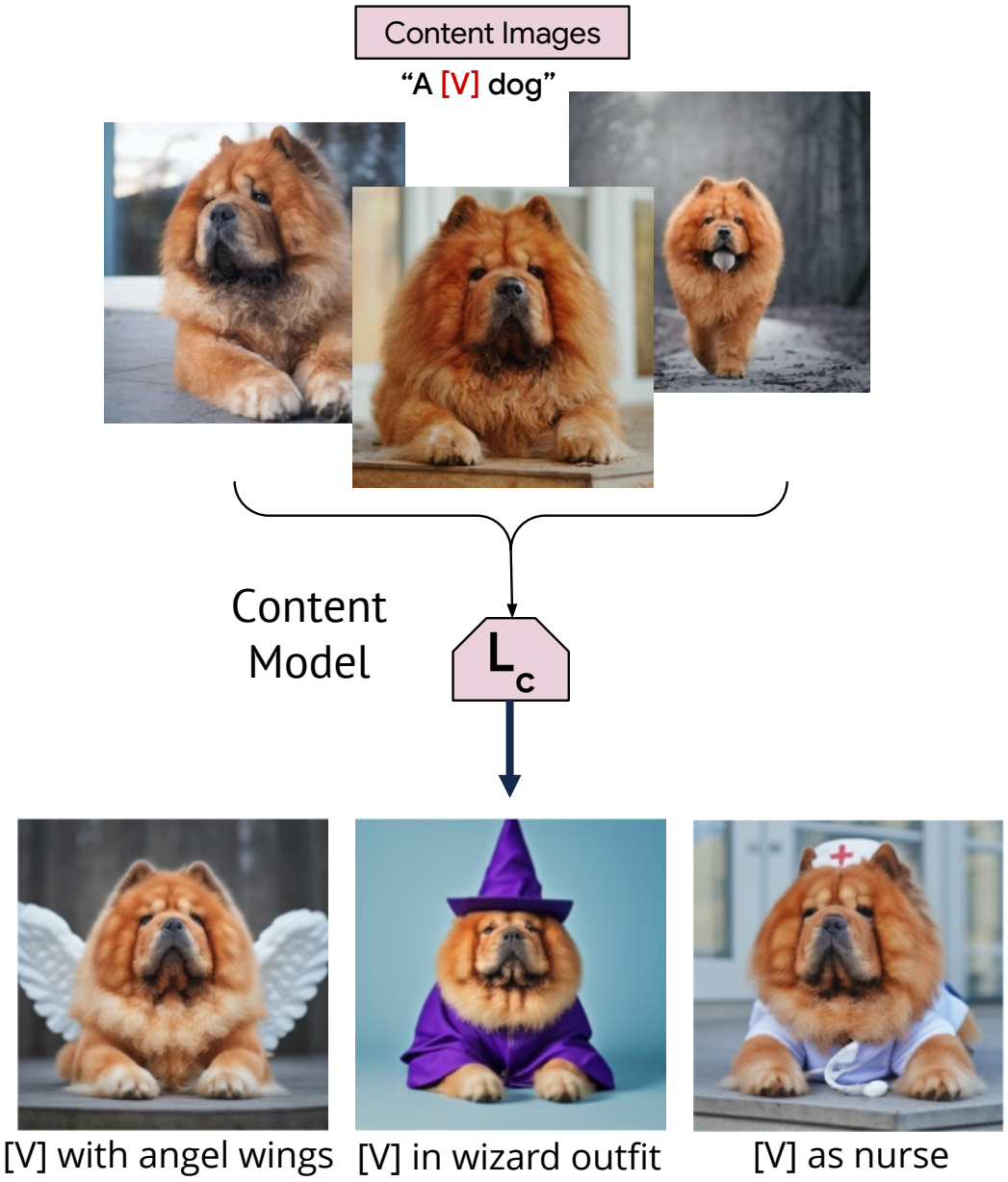
DreamBooth LoRA on SDXL for Stylizations



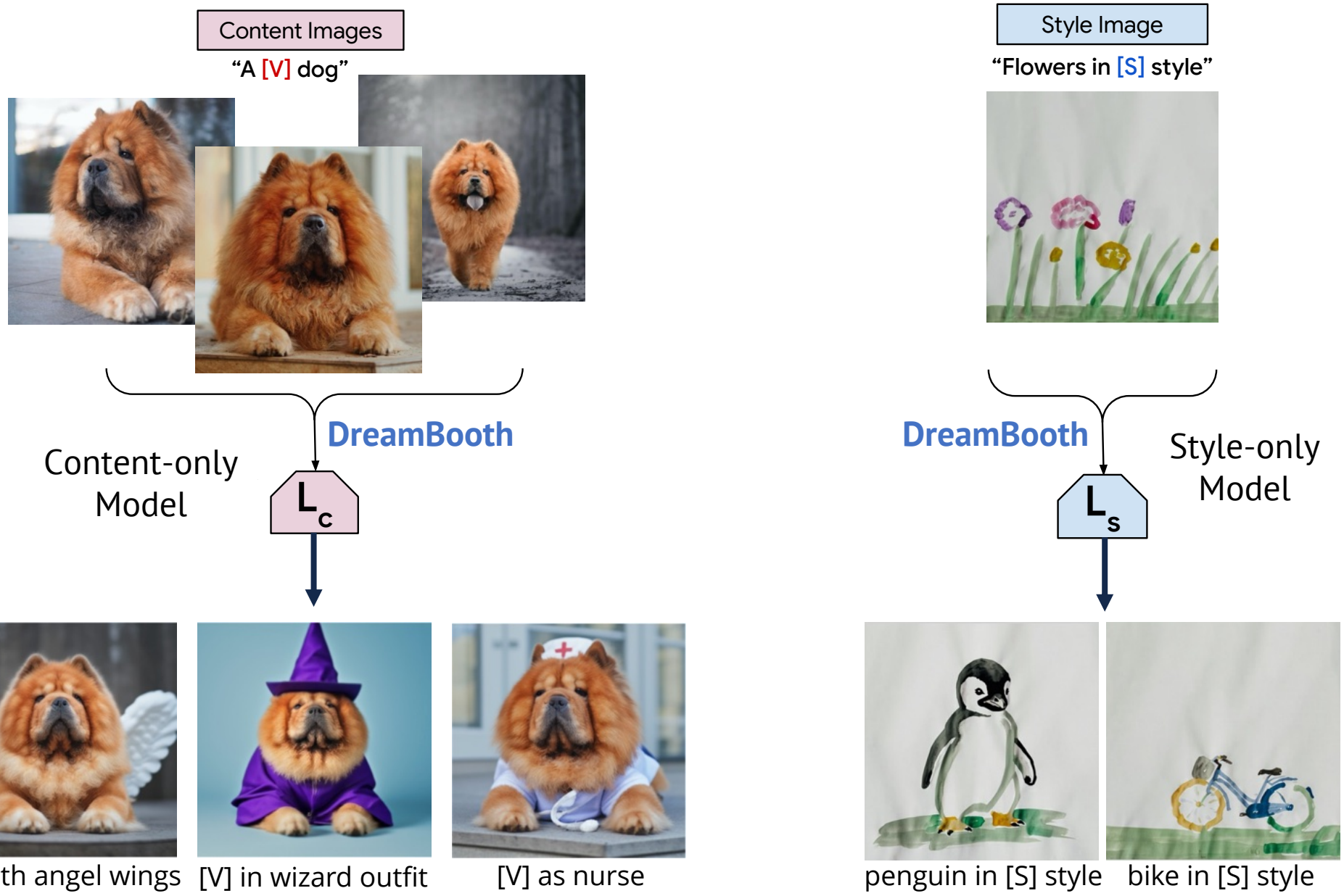
Stylizations obtained using DreamBooth on SDXL with LoRA

Slide from Jia-Bin Huang

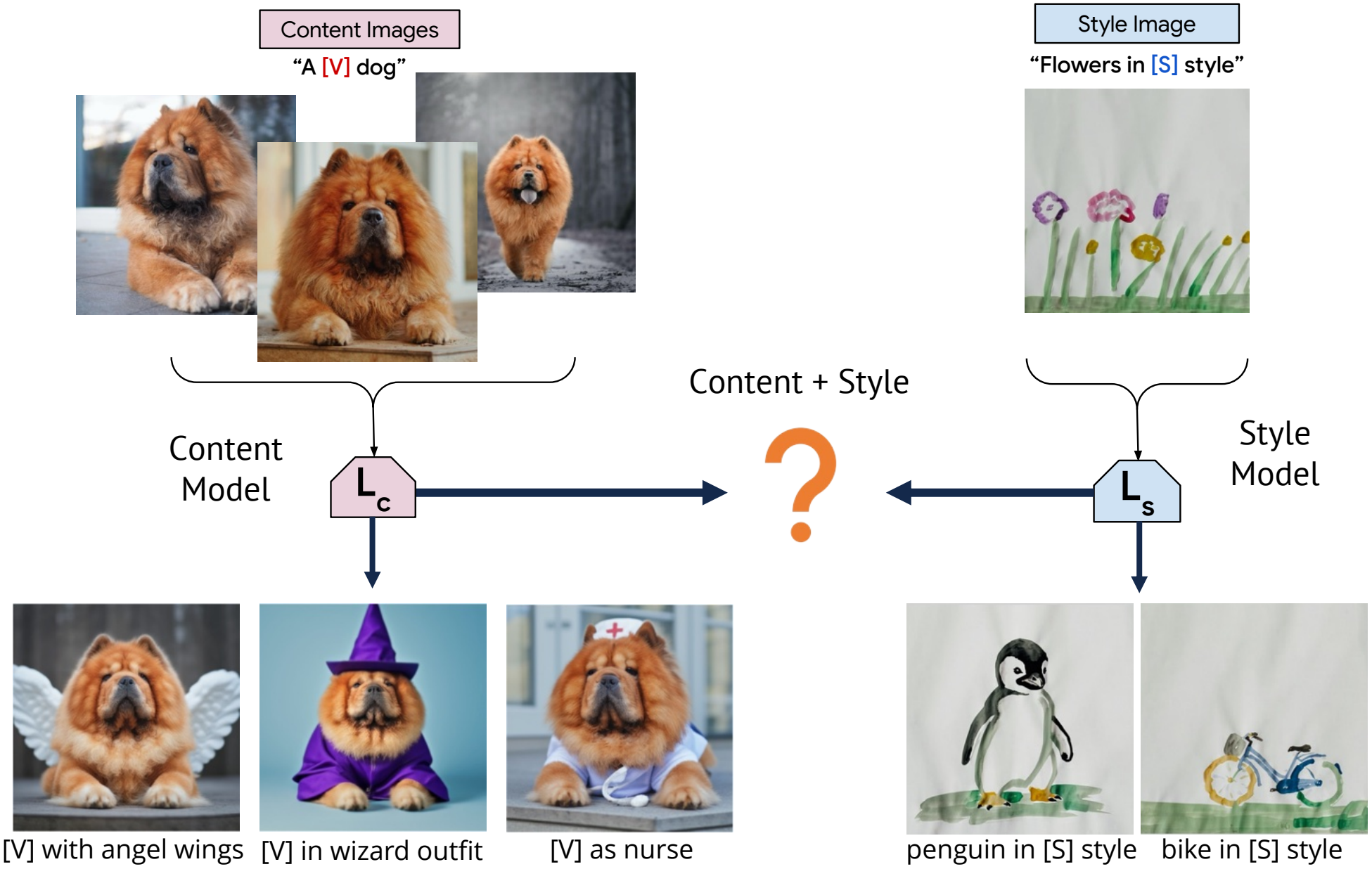
Can we Merge Content and Style Models?



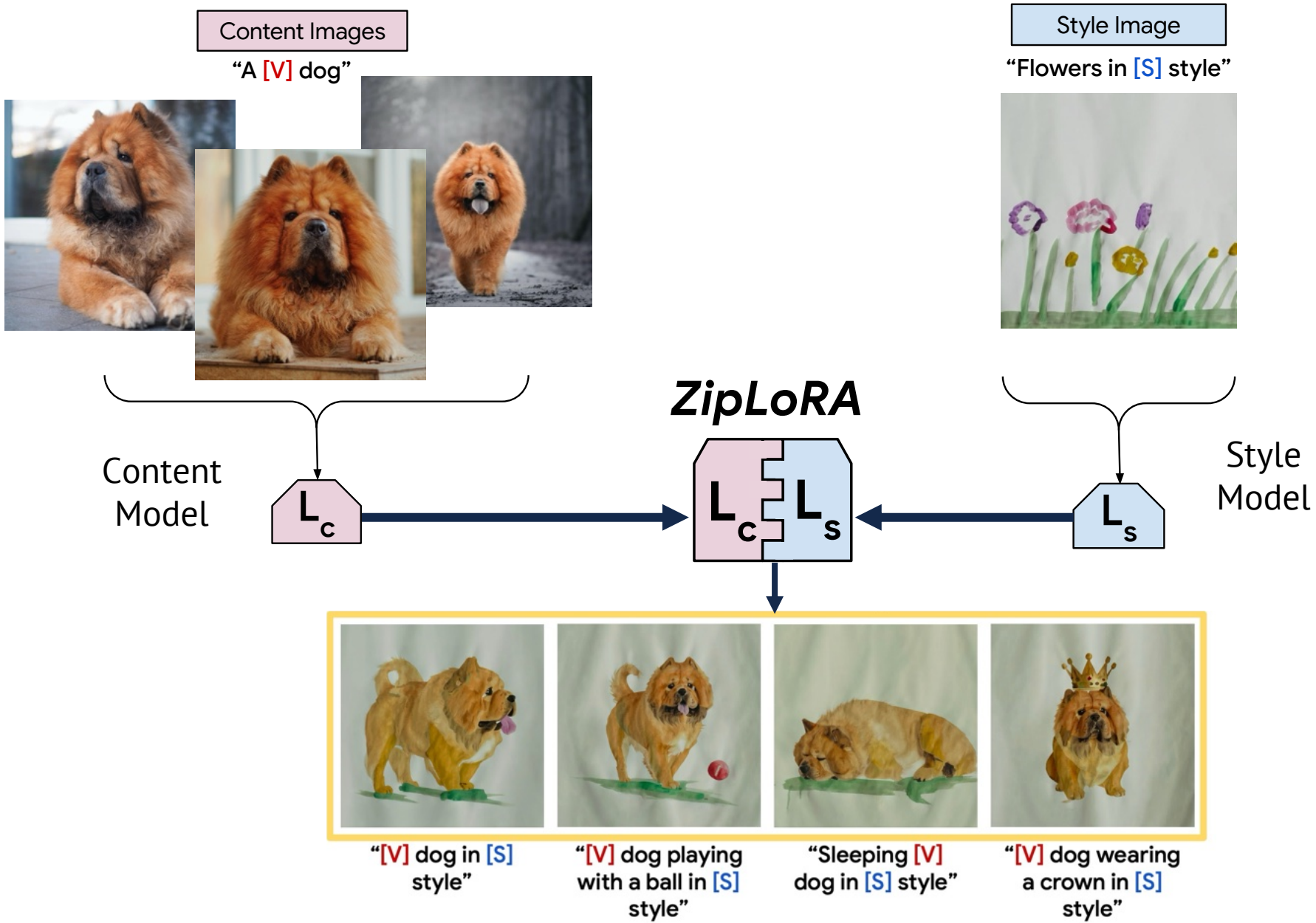
Can we Merge Content and Style Models?



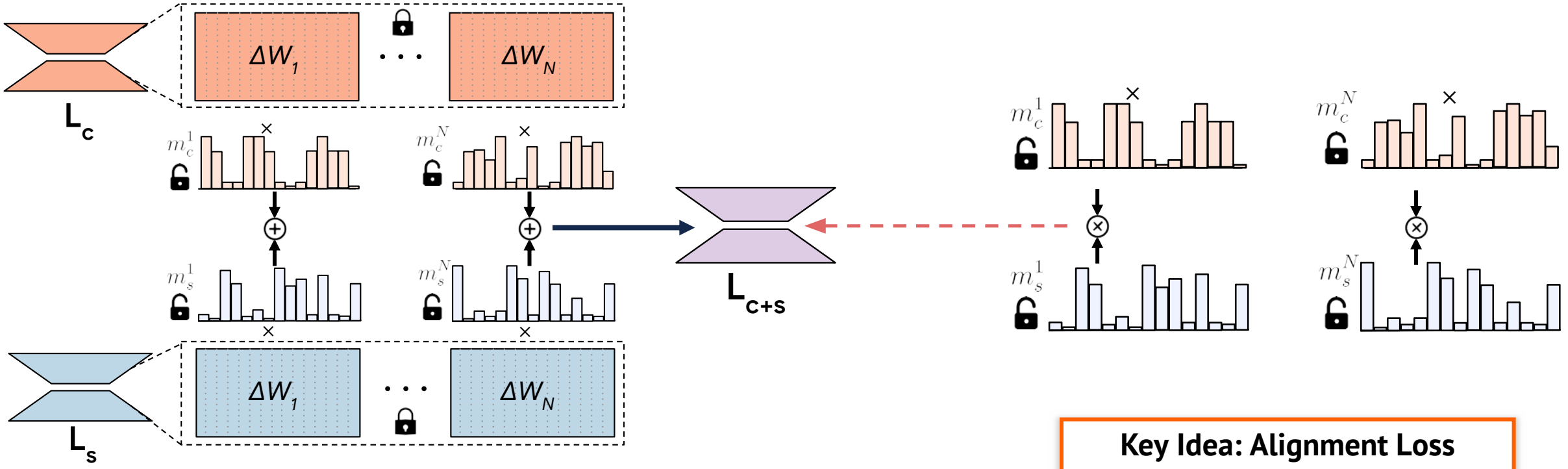
Can we Merge Content and Style Models?



Can we Merge Content and Style Models?



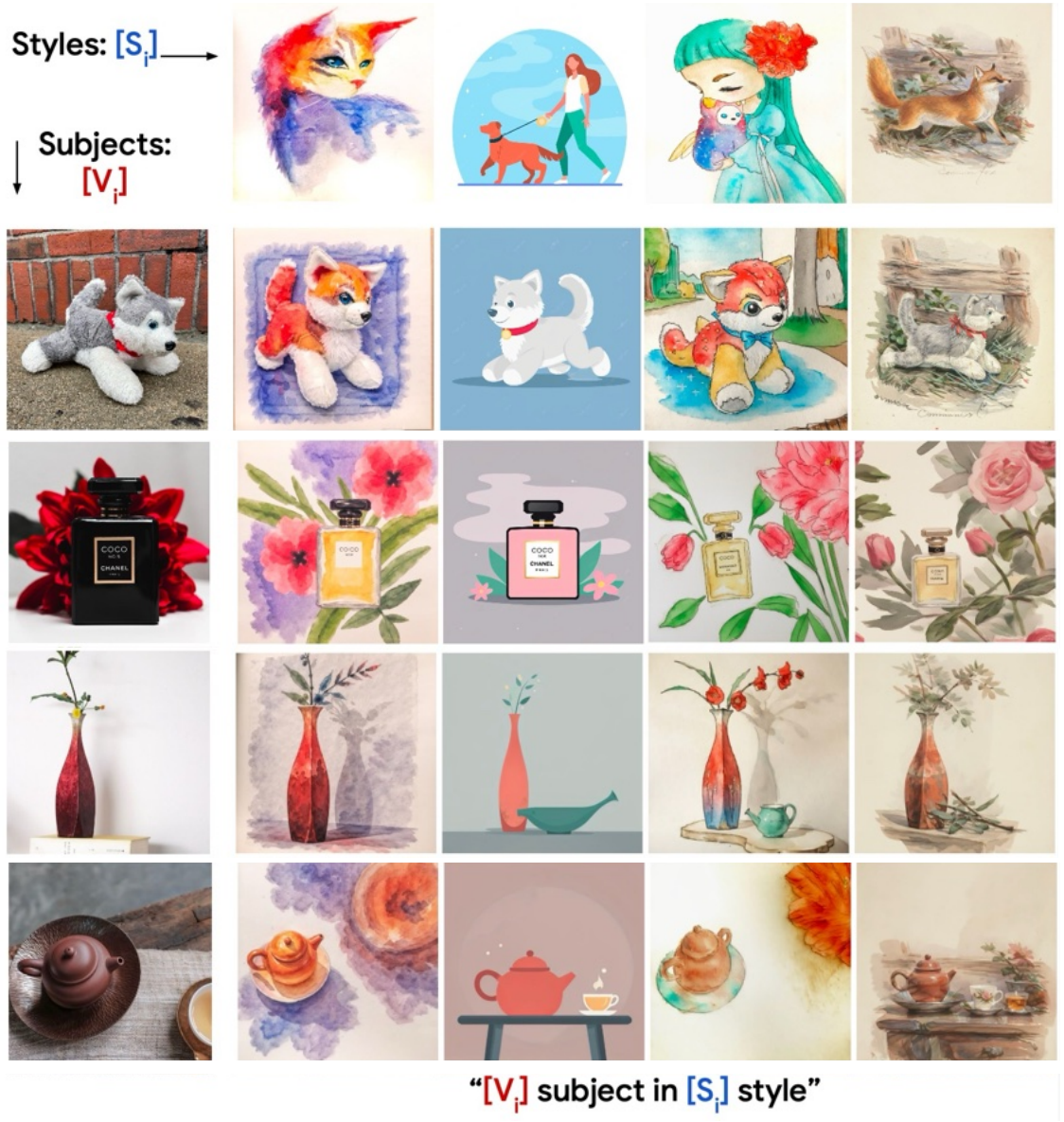
ZipLoRA: Alignment Loss



Key Idea: Alignment Loss

Make m_c and m_s orthogonal with each other!

Any Subject in Any Style



Ref: StyleDrop, Sohn et al., NeurIPS '23

Please fill in the Feedback Form at:

https://virajshah.com/sc395-feedback4

Thank You!