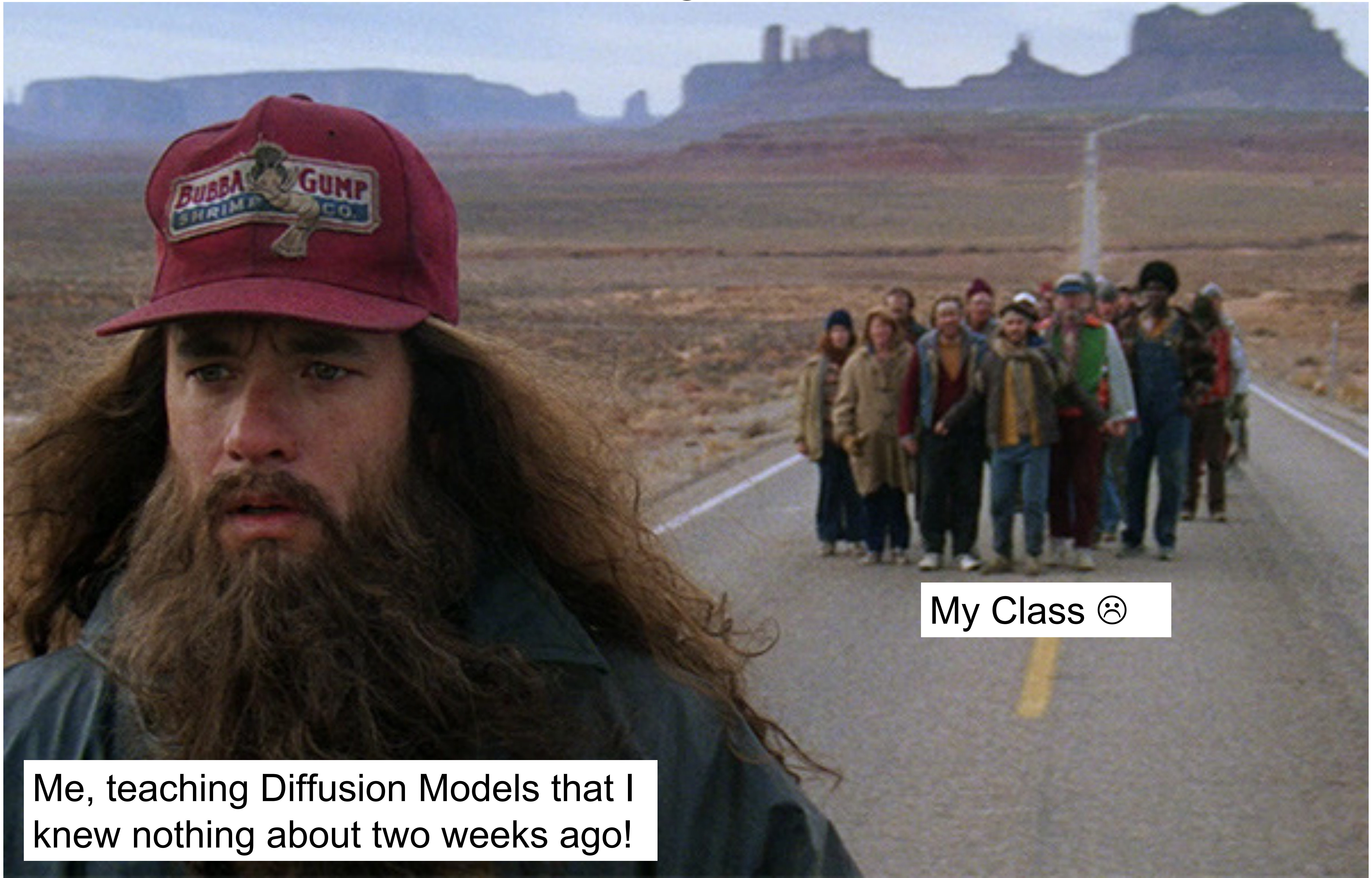
Lecture X: Denoising Diffusion Models



Next few lectures: Generative models for direct image based rendering.

Current Image

Vision

Encoder

Latent Space of Generative models

3D Intrinsic Components

Edit Components

Graphics

Decoder

New Image under different condition

Change:

- Viewpoint
- Lighting
- Reflectance
- Background
- Attributes
- Many others…

Implicit: Use a Neural Network (Conditional Generative networks) Often, end-to-end.

Slide Courtesy:

Denoising Diffusion-based Generative Modeling: Foundations and Applications, CVPR 2022 tutorial,
Karsten Kreis, Ruiqi Gao, Arash Vahdat

https://cvpr2022-tutorial-diffusion-models.github.io/

@karsten_kreis

@RuiqiGao

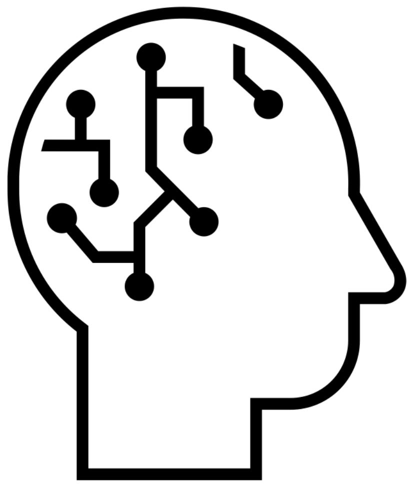
@ArashVahdat

Deep Generative Learning

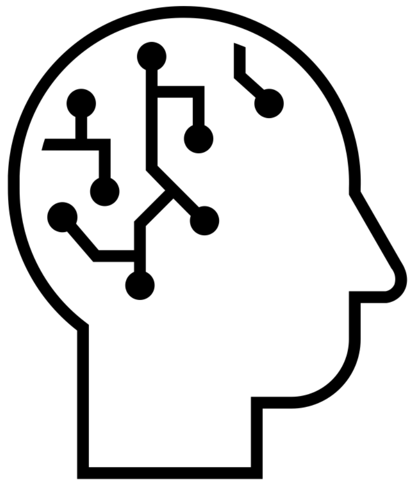
Learning to generate data

Samples from a Data Distribution

Train



Neural Network



Sample

The Landscape of Deep Generative Learning

Denoising Diffusion Models

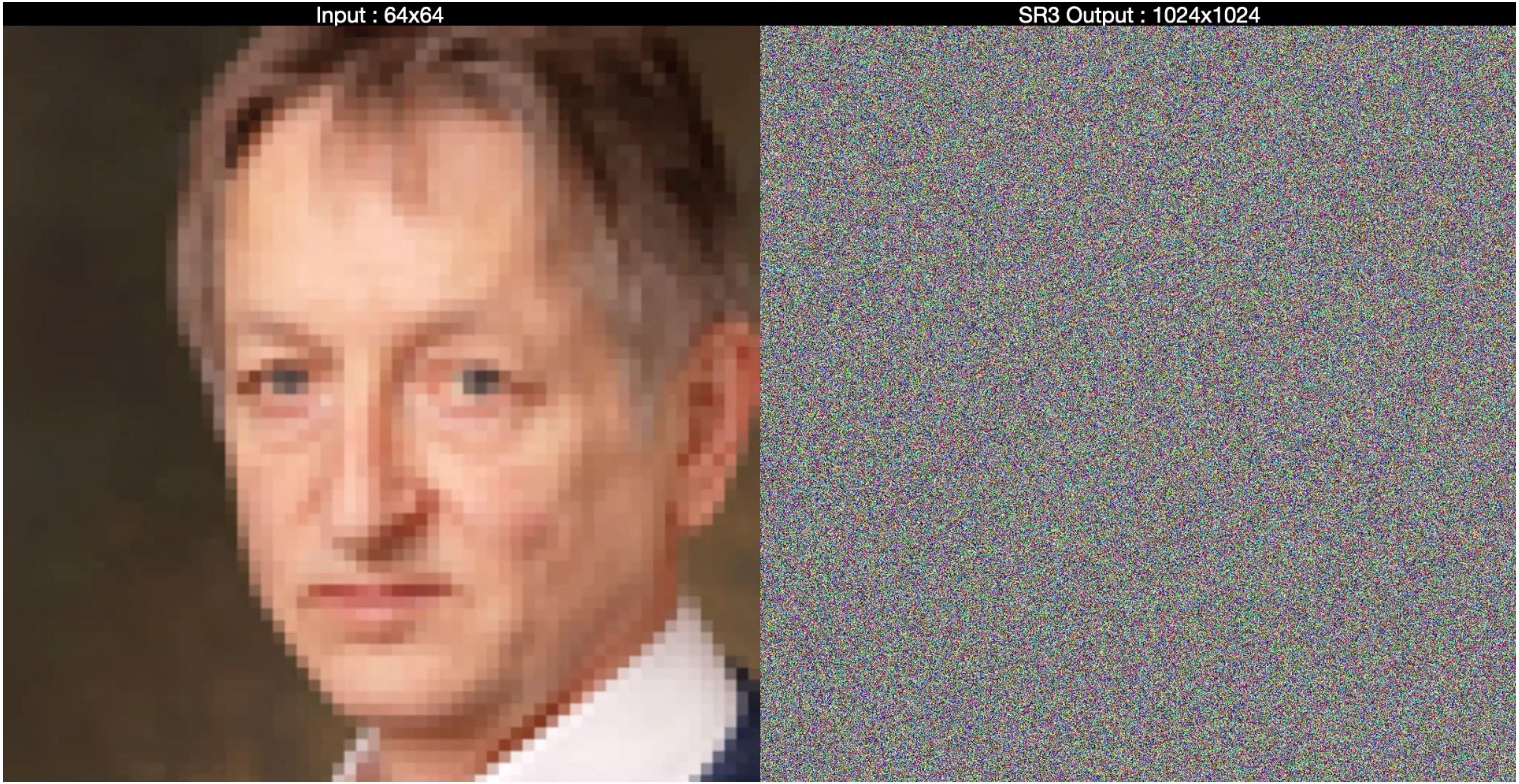
Emerging as powerful generative models, outperforming GANs

“Diffusion Models Beat GANs on Image Synthesis”
Dhariwal & Nichol, OpenAI, 2021

“Cascaded Diffusion Models for High Fidelity Image Generation”
Ho et al., Google, 2021

Image Super-resolution

Successful applications



[Saharia et al., Image Super-Resolution via Iterative Refinement, ICCV 2021]

Text-to-Image Generation

DALL·E 2

"a teddy bear on a skateboard in times square"

[“Hierarchical Text-Conditional Image Generation with CLIP Latents” Ramesh et al., 2022]

Imagen

A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.

[“Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, Saharia et al., 2022]

Text-to-Image Generation

Stable Diffusion

[Stable Diffusion Applications: Twitter Mega Thread]

["High-Resolution Image Synthesis with Latent Diffusion Models" Rombach et al., 2022]

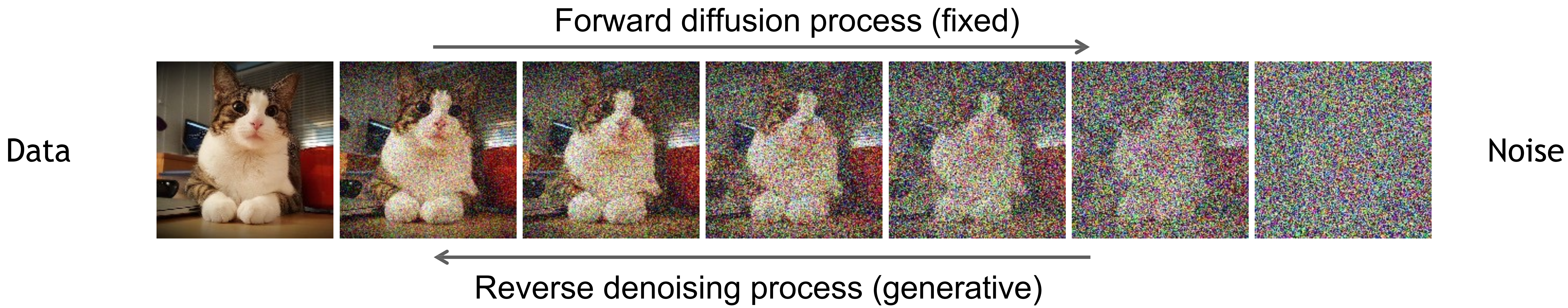
Q: What is a diffusion model?

Denoising Diffusion Models

Learning to generate by denoising

Denoising diffusion models consist of two processes:

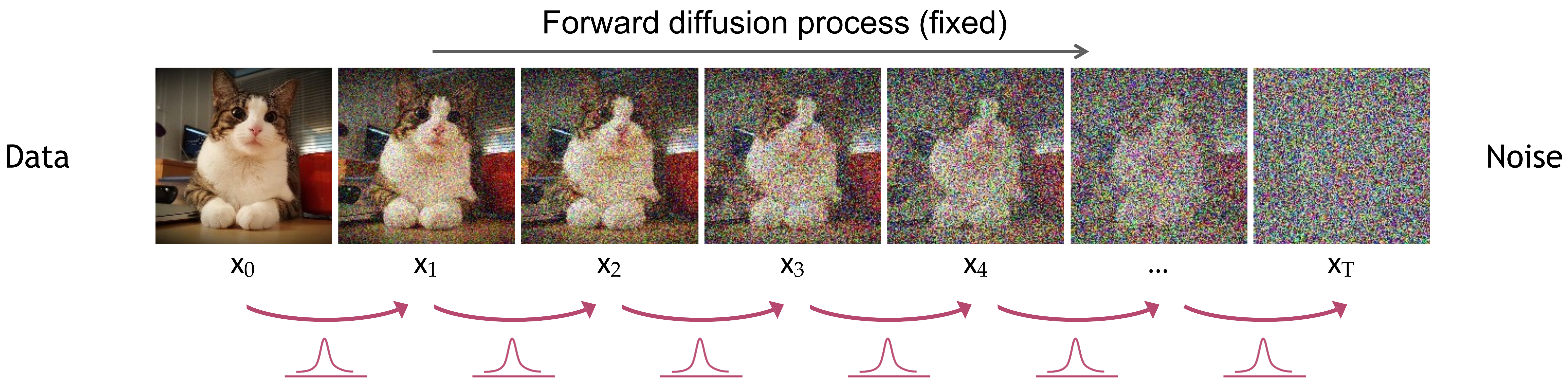
- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising



Sohl-Dickstein et al., Deep Unsupervised Learning using Nonequilibrium Thermodynamics, ICML 2015
Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021

Forward Diffusion Process

The formal definition of the forward process in T steps:

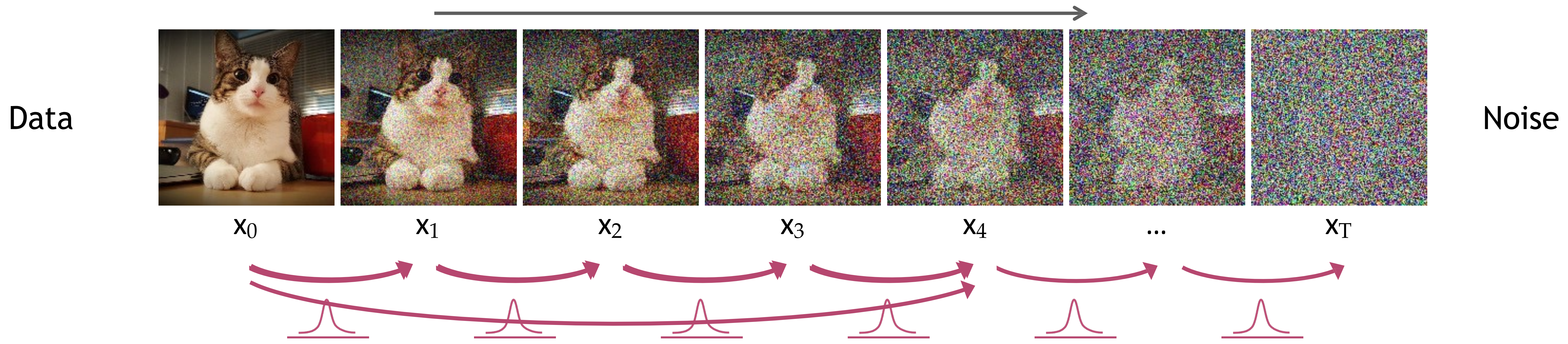


$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \underbrace{\sqrt{1-\beta_t}\mathbf{x}_{t-1}}_{\text{mean}}, \underbrace{\beta_t\mathbf{I}}_{\text{variance}})$$

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

where, $\epsilon_{t-1} \sim \mathcal{N}(0, \mathbf{I})$

Diffusion Kernel



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

➡ Sample: $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1}$

where, $\epsilon_{t-1} \sim \mathcal{N}(0, \mathbf{I})$

You will need to prove this in your assignment

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

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

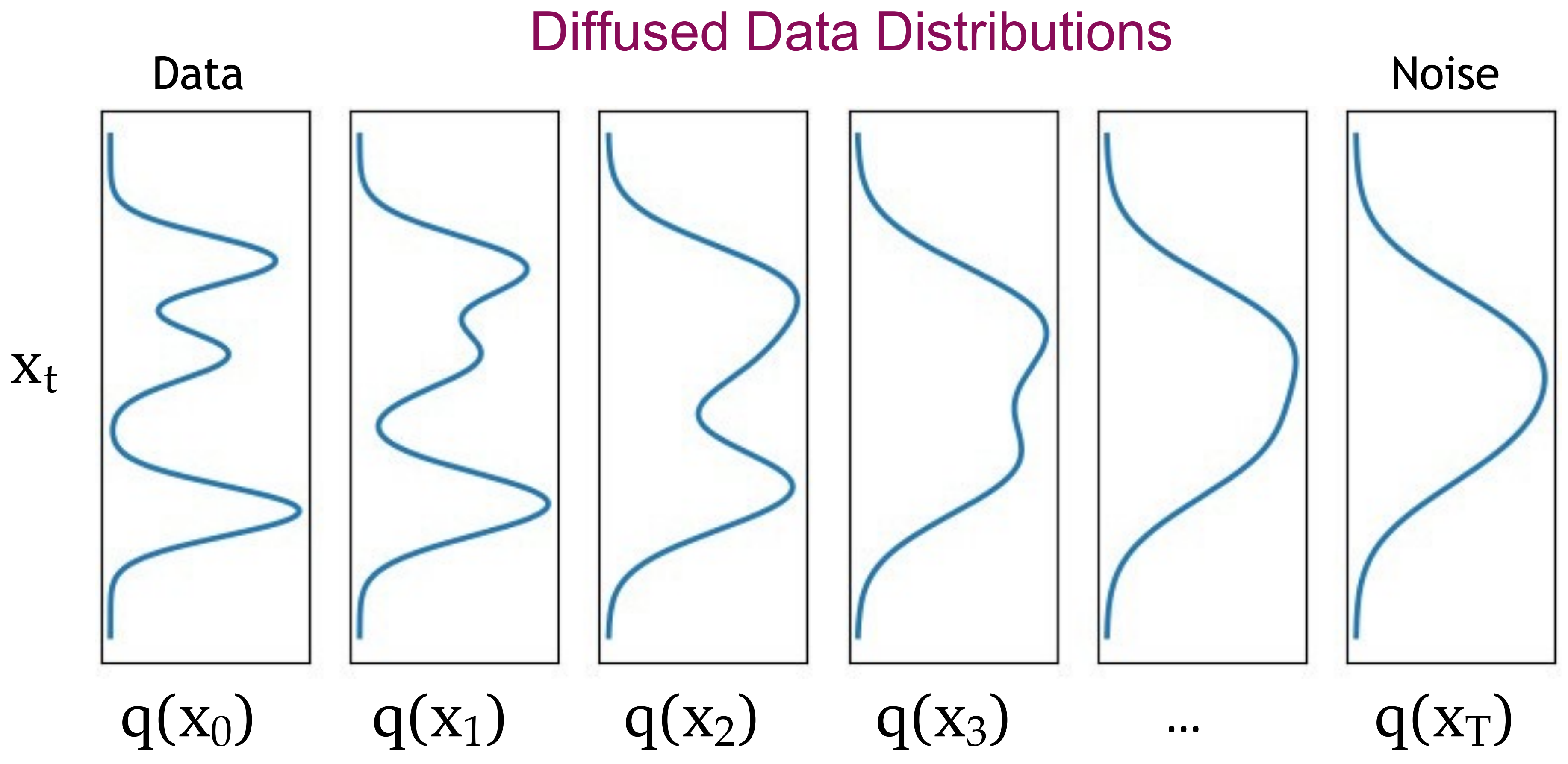
β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \to 0$ and $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

What happens to a distribution in the forward diffusion?

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

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

The diffusion kernel is Gaussian convolution.



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

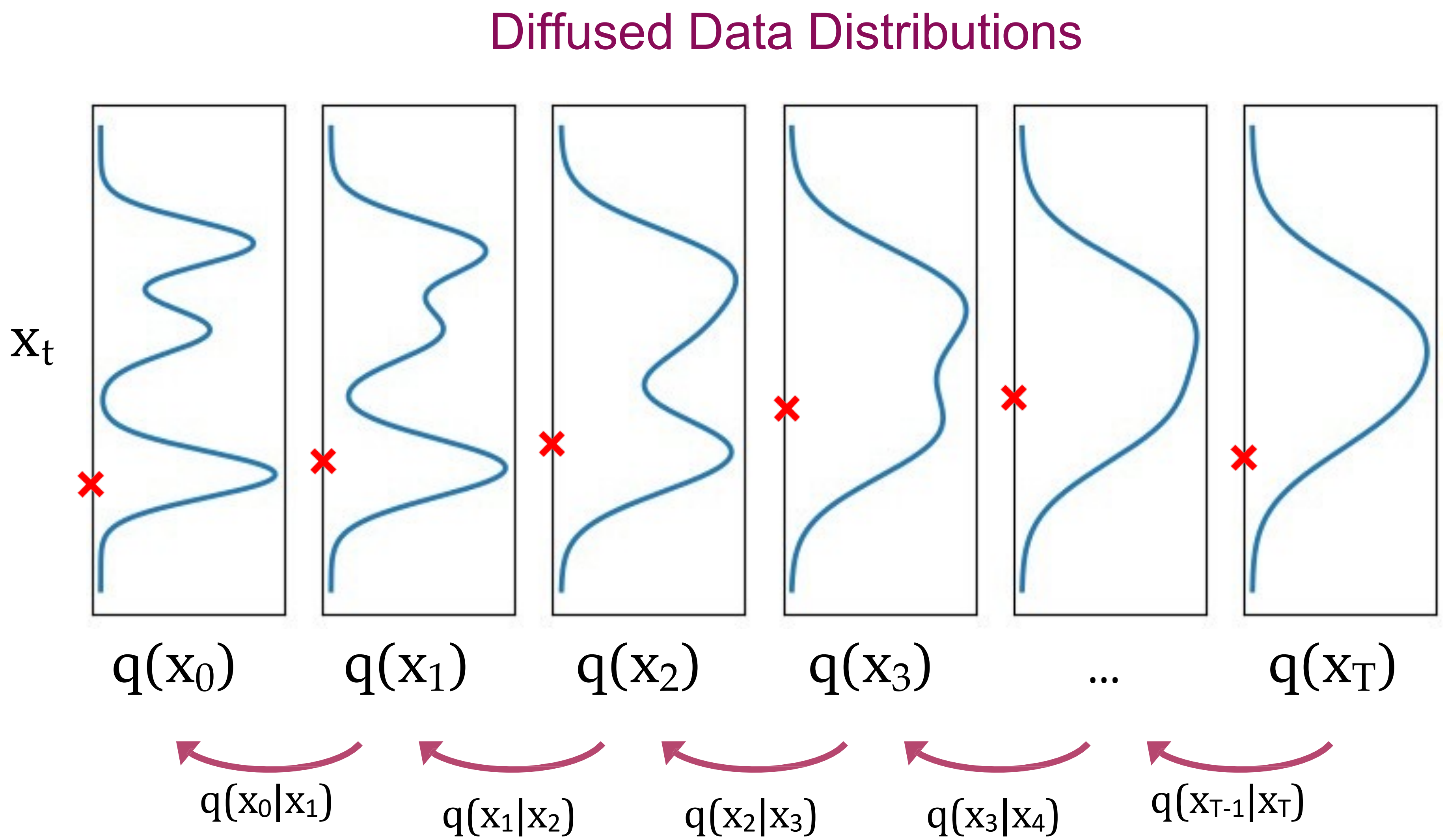
Generative Learning by Denoising

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

Generation:

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

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}_{\text{True Denoising Dist.}}$



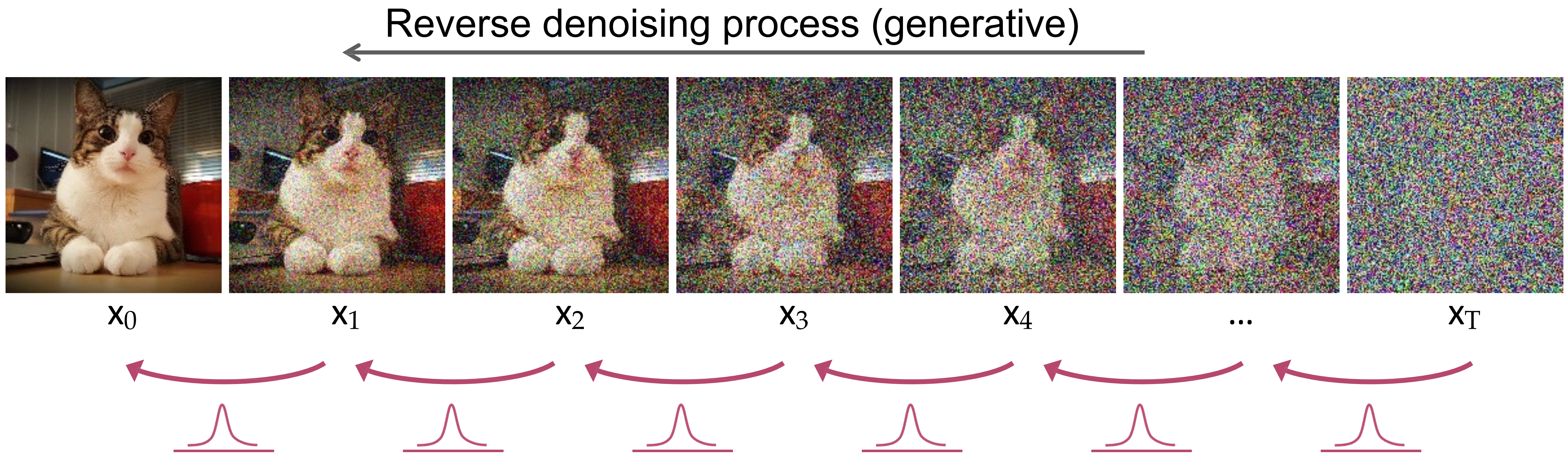
In general, $q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto q(\mathbf{x}_{t-1}) q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is intractable.

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

Reverse Denoising Process

Formal definition of forward and reverse processes in T steps:

Data



Noise

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

Trainable network
(U-net, Denoising Autoencoder)

Autoencoder predicts the mean of the denoised image x(t-1) given x(t).

How do we train? (summary version)

What is the loss function? (Ho et al. NeurIPS 2020)

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

Algorithm 1 Training

1: **repeat**
2: $\quad \mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $\quad t \sim \text{Uniform}(\{1, \ldots, T\})$
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $\quad$ Take gradient descent step on

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

6: **until** converged

U-Net autoencoder takes x(t) as input and simply predict a noise. The goal of the training is to generate a noise pattern that is unit normal. Very similar to VAE, right?

Summary

Training and Sample Generation

Algorithm 1 Training

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

Algorithm 2 Sampling

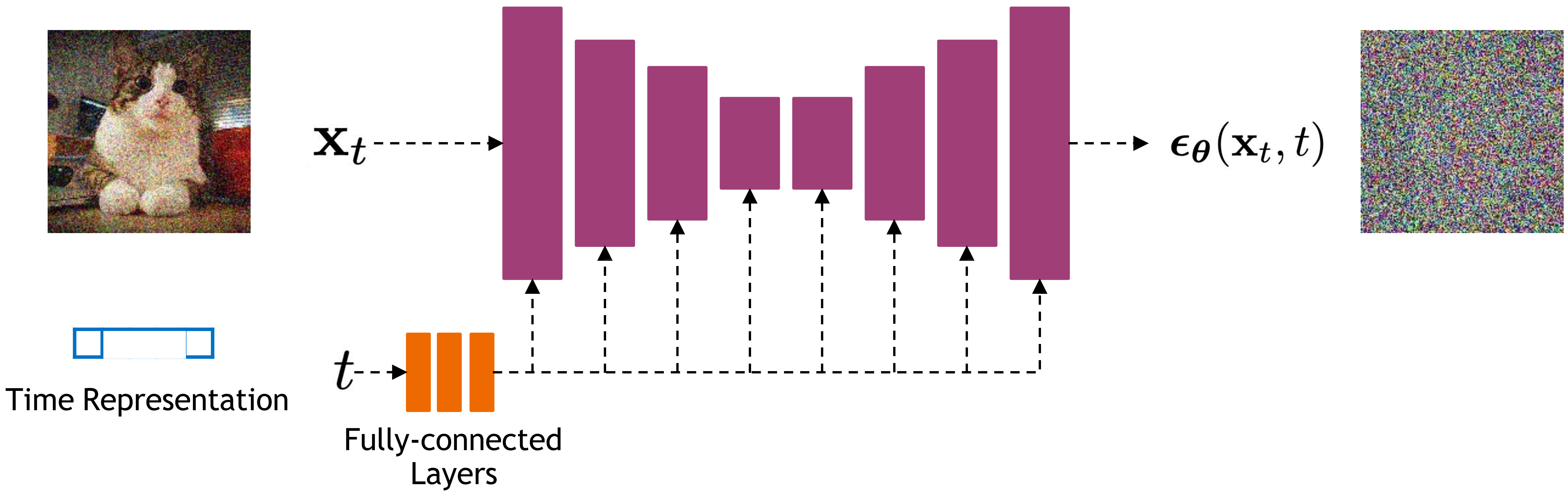
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
4: $\mathbf{x}_{t-1} = \boxed{\frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Intuitively: During forward process we add noise to image. During reverse process we try to predict that noise with a U-Net and then subtract it from the image to denoise it.

Implementation Considerations

Network Architectures

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



Time representation: sinusoidal positional embeddings or random Fourier features.

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

Diffusion Parameters

Noise Schedule

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

Data  Noise

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

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

Often a linear schedule is used for β_t, and σ_t^2 is set equal to β_t. Slowly increase the amount of added noise.

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

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

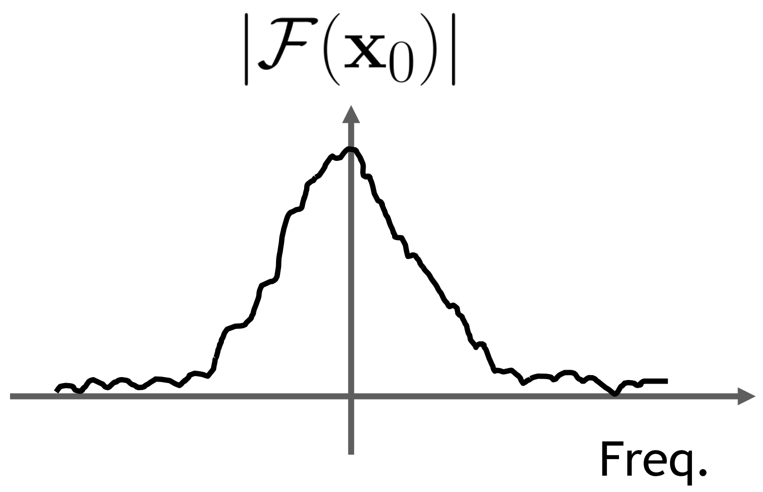
What happens to an image in the forward diffusion process?

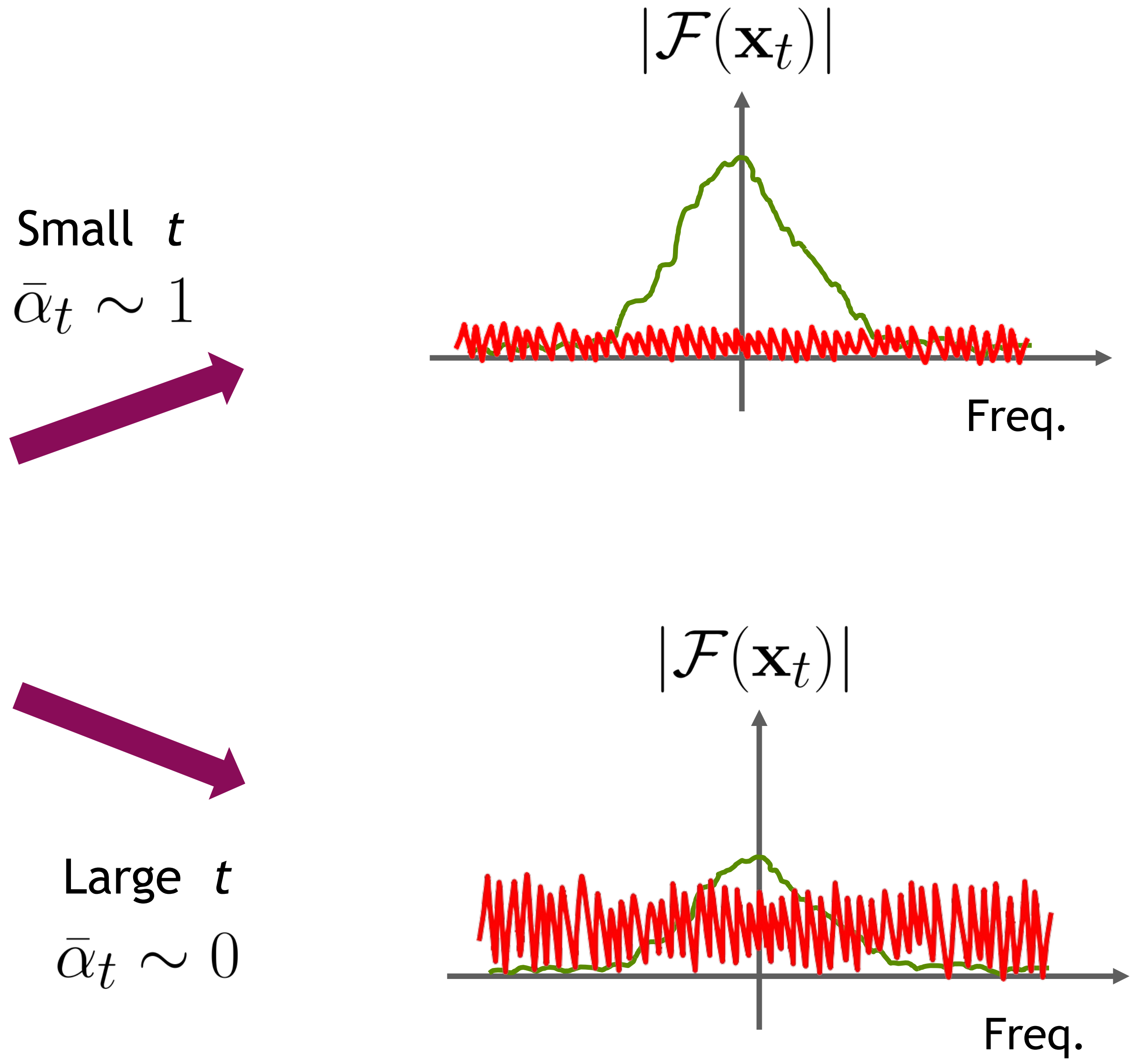
Recall that sampling from $q(\mathbf{x}_t|\mathbf{x}_0)$ is done using $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

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

Fourier Transform

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





In the forward diffusion, the high frequency content is perturbed faster.

Content-Detail Tradeoff

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

Reverse denoising process (generative)

Data

Noise

x_0 x_1 x_2 x_3 x_4 ... x_T

The denoising model is specialized for generating the high-frequency content (i.e., low-level details)

The denoising model is specialized for generating the low-frequency content (i.e., coarse content)

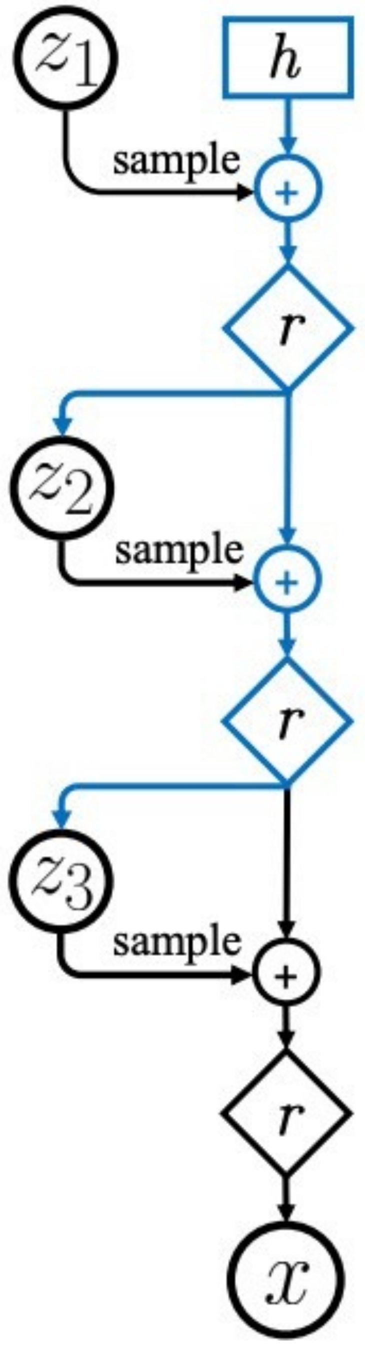
The weighting of the training objective for different timesteps is important!

Connection to VAEs

Diffusion models can be considered as a special form of hierarchical VAEs.

However, in diffusion models:

- The encoder is fixed
- The latent variables have the same dimension as the data
- The denoising model is shared across different timestep
- The model is trained with some reweighting of the variational bound.



[Vahdat and Kautz, NVAE: A Deep Hierarchical Variational Autoencoder, NeurIPS 2020]
[Sønderby, et al.. Ladder variational autoencoders, NeurIPS 2016.]

Summary

Denoising Diffusion Probabilistic Models

- Diffusion process can be reversed if the variance of the gaussian noise added at each step of the diffusion is small enough.
- To reverse the process we train a U-Net that takes input: current noisy image and timestamp, and predicts the noise map..
- Training goal is to make sure that the predicted noise map at each step is unit gaussian (Note that in VAE we also required the latent space to be unit gaussian).
- During sampling/generation, subtract the predicted noise from the noisy image at time t to generate the image at time t-1 (with some weighting).

The devil is in the details:

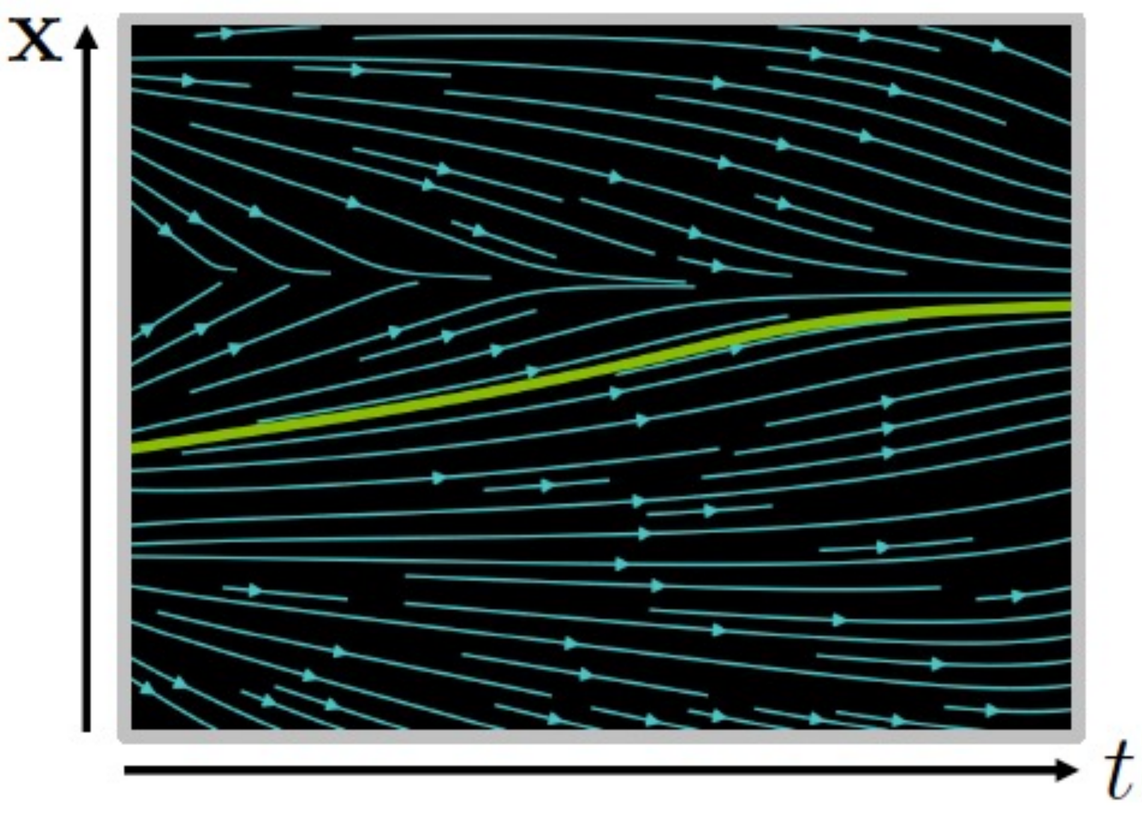
- Network architectures
- Objective weighting
- Diffusion parameters (i.e., noise schedule)

"Elucidating the Design Space of Diffusion-Based Generative Models" by Karras et al. for important design decisions.
To be presented in the class!

Crash Course in Differential Equations

Ordinary Differential Equation (ODE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) \quad \text{or} \quad \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t$$



Analytical Solution:

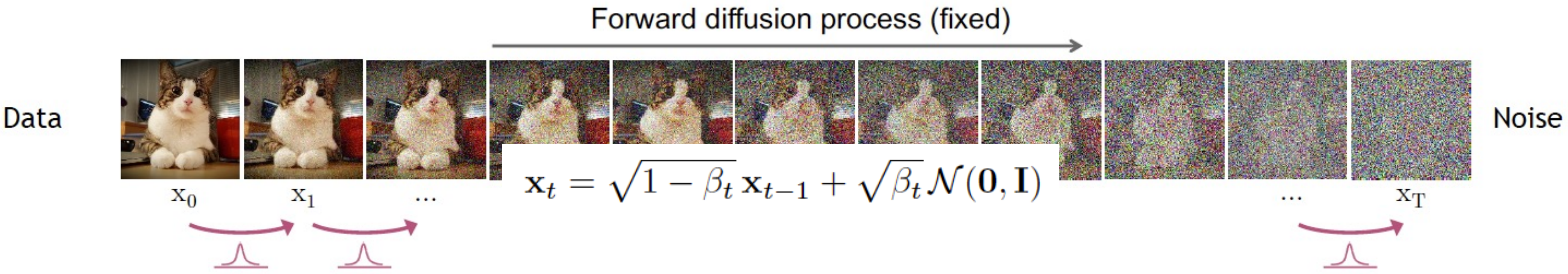
$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{f}(\mathbf{x}, \tau)\mathrm{d}\tau$$

Iterative Numerical Solution:

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

Forward Diffusion Process as Stochastic Differential Equation

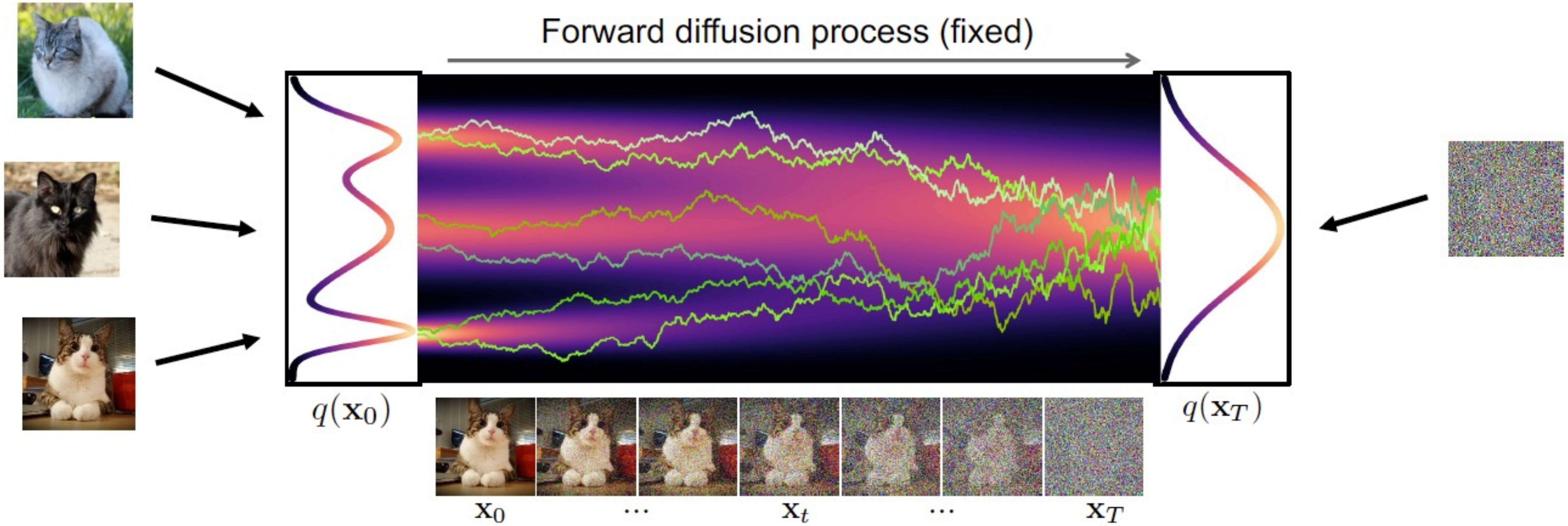
Consider the limit of **many small steps:** $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$



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

Song et al., “Score-Based Generative Modeling through Stochastic Differential Equations”, *ICLR*, 2021

Forward Diffusion Process as Stochastic Differential Equation



Forward Diffusion SDE:

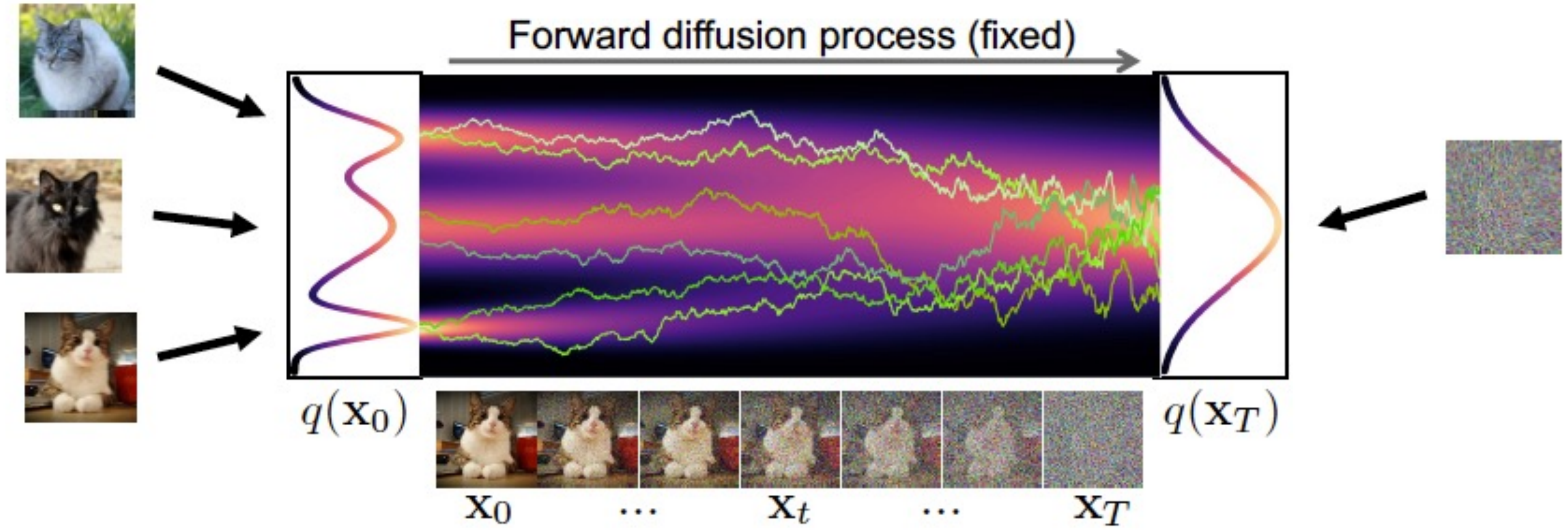
$$\mathrm{d}\mathbf{x}_t = \underbrace{-\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t}_{\substack{\text{drift term} \\ \text{(pulls towards mode)}}} + \underbrace{\sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t}_{\substack{\text{diffusion term} \\ \text{(injects noise)}}}$$

Song et al., *ICLR*, 2021

Special case of more general SDEs used in generative diffusion models:

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

The Generative Reverse Stochastic Differential Equation



Forward Diffusion SDE:

How do we obtain the ”Score Function”?

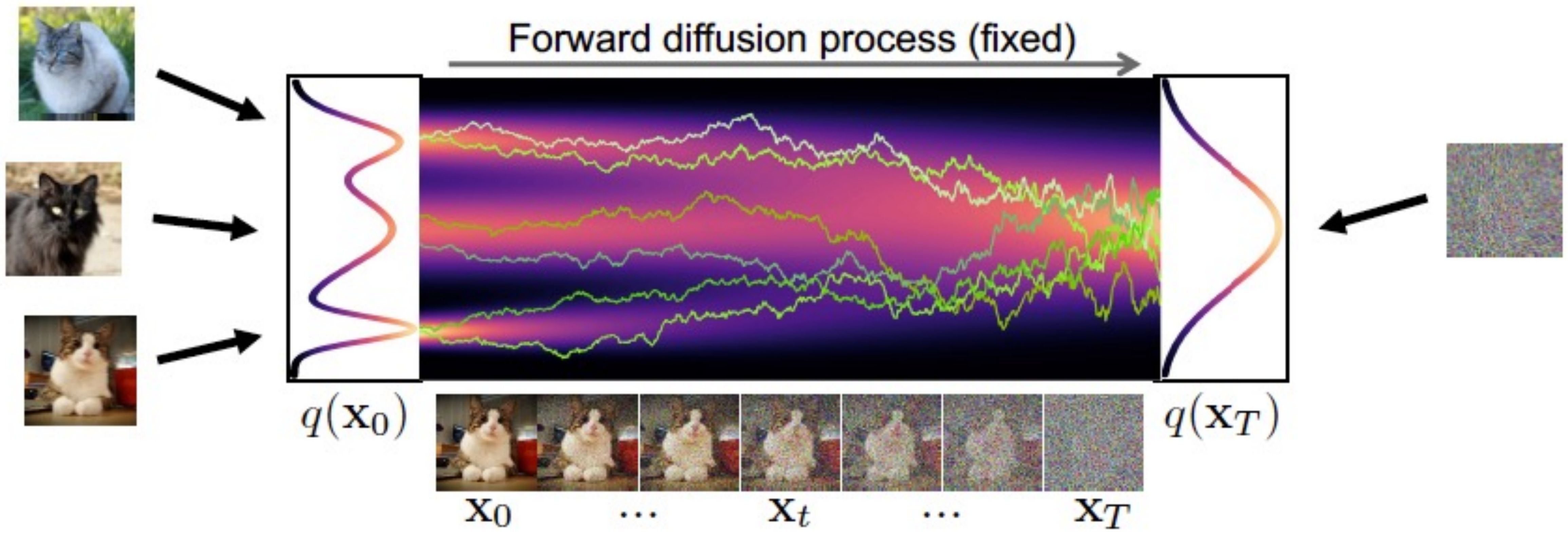
Reverse Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \overbrace{\left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)}_{\text{“Score Function”}}\right]}^{\text{drift term}}\mathrm{d}t + \overbrace{\sqrt{\beta(t)}\,\mathrm{d}\bar{\omega}_t}^{\text{diffusion term}}$$

➡ **Simulate reverse diffusion process: Data generation from random noise!**

Song et al., *ICLR*, 2021
Anderson, in *Stochastic Processes and their Applications*, 1982

Score Matching

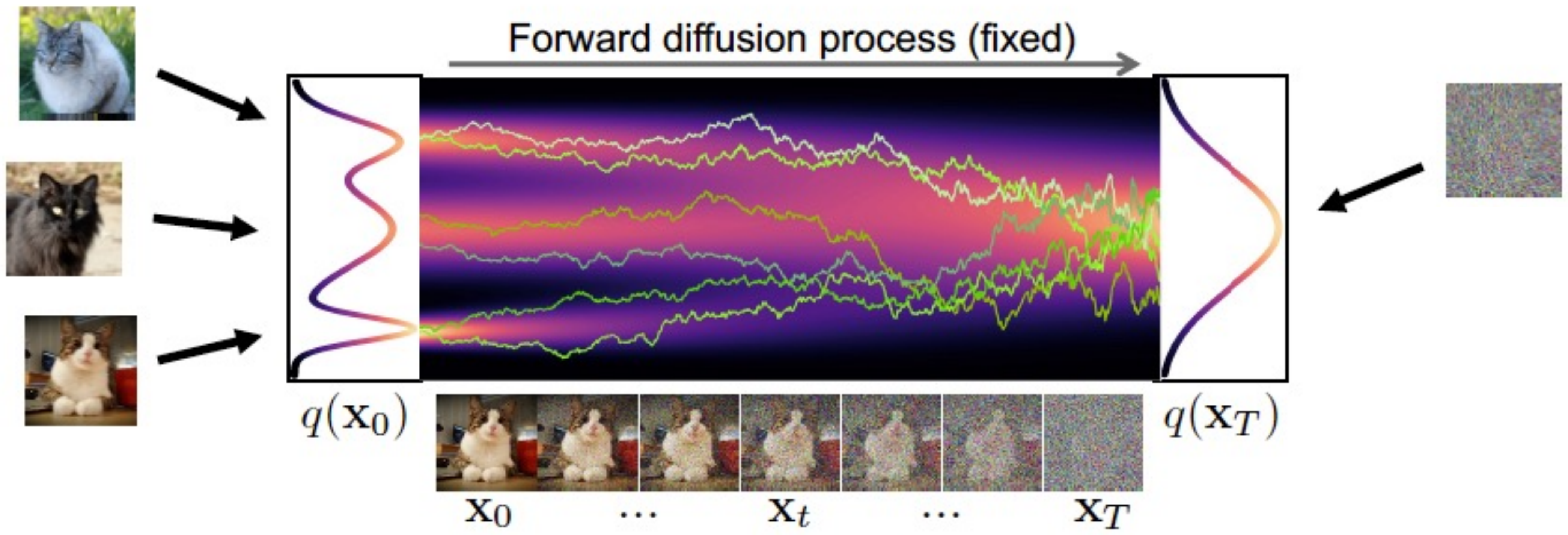


- Naïve idea, learn model for the score function by direct regression?

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\substack{\text{diffusion} \\ \text{time } t}} \underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t)}}_{\substack{\text{diffused} \\ \text{data } \mathbf{x}_t}} || \underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\substack{\text{neural} \\ \text{network}}} - \underbrace{\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)}_{\substack{\text{score of} \\ \text{diffused data} \\ \text{(marginal)}}} ||_2^2$$

➡ **But $\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$ (score of the *marginal diffused density* $q_t(\mathbf{x}_t)$) is not tractable!**

Denoising Score Matching



- Instead, diffuse individual data points $\mathbf{x}_0$. Diffused $q_t(\mathbf{x}_t|\mathbf{x}_0)$ ***is*** tractable!
- **Denoising Score Matching:**

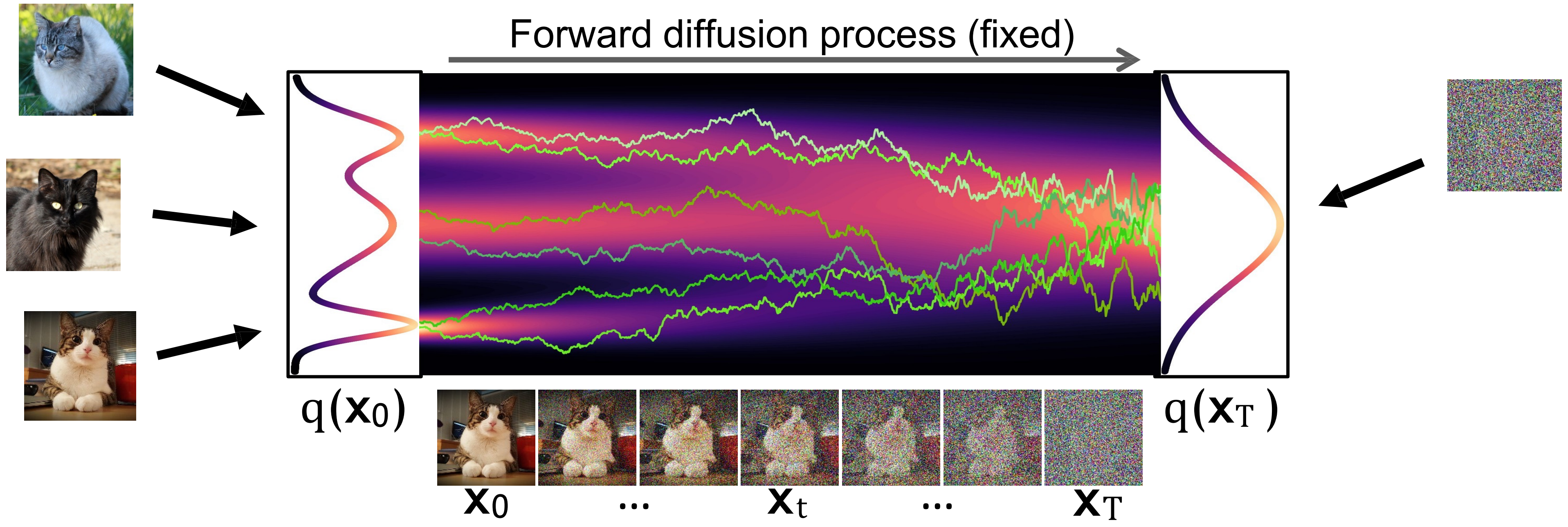
$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0} \underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t} ||\underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\text{neural network}} - \underbrace{\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|\mathbf{x}_0)}_{\text{score of diffused data sample}}||_2^2$$

→ **After expectations,** $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$**!**

Vincent, in *Neural Computation*, 2011
Song and Ermon, *NeurIPS*, 2019
Song et al. *ICLR*, 2021

Denoising Score Matching

Implementation Details



$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0} \underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t} ||\underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t,t)}_{\text{neural network}} - \underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t|\mathbf{x}_0)}_{\text{score of diffused data sample}}||_2^2$$

$$\Rightarrow \min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim\mathcal{U}(0,T)}\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\frac{1}{\sigma_t^2}||\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t,t)||_2^2$$

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim\mathcal{U}(0,T)}\mathbb{E}_{\mathbf{x}_0\sim q_0(\mathbf{x}_0)}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\frac{\lambda(t)}{\sigma_t^2}||\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t,t)||_2^2$$

More sophisticated model parametrizations and loss weightings are possible!

Karras et al., "Elucidating the Design Space of Diffusion-Based Generative Models", arXiv, 2022

To be discussed in detail in paper presentation

Different loss weightings trade off between model with good perceptual quality vs. high log-likelihood

- *Perceptual quality:* $\lambda(t) = \sigma_t^2$
- *Maximum log-likelihood:* $\lambda(t) = \beta(t)$ *(negative ELBO)*

Advanced Techniques

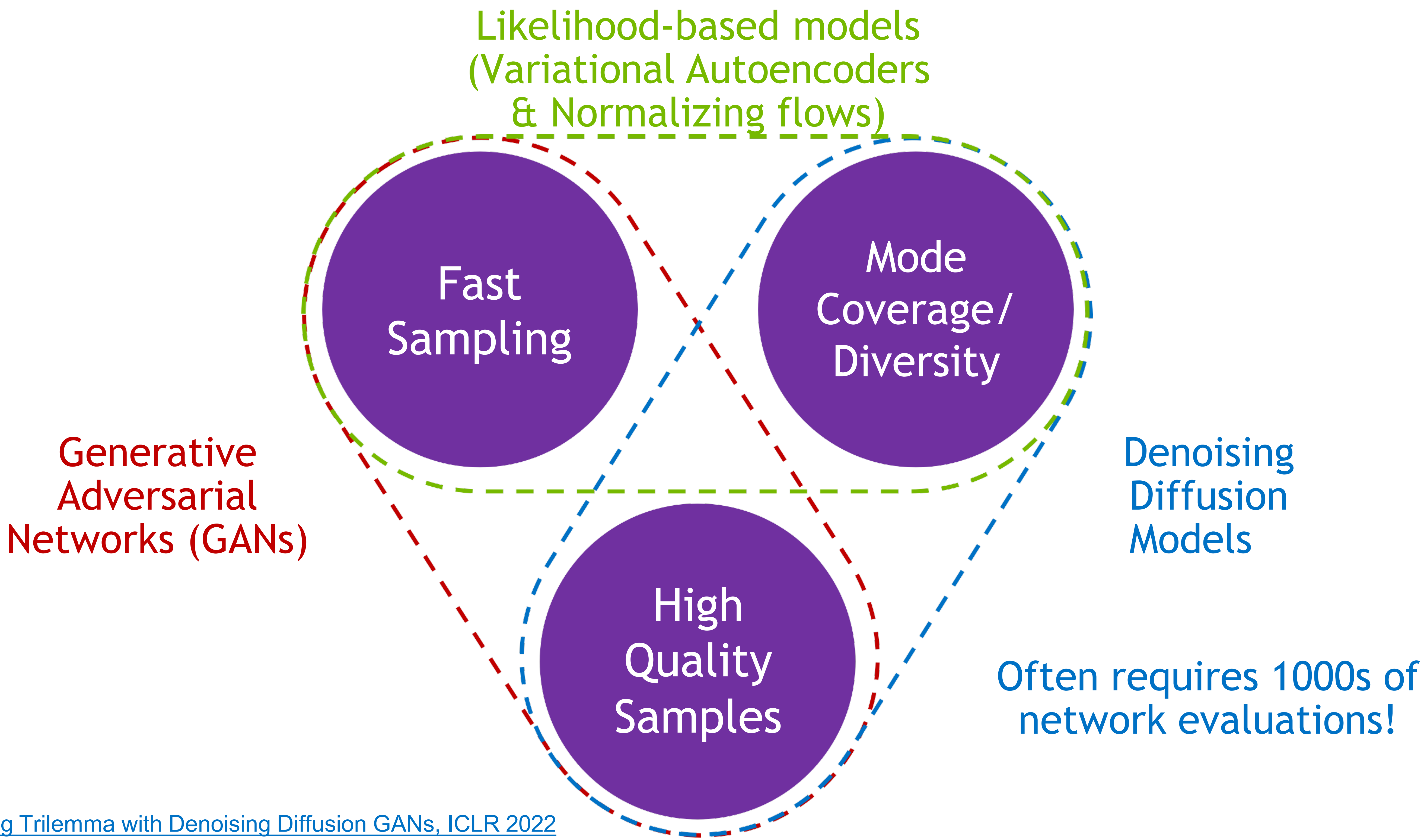
Questions to address with advanced techniques

- Q1: How to accelerate the sampling process?
  - Advanced forward diffusion process
  - Advanced reverse process
  - Hybrid models & model distillation
- Q2: How to do high-resolution (conditional) generation?
  - Conditional diffusion models
  - Classifier(-free) guidance
  - Cascaded generation

Q: How to accelerate sampling process?

What makes a good generative model?

The generative learning trilemma

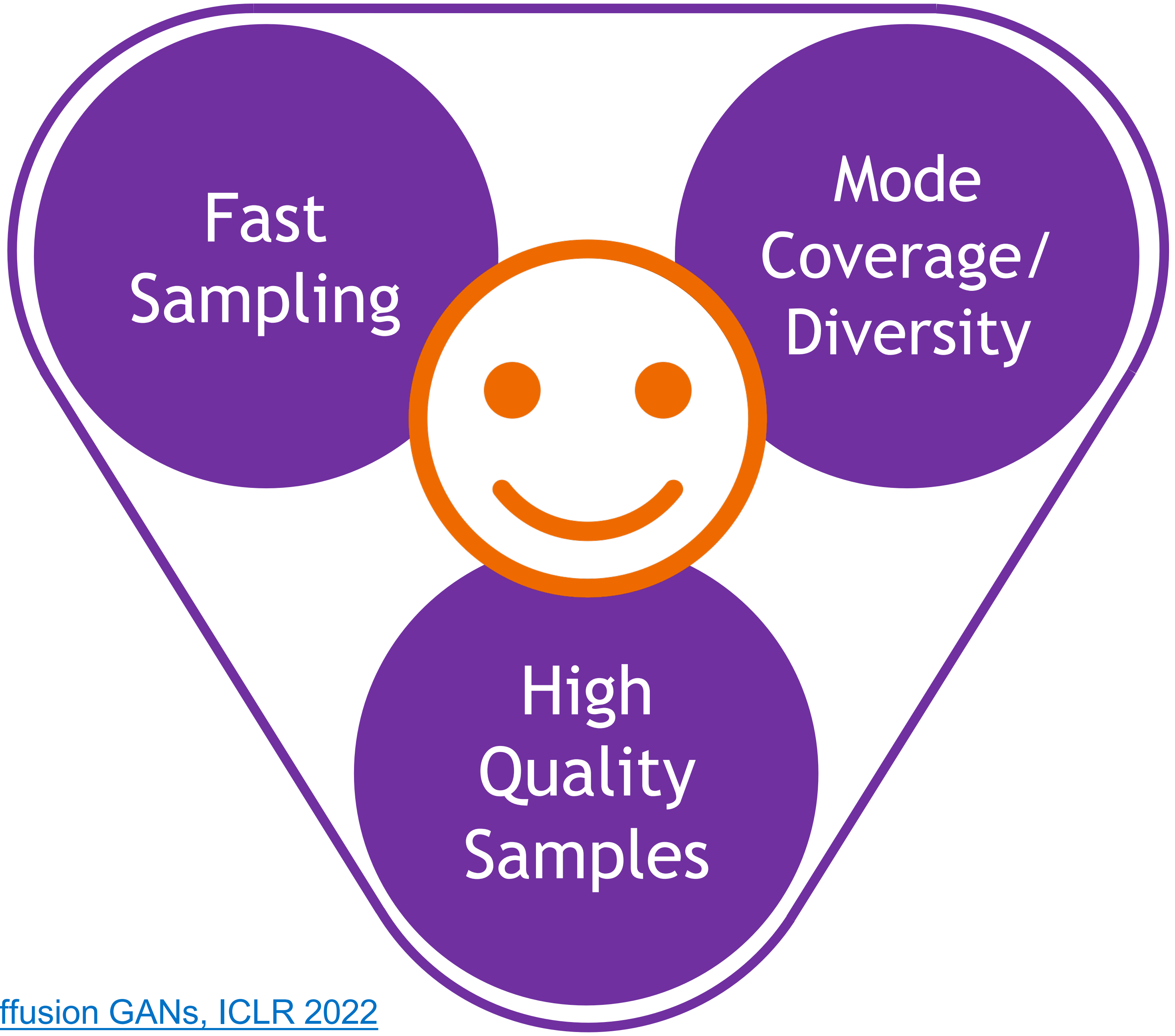


[Tackling the Generative Learning Trilemma with Denoising Diffusion GANs, ICLR 2022]

What makes a good generative model?

The generative learning trilemma

Tackle the trilemma by accelerating diffusion models



Tackling the Generative Learning Trilemma with Denoising Diffusion GANs, ICLR 2022

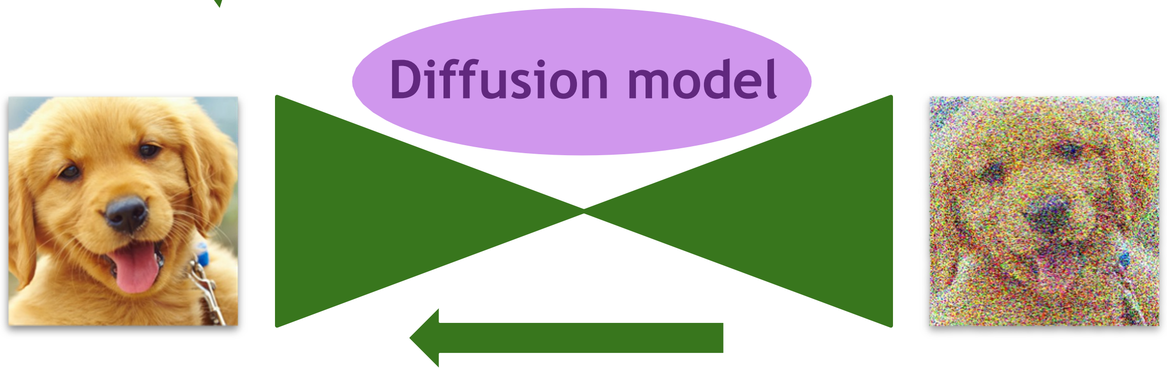
How to accelerate diffusion models?

[Image credit: Ben Poole, Mohammad Norouzi]

Simple **forward process** slowly maps data to noise

Reverse process maps noise back to data where **diffusion model** is trained

- Naïve acceleration methods, such as reducing diffusion time steps in training or sampling every k time step in inference, lead to immediate worse performance.

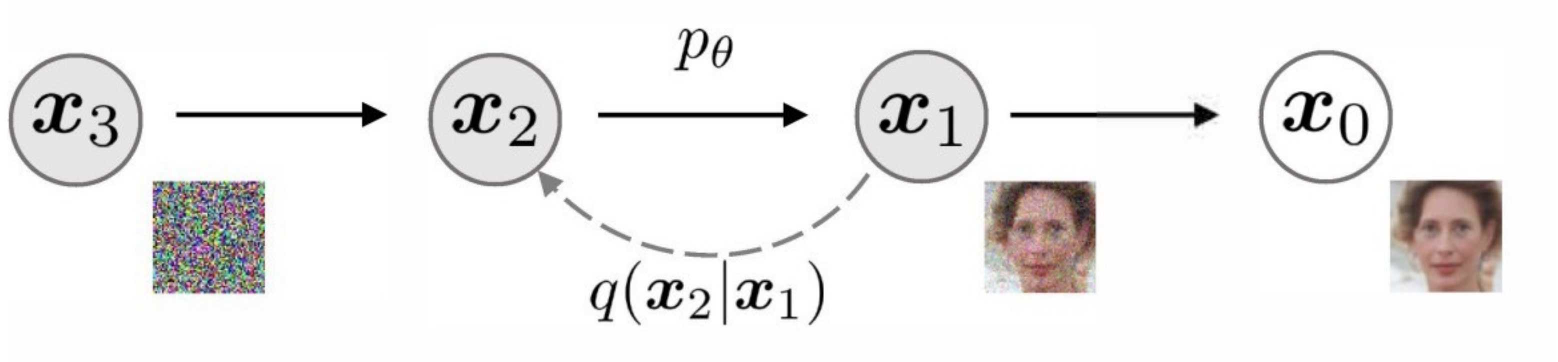


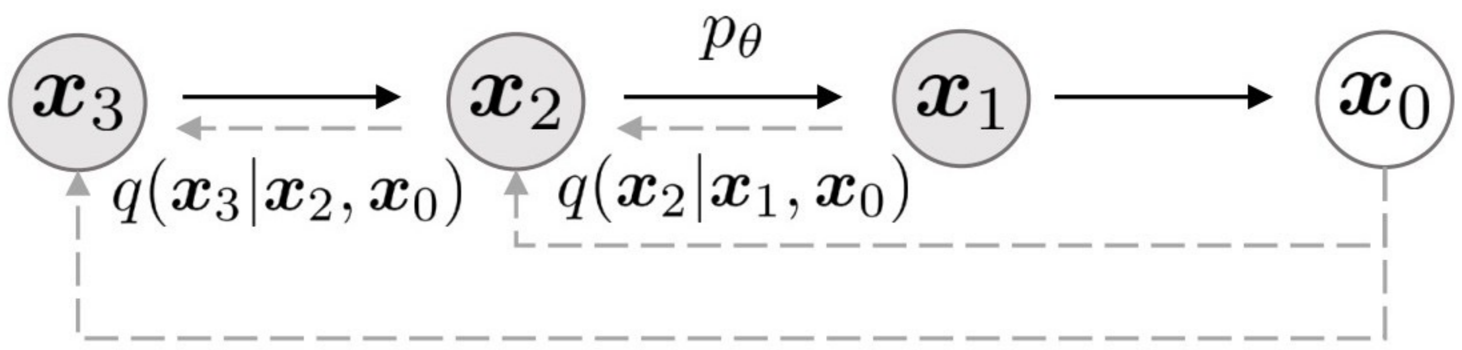
- We need something cleverer.

- Given a limited number of functional calls, usually much less than 1000s, how to improve performance?

Denoising diffusion implicit models (DDIM)

Non-Markovian diffusion process





Main Idea

Design a family of non-Markovian diffusion processes and corresponding reverse processes.

The process is designed such that the model can be optimized by the same surrogate objective as the original diffusion model.

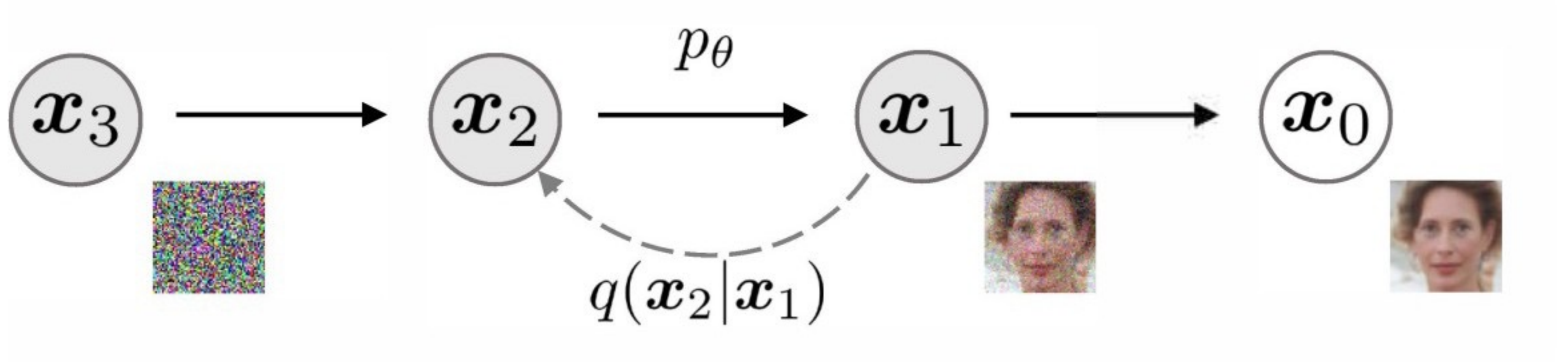
$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

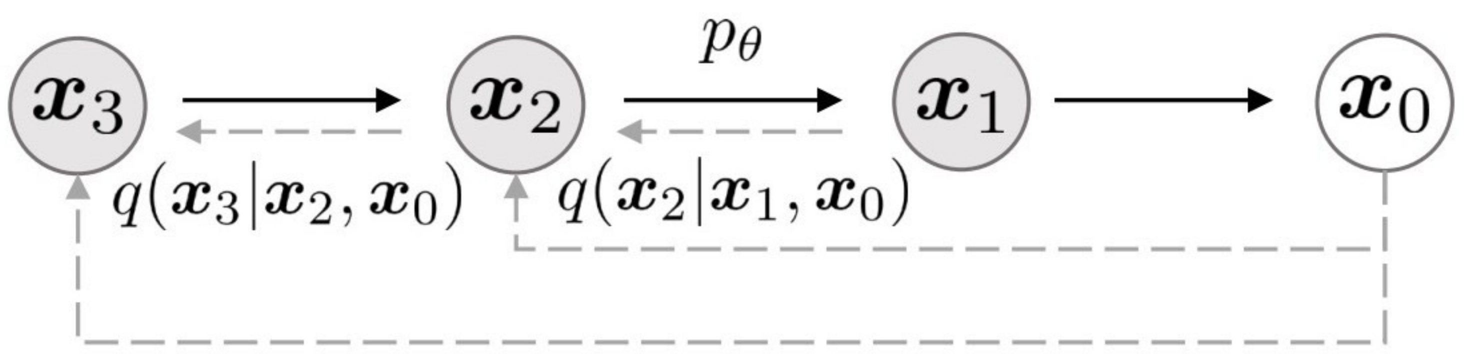
Therefore, can take a pretrained diffusion model but with more choices of sampling procedure.

Song et al., "Denoising Diffusion Implicit Models", ICLR 2021.

Denoising diffusion implicit models (DDIM)

Non-Markovian diffusion process





Define a family of forward processes that meets the above requirement:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \tilde{\sigma}_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1 - \bar{\alpha}_t}}, \tilde{\sigma}_t^2\mathbf{I}\right)$$

The corresponding reverse process is

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \tilde{\sigma}_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0}{\sqrt{1 - \bar{\alpha}_t}}, \tilde{\sigma}_t^2\mathbf{I}\right)$$

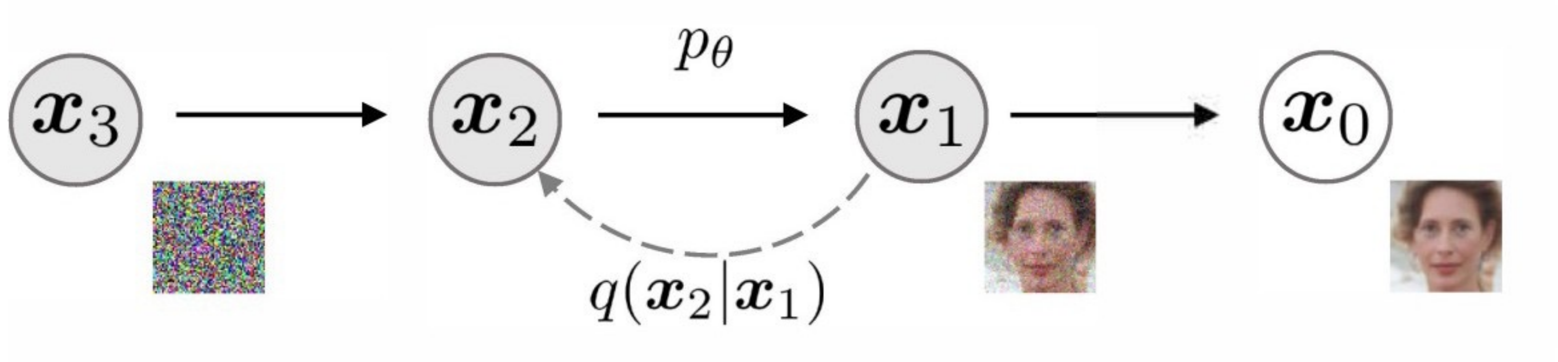
$$:= (\boldsymbol{x}_t - \sqrt{1 - \alpha_t} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t))/\sqrt{\alpha_t}.$$

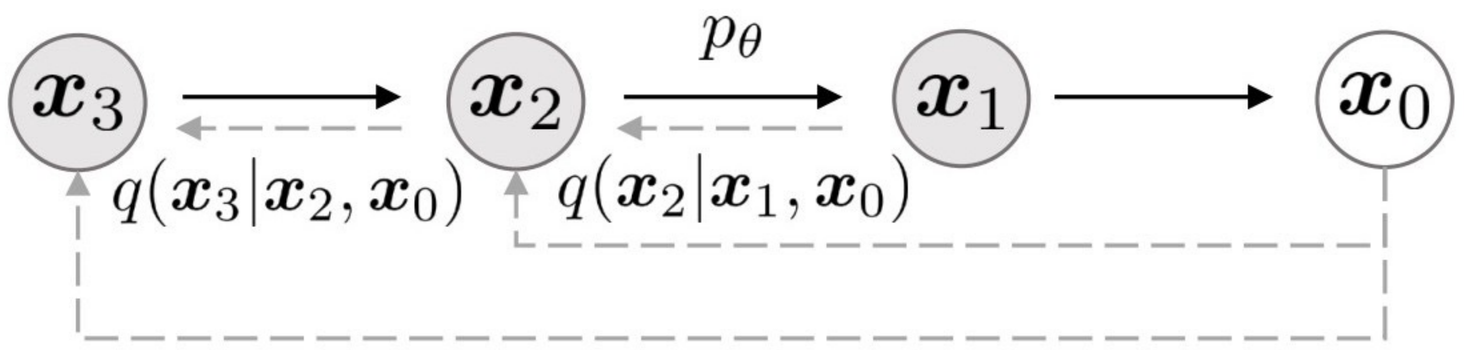
Intuitively, given noisy x_t we first predict the corresponding clean image x_0 and then use if to obtain a sample x_{t-1}

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

Denoising diffusion implicit models (DDIM)

Non-Markovian diffusion process





The corresponding reverse process is

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0 + \sqrt{1-\bar{\alpha}_{t-1}-\tilde{\sigma}_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0}{\sqrt{1-\bar{\alpha}_t}}, \tilde{\sigma}_t^2\mathbf{I}\right)$$

Intuitively, given noisy x_t we first predict the corresponding clean image x_0 and then use if to obtain a sample x_{t-1}

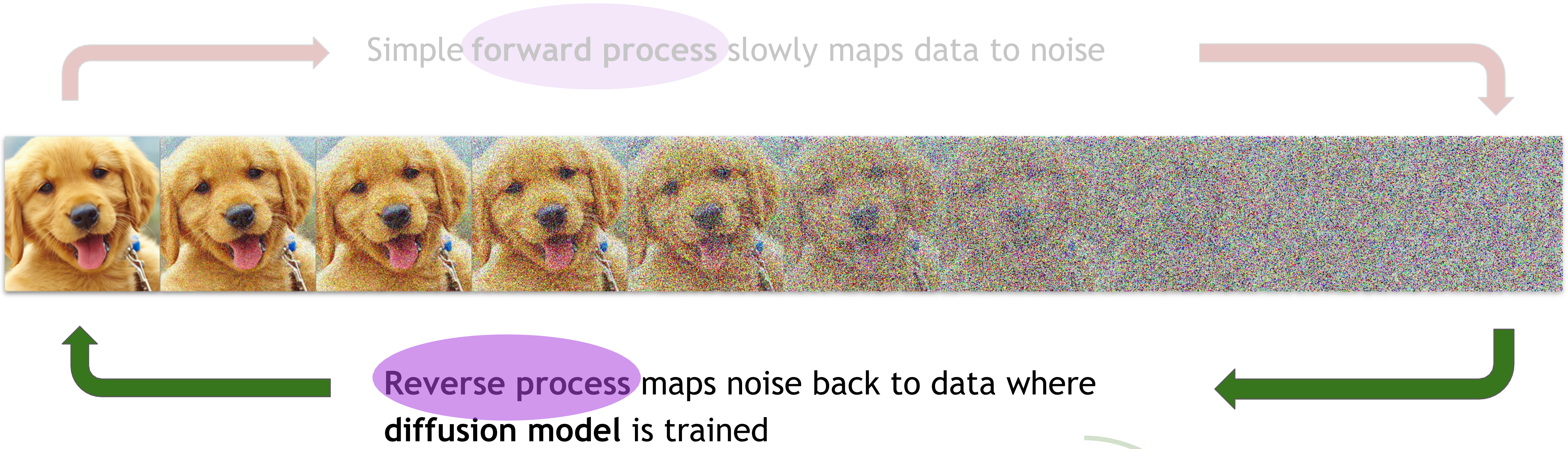
$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}}\right)}_{\text{“ predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\sigma_t\epsilon_t}_{\text{random noise}}$$

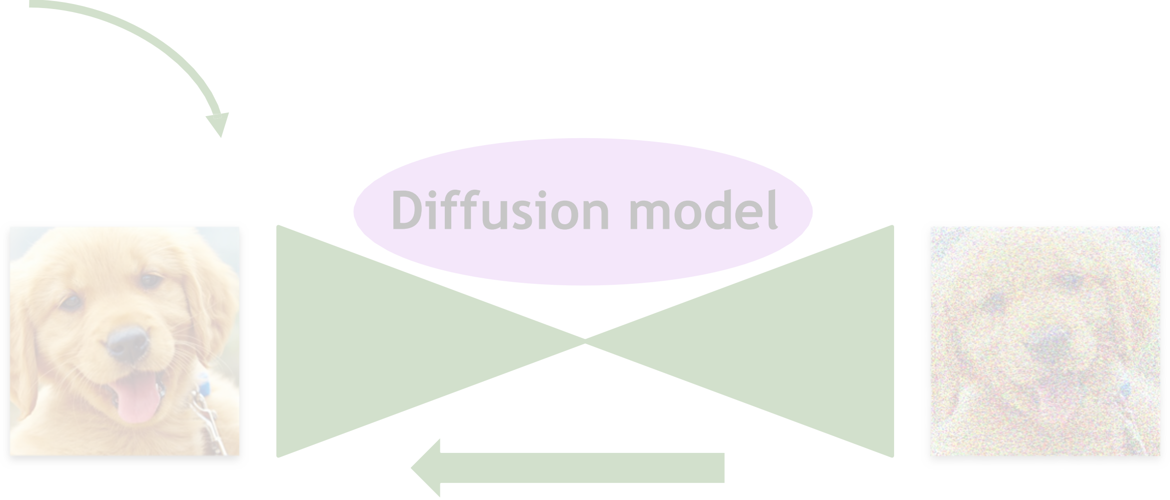
- Different choice of σ results in different generative process without re-training the model

- When $\sigma = 0$ for all t, we have a deterministic generative process, with randomness from only t=T (the last step).

Advanced reverse process

Approximate reverse process with more complicated distributions

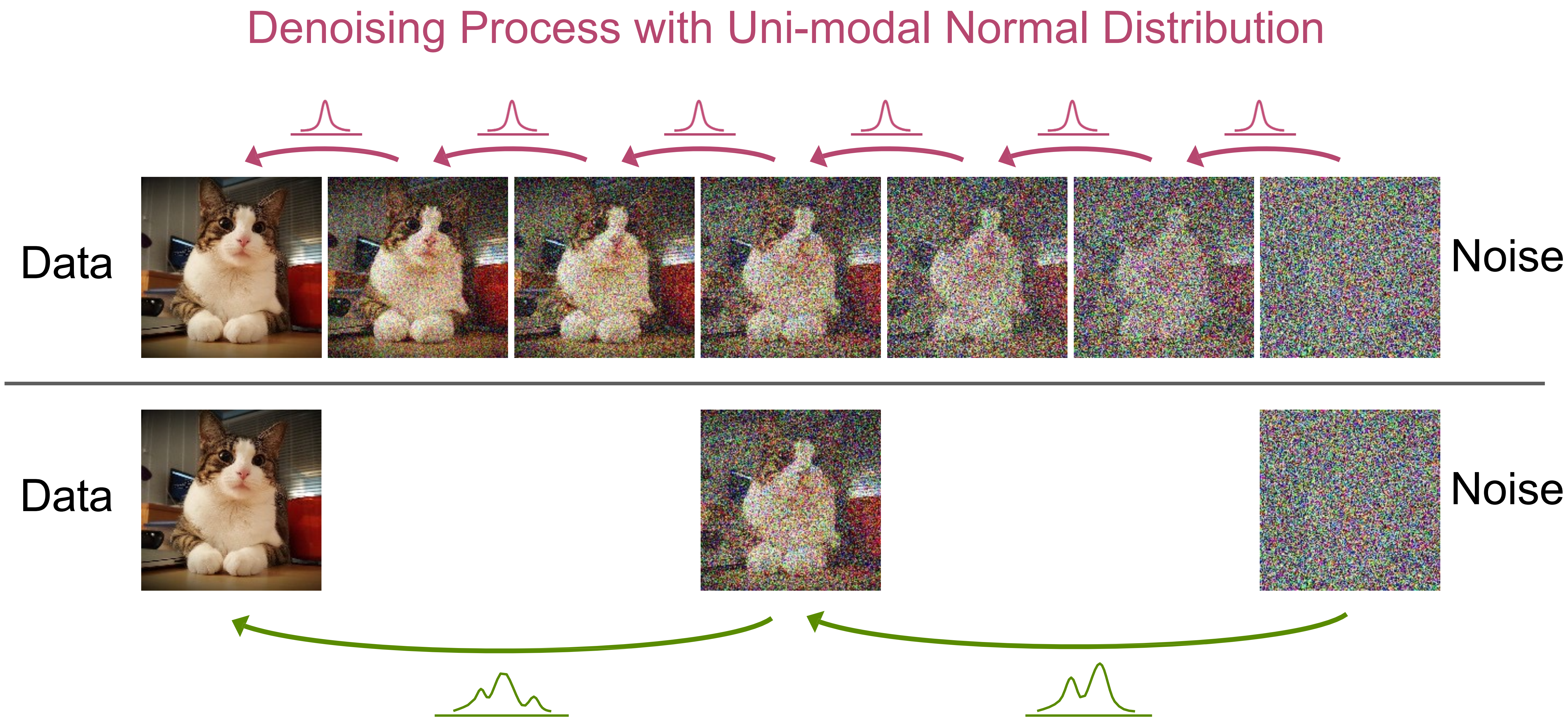




- Q: is normal approximation of the reverse process still accurate when there’re less diffusion time steps?

Advanced approximation of reverse process

Normal assumption in denoising distribution holds only for small step



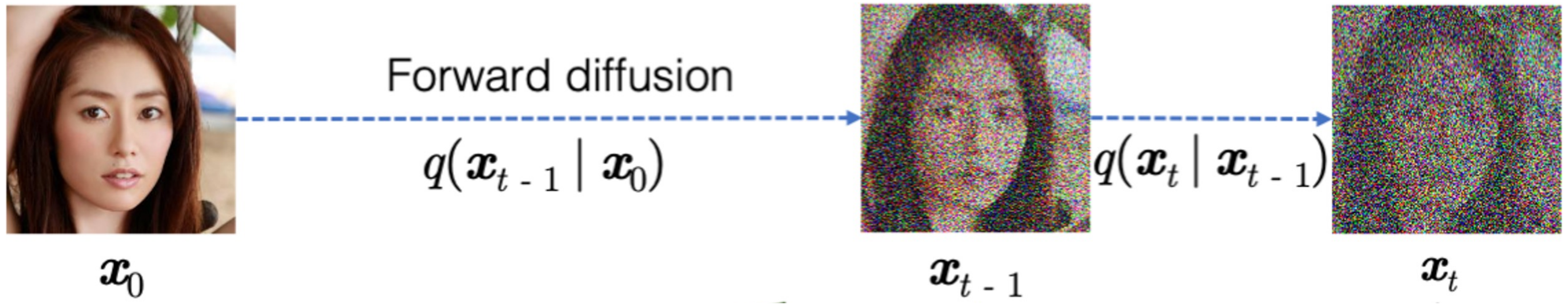
Requires more complicated functional approximators!

Xiao et al., "Tackling the Generative Learning Trilemma with Denoising Diffusion GANs", ICLR 2022.
Gao et al., "Learning energy-based models by diffusion recovery likelihood", ICLR 2021.

Denoising diffusion GANs

Approximating reverse process by conditional GANs

$$\min_{\theta} \sum_{t \geq 1} \mathbb{E}_{q(\mathbf{x}_t)} \left[D_{\mathrm{adv}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) \right]$$



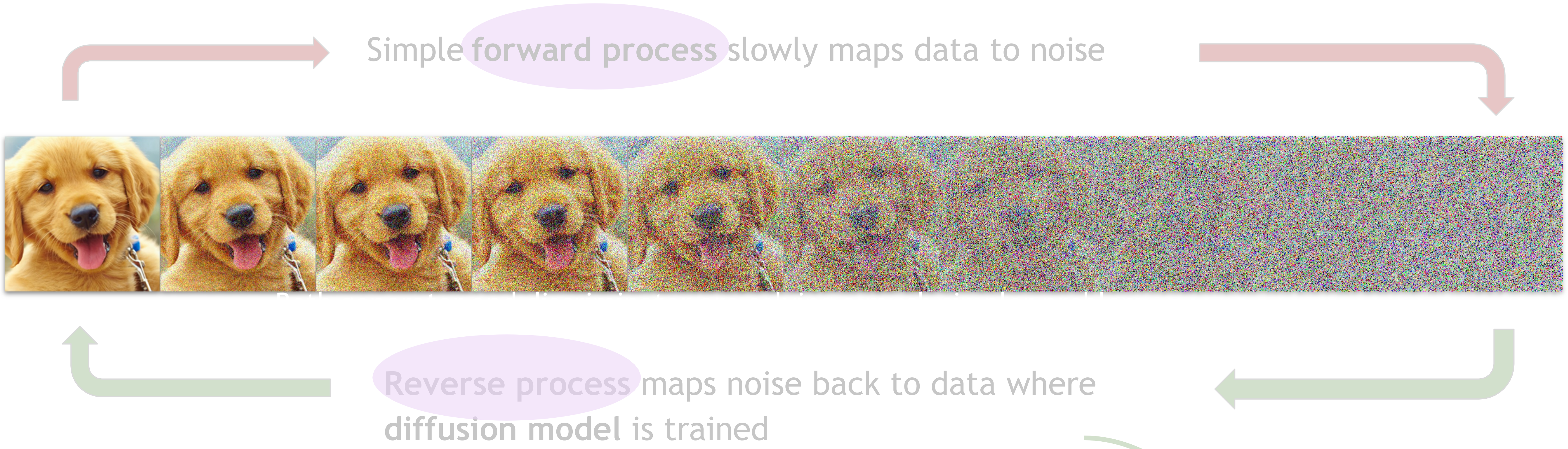
Compared to a one-shot GAN generator:

- Both generator and discriminator are solving a much simpler problem.
- Stronger mode coverage
- Better training stability

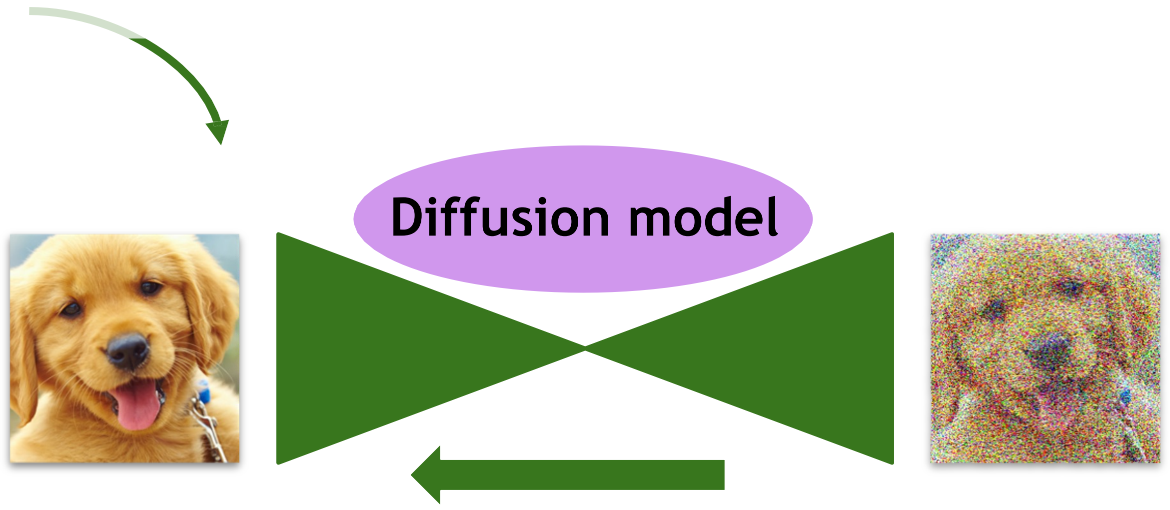
Xiao et al., "Tackling the Generative Learning Trilemma with Denoising Diffusion GANs", ICLR 2022.

Advanced modeling

Latent space modeling & model distillation

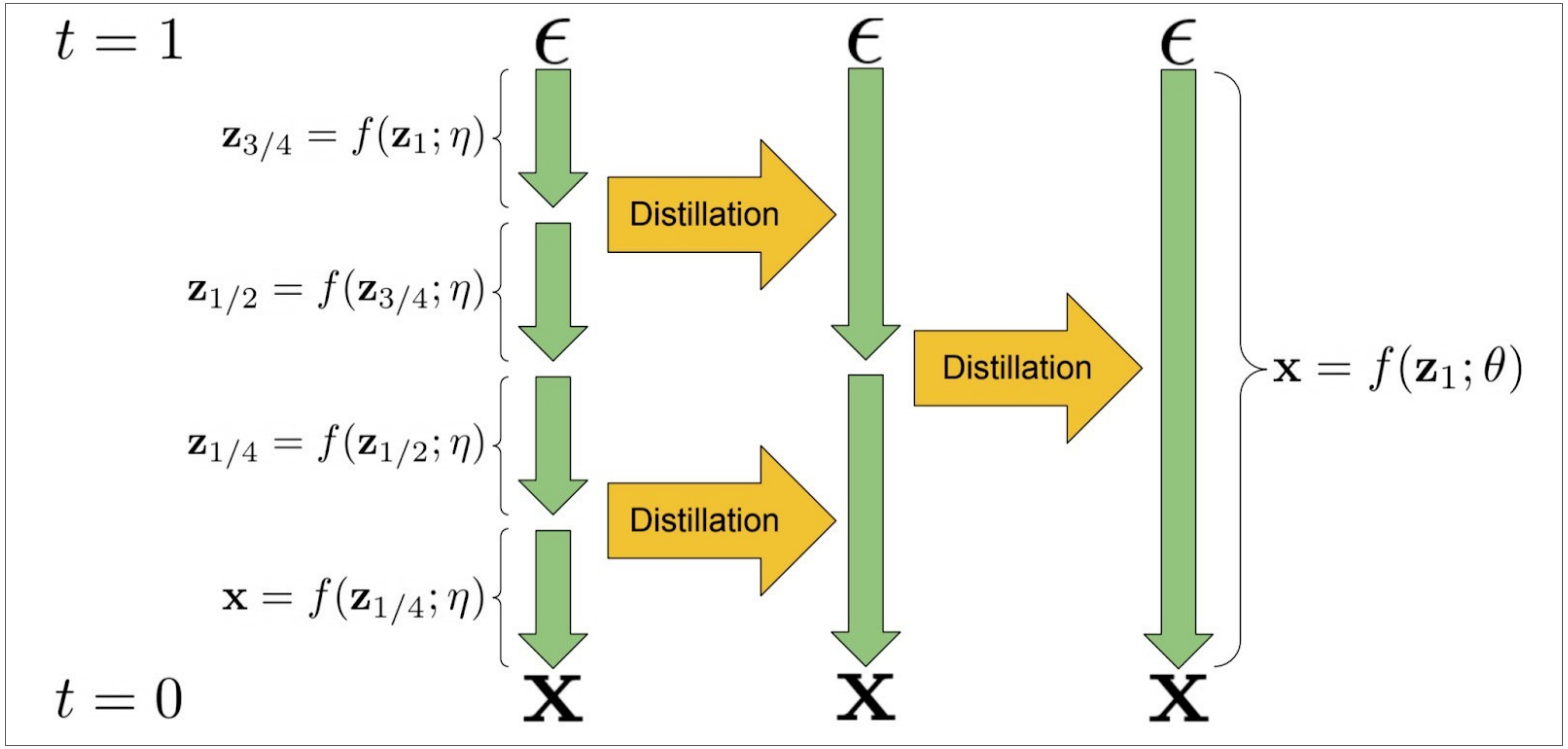


- Can we do model distillation for fast sampling?
- Can we lift the diffusion model to a latent space that is faster to diffuse?



Progressive distillation

- Distill a deterministic DDIM sampler to the same model architecture.
- At each stage, a “student” model is learned to distill two adjacent sampling steps of the “teacher” model to one sampling step.
- At next stage, the “student” model from previous stage will serve as the new “teacher” model.

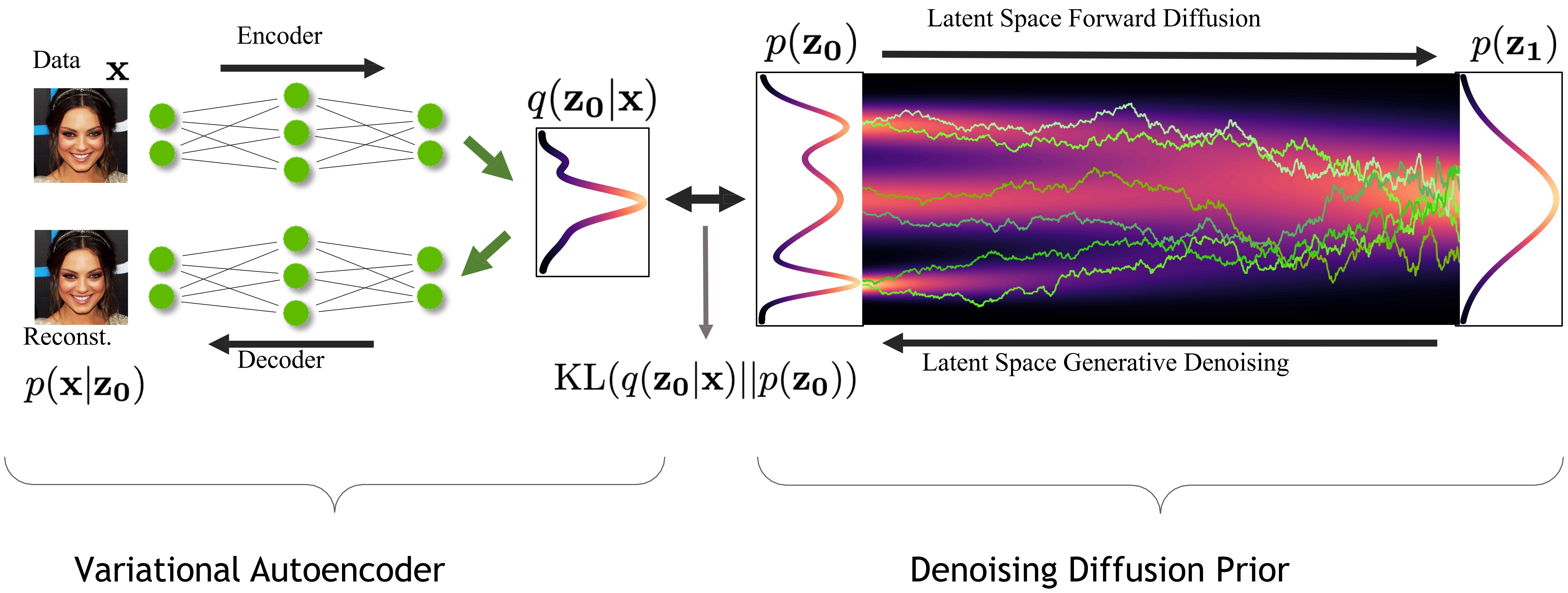


Distillation stage →

Salimans & Ho, “Progressive distillation for fast sampling of diffusion models”, ICLR 2022.

Latent-space diffusion models

Variational autoencoder + score-based prior



Main Idea

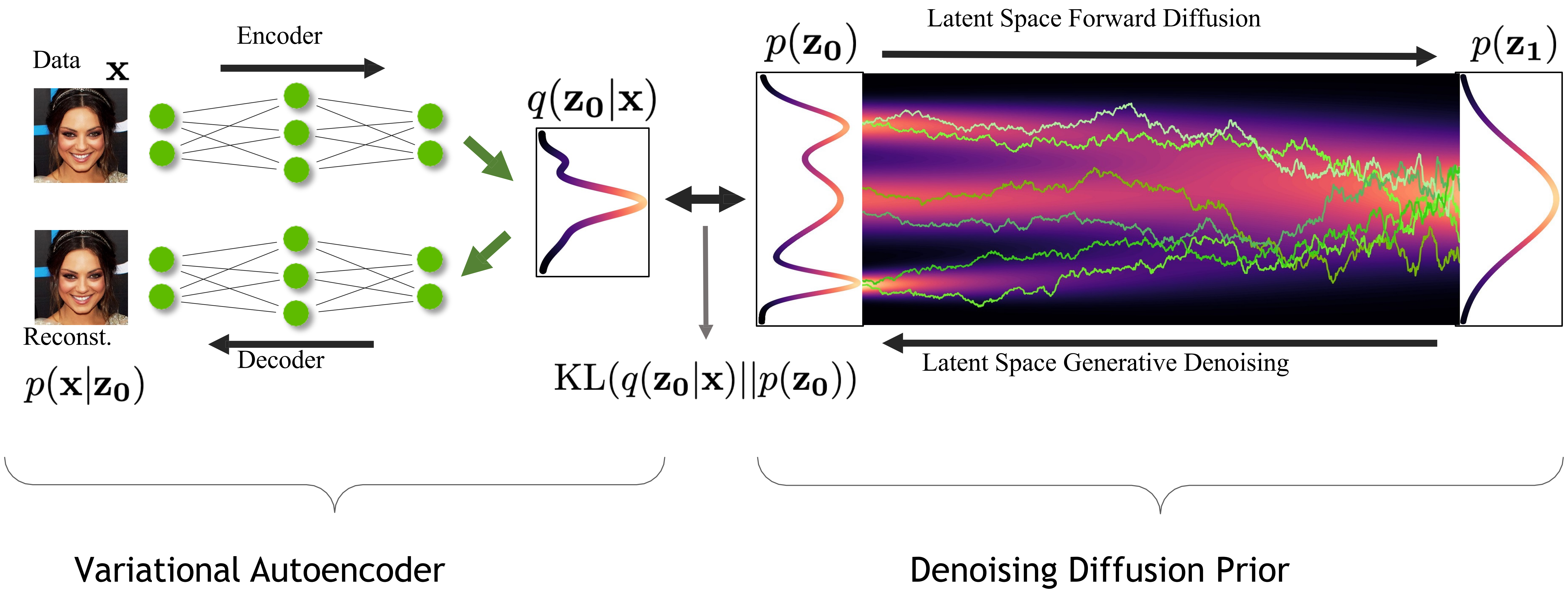
Encoder maps the input data to an embedding space

Denoising diffusion models are applied in the latent space

Vahdat et al., "Score-based generative modeling in latent space", NeurIPS 2021.
Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models", CVPR 2022.

Latent-space diffusion models

Variational autoencoder + score-based prior



Advantages:

(1) The distribution of latent embeddings close to Normal distribution ➔ *Simpler denoising, Faster Synthesis!*

(2) Augmented latent space ➔ *More expressivity!*

(3) Tailored Autoencoders ➔ *More expressivity, Application to any data type (graphs, text, 3D data, etc.) !*

Q: How to do high-resolution conditional generation?

Impressive conditional diffusion models

Text-to-image generation

DALL·E 2

“a propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese”

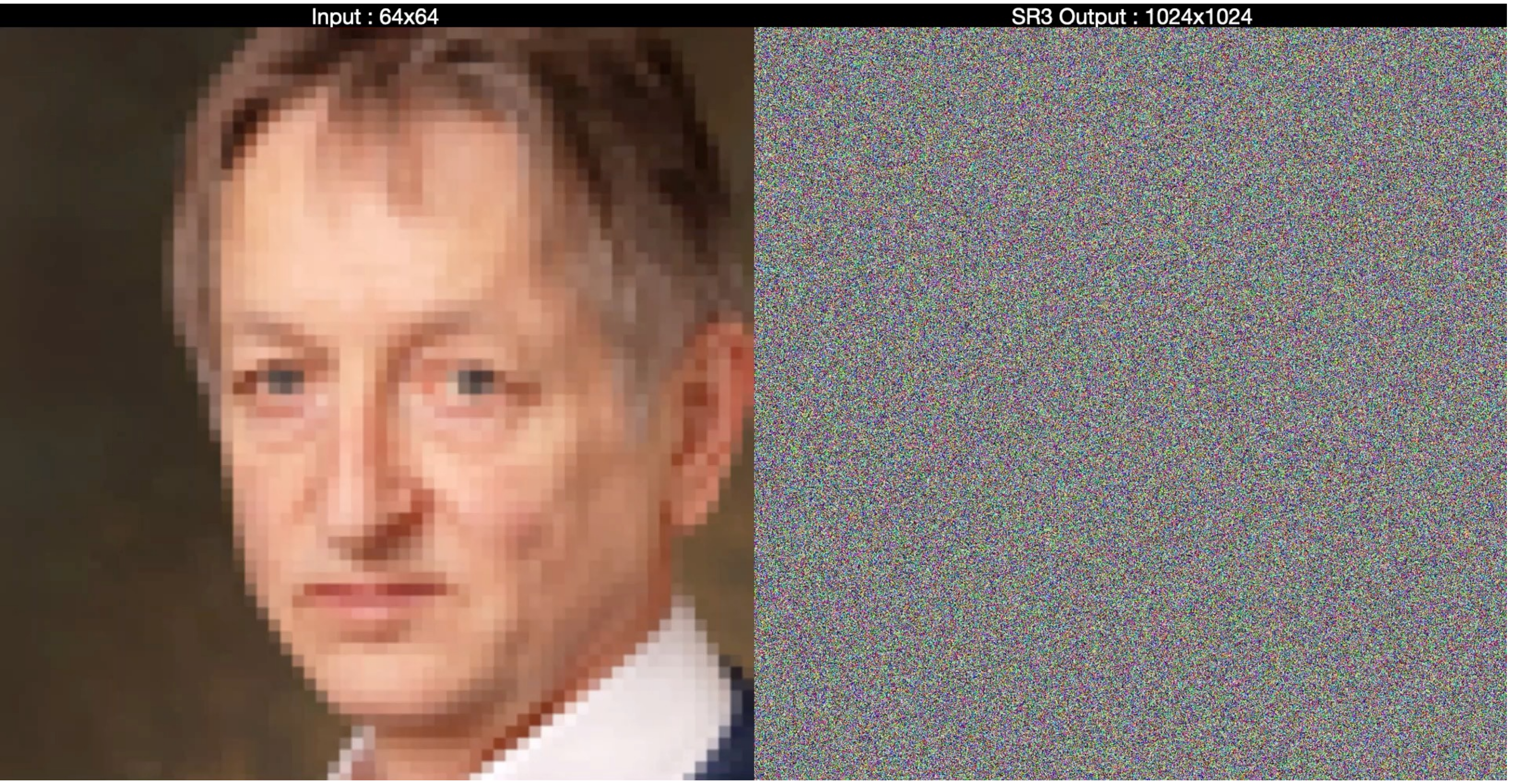
IMAGEN

“A photo of a raccoon wearing an astronaut helmet, looking out of the window at night.”

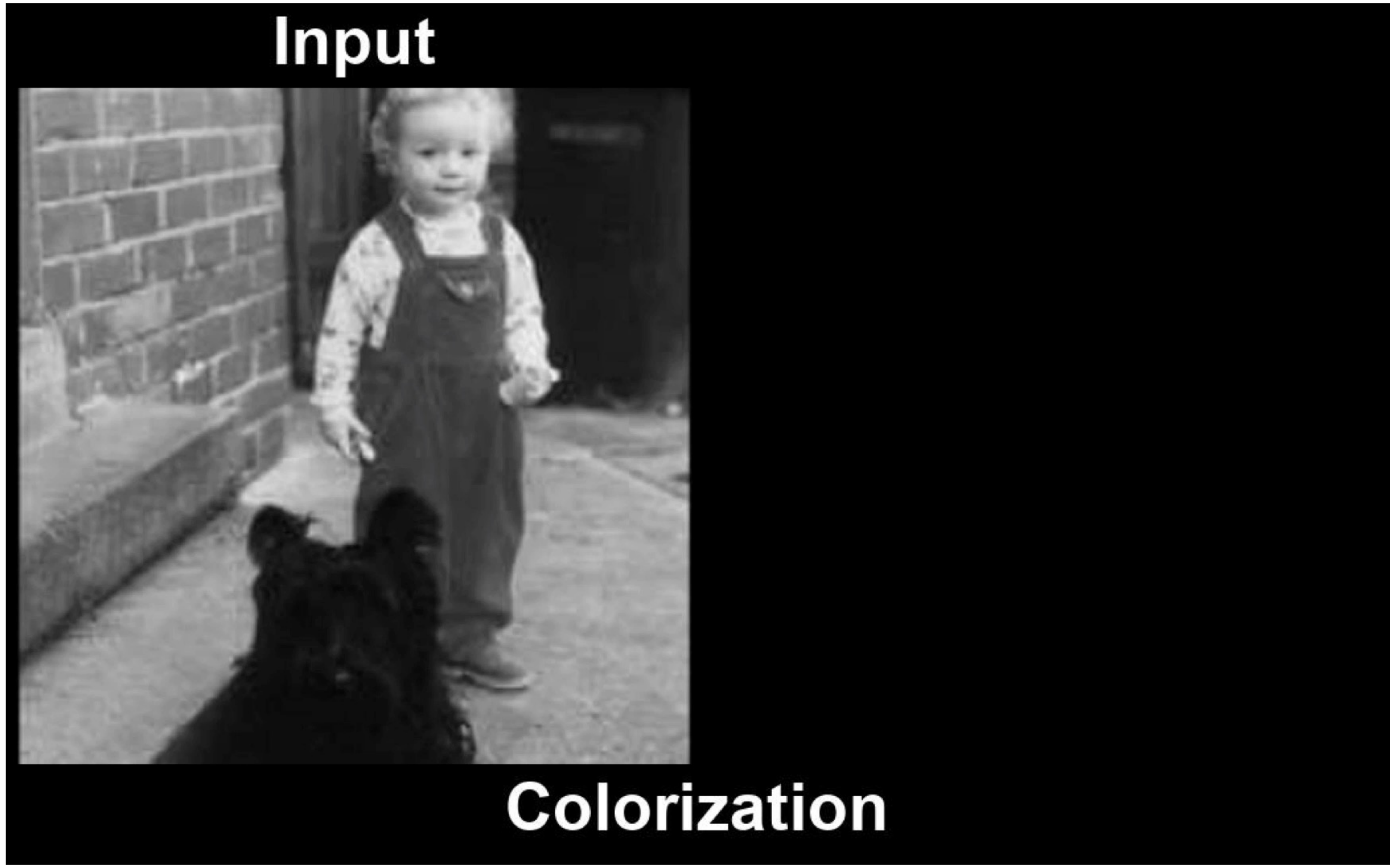
[Ramesh et al., “Hierarchical Text-Conditional Image Generation with CLIP Latents”, arXiv 2022.]()
[Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.]()

Impressive conditional diffusion models

Super-resolution & colorization



Super-resolution



Colorization

Saharia et al., "Palette: Image-to-Image Diffusion Models", arXiv 2021.

Impressive conditional diffusion models

Panorama generation

← Generated Input Generated →

Conditional diffusion models

Include condition as input to reverse process

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

Variational upper bound: $$L_\theta(\mathbf{x}_0|\mathbf{c}) = \mathbb{E}_q\left[L_T(\mathbf{x}_0) + \sum_{t>1} D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel \boxed{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{c})) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1,\mathbf{c})}\right].$$

Incorporate conditions into U-Net

- Scalar conditioning: encode scalar as a vector embedding, simple spatial addition or adaptive group normalization layers.
- Image conditioning: channel-wise concatenation of the conditional image.
- Text conditioning: single vector embedding – spatial addition or adaptive group norm / a seq of vector embeddings - cross-attention.

Classifier guidance

Using the gradient of a trained classifier as guidance

Recap: What is a score function?

Forward Diffusion SDE:

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

Reverse Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \overbrace{\left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)}_{\text{“Score Function”}}\right]}^{\text{drift term}}\mathrm{d}t + \overbrace{\sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t}^{\text{diffusion term}}$$

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t}\underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t)}}_{\text{diffused data } \mathbf{x}_t}||\underbrace{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\text{neural network}} - \underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)}_{\text{score of diffused data (marginal)}}||_2^2$$

Classifier guidance

Using the gradient of a trained classifier as guidance

Applying Bayes rule to obtain conditional score function

$$p(x \mid y) = \frac{p(y \mid x) \cdot p(x)}{p(y)}$$

$$\implies \log p(x \mid y) = \log p(y \mid x) + \log p(x) - \log p(y)$$

$$\implies \nabla_x \log p(x \mid y) = \nabla_x \log p(y \mid x) + \nabla_x \log p(x),$$

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x).$$ ← Classifier

Guidance scale: value >1 amplifies the influence of classifier signal.

$$p_\gamma(x \mid y) \propto p(x) \cdot p(y \mid x)^\gamma.$$

Slide Credits of guidance: https://benanne.github.io/2022/05/26/guidance.html

Classifier guidance

Using the gradient of a trained classifier as guidance

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x).$$

Samples from an unconditional diffusion model with classifier guidance, for guidance scales 1.0 (left) and 10.0 (right), taken from Dhariwal & Nichol (2021).

Classifier guidance

Using the gradient of a trained classifier as guidance

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s

$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$

for all t from T to 1 **do**

$\quad \mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$

$\quad x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma\, \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$

end for

return x_0

Score model

Classifier gradient

- Train unconditional Diffusion model

- Take your favorite classifier, depending on the conditioning type

- During inference / sampling mix the gradients of the classifier with the predicted score function of the unconditional diffusion model.

Classifier guidance

Problems of classifier guidance

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x).$$

Classifier

Guidance scale: value >1 amplifies the influence of classifier signal.

- At each step of denoising the input to the classifier is a noisy image x_t . Classifier is never trained on noisy image. So one needs to re-train classifier on noisy images! Can't use existing pre-trained classifiers.

- Most of the information in the input x is not relevant to predicting y, and as a result, taking the gradient of the classifier w.r.t. its input can yield arbitrary (and even adversarial) directions in input space.

Classifier-free guidance

Get guidance by Bayes' rule on conditional diffusion models

$$p(y \mid x) = \frac{p(x \mid y) \cdot p(y)}{p(x)}$$

$$\implies \log p(y \mid x) = \log p(x \mid y) + \log p(y) - \log p(x)$$

$$\implies \boxed{\nabla_x \log p(y \mid x) = \nabla_x \log p(x \mid y) - \nabla_x \log p(x).}$$

We proved this in classifier guidance.

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x).$$

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \left(\nabla_x \log p(x \mid y) - \nabla_x \log p(x)\right),$$

$$\nabla_x \log p_\gamma(x \mid y) = (1-\gamma)\nabla_x \log p(x) + \gamma \nabla_x \log p(x \mid y).$$

Score function for unconditional diffusion model

Score function for conditional diffusion model

Classifier-free guidance

Get guidance by Bayes' rule on conditional diffusion models

$$\nabla_x \log p_\gamma(x \mid y) = (1-\gamma)\nabla_x \log p(x) + \gamma \nabla_x \log p(x \mid y).$$

This is a barycentric combination of the conditional and the unconditional score function. For $\gamma = 0$, we recover the unconditional model, and for $\gamma = 1$ we get the standard conditional model. But $\gamma > 1$ is where the magic happens. Below are some examples from OpenAI's GLIDE model[8], obtained using classifier-free guidance.

Score function for unconditional diffusion model

Score function for conditional diffusion model

Two sets of samples from OpenAI's GLIDE model, for the prompt *'A stained glass window of a panda eating bamboo.'*, taken from their paper. Guidance scale 1 (no guidance) on the left, guidance scale 3 on the right.

Classifier-free guidance

Get guidance by Bayes' rule on conditional diffusion models

$$\nabla_x \log p_\gamma(x \mid y) = (1-\gamma)\nabla_x \log p(x) + \gamma\nabla_x \log p(x \mid y).$$

Score function for unconditional diffusion model

Score function for conditional diffusion model

In practice:

- Train a conditional diffusion model p(x|y), with *conditioning dropout*: some percentage of the time, the conditioning information y is removed (10-20% tends to work well).

- The conditioning is often replaced with a special input value representing the absence of conditioning information.

- The resulting model is now able to function both as a conditional model p(x|y), and as an unconditional model p(x), depending on whether the conditioning signal is provided.

- During inference / sampling simply mix the score function of conditional and unconditional diffusion model based on guidance scale.

Classifier-free guidance

Trade-off for sample quality and sample diversity

Non-guidance

Guidance scale = 1

Guidance scale = 3

Large guidance weight (ω) usually leads to better individual sample quality but less sample diversity.

Ho & Salimans, "Classifier-Free Diffusion Guidance", 2021.

Classifier guidance

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x).$$

Guidance scale

Classifier

X Need to train a separate "noise-robust" classifier + unconditional diffusion model.

X Gradient of the classifier w.r.t. input yields arbitrary values.

Classifier-free guidance

$$\nabla_x \log p_\gamma(x \mid y) = (1-\gamma) \nabla_x \log p(x) + \gamma \nabla_x \log p(x \mid y).$$

Score function for unconditional diffusion model

Score function for conditional diffusion model

+ Train conditional & unconditional diffusion model jointly via drop-out.

+ All pixels in input receive equally 'good' gradients.

Rather than constructing a generative model from classifier, we construct a classifier from a generative model!

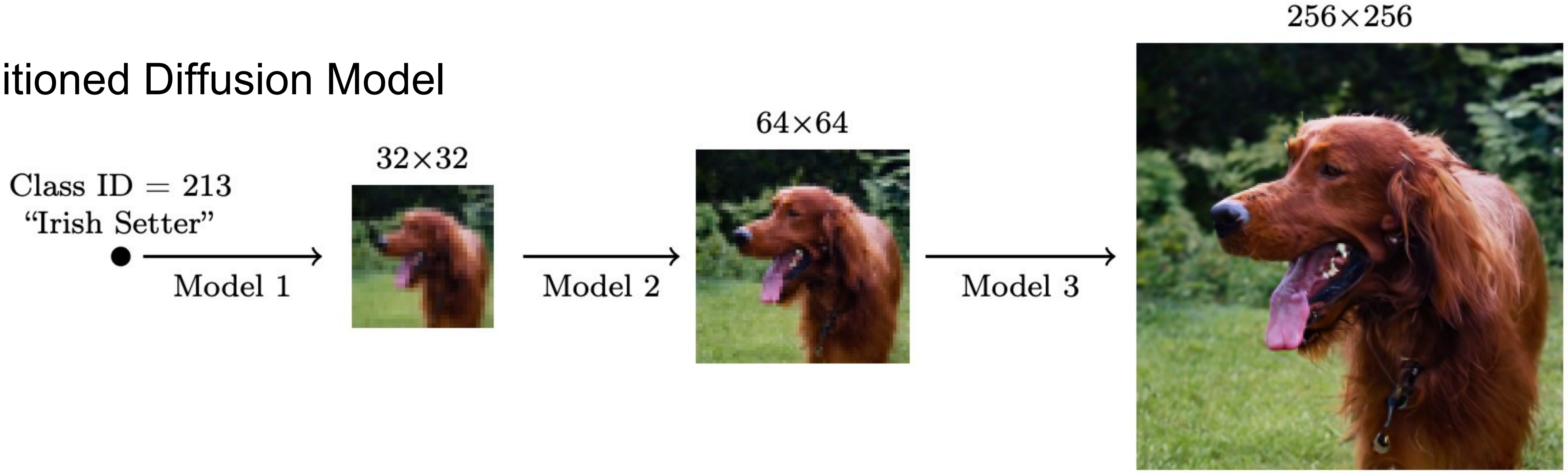
Most recent papers use classifier-free guidance! Very simple yet very powerful idea!

Cascaded generation

Pipeline

Super-Resolution Diffusion Models

Class Conditioned Diffusion Model



Similar cascaded / multi-resolution image generation also exist in GAN (Big-GAN & StyleGAN)

Cascaded Diffusion Models outperform Big-GAN in FID and IS and VQ-VAE2 in Classification Accuracy Score.

[Ho et al., “Cascaded Diffusion Models for High Fidelity Image Generation”, 2021.]

Noise conditioning augmentation

Reduce compounding error

Problem:

- During training super-resolution models are trained on original low-res images from the dataset.
- During inference, these low-res images are generated by class conditioned diffusion model, which has artifacts and poor quality than original low-res images used for training.

Mismatch issue

Solution: Noise conditioning augmentation.

- During training, add varying amounts of Gaussian noise (or blurring by Gaussian kernel) to the low-res images.
- During inference, sweep over the optimal amount of noise added to the low-res images.
- BSR-degradation process: applies JPEG compressions noise, camera sensor noise, different image interpolations for downsampling, Gaussian blur kernels and Gaussian noise in a random order to an image.

Ho et al., "Cascaded Diffusion Models for High Fidelity Image Generation", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

Summary

Questions to address with advanced techniques

- Q1: How to accelerate the sampling process?
  - Advanced forward diffusion process
  - Advanced reverse process
  - Hybrid models & model distillation
- Q2: How to do high-resolution (conditional) generation?
  - Conditional diffusion models
  - Classifier(-free) guidance
  - Cascaded generation

Applications (1):
Image Synthesis, Controllable Generation, Text-to-Image

GLIDE

OpenAI

- A 64x64 base model + a 64x64 → 256x256 super-resolution model.
- Tried classifier-free and CLIP guidance. Classifier-free guidance works better than CLIP guidance.

"a hedgehog using a calculator"

"a corgi wearing a red bowtie and a purple party hat"

"robots meditating in a vipassana retreat"

"a fall landscape with a small cottage next to a lake"

Samples generated with classifier-free guidance (256x256)

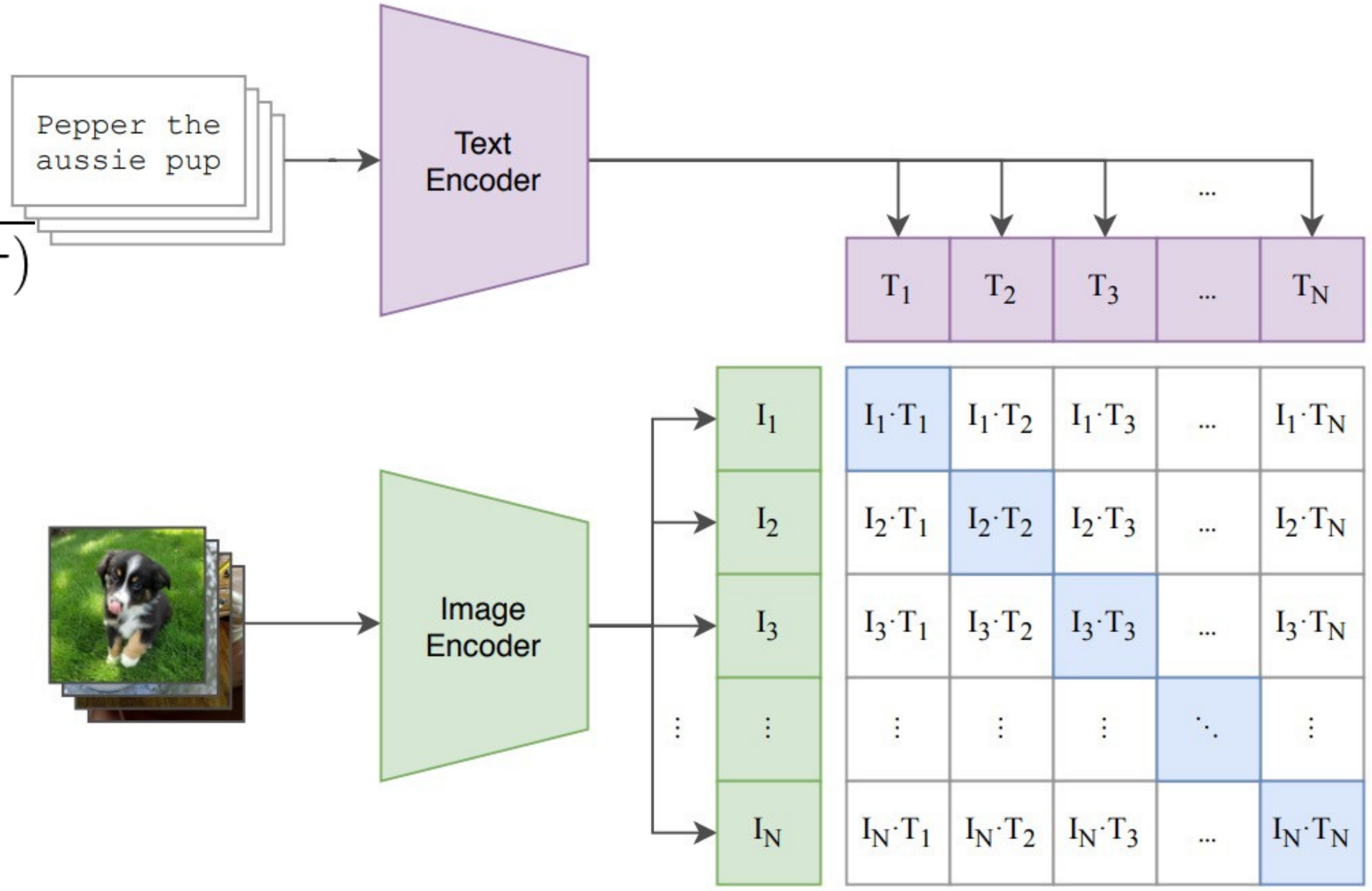
[Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.]

CLIP guidance

What is a CLIP model?

- Trained by contrastive cross-entropy loss:

$$-\log \frac{\exp(f(\mathbf{x}_i) \cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_i) \cdot g(\mathbf{c}_k)/\tau)} - \log \frac{\exp(f(\mathbf{x}_i) \cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_k) \cdot g(\mathbf{c}_j)/\tau)}$$



- The optimal value of $f(\mathbf{x}) \cdot g(\mathbf{c})$ is

$$\log \frac{p(\mathbf{x}, \mathbf{c})}{p(\mathbf{x})p(\mathbf{c})} = \log p(\mathbf{c}|\mathbf{x}) - \log p(\mathbf{c})$$

Radford et al., “Learning Transferable Visual Models From Natural Language Supervision”, 2021.
Nichol et al., “GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models”, 2021.

CLIP guidance

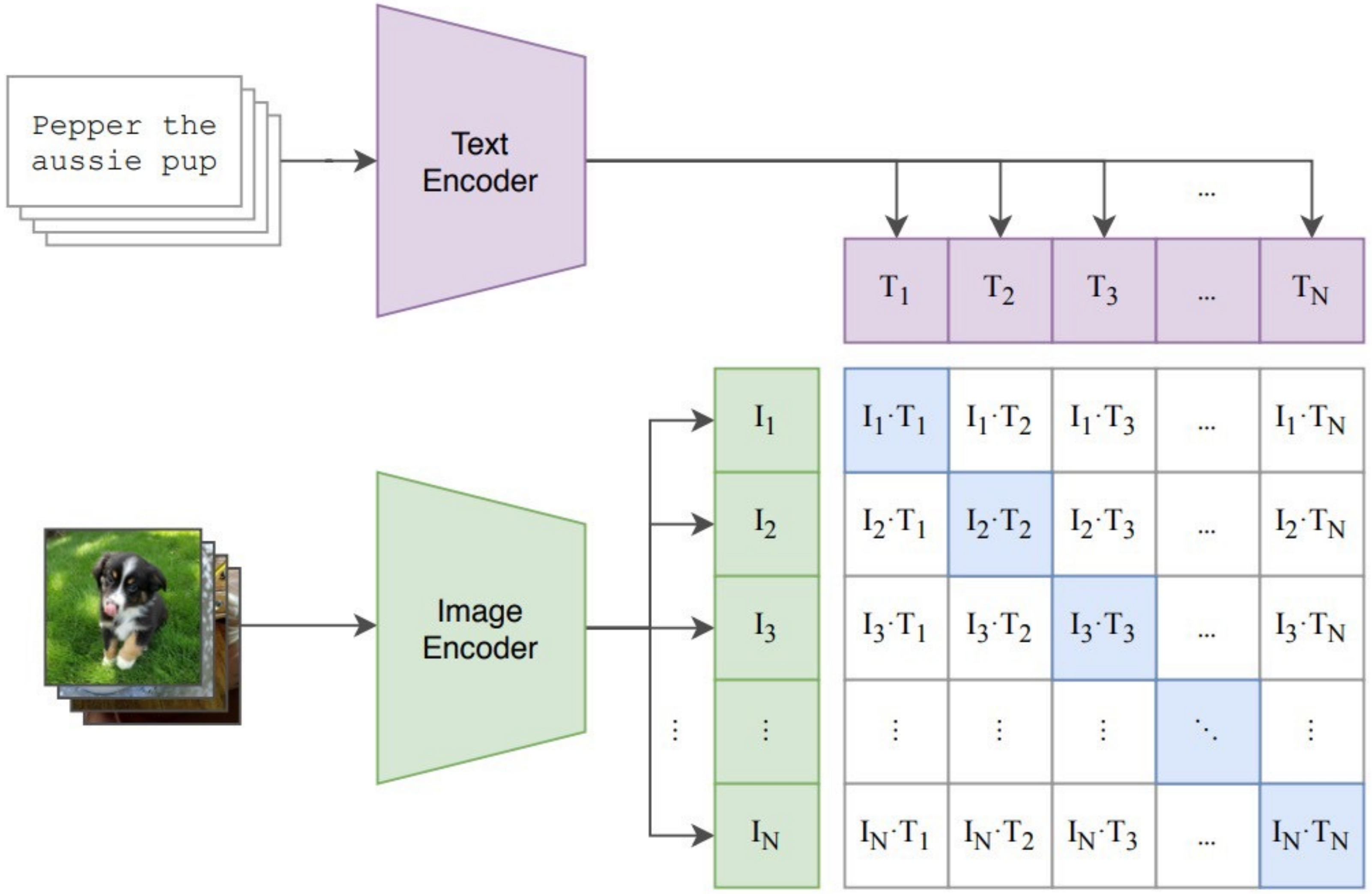
Replace the classifier in classifier guidance with a CLIP model

- Sample with a modified score:

$$\nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega\underbrace{(\log p(\mathbf{c}|\mathbf{x}_t) - \log p(\mathbf{c}))}_{\text{CLIP model}}]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(f(\mathbf{x}_t) \cdot g(\mathbf{c}))]$$



Radford et al., "Learning Transferable Visual Models From Natural Language Supervision", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

GLIDE

OpenAI

- Fine-tune the model especially for inpainting: feed randomly occluded images with an additional mask channel as the input.

“an old car in a snowy forest”

“a man wearing a white hat”

Text-conditional image inpainting examples

Nichol et al., “GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models”, 2021.

DALL·E 2

OpenAI

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

1kx1k Text-to-image generation.

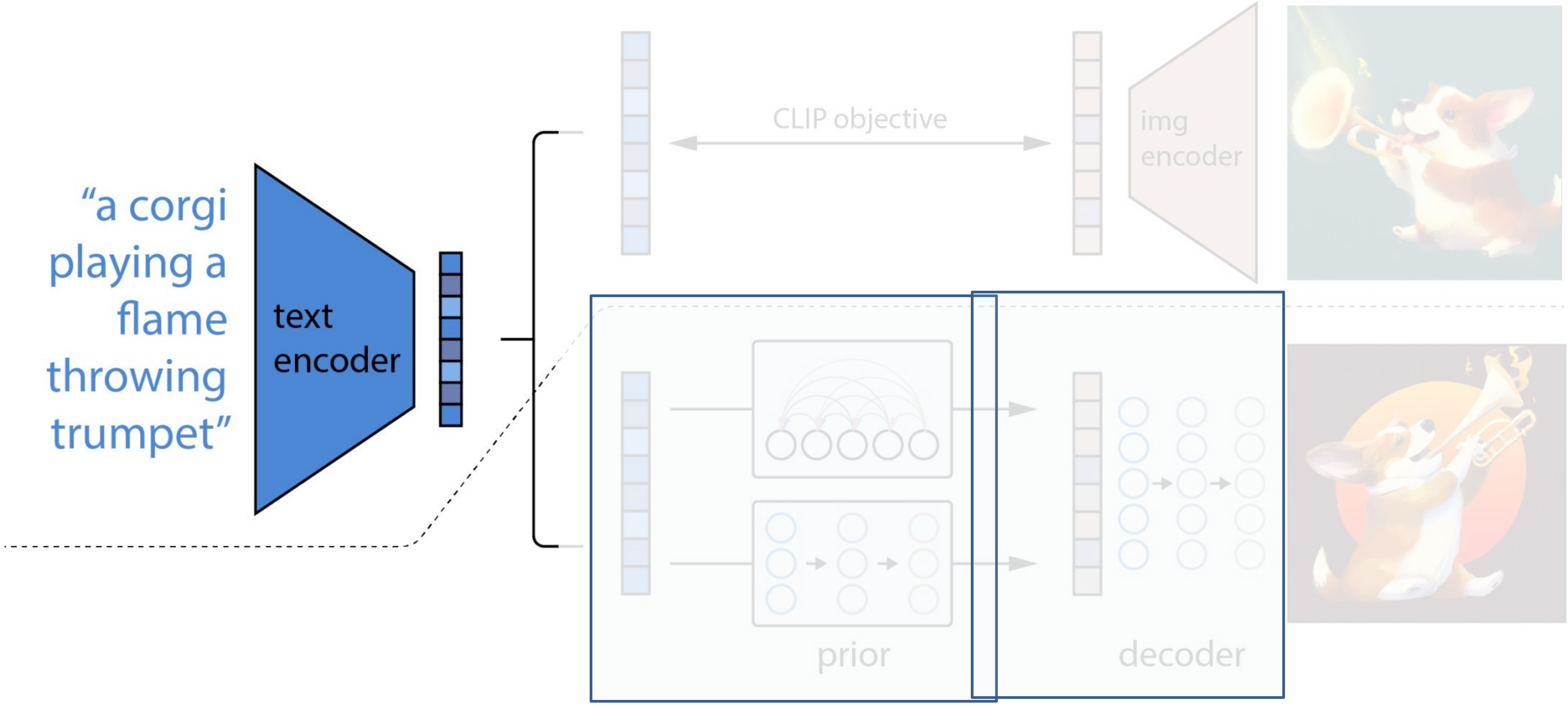
Outperform DALL-E (autoregressive transformer).

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Model components

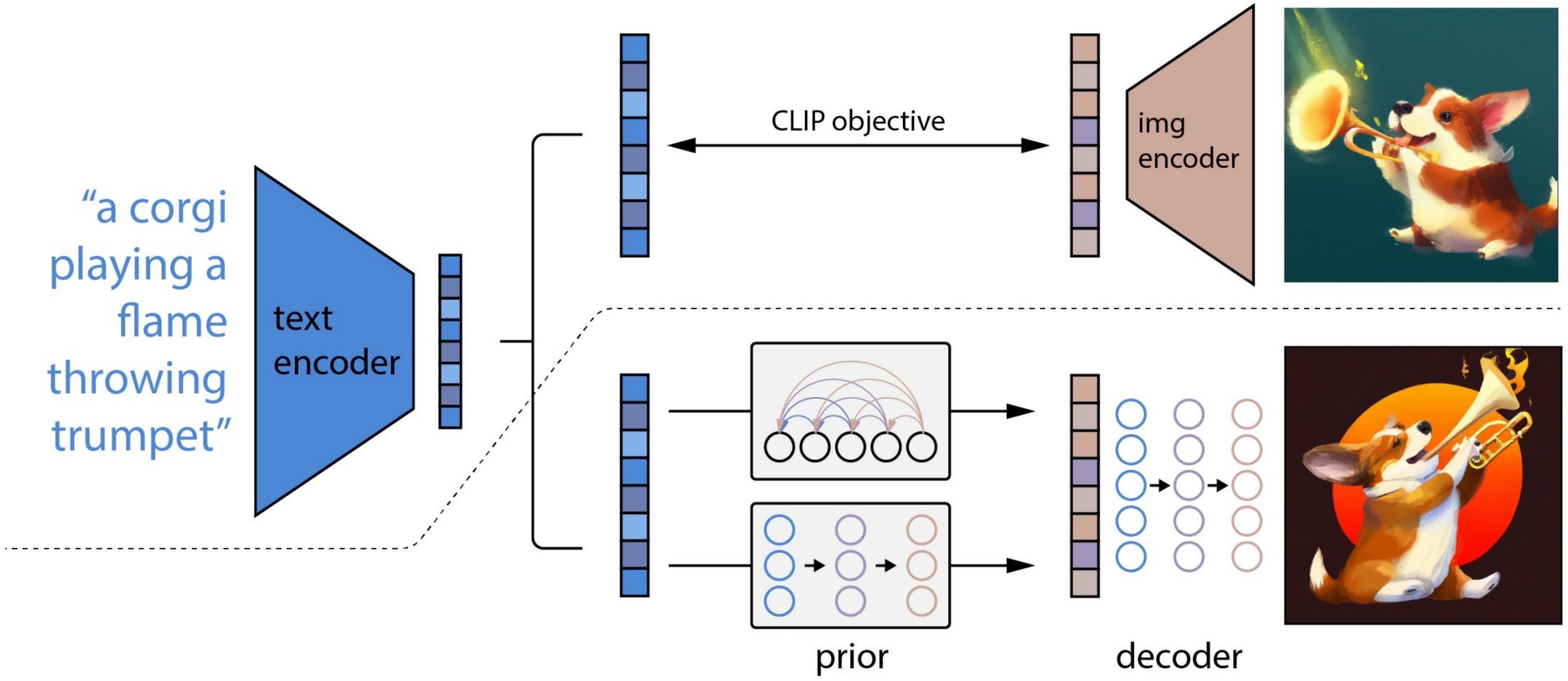
Prior: produces CLIP image embeddings conditioned on the caption.
Decoder: produces images conditioned on CLIP image embeddings and text.



Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Model components



Why conditional on CLIP image embeddings?

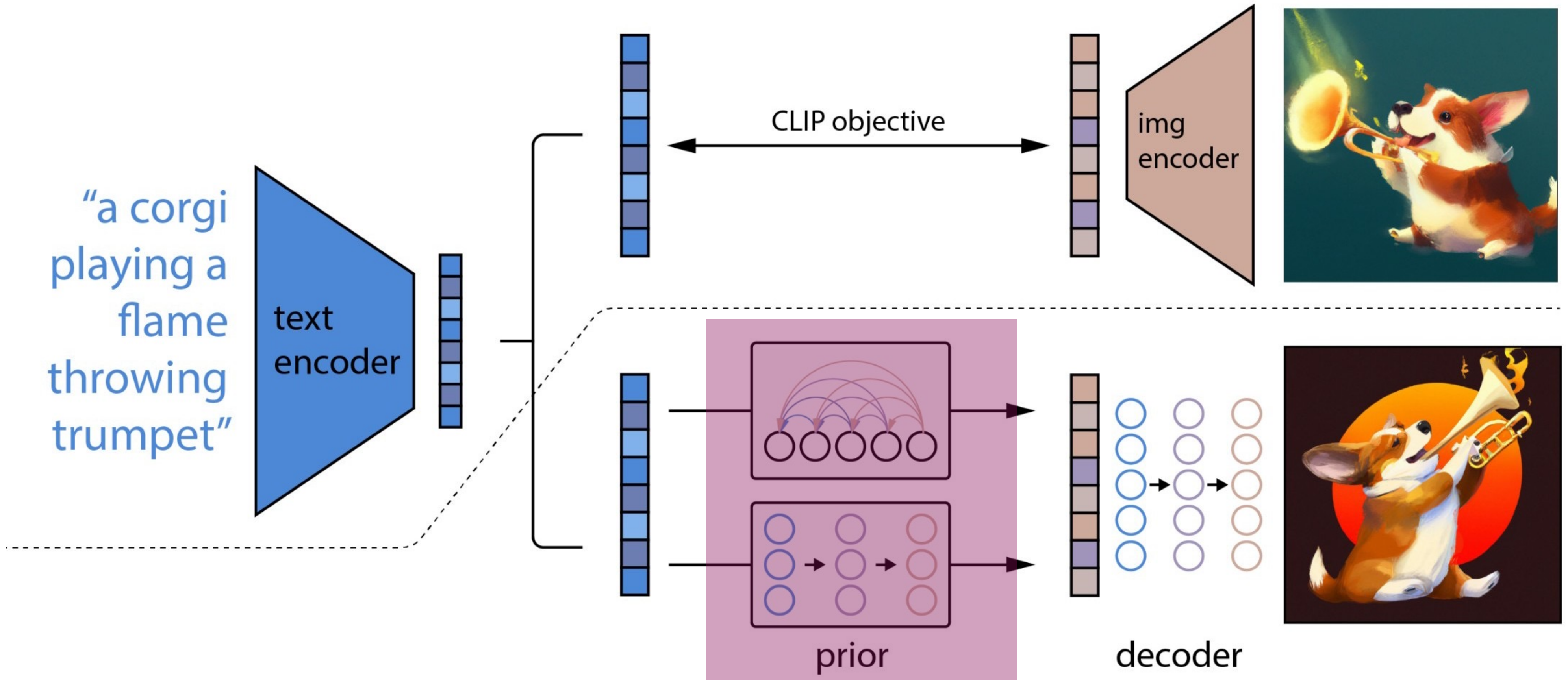
CLIP image embeddings capture high-level semantic meaning.

Latents in the decoder model take care of the rest.

The bipartite latent representation enables several text-guided image manipulation tasks.

DALL·E 2

Model components (1/2): prior model



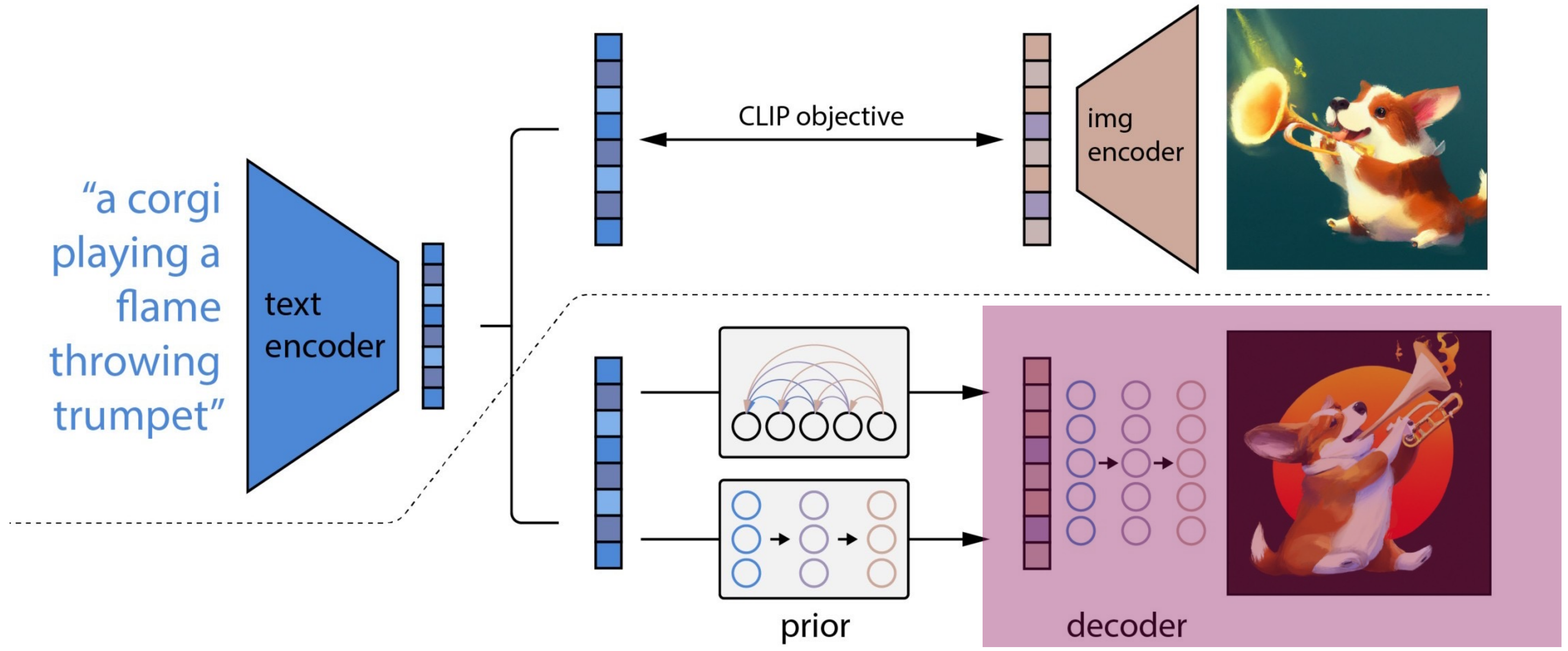
Prior: produces CLIP image embeddings conditioned on the caption.

- Option 1. autoregressive prior: quantize image embedding to a seq. of discrete codes and predict them autoregressively.
- Option 2. diffusion prior: model the continuous image embedding by diffusion models conditioned on caption.

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Model components (2/2): decoder model



Decoder: produces images conditioned on CLIP image embeddings (and text).

- Cascaded diffusion models: 1 base model (64x64), 2 super-resolution models (64x64 → 256x256, 256x256 → 1024x1024).
- Largest super-resolution model is trained on patches and takes full-res inputs at inference time.
- Classifier-free guidance & noise conditioning augmentation are important.

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

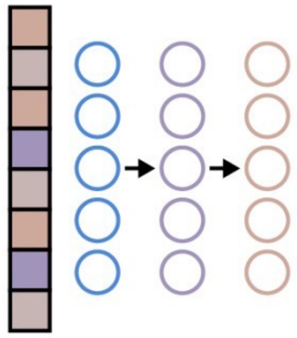
DALL·E 2

Bipartite latent representations

Bipartite latent representations $(\mathbf{z}, \mathbf{x}_T)$

$\mathbf{z}$: CLIP image embeddings

$\mathbf{x}_T$: inversion of DDIM sampler
(latents in the decoder model)



decoder

Near exact
reconstruction

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

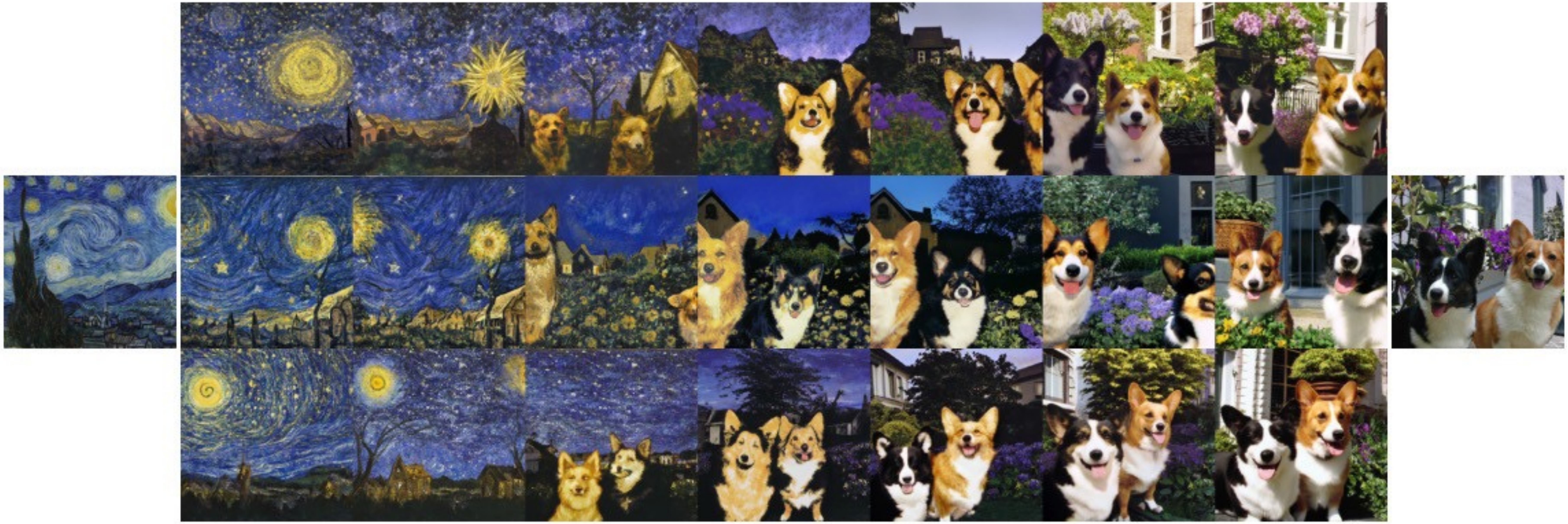
Image variations

Fix the CLIP embedding $\mathbf{z}$.

Decode using different decoder latents $\mathbf{x}_T$

DALL·E 2

Image interpolation



Interpolate image CLIP embeddings $\mathbf{z}$.

Use different $\mathbf{x}_T$ to get different interpolation trajectories.

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Text Diffs

a photo of a cat → an anime drawing of a super saiyan cat, artstation

a photo of a victorian house → a photo of a modern house

a photo of an adult lion → a photo of lion cub

Change the image CLIP embedding towards the difference of the text CLIP embeddings of two prompts.

Decoder latent is kept as a constant.

Imagen

Google Research, Brain team

Input: text;   Output: 1kx1k images

- An unprecedented degree of photorealism
    - SOTA automatic scores & human ratings
- A deep level of language understanding
- Extremely simple
    - no latent space, no quantization

A brain riding a rocketship heading towards the moon.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Google Research, Brain team

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

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Google Research, Brain team

A dragon fruit wearing karate belt in the snow.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Google Research, Brain team

A relaxed garlic with a blindfold reading a newspaper while floating in a pool of tomato soup.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

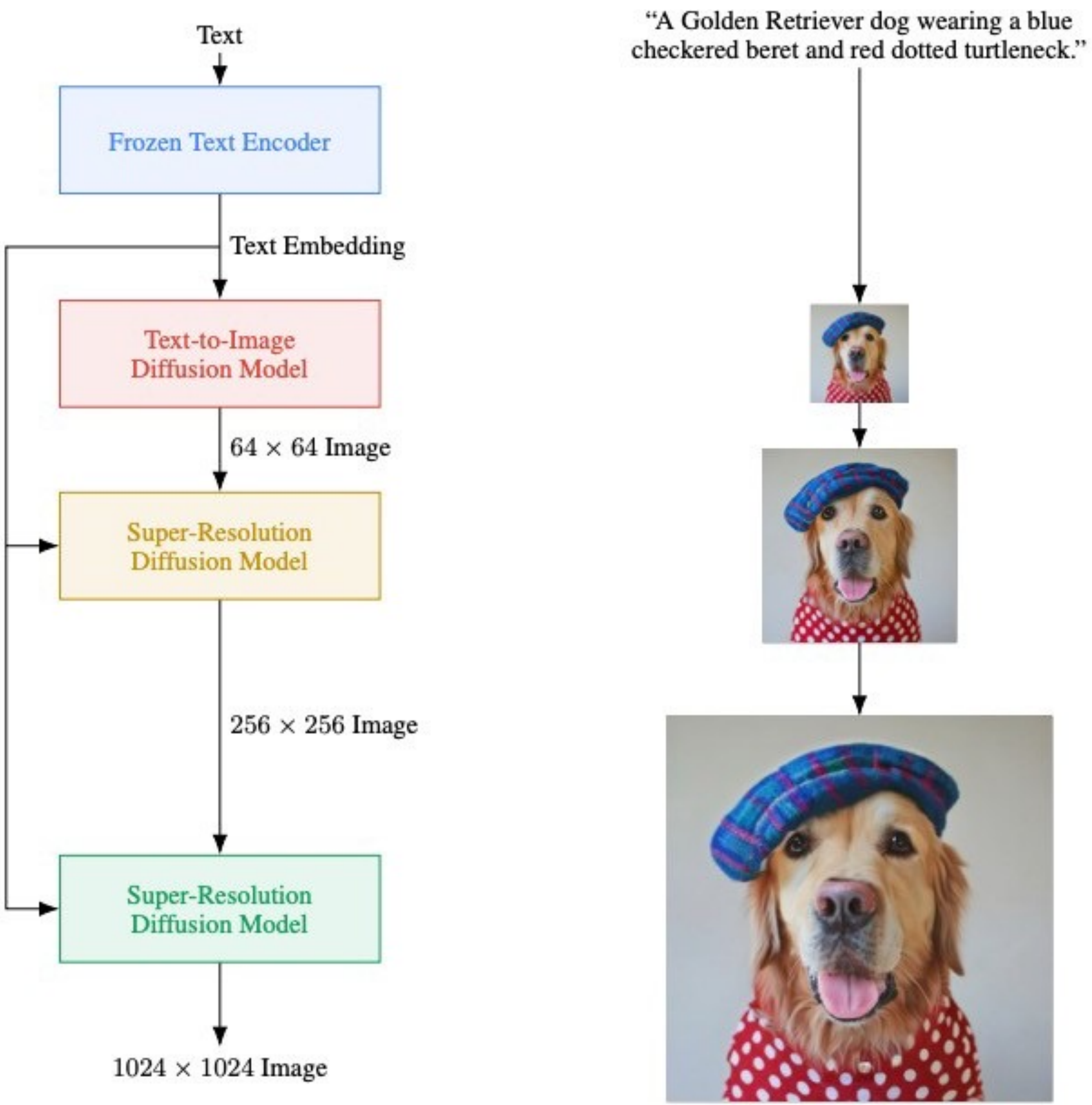
Imagen

Google Research, Brain team

A cute hand-knitted koala wearing a sweater with 'CVPR' written on it.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen



Key modeling components:

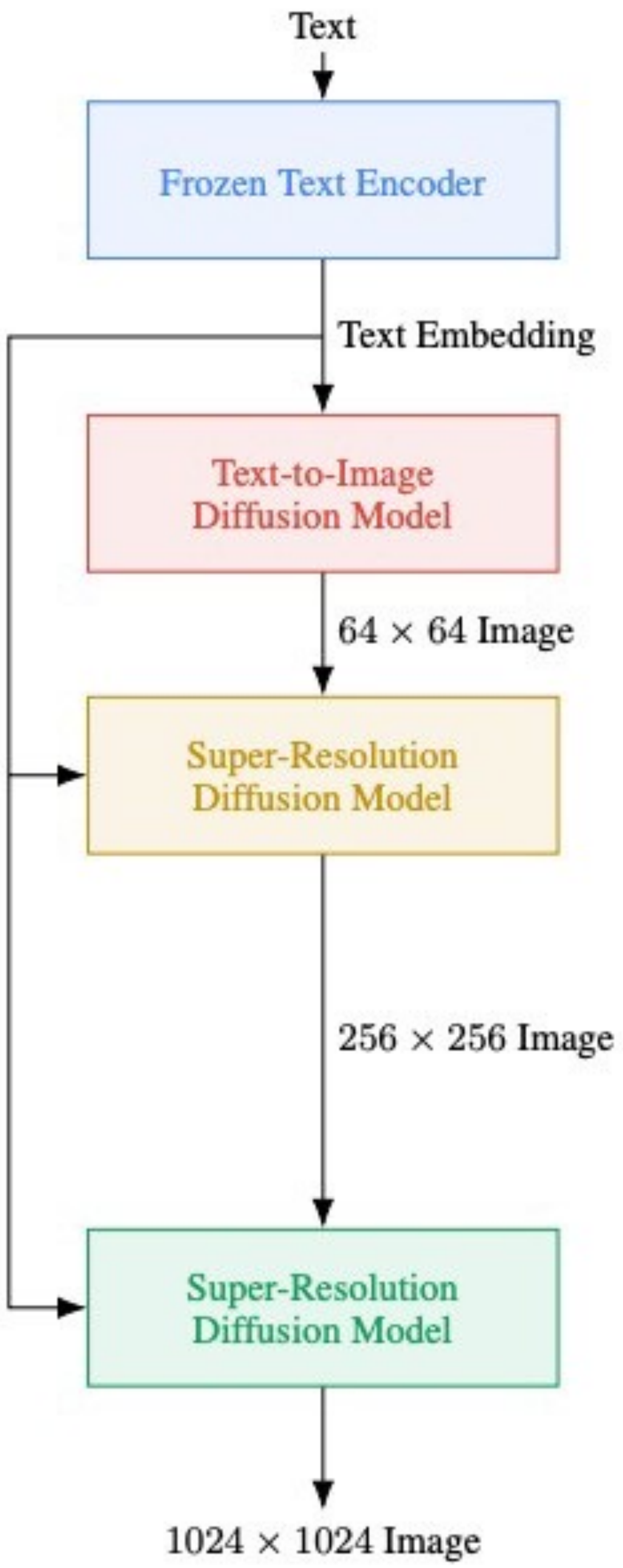
- Cascaded diffusion models
- Classifier-free guidance and dynamic thresholding.
- Frozen large pretrained language models as text encoders. (T5-XXL)

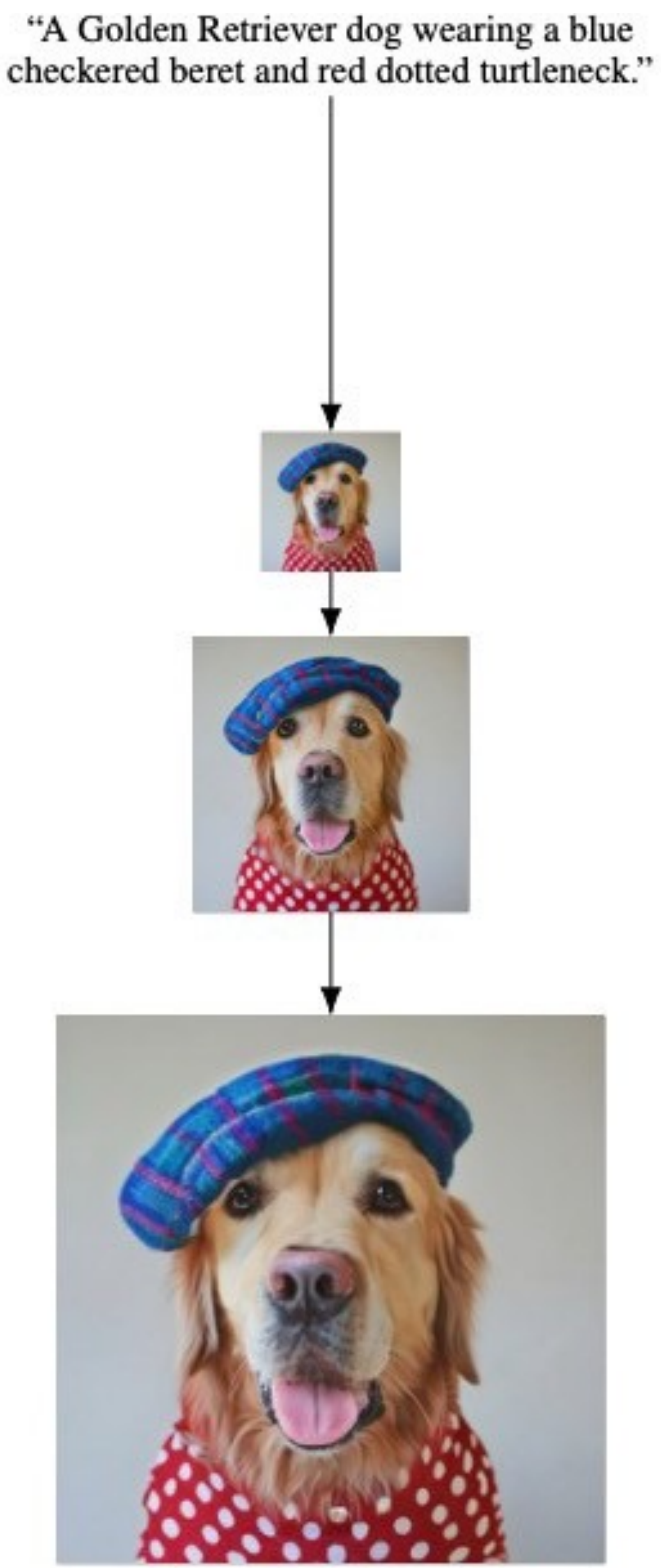
Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

Imagen

Key observations:

- Beneficial to use text conditioning for all super-res models.
    - Noise conditioning augmentation weakens information from low-res models, thus needs text conditioning as extra information input.
- Scaling text encoder is extremely efficient.
    - More important than scaling diffusion model size.
- Human raters prefer T5-XXL as the text encoder over CLIP encoder on DrawBench.



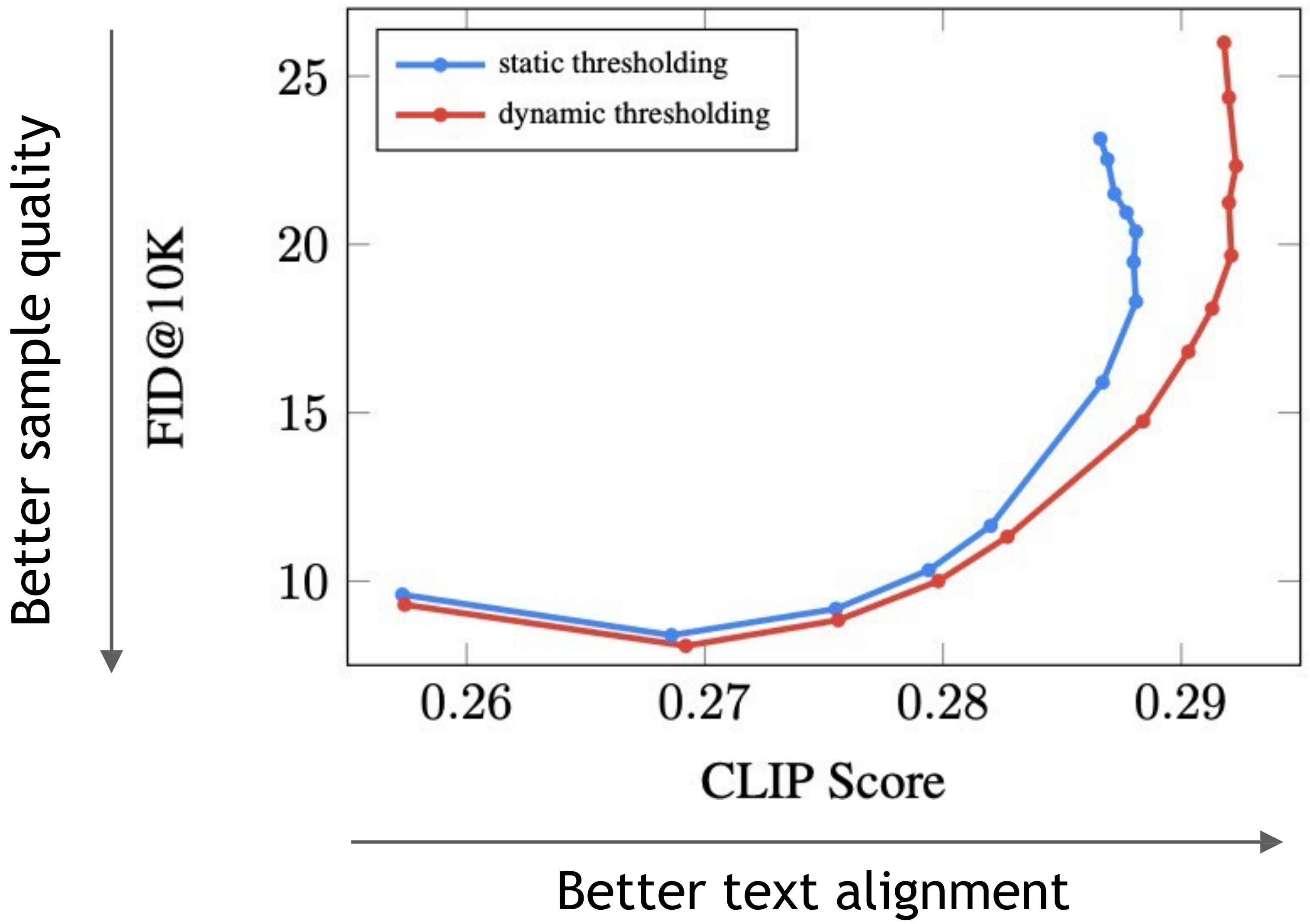


Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Dynamic thresholding

- Large classifier-free guidance weights → better text alignment, worse image quality



Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Dynamic thresholding

- Large classifier-free guidance weights → better text alignment, worse image quality
- Hypothesis : at large guidance weight, the generated images are saturated due to the very large gradient updates during sampling
- Solution – dynamic thresholding: adjusts the pixel values of samples at each sampling step to be within a dynamic range computed over the statistics of the current samples.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Dynamic thresholding

- Large clas
- Hypothesi dates during sampling
- Solution – ynamic range com

Static thresholding

Dynamic thresholding

[Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.]

Imagen

DrawBench: new benchmark for text-to-image evaluations

- A set of 200 prompts to evaluate text-to-image models across multiple dimensions.
    - E.g., the ability to faithfully render different colors, numbers of objects, spatial relations, text in the scene, unusual interactions between objects.
    - Contains complex prompts, e.g, long and intricate descriptions, rare words, misspelled prompts.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

Imagen

DrawBench: new benchmark for text-to-image evaluations

- A set of 200 pro
  - E.g., the
    interactio
    cene, unusual
  - Contains

A brown bird and a blue bear.
One cat and two dogs sitting on the grass.
A sign that says 'NeurIPS'.
A small blue book sitting on a large red book.
A blue coloured pizza.
A wine glass on top of a dog.
A pear cut into seven pieces arranged in a ring.
A photo of a confused grizzly bear in calculus class.
A small vessel propelled on water by oars, sails, or an engine.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

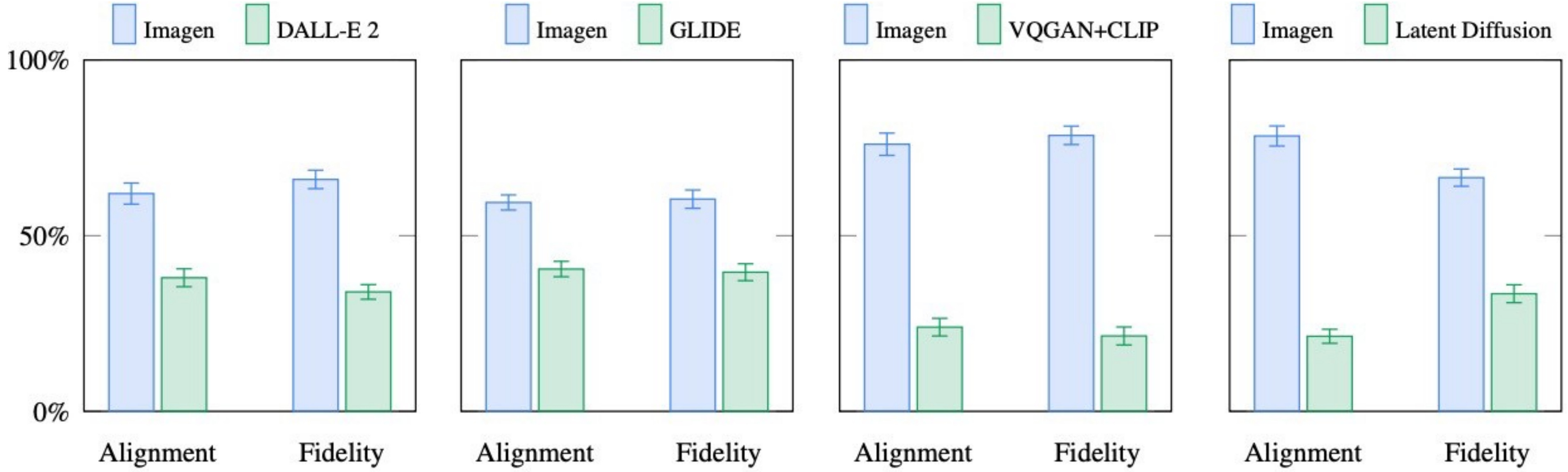
Imagen

Evaluations

Imagen got SOTA automatic evaluation scores on COCO dataset

Model	FID-30K	Zero-shot FID-30K
AttnGAN [76]	35.49	
DM-GAN [83]	32.64	
DF-GAN [69]	21.42	
DM-GAN + CL [78]	20.79	
XMC-GAN [81]	9.33	
LAFITE [82]	8.12	
Make-A-Scene [22]	7.55	
DALL-E [53]		17.89
LAFITE [82]		26.94
GLIDE [41]		12.24
DALL-E 2 [54]		10.39
Imagen (Our Work)		**7.27**

Imagen is preferred over recent work by human raters in sample quality & image-text alignment on DrawBench.



Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Stable Diffusion

Latest & Publicly available text-to-image generation

To be discussed in detail in paper presentation

High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach*, Andreas Blattmann*, Dominik Lorenz, Patrick Esser, Björn Ommer

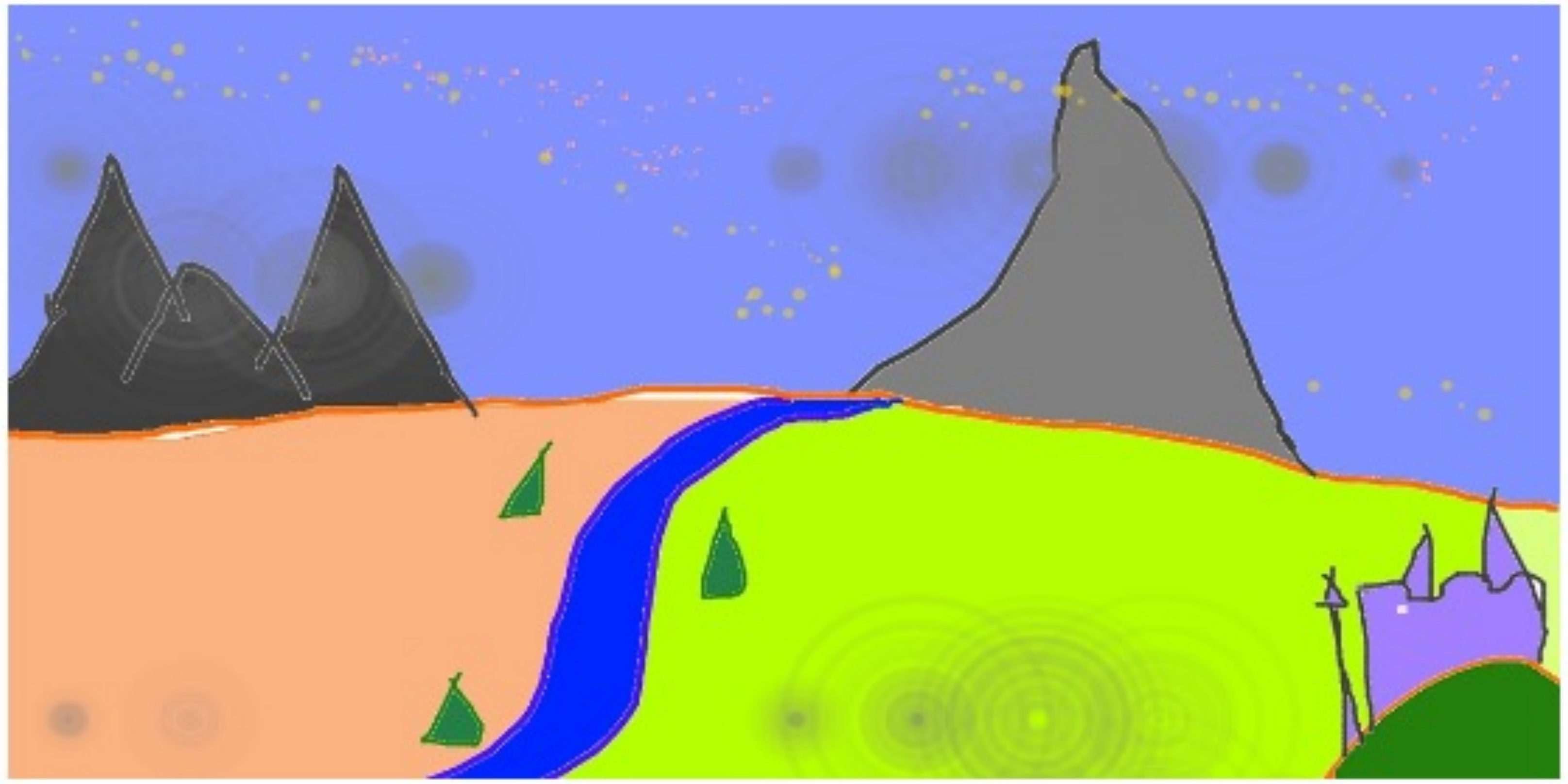
CVPR '22 Oral | *GitHub* | *arXiv* | *Project page*

Stable Diffusion is a latent text-to-image diffusion model. Thanks to a generous compute donation from Stability AI and support from LAION, we were able to train a Latent Diffusion Model on 512x512 images from a subset of the LAION-5B database. Similar to Google's Imagen, this model uses a frozen CLIP ViT-L/14 text encoder to condition the model on text prompts. With its 860M UNet and 123M text encoder, the model is relatively lightweight and runs on a GPU with at least 10GB VRAM. See this section below and the model card.

Stable Diffusion

Latest & Publicly available text-to-image generation

Input



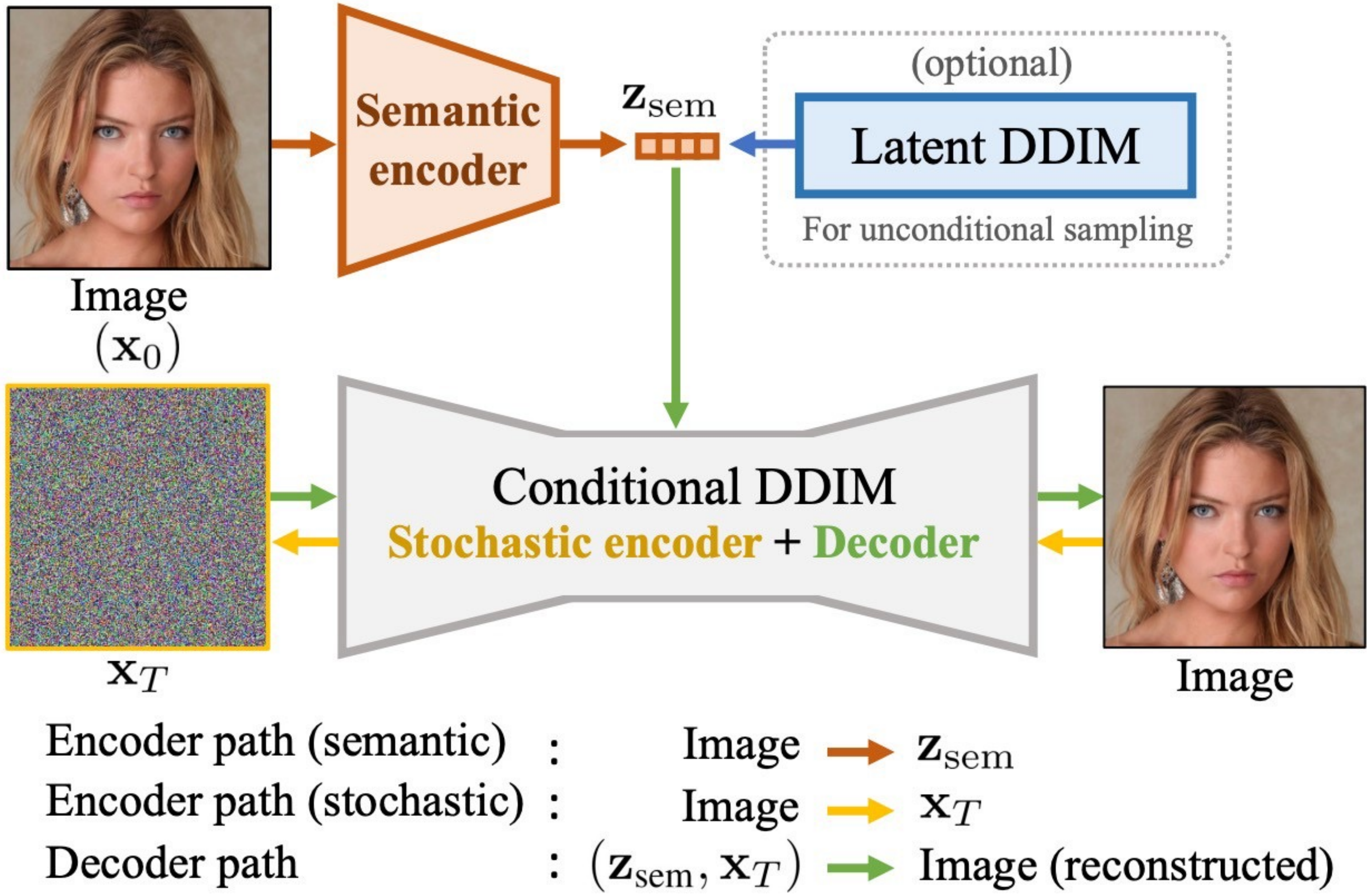
Outputs

HW assignment: Use stable diffusion API to generate ‘interesting’ image from text prompt. All submissions will be rated for top 3!

Applications (2):
Image Editing, Image-to-Image, Super-resolution, Segmentation

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models

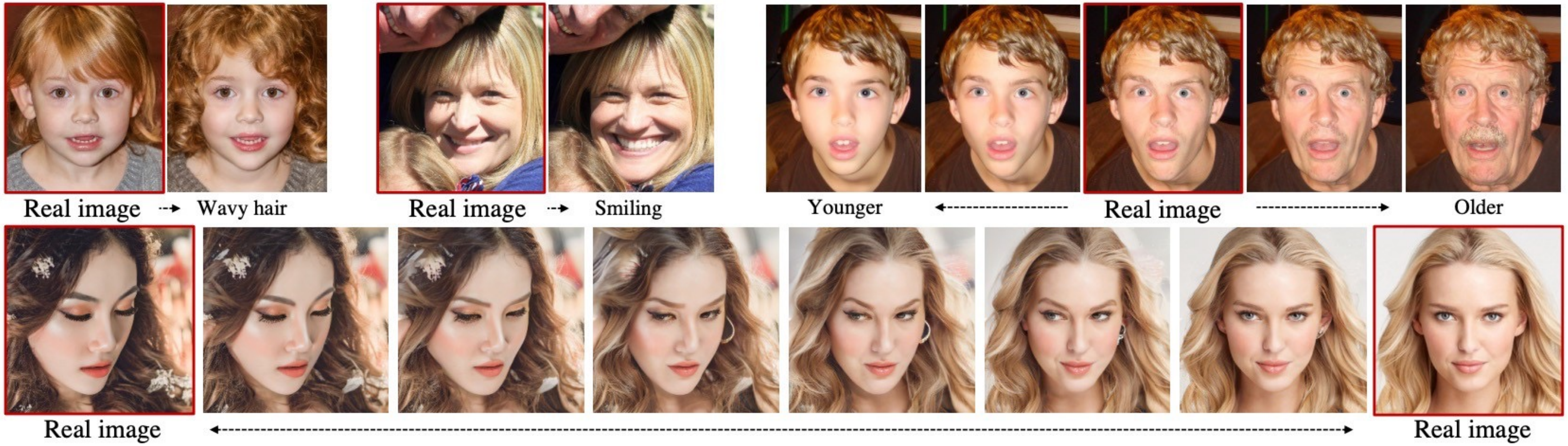


To be discussed in detail in paper presentation

[Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.](#)

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models



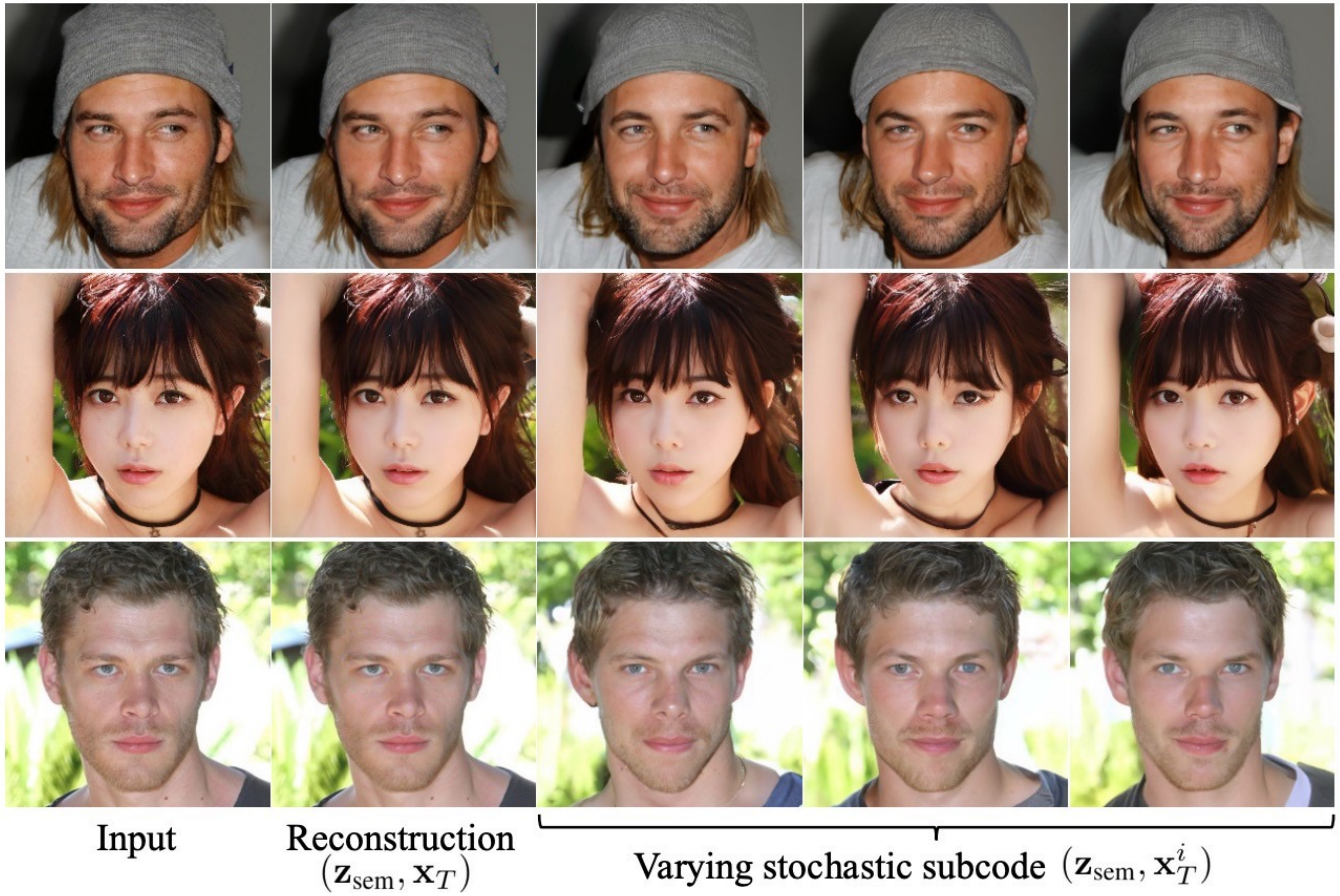
Changing the semantic latent $\mathbf{z}_{\text{sem}}$

Very similar to StyleGAN based editing. Zsem is the latent representation similar to the W/W+ space of StyleGAN

Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models



[Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.]

Super-Resolution

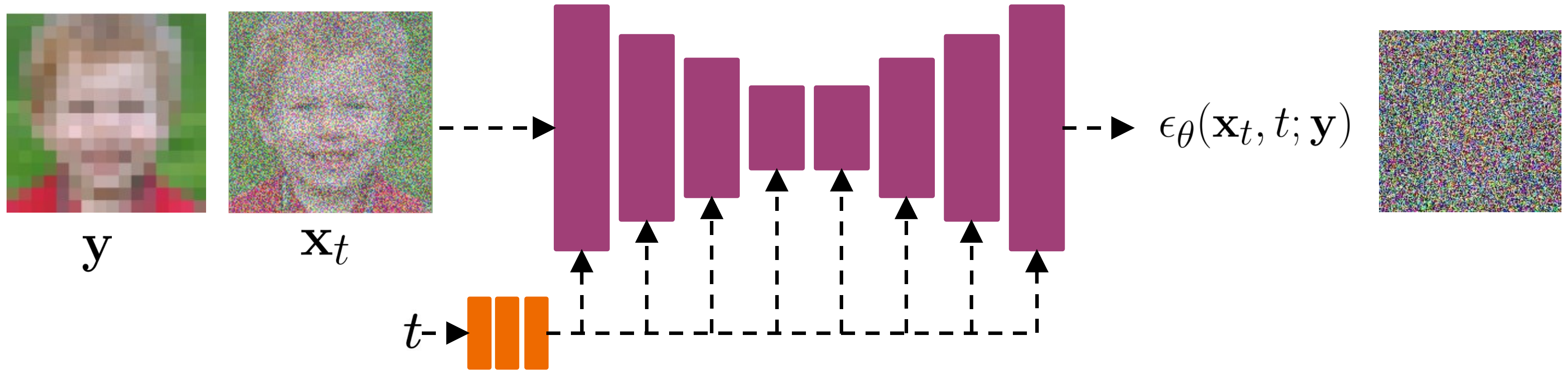
Super-Resolution via Repeated Refinement (SR3)

Image super-resolution can be considered as training $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is a low-resolution image and $\mathbf{x}$ is the corresponding high-resolution image

Train a score model for $\mathbf{x}$ conditioned on $\mathbf{y}$ using:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\, \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\, \mathbb{E}_t\, ||\epsilon_\theta(\mathbf{x}_t, t; \mathbf{y}) - \epsilon||_p^p$$

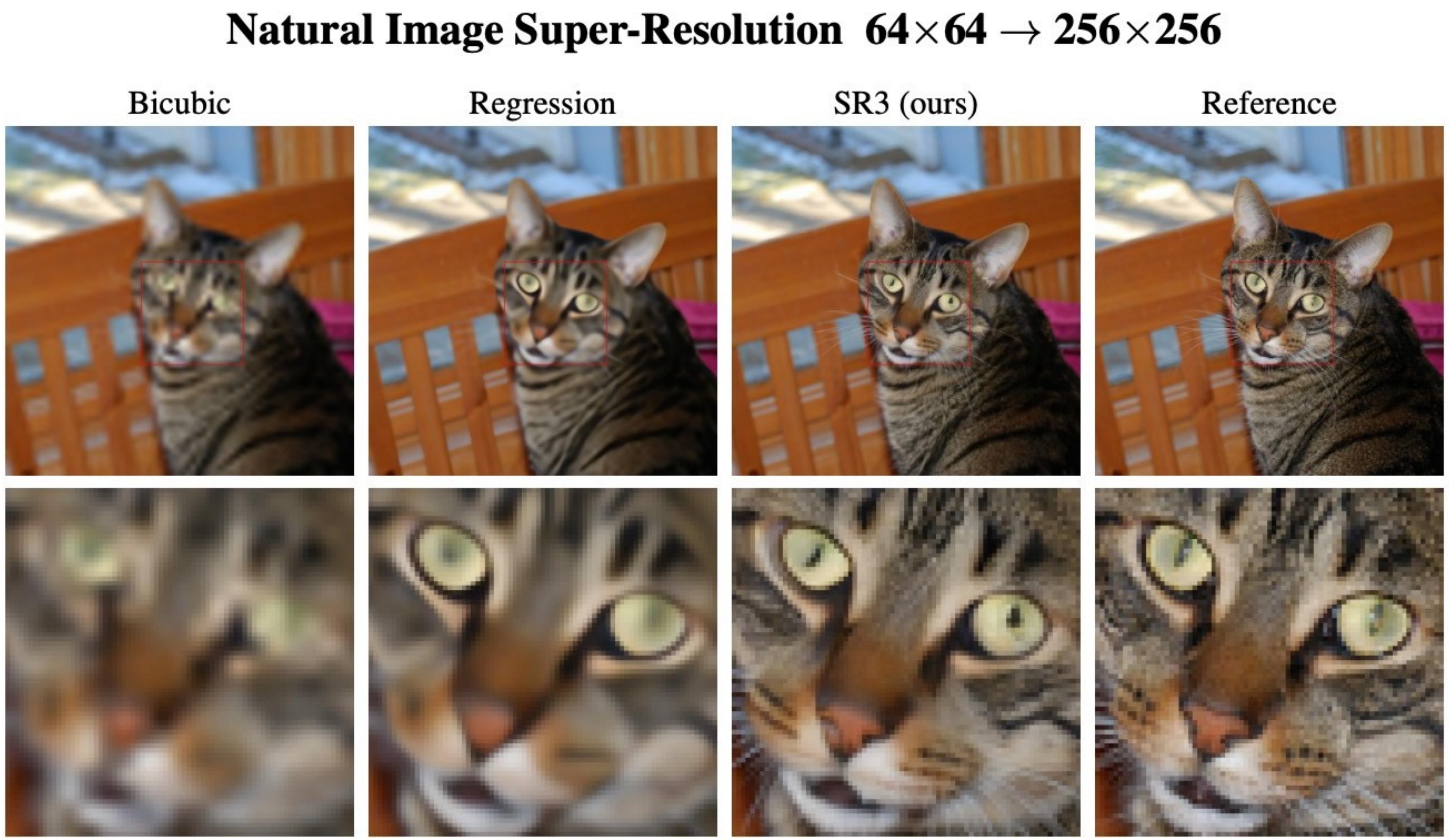
The conditional score is simply a U-Net with $\mathbf{x}_t$ and $\mathbf{y}$ (resolution image) concatenated.



Saharia et al., Image Super-Resolution via Iterative Refinement, 2021

Super-Resolution

Super-Resolution via Repeated Refinement (SR3)



[Saharia et al., Image Super-Resolution via Iterative Refinement, 2021]

Image-to-Image Translation

Palette: Image-to-Image Diffusion Models

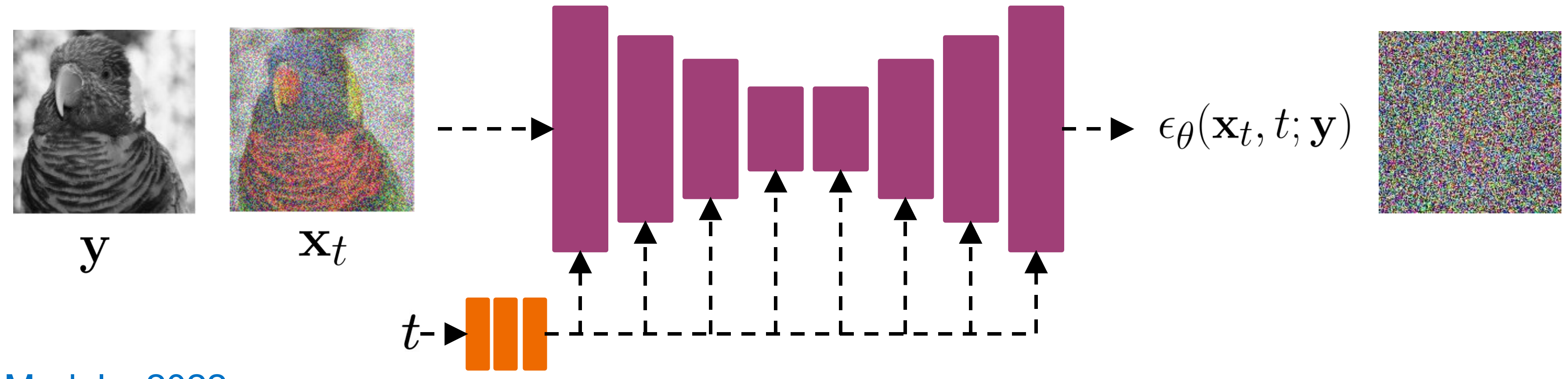
Many image-to-image translation applications can be considered as training $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is the input image.

For example, for colorization, $\mathbf{x}$ is a colored image and $\mathbf{y}$ is a gray-level image.

Train a score model for $\mathbf{x}$ conditioned on $\mathbf{y}$ using:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\, \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\, \mathbb{E}_t\, ||\epsilon_\theta(\mathbf{x}_t, t; \mathbf{y}) - \epsilon||_p^p$$

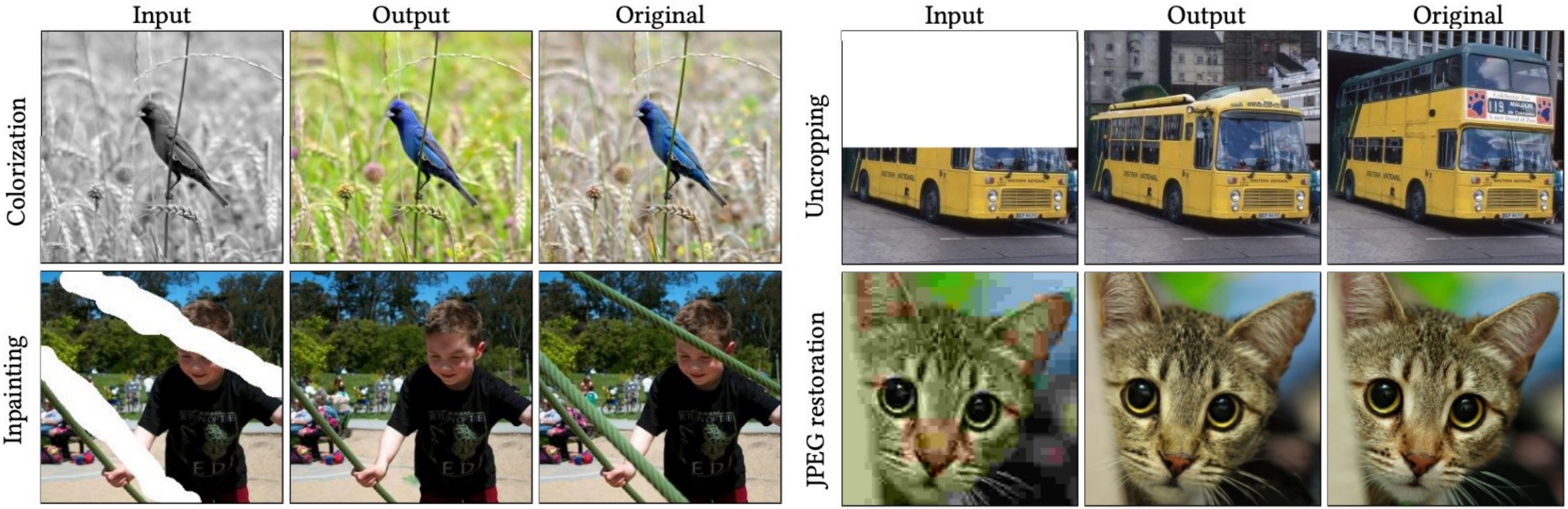
The conditional score is simply a U-Net with $\mathbf{x}_\mathbf{t}$ and $\mathbf{y}$ concatenated.



Saharia et al., Palette: Image-to-Image Diffusion Models, 2022

Image-to-Image Translation

Palette: Image-to-Image Diffusion Models



Saharia et al., Palette: Image-to-Image Diffusion Models, 2022

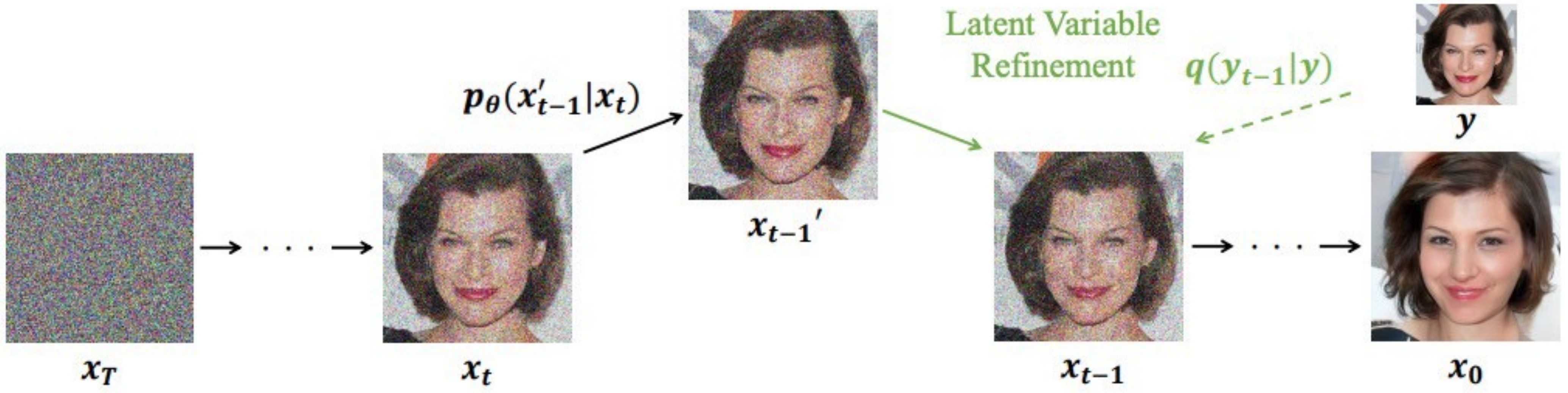
Conditional Generation

Iterative Latent Variable Refinement (ILVR)

To be discussed in detail in paper presentation

A simple technique to guide the generation process of an unconditional diffusion model using a reference image.

Given the conditioning (reference) image $\mathbf{y}$ the generation process is modified to pull the samples towards the reference image.



for $t = T, ..., 1$ **do**

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

$\quad x'_{t-1} \sim p_\theta(x'_{t-1}|x_t)$ ▷ unconditional proposal

$\quad y_{t-1} \sim q(y_{t-1}|y)$ ▷ condition encoding

$\quad x_{t-1} \leftarrow \phi_N(y_{t-1}) + x'_{t-1} - \phi_N(x'_{t-1})$

end for

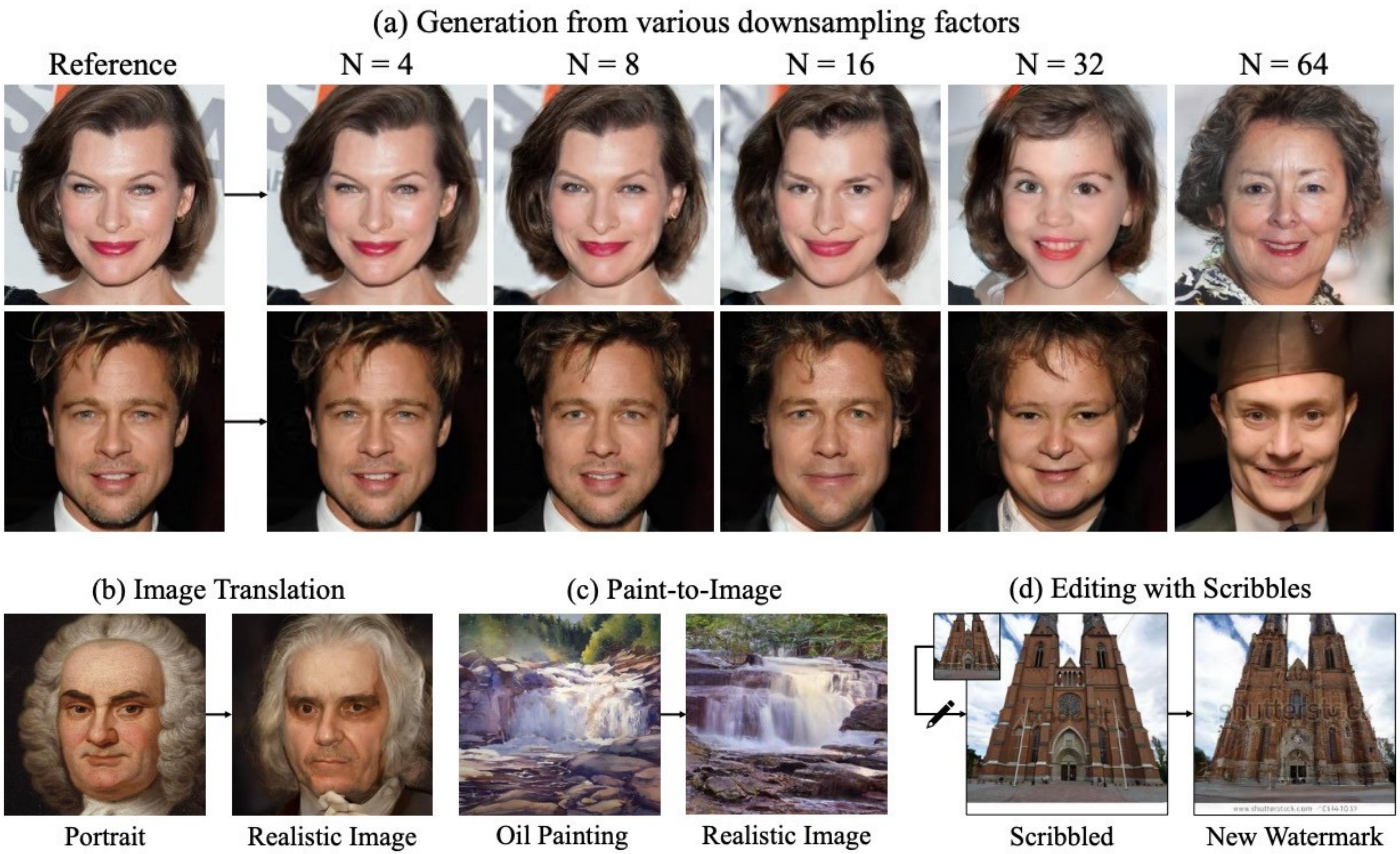
Low-pass filter

Downsampling / Upsampling by a factor of N

Choi et al., ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models, ICCV 2021

Conditional Generation

Iterative Latent Variable Refinement (ILVR)

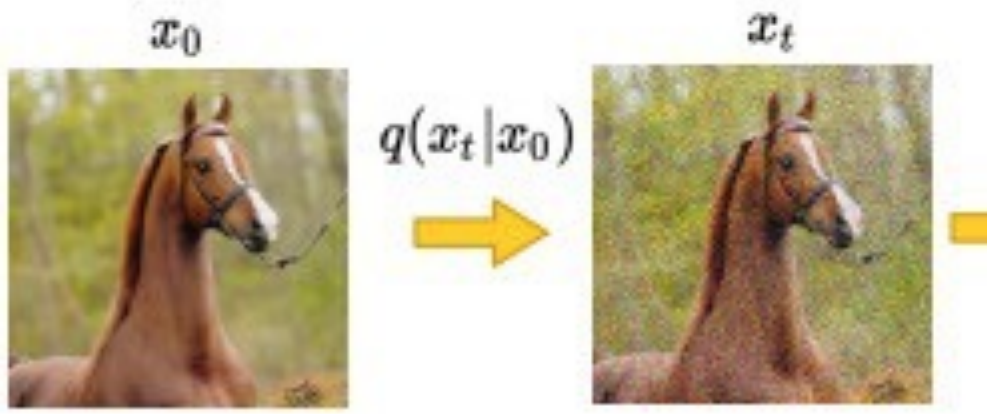


Choi et al., ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models, ICCV 2021

Semantic Segmentation

Label-efficient semantic segmentation with diffusion models

Can we use representation learned from diffusion models for downstream applications such as semantic segmentation?

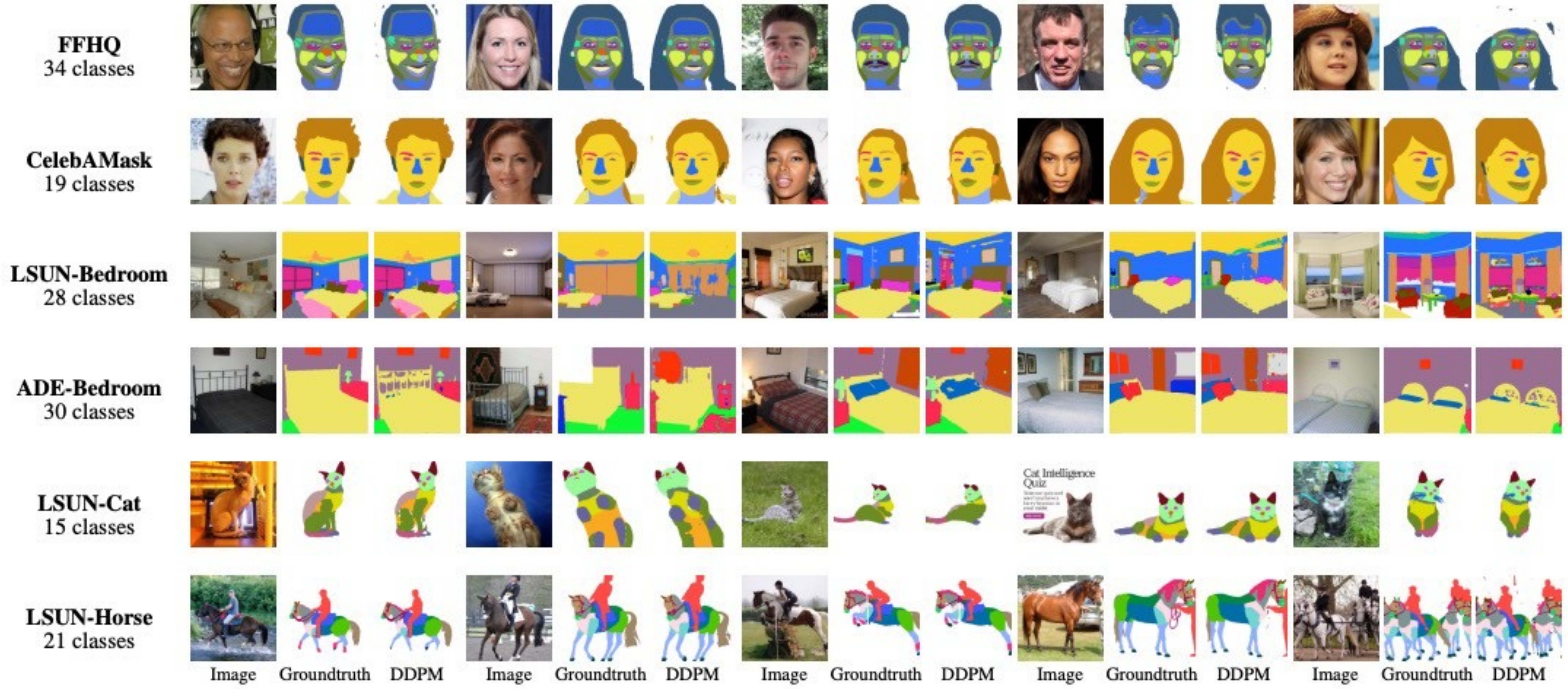


Baranchuk et al., Label-Efficient Semantic Segmentation with Diffusion Models, ICLR 2022

Semantic Segmentation

Label-efficient semantic segmentation with diffusion models

The experimental results show that the proposed method outperforms Masked Autoencoders, GAN and VAE-based models.

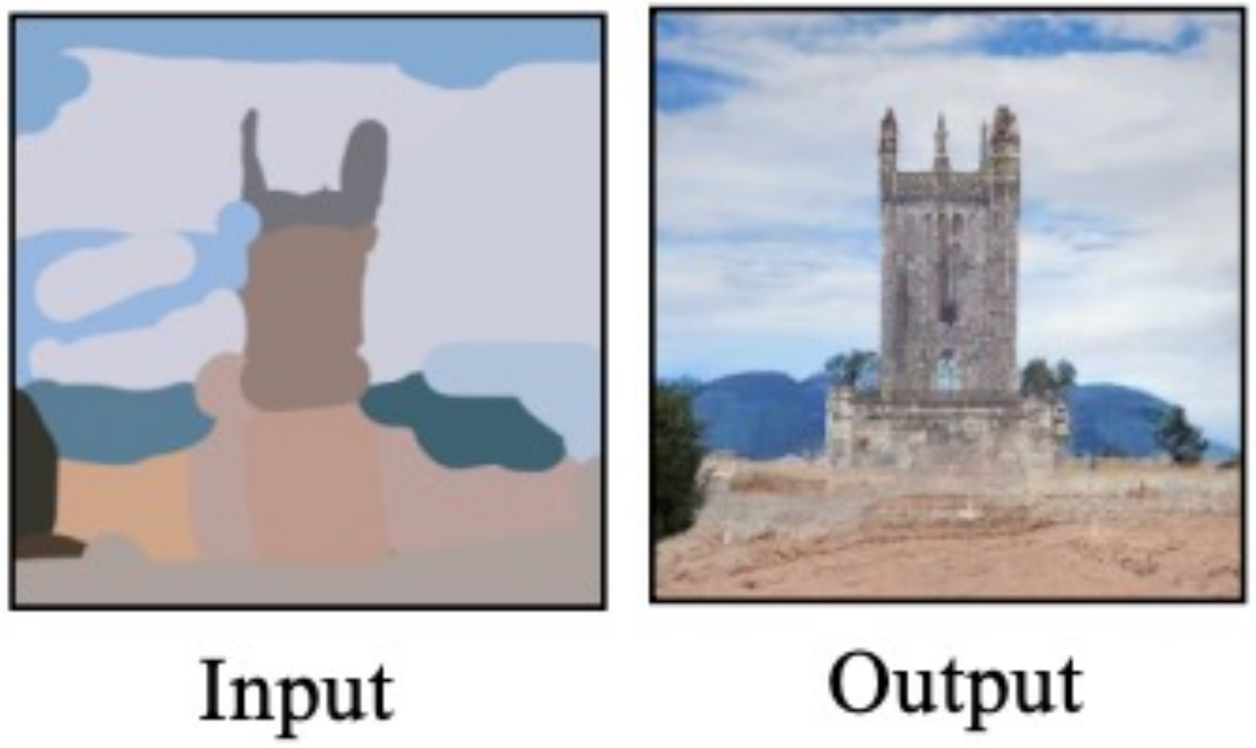


[Baranchuk et al., Label-Efficient Semantic Segmentation with Diffusion Models, ICLR 2022]

Image Editing

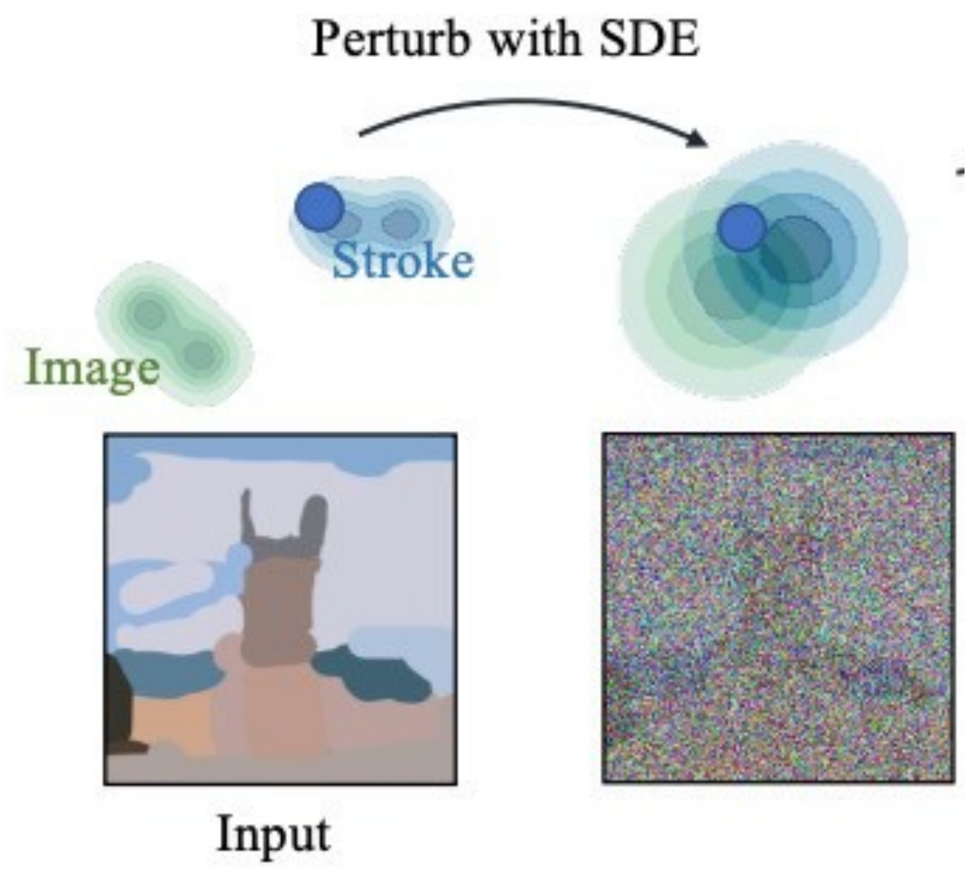
SDEdit

Goal: Given a stroke painting with color, generate a photorealistic image



Key Idea:

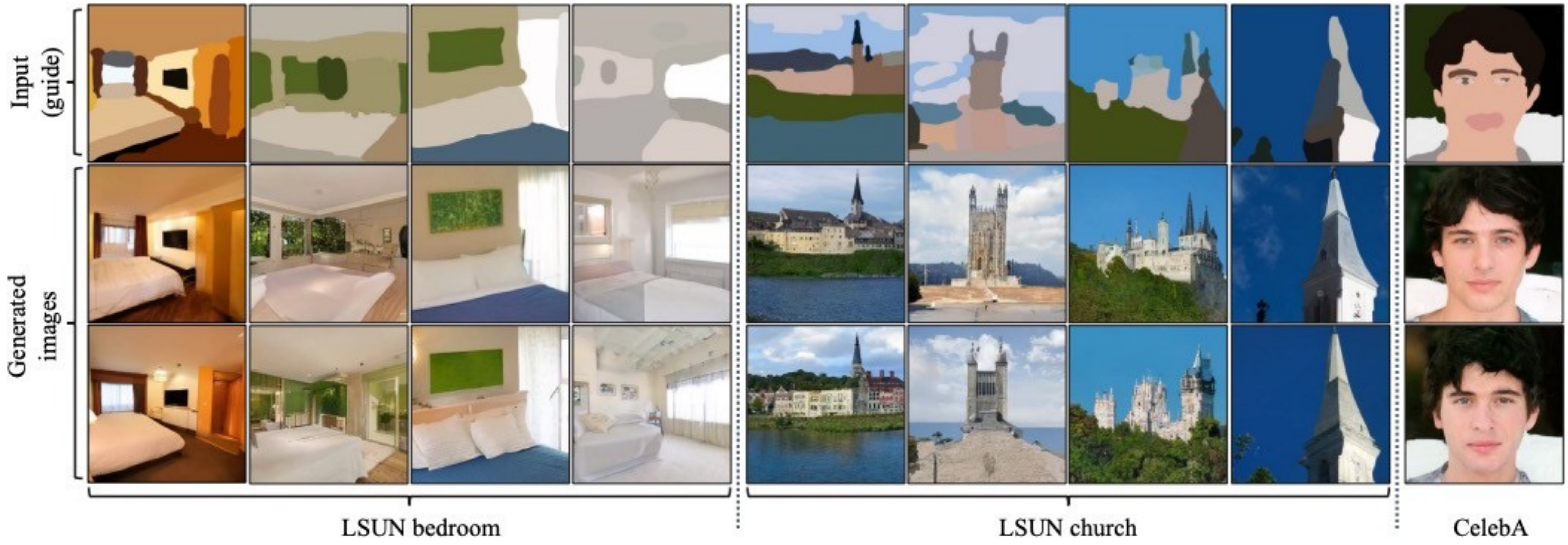
- Latent Distribution of stroke and real images do not overlap.
- But once we apply forward diffusion on them, their distribution start overlapping as finally it becomes gaussian noise.



[Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022]

Image Editing

SDEdit



Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022

Video Synthesis, Medical Imaging, 3D Generation, Discrete State Models

Video Generation

Samples from a text-conditioned video diffusion model, conditioned on the string ***fireworks***.

Ho et al., "Video Diffusion Models", *arXiv*, 2022
Harvey et al., "Flexible Diffusion Modeling of Long Videos", arXiv, 2022

Yang et al., "Diffusion Probabilistic Modeling for Video Generation", *arXiv*, 2022
Höppe et al., "Diffusion Models for Video Prediction and Infilling", arXiv, 2022
Voleti et al., "MCVD: Masked Conditional Video Diffusion for Prediction, Generation, and Interpolation", *arXiv*, 2022

Video Generation

Video Generation Tasks:

- Unconditional Generation (Generate all frames)
- Future Prediction (Generate future from past fames)
- Past Prediction (Generate past from future fames)
- Interpolation (Generate intermediate frames)

➡ Learn a model of the form:

$$p_{\boldsymbol{\theta}}(\mathbf{x}^{t_1}, \cdots, \mathbf{x}^{t_K} | \mathbf{x}^{\tau_1}, \cdots, \mathbf{x}^{\tau_M})$$

Given frames: $\mathbf{x}^{\tau_1}, \cdots, \mathbf{x}^{\tau_M}$

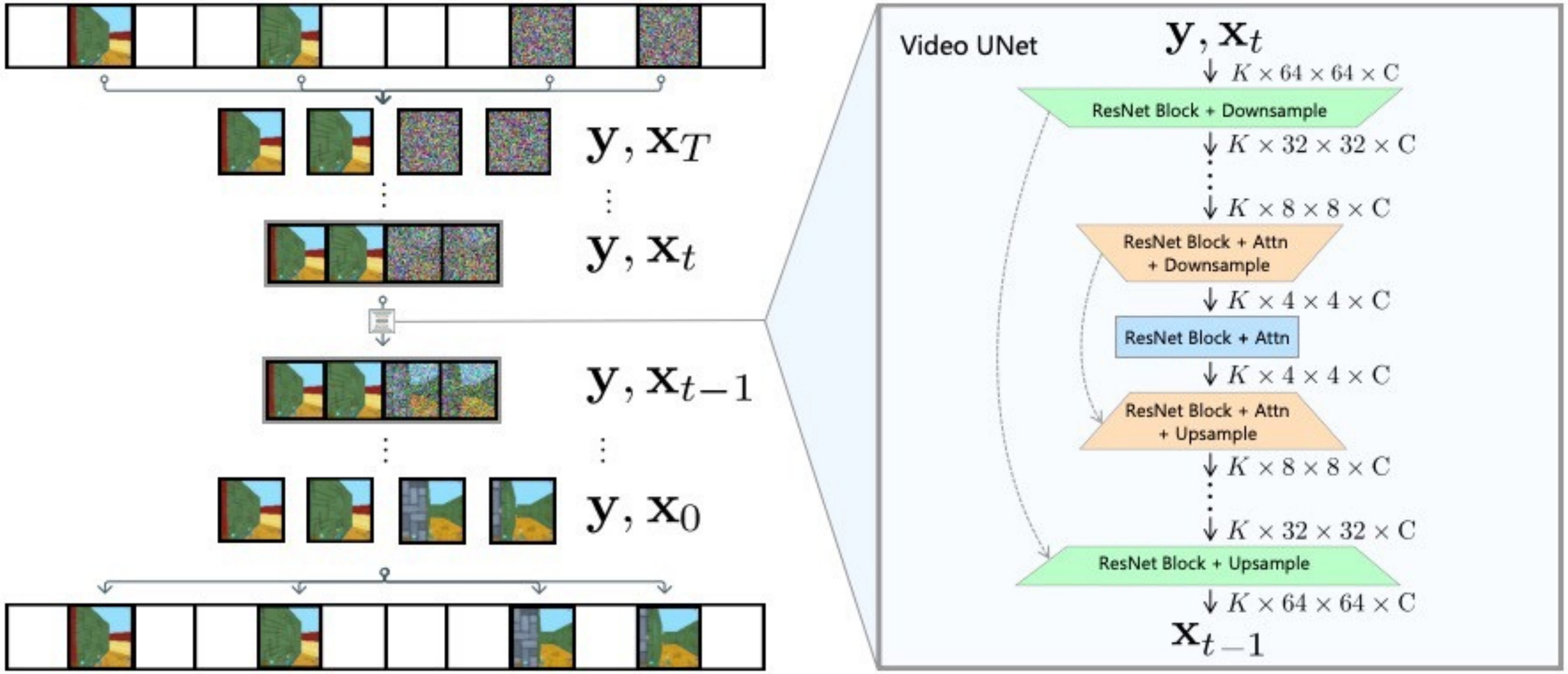
Frames to be predicted: $\mathbf{x}^{t_1}, \cdots, \mathbf{x}^{t_K}$

Ho et al., "Video Diffusion Models", *arXiv*, 2022
Harvey et al., "Flexible Diffusion Modeling of Long Videos", arXiv, 2022
Yang et al., "Diffusion Probabilistic Modeling for Video Generation", *arXiv*, 2022
Höppe et al., "Diffusion Models for Video Prediction and Infilling", arXiv, 2022
Voleti et al., "MCVD: Masked Conditional Video Diffusion for Prediction, Generation, and Interpolation", *arXiv*, 2022

Video Generation

Learn one model for everything:

- Architecture as **one diffusion model** over **all frames concatenated**.
- Mask frames to be predicted; provide conditioning frames; vary applied masking/conditioning for different tasks during training.
- Use **time position encodings** to encode times.



(image from: Harvey et al., "Flexible Diffusion Modeling of Long Videos", *arXiv*, 2022)

Video Generation

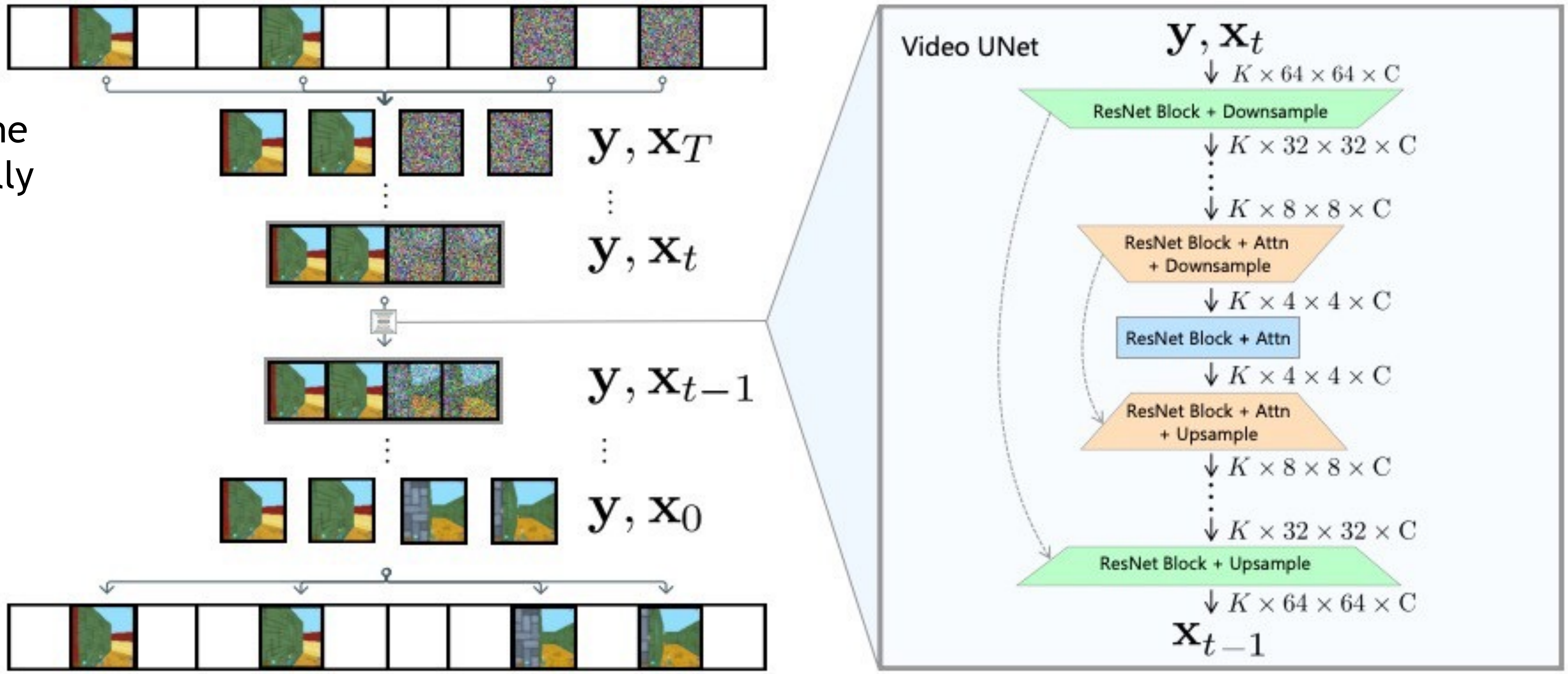
Architecture Details:

Data is 4D (image height, image width, #frames, channels)

- Option (1): 3D Convolutions. Can be computationally expensive.
- Option (2): Spatial 2D Convolutions + Attention Layers along frame axis.

➡ Additional Advantage:

Ignoring the attention layers, the model can be trained additionally on pure image data!



(image from: Harvey et al., “Flexible Diffusion Modeling of Long Videos”, *arXiv*, 2022)

Video Generation

Results

Long term video generation in hierarchical manner:

- 1. Generate future frames in sparse manner, conditioning on frames far back
- 2. Interpolate in-between frames

1+ hour coherent video generation possible!

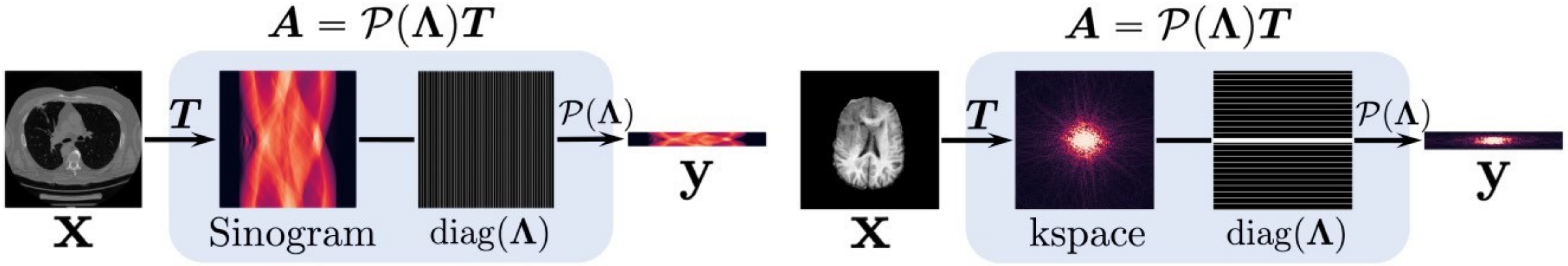
Test Data:

Generated:

(video from: Harvey et al., "Flexible Diffusion Modeling of Long Videos", *arXiv*, 2022, https://plai.cs.ubc.ca/2022/05/20/flexible-diffusion-modeling-of-long-videos/)

Solving Inverse Problems in Medical Imaging

Forward CT or MRI imaging process (simplified):



sparse-view CT undersampled MRI

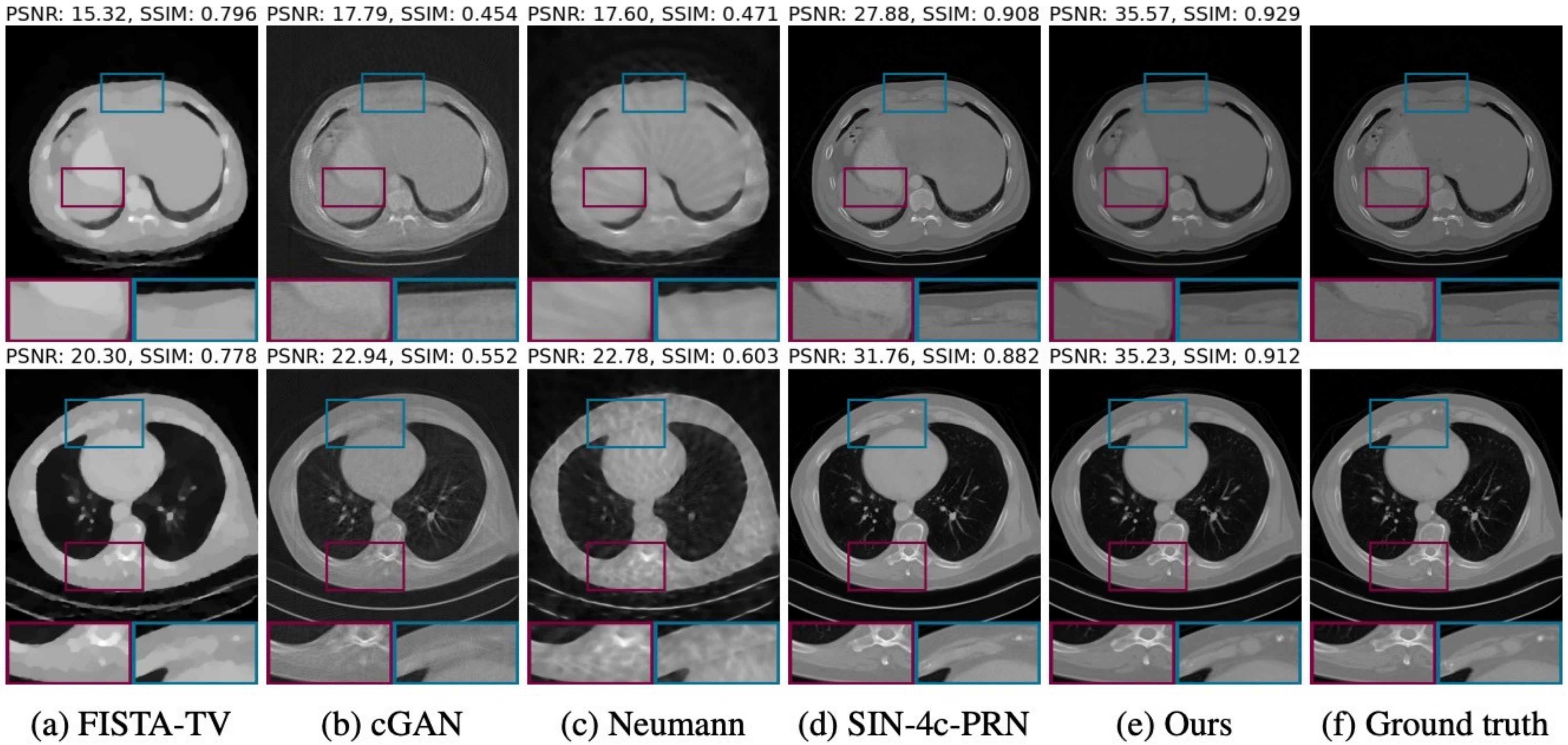
(image from: Song et al., "Solving Inverse Problems in Medical Imaging with Score-Based Generative Models", *ICLR*, 2022)

➡ *Inverse Problem:*
Reconstruct original image from sparse measurements.

[Song et al., "Solving Inverse Problems in Medical Imaging with Score-Based Generative Models", *ICLR*, 2022]

Solving Inverse Problems in Medical Imaging

High-level idea: Learn Generative Diffusion Model as "prior"; then guide synthesis conditioned on sparse observations:



(image from: Song et al., "Solving Inverse Problems in Medical Imaging with Score-Based Generative Models", *ICLR*, 2022)

➡ *Outperforms even fully-supervised methods.*

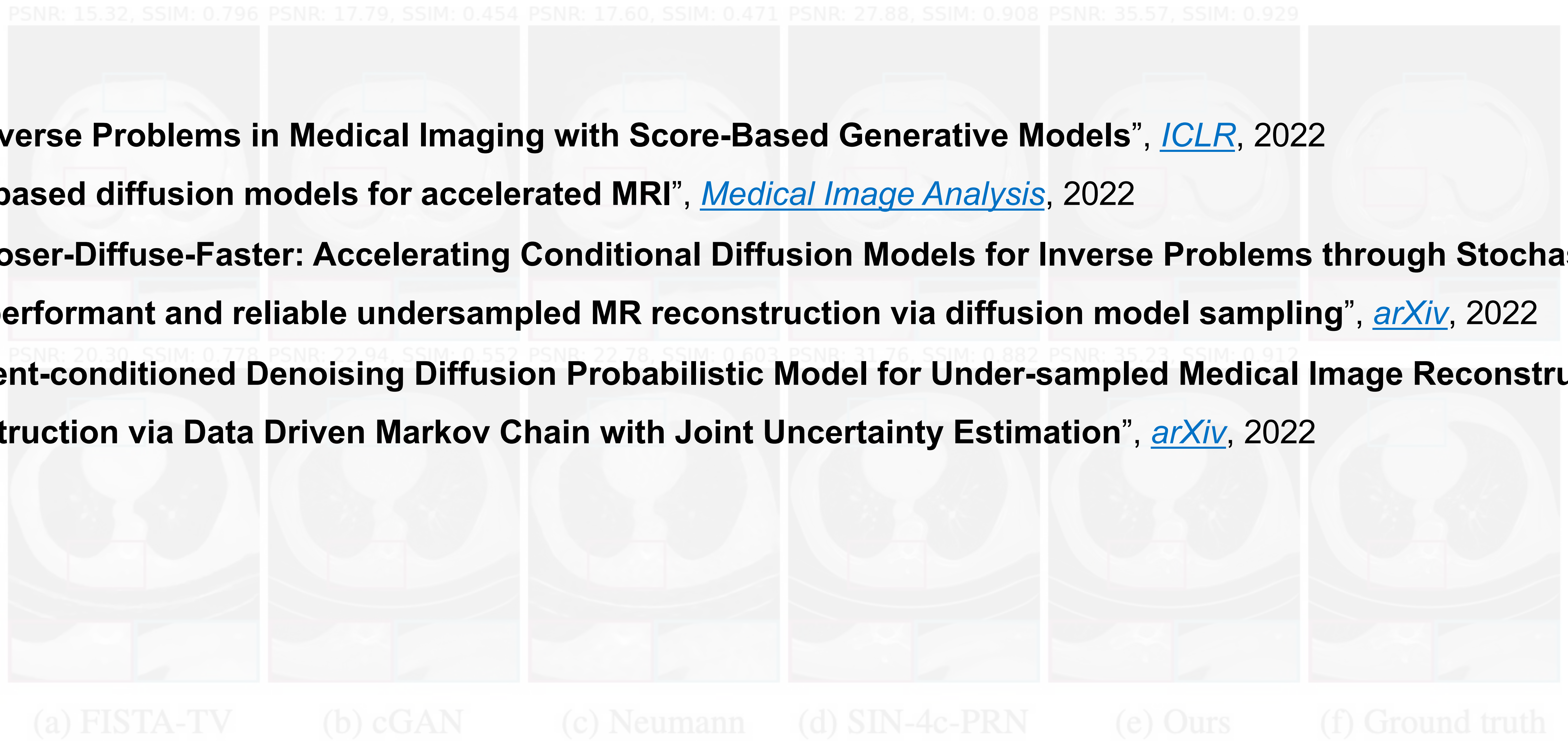
[Song et al., "Solving Inverse Problems in Medical Imaging with Score-Based Generative Models", *ICLR*, 2022]

Solving Inverse Problems in Medical Imaging

Lots of Literature

High-level idea: Learn Generative Diffusion Model as "prior"; then guide synthesis conditioned on sparse observations:

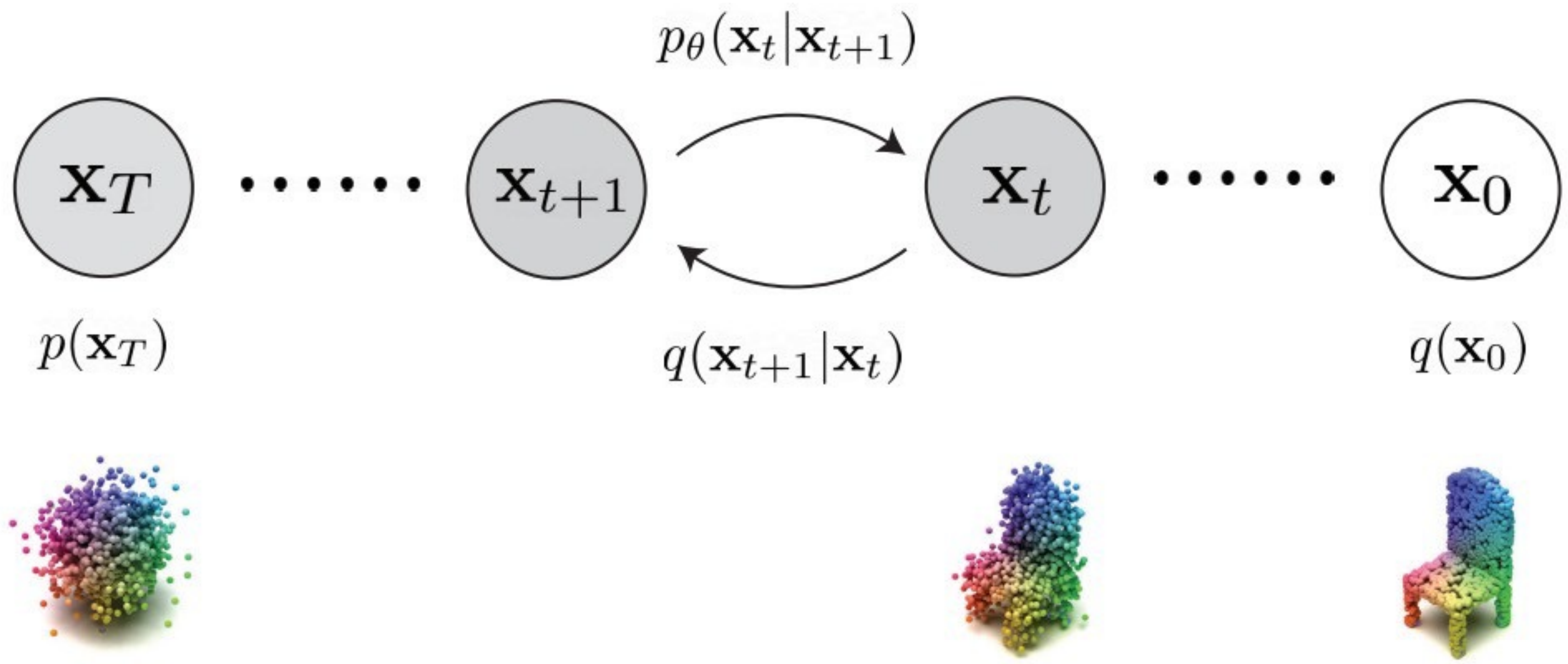
- Song et al., "**Solving Inverse Problems in Medical Imaging with Score-Based Generative Models**", *ICLR*, 2022
- Chung and Ye, "**Score-based diffusion models for accelerated MRI**", *Medical Image Analysis*, 2022
- Chung et al., "**Come-Closer-Diffuse-Faster: Accelerating Conditional Diffusion Models for Inverse Problems through Stochastic Contraction**", *CVPR*, 2022
- Peng et al., "**Towards performant and reliable undersampled MR reconstruction via diffusion model sampling**", *arXiv*, 2022
- Xie and Li, "**Measurement-conditioned Denoising Diffusion Probabilistic Model for Under-sampled Medical Image Reconstruction**", *arXiv*, 2022
- Luo et al, "**MRI Reconstruction via Data Driven Markov Chain with Joint Uncertainty Estimation**", *arXiv*, 2022
- …



(Song et al., "Solving Inverse Problems in Medical Imaging with Score-Based Generative Models", *ICLR*, 2022)

3D Shape Generation

- Point clouds as 3D shape representation can be diffused easily and intuitively
- Denoiser implemented based on modern point cloud-processing networks (PointNets & Point-VoxelCNNs)



(image from: Zhou et al., "3D Shape Generation and Completion through Point-Voxel Diffusion", *ICCV*, 2021)

Zhou et al., "3D Shape Generation and Completion through Point-Voxel Diffusion", *ICCV*, 2021
Luo and Hu, "Diffusion Probabilistic Models for 3D Point Cloud Generation", *CVPR*, 2021

3D Shape Generation

- Point clouds as 3D shape representation can be diffused easily and intuitively
- Denoiser implemented based on modern point cloud-processing networks (PointNets & Point-VoxelCNNs)

(video from: Zhou et al., "3D Shape Generation and Completion through Point-Voxel Diffusion", *ICCV*, 2021, https://alexzhou907.github.io/pvd)

Zhou et al., "3D Shape Generation and Completion through Point-Voxel Diffusion", *ICCV*, 2021

3D Shape Generation

Shape Completion

- Can train conditional shape completion diffusion model (subset of points fixed to given conditioning points):

(video from: Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021, https://alexzhou907.github.io/pvd)

Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021

3D Shape Generation

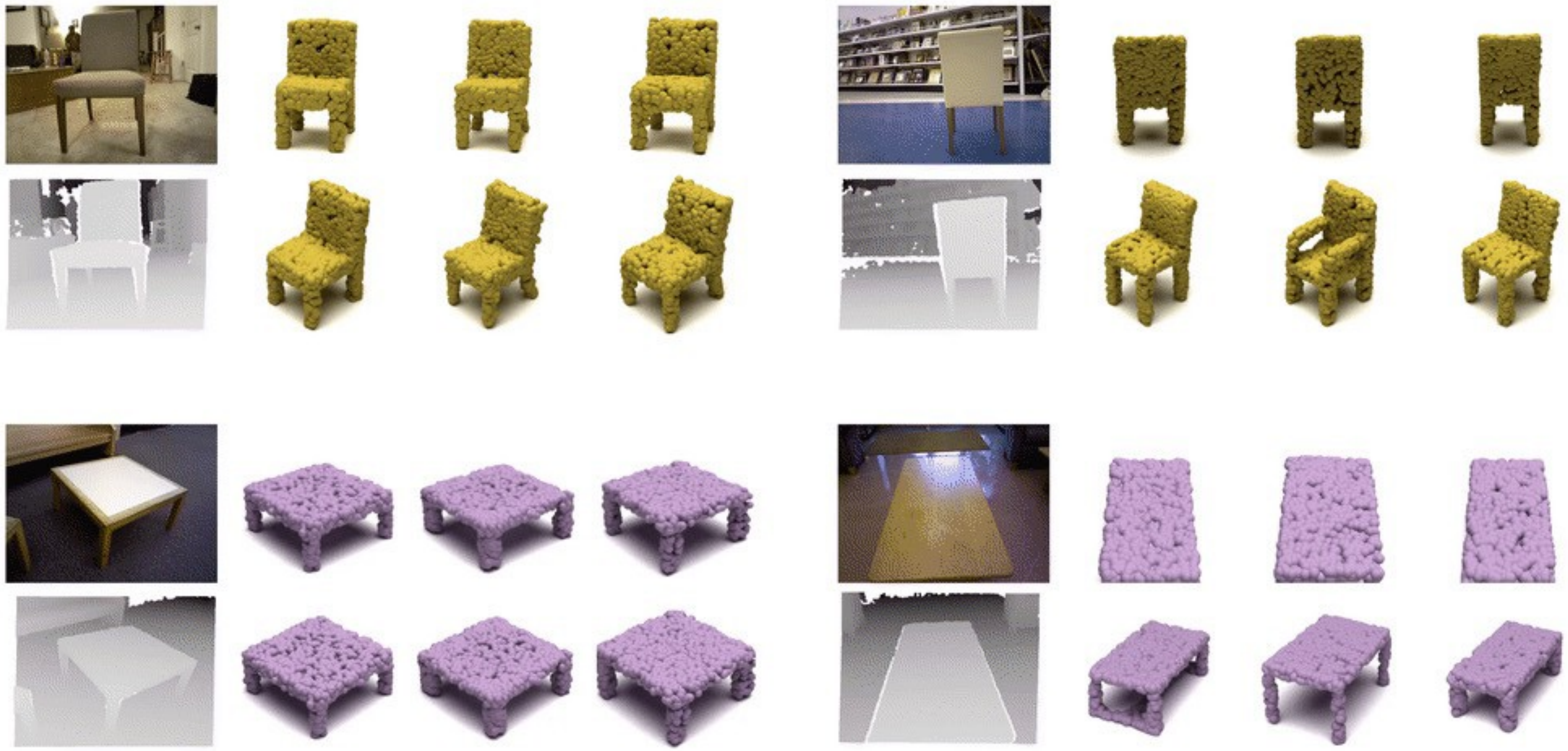
Shape Completion – Multimodality

(video from: Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021, https://alexzhou907.github.io/pvd)

[Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021]

3D Shape Generation

Shape Completion – Multimodality – On Real Data



(video from: Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021, https://alexzhou907.github.io/pvd)

Zhou et al., “3D Shape Generation and Completion through Point-Voxel Diffusion”, *ICCV*, 2021

Towards Discrete State Diffusion Models

- So far:

  Continuous diffusion and denoising processes.

Data

Noise

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

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

➡ *But what if data is discrete? Categorical?*
Continuous perturbations are not possible!

(Text, Pixel-wise Segmentation Labels,
Discrete Image Encodings, etc.)

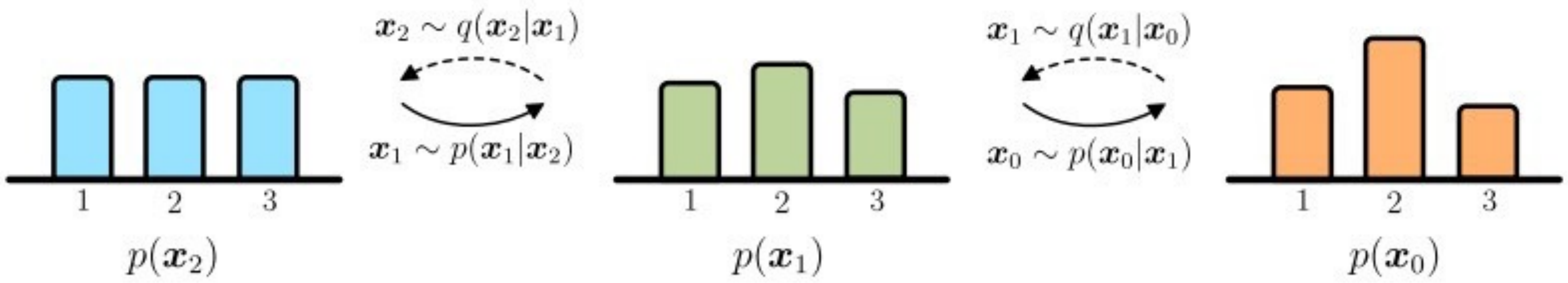
Discrete State Diffusion Models

- **Categorical diffusion:** $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{Cat}(\mathbf{x}_t; \mathbf{p} = \mathbf{x}_{t-1}\mathbf{Q}_t)$

  $\mathbf{x}_t$: one-hot state vector

  $\mathbf{Q}_t$: transition matrix $[\mathbf{Q}_t]_{ij} = q(x_t = j | x_{t-1} = i)$

- Reverse process can be parametrized categorical distribution.



(image from: Hoogeboom et al., "Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions", *NeurIPS*, 2022)

Austin et al., "Structured Denoising Diffusion Models in Discrete State-Spaces", *NeurIPS*, 2021
Hoogeboom et al., "Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions", *NeurIPS*, 2022

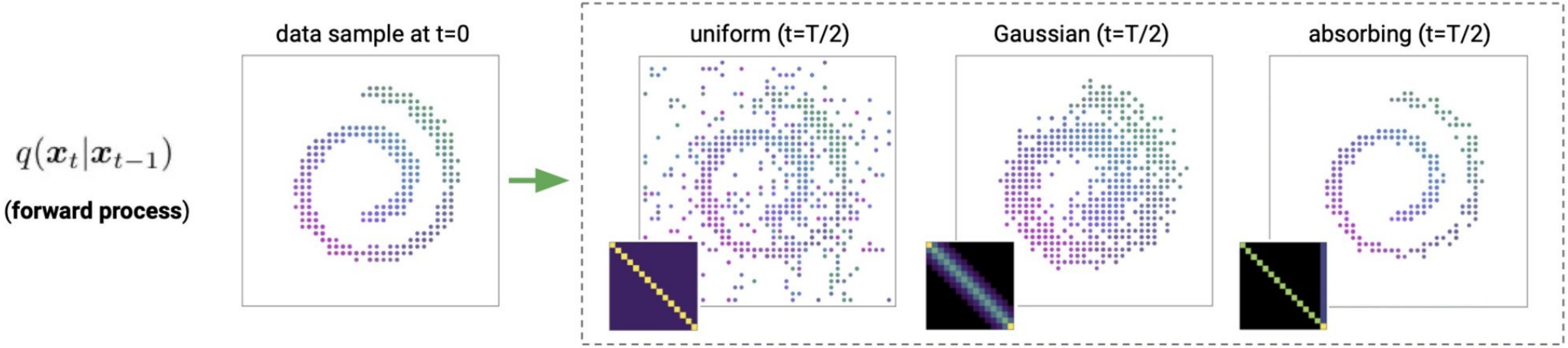
Discrete State Diffusion Models

Options for forward process:

- Uniform categorical diffusion:

$$\mathbf{Q}_t = (1 - \beta_t)\mathbf{I} + \frac{\beta_t}{K}\mathbb{1}\mathbb{1}^\top$$

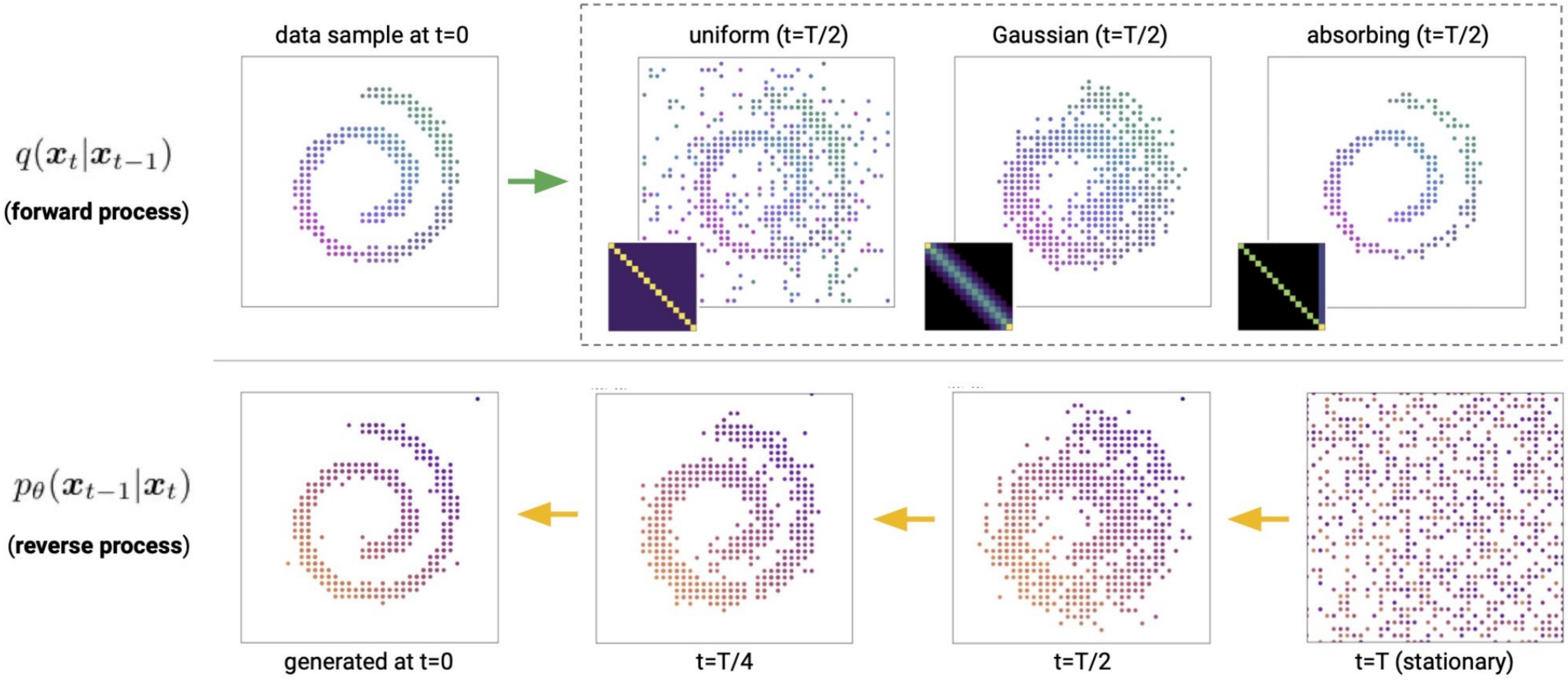
- Progressive masking out of data (generation is "de-masking")
- Tailored to ordinal data (e.g. discretized Gaussian)



(image from: Austin et al., "Structured Denoising Diffusion Models in Discrete State-Spaces", *NeurIPS*, 2021)

Austin et al., "Structured Denoising Diffusion Models in Discrete State-Spaces", *NeurIPS*, 2021

Discrete State Diffusion Models



(image from: Austin et al., "Structured Denoising Diffusion Models in Discrete State-Spaces", *NeurIPS*, 2021)

Austin et al., "Structured Denoising Diffusion Models in Discrete State-Spaces", *NeurIPS*, 2021

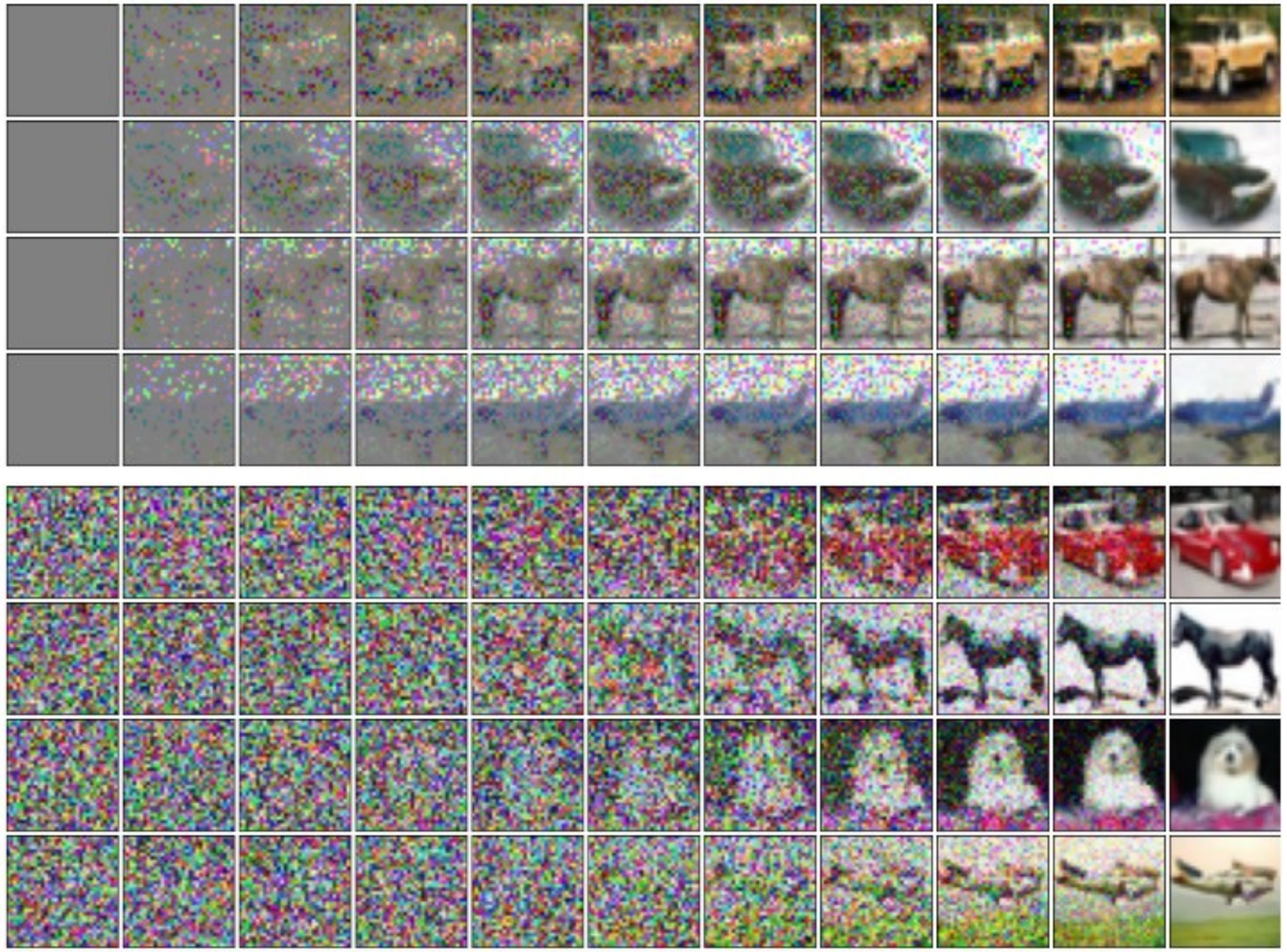
Discrete State Diffusion Models

Modeling Categorical Image Pixel Values

Progressive denoising starting from all-masked state.

Progressive denoising starting from random uniform state.

(with discretized Gaussian denoising model)

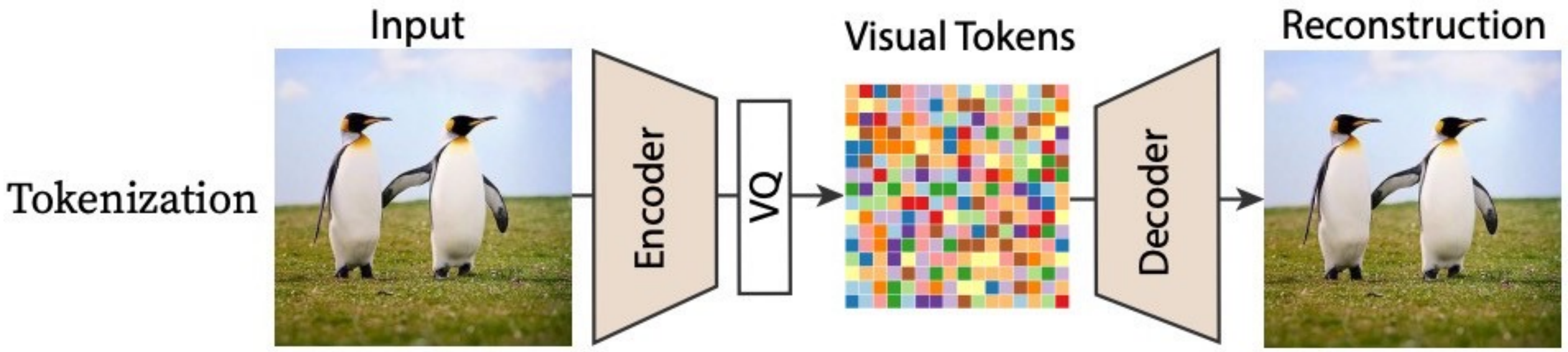


(image from: Austin et al., “Structured Denoising Diffusion Models in Discrete State-Spaces”, *NeurIPS*, 2021)

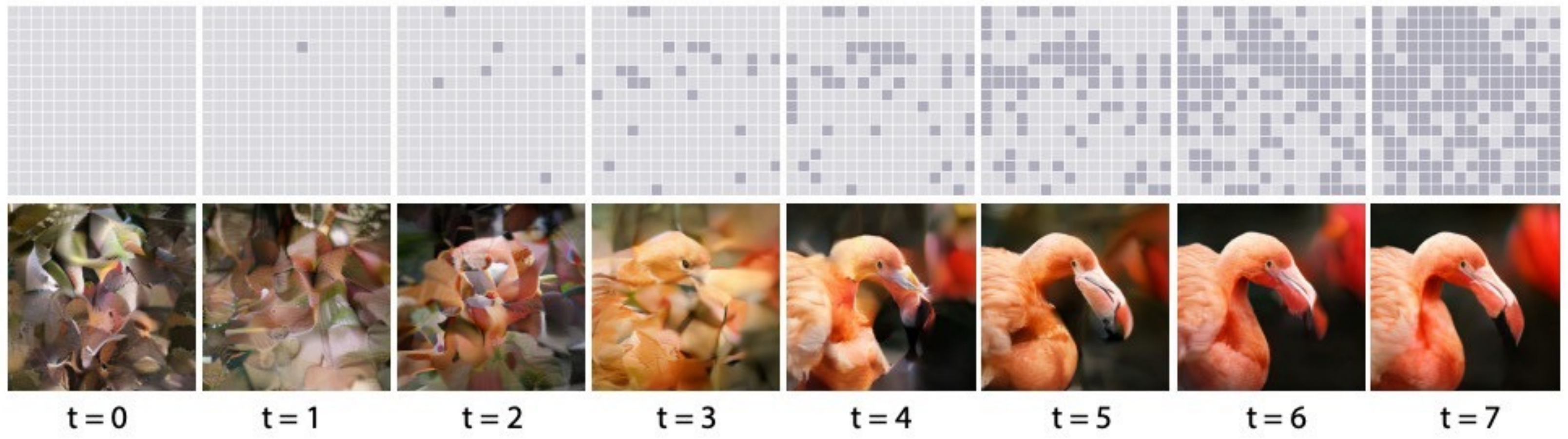
Austin et al., “Structured Denoising Diffusion Models in Discrete State-Spaces”, *NeurIPS*, 2021

Discrete State Diffusion Models

Modeling Discrete Image Encodings



Encoding images into latent space with discrete tokens, and modeling discrete token distribution



Iterative generation

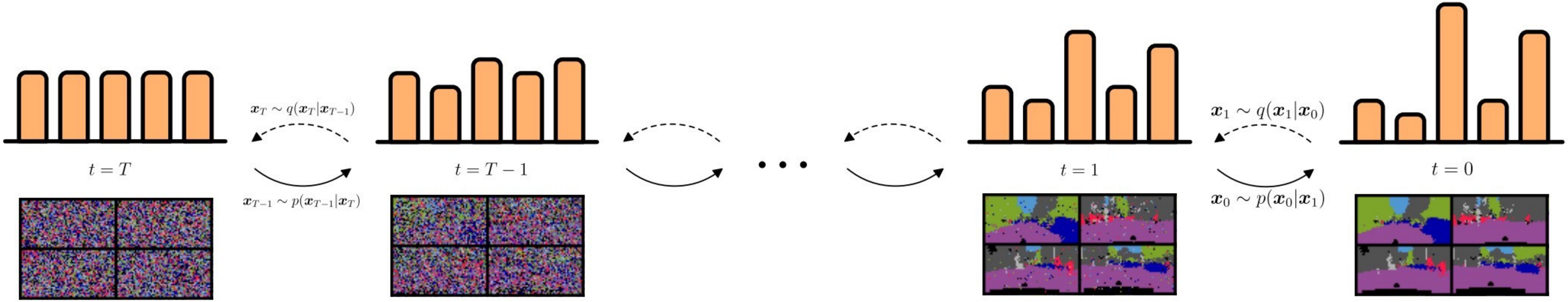
Class-conditional model samples

(images from: Chang et al., "MaskGIT: Masked Generative Image Transformer", *CVPR*, 2022)

Chang et al., "MaskGIT: Masked Generative Image Transformer", *CVPR*, 2022
Esser et al., "ImageBART: Bidirectional Context with Multinomial Diffusion for Autoregressive Image Synthesis", *NeurIPS*, 2021

Discrete State Diffusion Models

Modeling Pixel-wise Segmentations



(image from: Hoogeboom et al., “Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions”, *NeurIPS*, 2022)

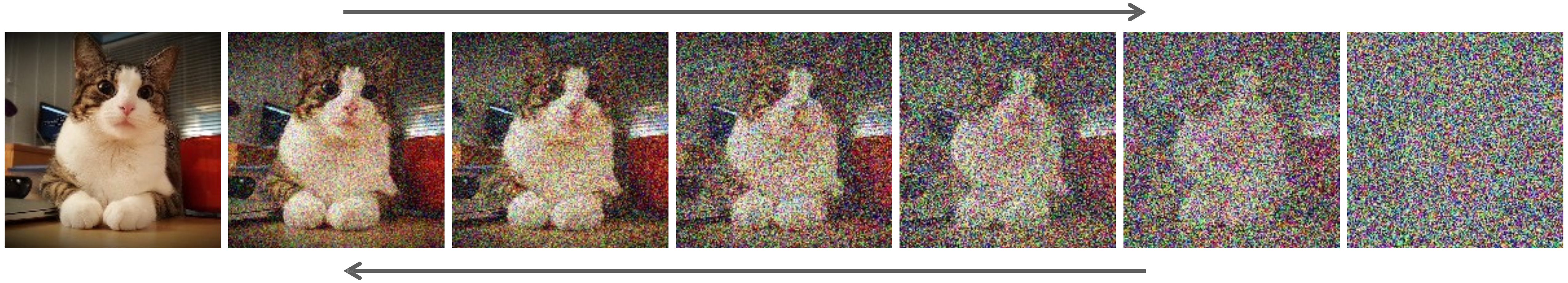
[Hoogeboom et al., “Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions”, *NeurIPS*, 2022]

Conclusions, Open Problems and Final Remarks

Summary: Denoising Diffusion Probabilistic Models

“Discrete-time” Diffusion Models

We started with denoising diffusion probabilistic models:



We showed how the denoising model can be trained by predicting noise injected in each diffused image:

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

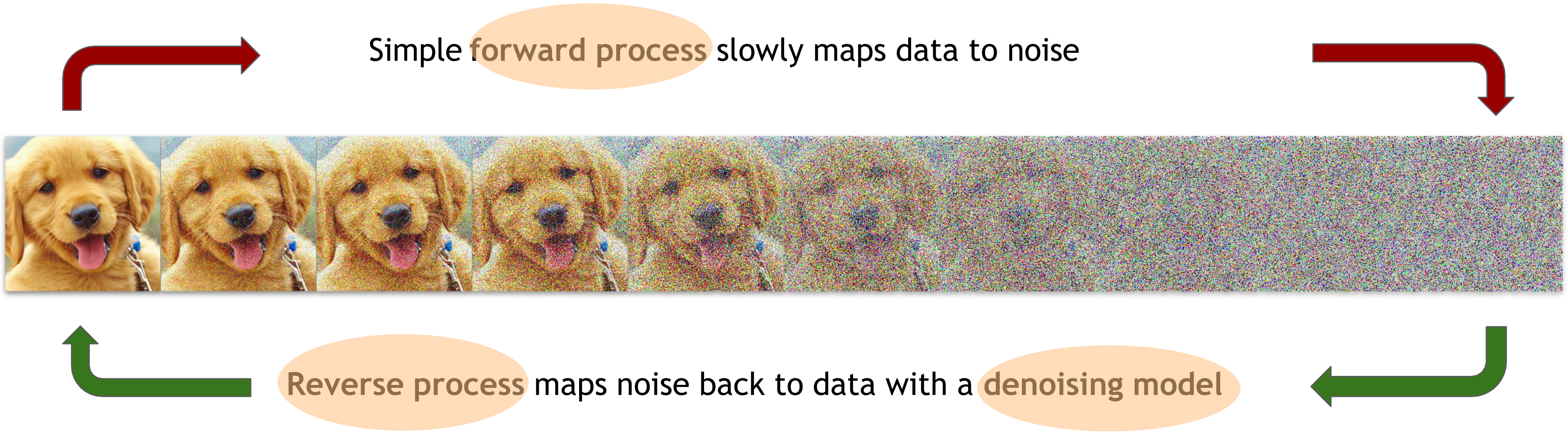
Summary: Advanced Techniques

Acceleration, Guidance and beyond

In the third part, we discussed several advanced topics in diffusion models.

How can we accelerate the sample generation?

[Image credit: Ben Poole, Mohammad Norouzi]



How to scale up diffusion models to high-resolution (conditional) generation?

- Cascaded models
- Guided diffusion models

Summary: Applications

We covered many successful applications of diffusion models:

- Image generation, text-to-image generation, controllable generation
- Image editing, image-to-image translation, super-resolution, segmentation, adversarial robustness
- Discrete models, 3D generation, medical imaging, video synthesis

Open Problems (1)

- Diffusion models are a special form of VAEs and continuous normalizing flows
    - Why do diffusion models perform so much better than these models?
    - How can we improve VAEs and normalizing flows with lessons learned from diffusion models?

- Sampling from diffusion models is still slow especially for interactive applications
    - The best we could reach is 4-10 steps. How can we have one step samplers?
    - Do we need new diffusion processes?

- **Diffusion models can be considered as latent variable models, but their latent space lacks semantics**
    - **How can we do latent-space semantic manipulations in diffusion models**

Open Problems (2)

- How can diffusion models help with discriminative applications?
  - Representation learning (high-level vs low-level)
  - Uncertainty estimation
  - Joint discriminator-generator training

- What are the best network architectures for diffusion models?
  - Can we go beyond existing U-Nets?
  - How can we feed the time input and other conditioning?
  - How can we improve the sampling efficiency using better network designs?

Open Problems (3)

- How can we apply diffusion models to other data types?
    - 3D data (e.g., distance functions, meshes, voxels, volumetric representations), video, text, graphs, etc.
    - How should we change diffusion models for these modalities?

- **Compositional and controllable generation**
    - **How can we go beyond images and generate scenes?**
    - **How can we have more fine-grained control in generation?**

- Diffusion models for X
    - Can we better solve applications that were previously addressed by GANs and other generative models?
    - Which applications will benefit most from diffusion models?

Thanks!

https://cvpr2022-tutorial-diffusion-models.github.io/

@karsten_kreis

@RuiqiGao

@ArashVahdat