Introduction to Deep Generative Modeling

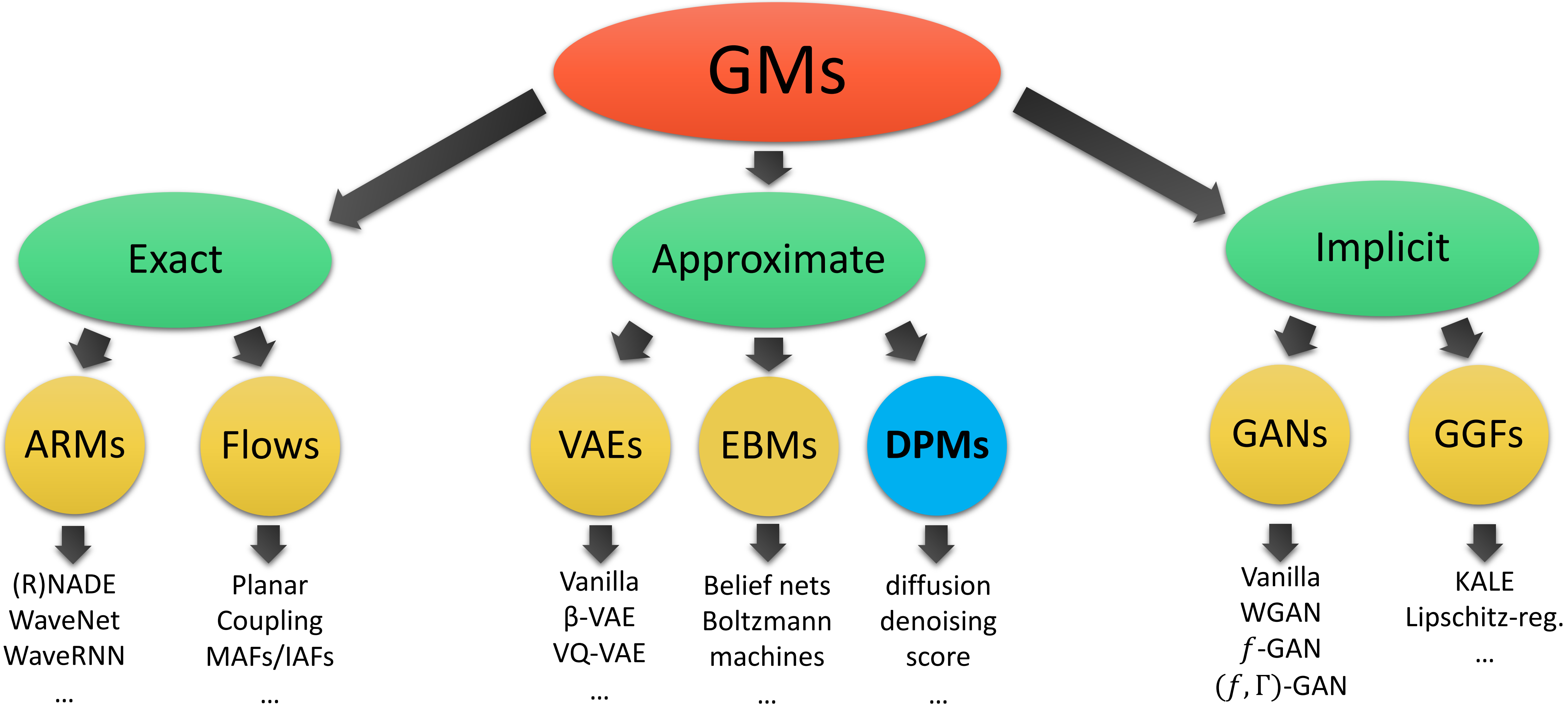
Lecture #14

HY-673 – Computer Science Dep., University of Crete

Professors: Yannis Pantazis, Yannis Stylianou

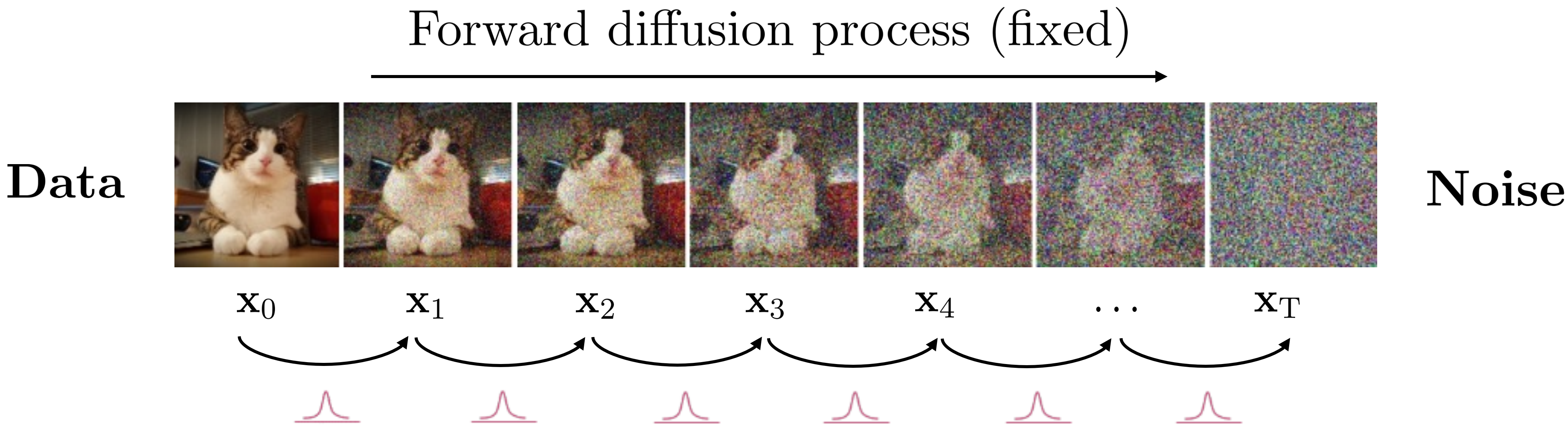
Teaching Assistant: Michail Raptakis

Taxonomy of GMs



Recap: Forward Diffusion Process

The forward diffusion process:

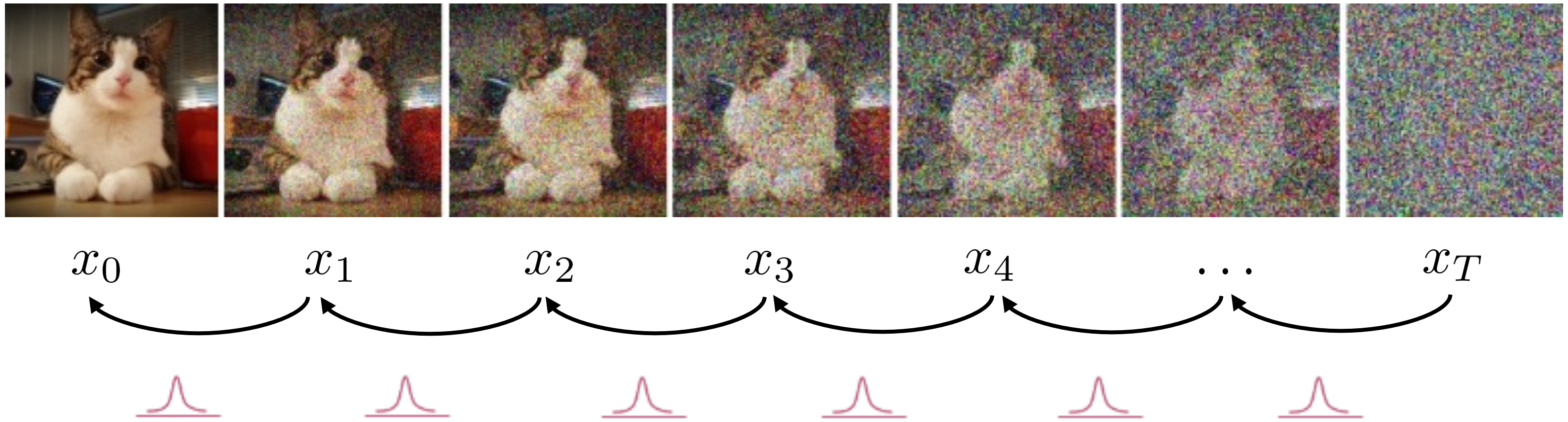


$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

Recap: Reverse Denoising Process

The formal definition of the reverse process in T steps:

Reverse Denoising Process (generative)

Data  **Noise**

$$p(x_T) = \mathcal{N}(x_T; 0, I_d)$$

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \underbrace{\mu_\theta(x_t, t)}_{\substack{\text{Trainable network} \\ \text{(U-net, Denoising Autoencoder)}}}, \sigma_t^2 I_d) \quad \Rightarrow \quad p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t).$$

$\approx q(x_{t-1}|x_t)$ (true posterior; intractable)

Recap: Training and Sampling

Minimize a simplification of negative ELBO:

$$\mathrm{L}_{\text{simple}} = \mathbb{E}_{x_0 \sim p_d(x_0), \epsilon \sim \mathcal{N}(0, I_d), t \sim \mathcal{U}(1, \mathrm{T})} \left[||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon}_{x_t}, t)||^2 \right].$$

Algorithm 1 Training

repeat
$x_0 \sim p_d(x_0)$
$t \sim \text{Uniform}(1, \ldots, T)$
$\epsilon \sim \mathcal{N}(0, I_d)$
Take gradient descent step on

$$\boxed{\nabla_\theta ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t)||^2}$$

until converged

Algorithm 2 Sampling

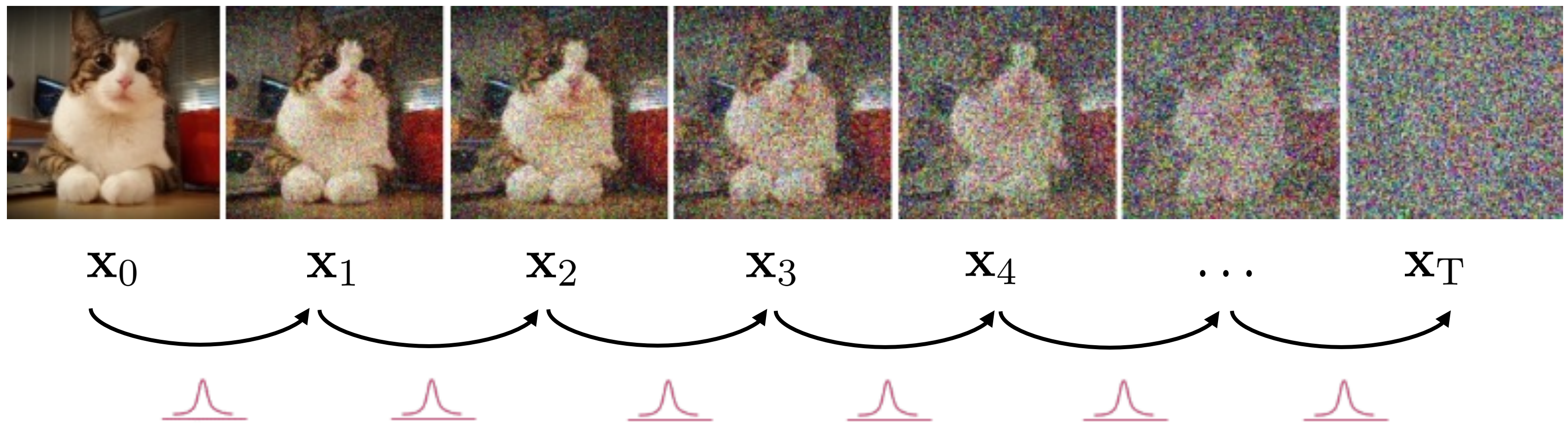
$x_\mathrm{T} \sim \mathcal{N}(0, I_d)$
for $t = T, \ldots, 1$ **do**
  $z \sim \mathcal{N}(0, I_d)$
  $\boxed{x_{t-1} = \frac{1}{\sqrt{1-\beta_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t)\right) + \sigma_t z}$
end for
return x_0

Forward Diffusion Process Limit

Consider the limit of **many small steps:**

Forward diffusion process (fixed)

Data



Noise

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

$$\begin{aligned} x_t &= \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t. && (z_t \sim \mathcal{N}(0, I)) \\ &= \sqrt{1-\beta(t)\Delta t}x_{t-1} + \sqrt{\beta(t)\Delta t}z_t && (\beta_t := \beta(t)\Delta t) \\ &\approx x_{t-1} - \frac{\beta(t)\Delta(t)}{2}x_{t-1} + \sqrt{\beta(t)\Delta t}z_t. && \text{(Taylor expansion)} \end{aligned}$$

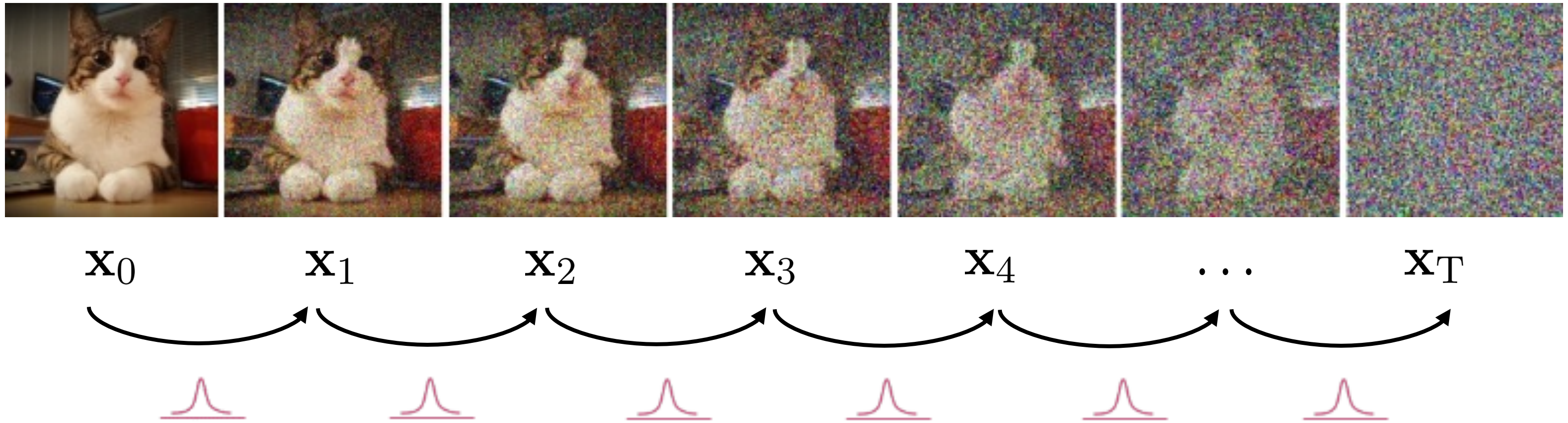
https://arxiv.org/abs/2011.13456

Forward Diffusion Process as an SDE

Consider the limit of **many small steps:**

Forward diffusion process (fixed)

Data



Noise

$$x_t - x_{t-1} \approx \frac{\beta(t)\Delta(t)}{2} x_{t-1} + \sqrt{\beta(t)\Delta t} z_t$$

Stochastic Differential Equation (SDE)

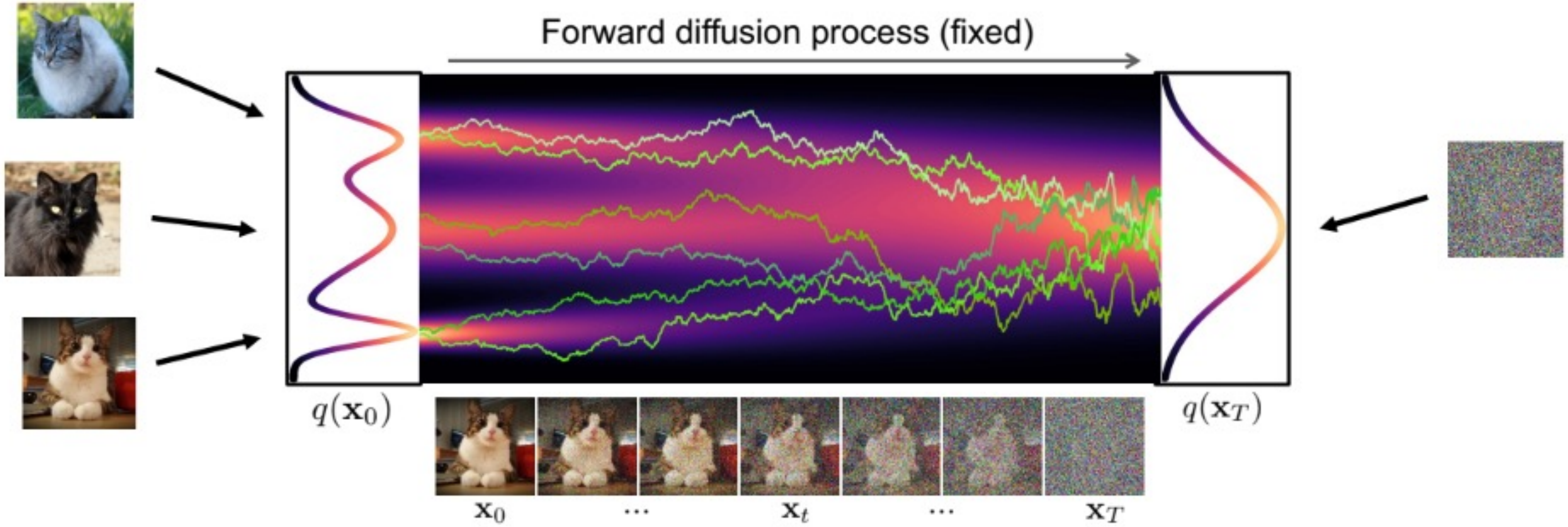
describing the diffusion process in the infinitesimal limit

$$dx_t = -\frac{1}{2}\beta(t) x_t dt + \sqrt{\beta(t)} dW_t$$

W_t: Weiner process

https://arxiv.org/abs/2011.13456

Forward Diffusion Process as an SDE

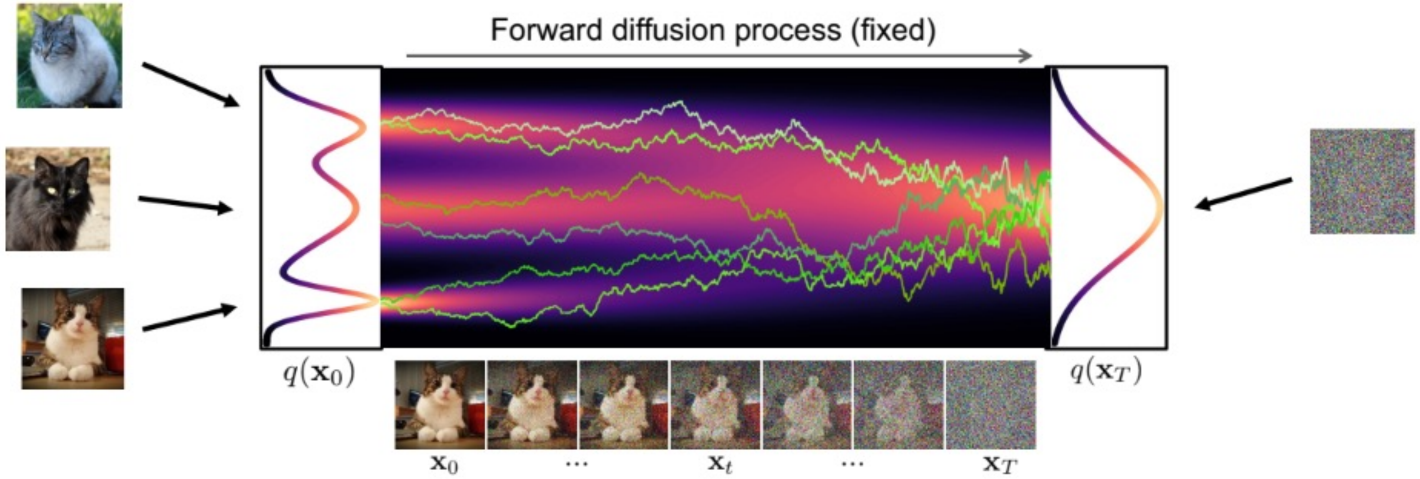


Forward Diffusion SDE:

$$dx_t = \underbrace{-\frac{1}{2}\beta(t)x_t dt}_{\substack{\text{drift term}\\\text{(pulls towards mode)}}} + \underbrace{\sqrt{\beta(t)}dW_t}_{\substack{\text{diffusion term}\\\text{(injects noise)}}}.$$

https://arxiv.org/abs/2011.13456

Denoising Score Matching



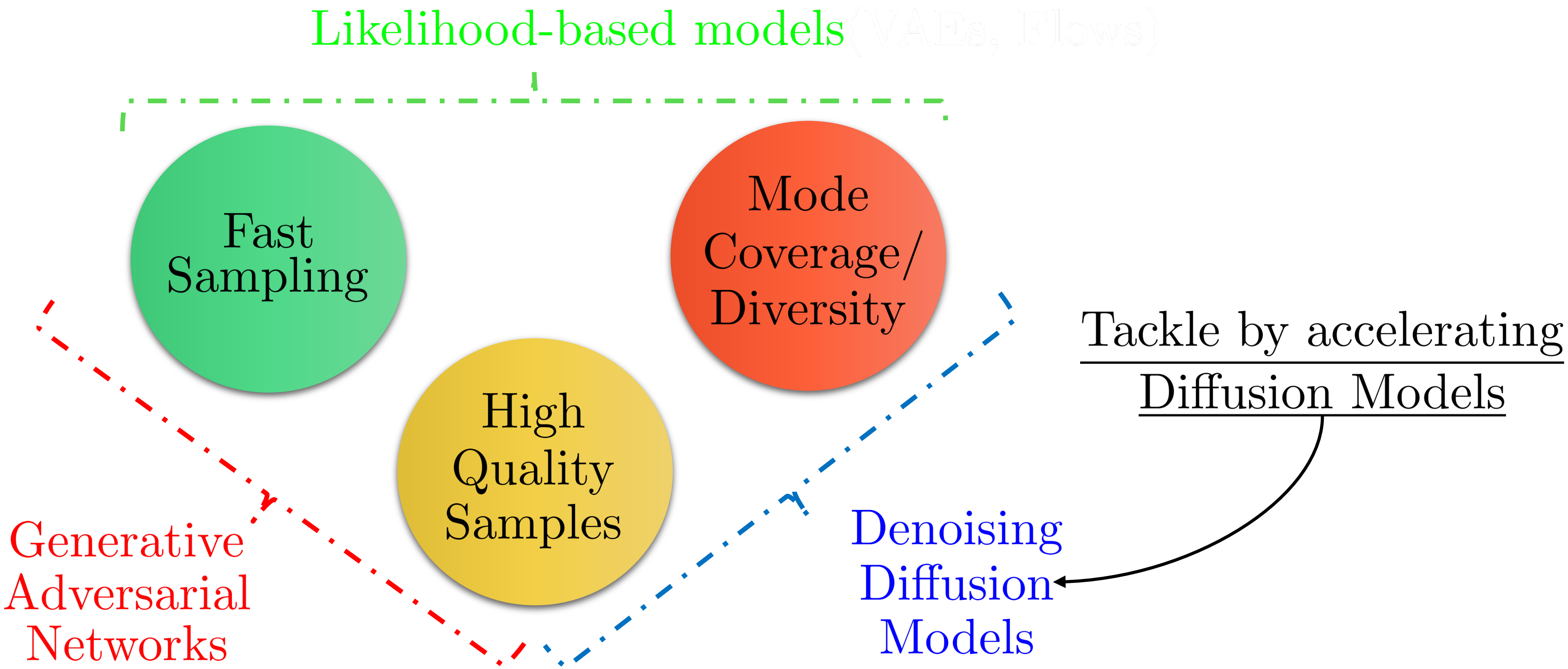
- Instead, diffuse individual data points x_0. Conditional $q_t(x_t|x_0)$ ***is*** tractable!
- **Denoising Score Matching:**

$$\min_\theta \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{x_0\sim q_0(x_0)}}_{\text{diffused data } x_0} \underbrace{\mathbb{E}_{x_t\sim q_t(x_t|x_0)}}_{\text{diffused data sample } x_t|x_0} || \underbrace{s_\theta(x_t,t)}_{\text{neural network}} - \underbrace{\nabla_x \log q_t(x_t|x_0)}_{\text{score of diffused data sample}} ||_2^2.$$

⇨ **After expectations,** $s_\theta(x_t,t) \approx \nabla_x \log q_t(x_t)$**!**

https://ieeexplore.ieee.org/document/6795935
https://arxiv.org/abs/1907.05600
https://arxiv.org/abs/2011.13456

The Generative Learning Trilemma



How to Accelerate Diffusion Models

- Naive acceleration methods: Reduce diffusion time steps in training every k-th time step in inference.
  Unfortunately, it leads to immediate worse performance.
- We need something more clever.
- Given a limited number of functional calls, usually much less than 1000, how to improve the performance?

Advanced Forward Process

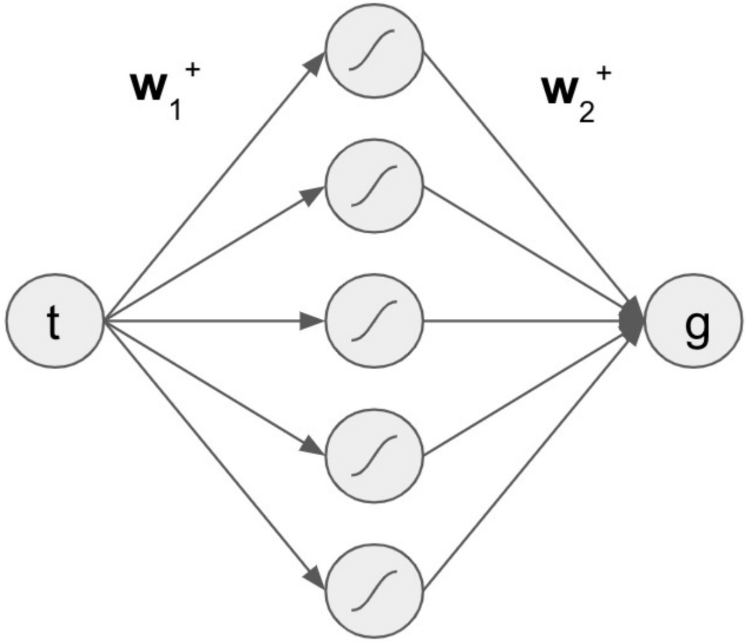
- Does the noise schedule have to be predefined?
- Does it have to be a Markovian process?
- Is there any faster mixing diffusion process?

Variational Diffusion Models: Learnable Diffusion Process

- Given the forward process $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I))$:
- Directly parametrize the variance through a learned function

$$1 - \bar{\alpha}_t = \text{sigmoid}(\gamma_\eta(t)).$$

- $\gamma_\eta(t)$: A monotonic MLP.
  - Strictly positive weights & monotonic activations (e.g., sigmoid).
- Analogous to hierarchical VAE: unlike diffusion models using a fixed encoder, we include learnable parameters in the encoder.



Variational Diffusion Models: New Parametrization of Training Objectives

- Optimizing variational upper bound of diffusion models can be simplified to the following training objective:

$$\mathcal{L}_T = \frac{T}{2}\mathbb{E}_{x_0,\epsilon,t}\left[(\exp(\gamma_\eta(t) - \gamma_\eta(t-1)) - 1)||\epsilon - \epsilon_\theta(x_t, t)||_2^2\right]$$

  - Learning noise schedule improves likelihood estimation of diffusion models, given fewer diffusion steps.

Kingma et al., "Variational diffusion models", NeurIPS 2021.

Variational Diffusion Models: New Parametrization of Training Objectives

- Letting $T \to \infty$ leads to variational upper bound in continuous time:

$$\mathcal{L}_\infty = \frac{1}{2}\mathbb{E}_{x_0,\epsilon,t}\left[\gamma'_\eta ||\epsilon - \epsilon_\theta(x_t, t)||_2^2\right], \ \gamma'_\eta(t) = \frac{d}{dt}\gamma_\eta(t).$$

  - It is shown to be only related to the signa-to-noise ratio (SNR):

$$\mathrm{SNR}(t) = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t} = \exp(-\gamma_\eta(t)).$$

    at endpoints, invariant to the noise schedule in-between.
  - The continuous time noise schedule can be learned to minimize the variance of the training objective for faster training.

Kingma et al., “Variational diffusion models”, NeurIPS 2021.

Variational Diffusion Models: SOTA Likelihood Estimation

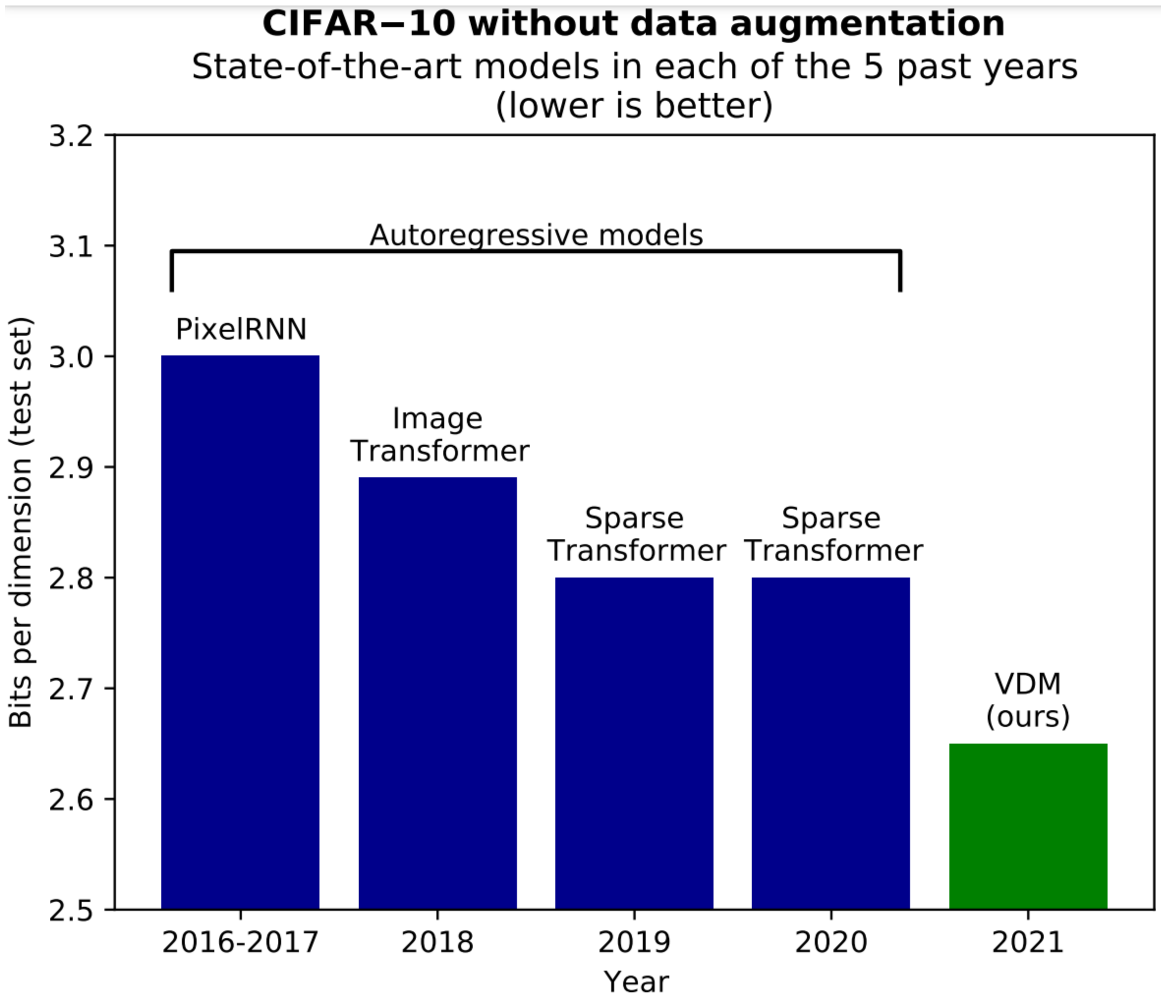
- Key factor: Appending Fourier features to the input of U-Net:

$$f_{i,j,k}^{n} = \sin\left(x_{i,j,k} 2^{n} \pi\right), \; g_{i,j,k}^{n} = \cos\left(x_{i,j,k} 2^{n} \pi\right), \; n = 7, 8.$$

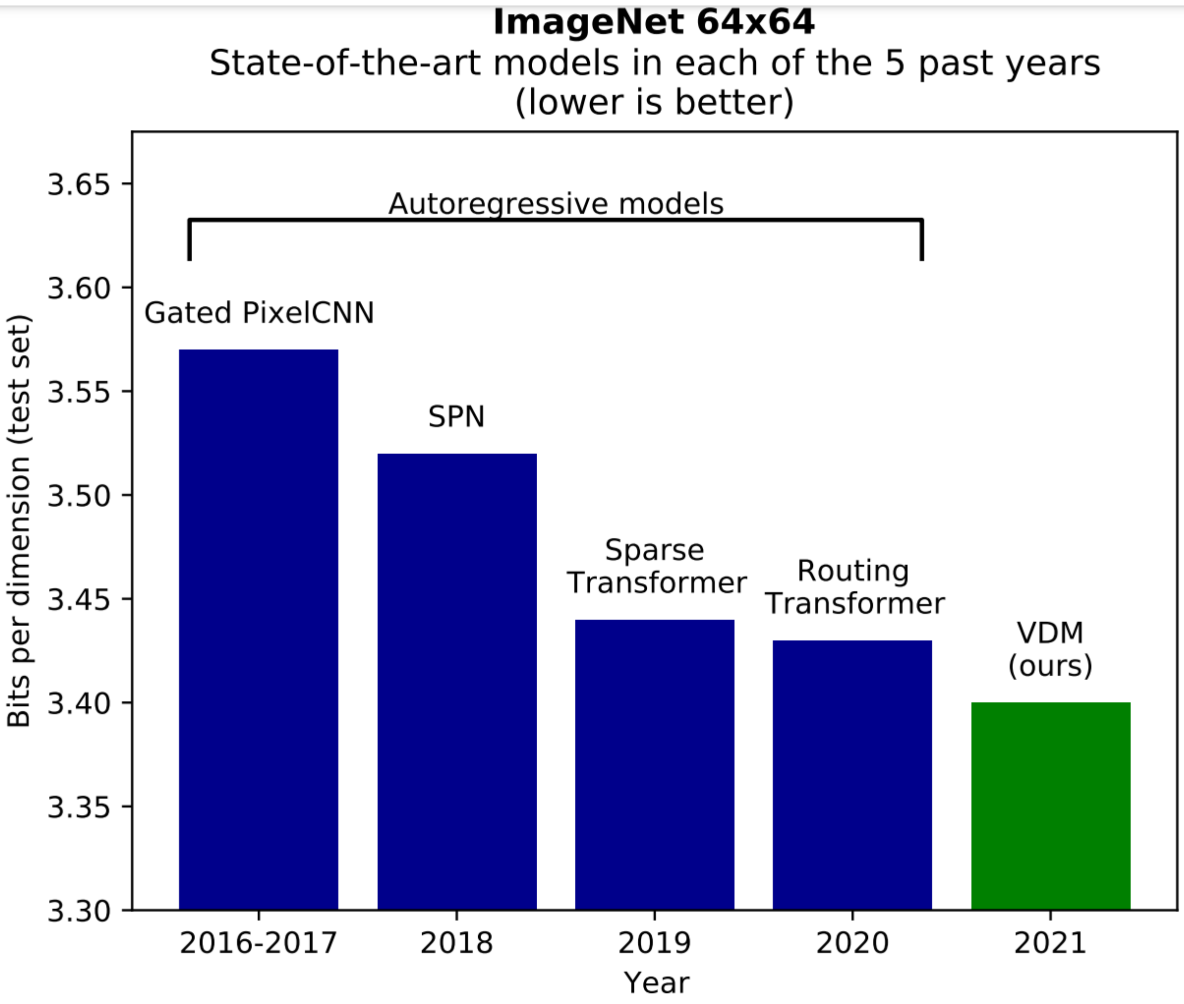
- Good likelihoods require modeling all bits, even the ones corresponding to very small changes in the input.
- But: Neural Networkds are usually bad at modeling small changes to the inputs.
- Significant improvements in log-likelihoods.

Kingma et al., "Variational diffusion models", NeurIPS 2021.

Variational Diffusion Models: SOTA Likelihood Estimation



(a) CIFAR-10 without data augmentation

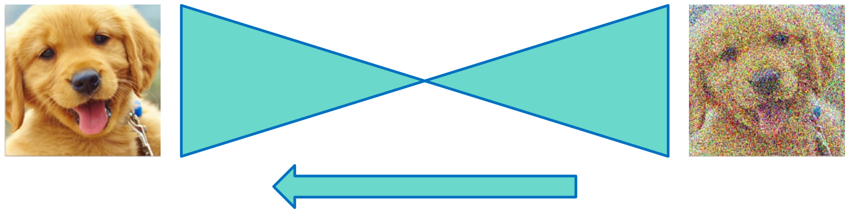


(b) ImageNet 64x64

Kingma et al., “Variational diffusion models”, NeurIPS 2021.

Advanced Reverse Process: Approximating transition probabilities with more complicated distributions

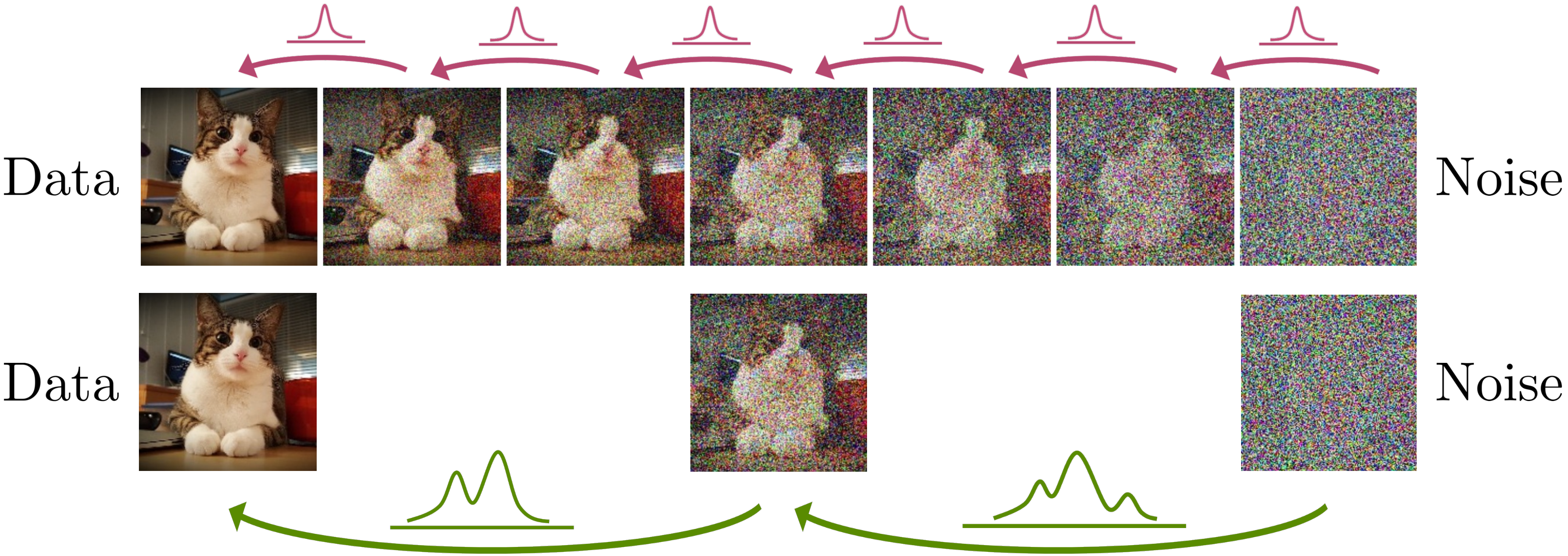
Reverse process maps noise back to data
where the diffusion model is trained



- Question: Is normal approximation of the reverse process still accurate when there are less diffusion time steps?

Advanced Reverse Process: Normal Assumption in Denoising Distribution Holds Only for Small Step

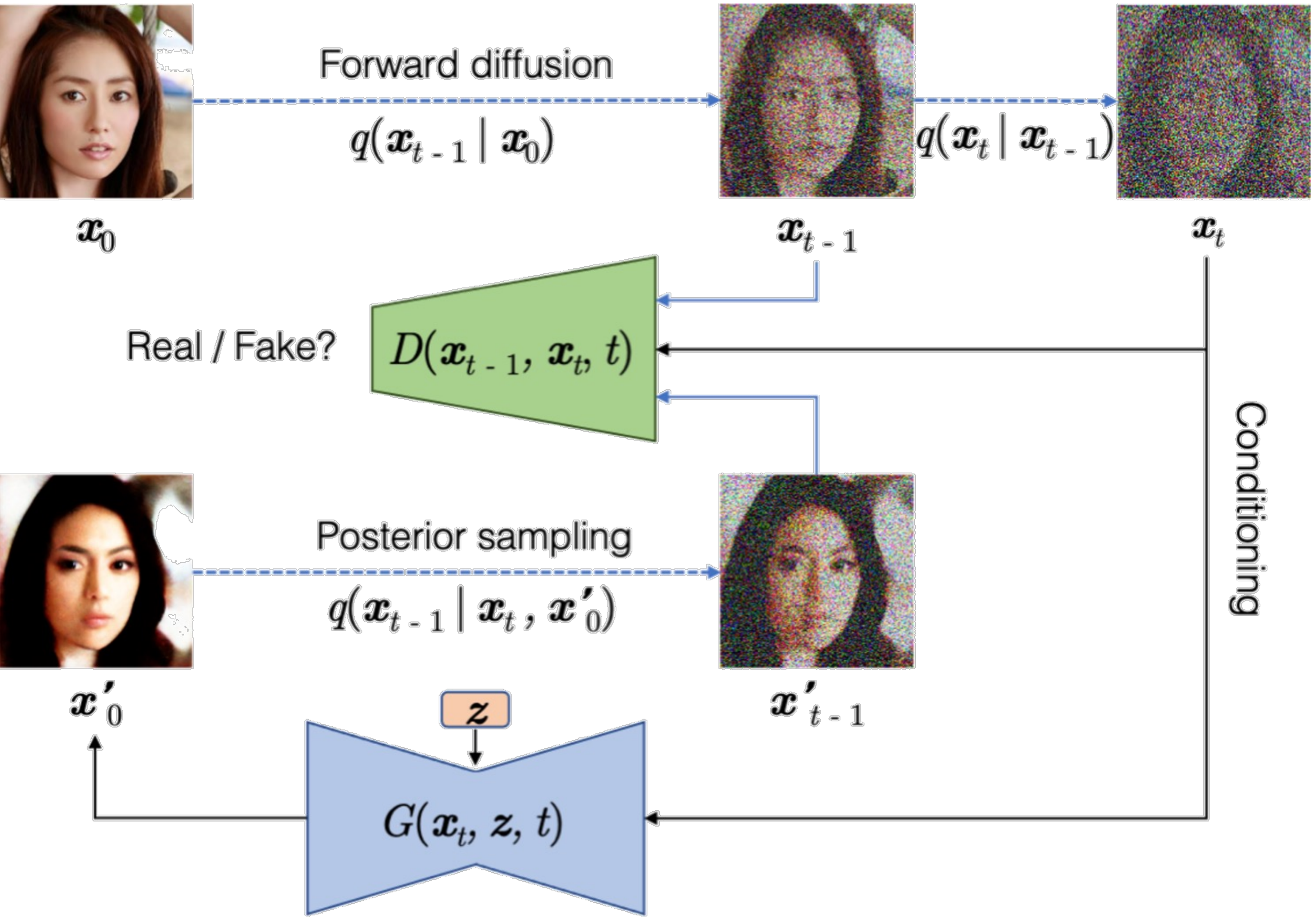
- Denoising Process with unimodal normal distribution:



- Requires more complicated functional approximators!

Denoising Diffusion GANs: Approximating Reverse Process by Conditional GANs

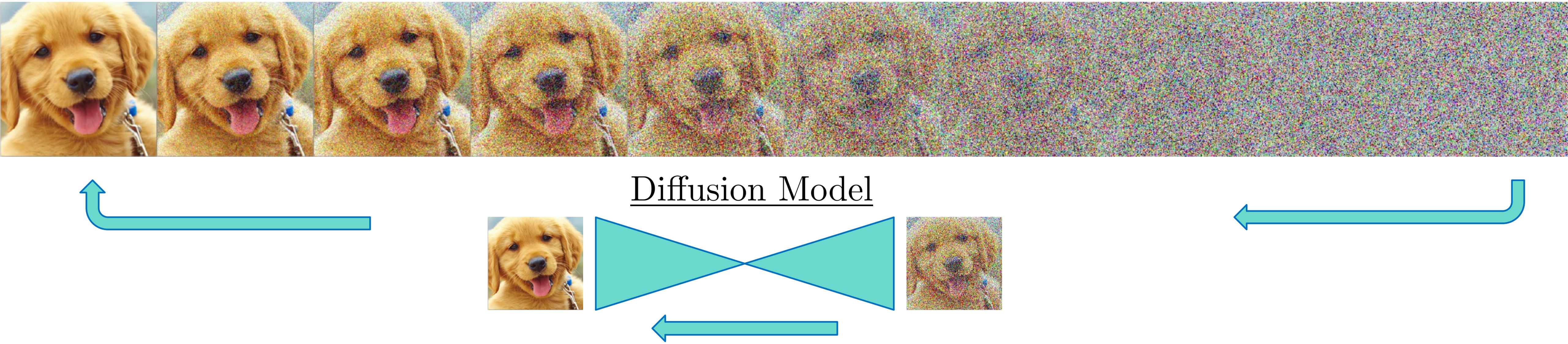
$$\min_{\theta} \sum_{t \geq 1} \mathbb{E}_{q(x_t)} \left[D_{\mathrm{adv}} \left(q(x_{t-1}|x_t) || p_\theta(x_{t-1}|x_t) \right) \right].$$



Compared to a one-shot GAN generator:

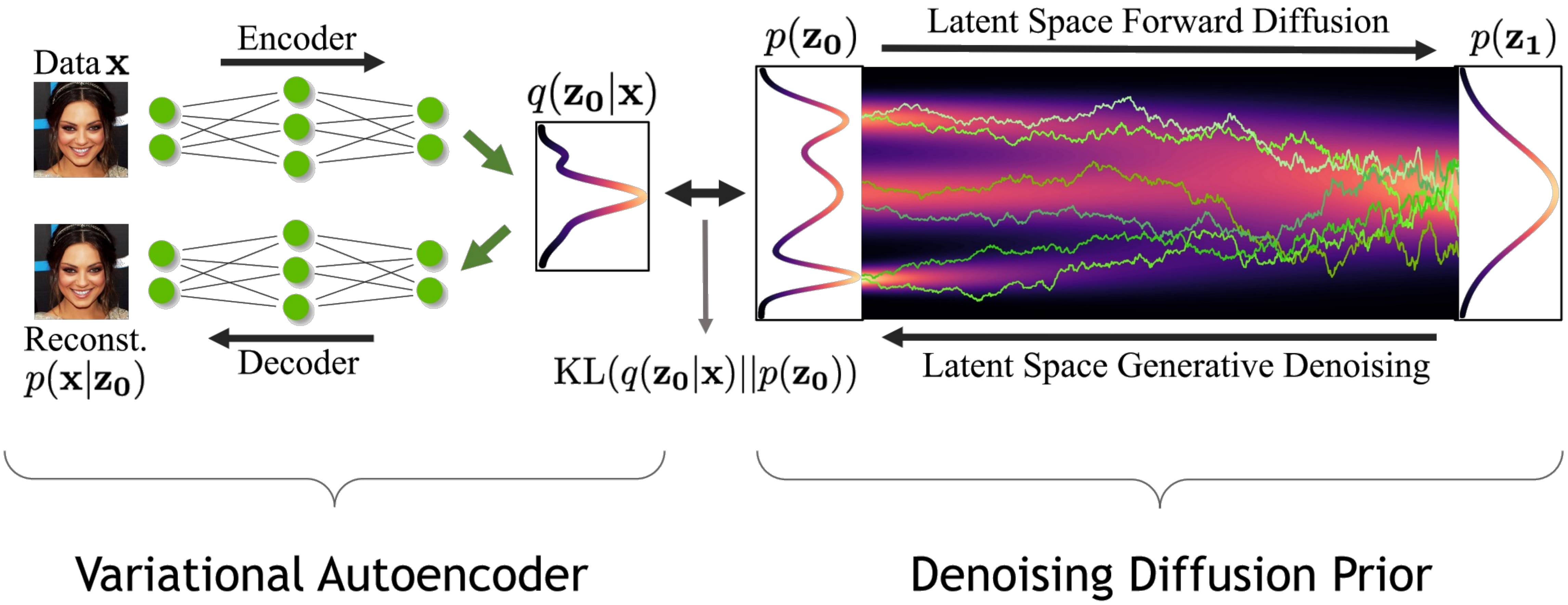
- Both generator and discriminator are solving a much simpler problem.
- Stronger mode coverage.
- Better training stability.

Advanced Modeling: Latent Space Modeling



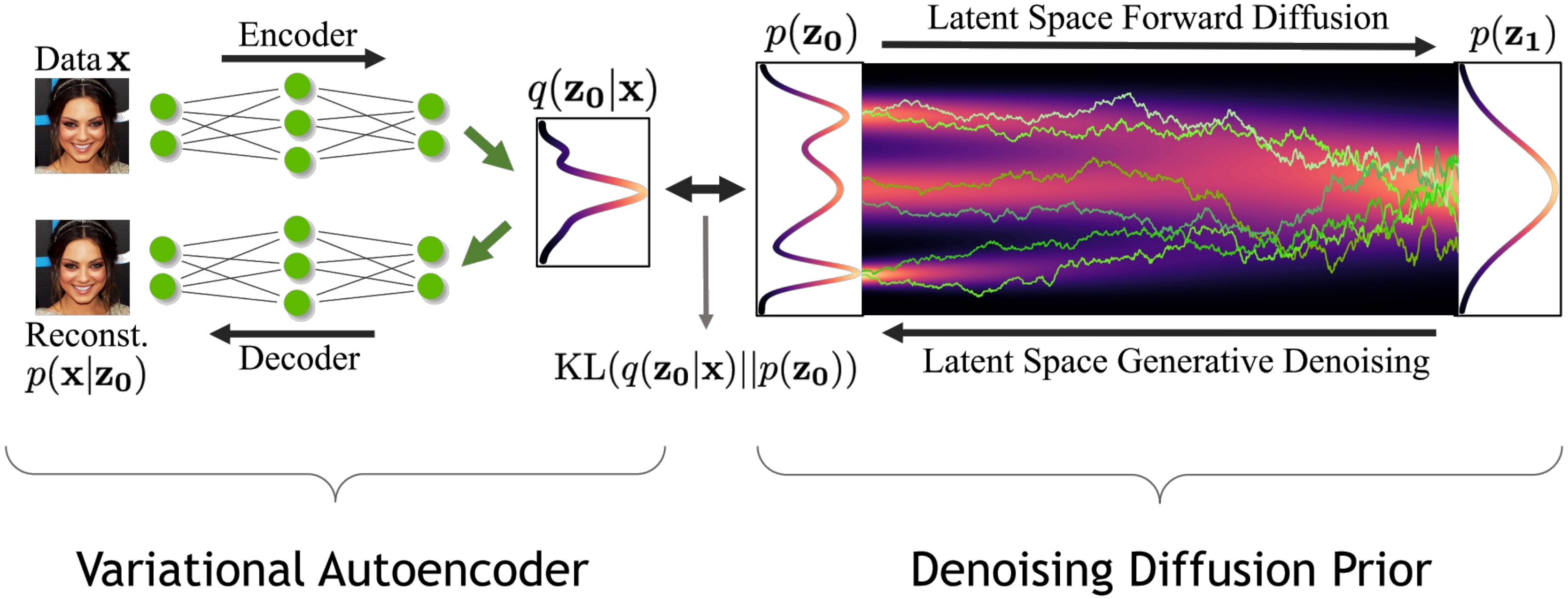
- Q: Can we lift the diffusion model to a latent space that is faster to diffuse?

Latent-Space Diffusion Models: Variational Autoencoder + Score-Based Prior



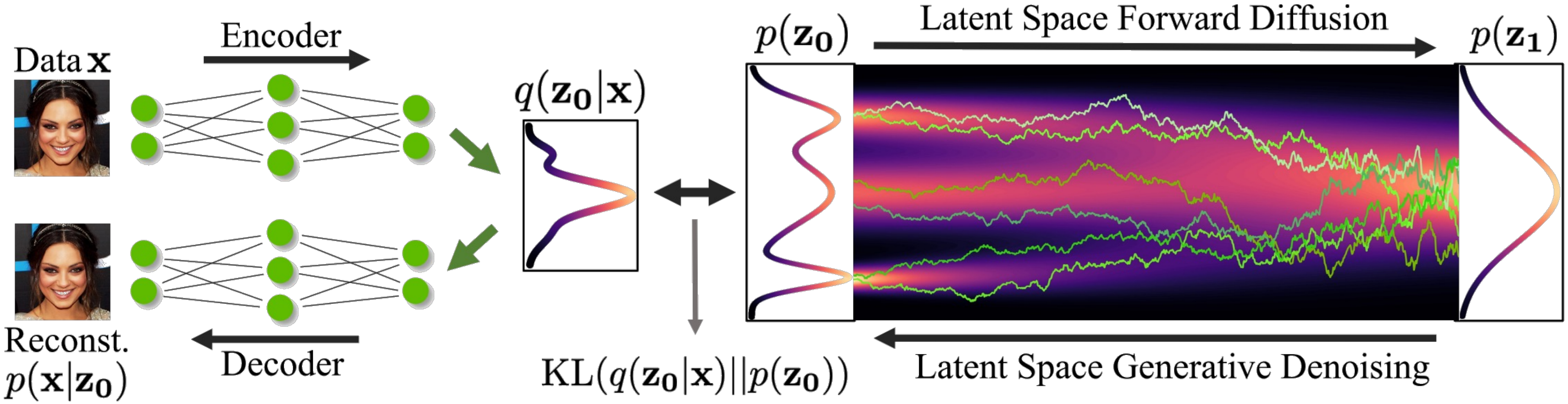
- Encoder maps the input data to an embedding space.
- Denoising diffusion models are applied in the latent space.

Latent-Space Diffusion Models: Variational Autoencoder + Score-Based Prior



- Advantages:
  - The distribution of latent embeddings close to Normal distribution $\rightarrow$ Simpler denoising and faster synthesis.
  - Augmented latent space $\rightarrow$ More expressivity.
  - Tailored Autoencoders $\rightarrow$ More expressivity, application to any data type, e.g., graphs, text, 3D data, etc.

Latent-Space Diffusion Models: Variational Autoencoder + Score-Based Prior



Variational Autoencoder | **Denoising Diffusion Prior**

$$\begin{aligned}\mathcal{L}(x,\phi,\theta,\psi) &= \mathbb{E}_{q\phi(z_0|x)}\left[-\log p_\psi(x|z_0)\right] + D_{\mathrm{KL}}(q_\phi(z_0|x)||p_\theta(z_0)) \\ &= \underbrace{\mathbb{E}_{q\phi(z_0|x)}\left[-\log p_\psi(x|z_0)\right]}_{\text{reconstruction term}} + \underbrace{\mathbb{E}_{q\phi(z_0|x)}\left[\log q_\phi(z_0|x)\right]}_{\text{negative encoder entropy}} + \underbrace{\mathbb{E}_{p_\theta(z_0)}\left[-\log p_\psi(x|z_0)\right]}_{\text{cross entropy}}.\end{aligned}$$

Applications

- There are many successful applications of diffusion models (in constantly growing numbers):
  - **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.
- Key enabler by diffusion models: Perform high-resolution conditional generation!

High Resolution Conditional Generation: Impressive Text-to-Image Conditional Diffusion Models

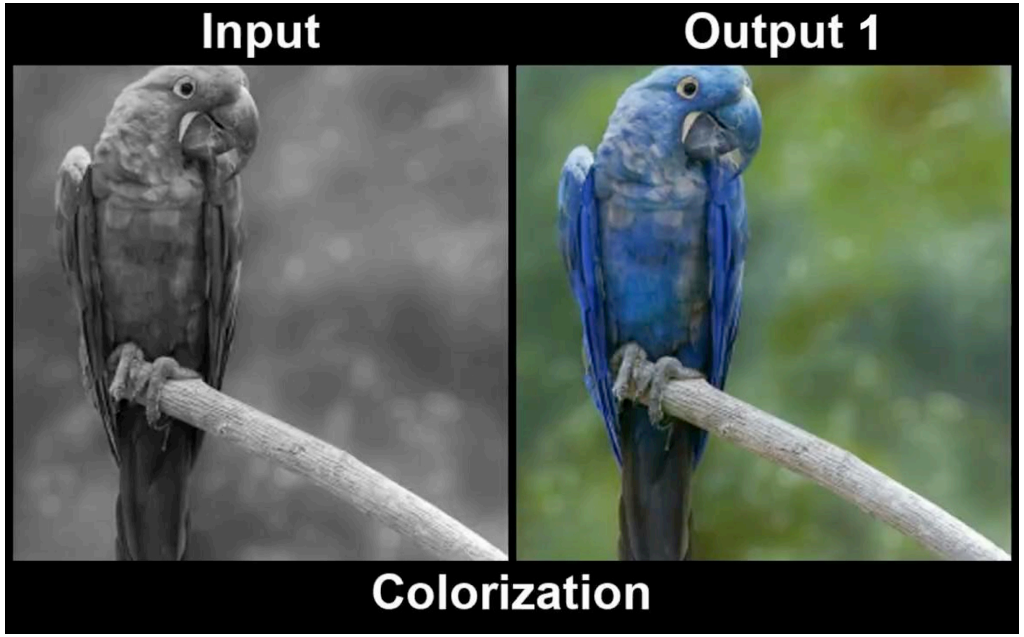
"a propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese"

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

High Resolution Conditional Generation:
Impressive Super-Resolution & Colorization Diffusion Mode

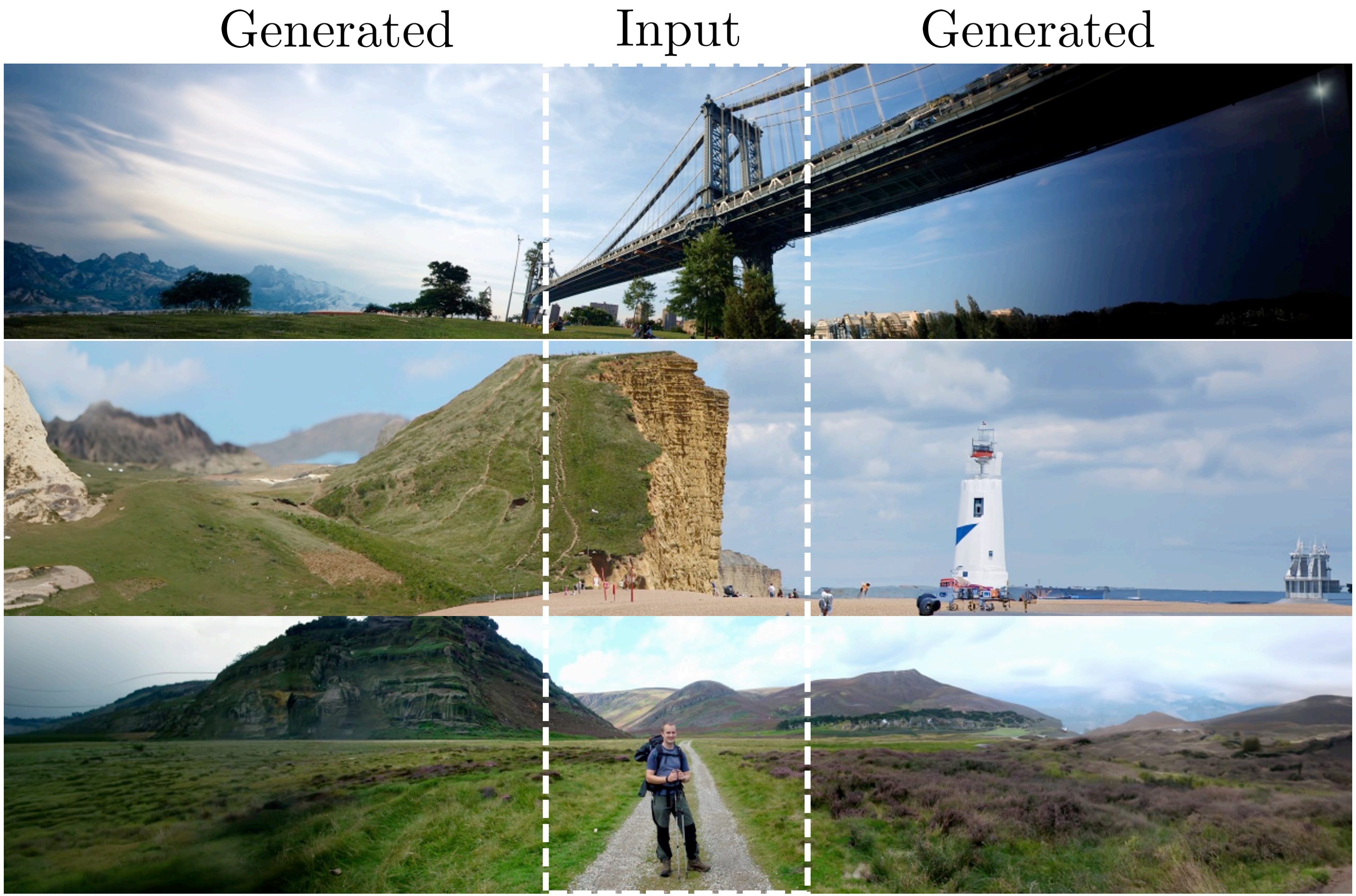
Lecture #14

Super-Resolution



Colorization

High Resolution Conditional Generation: Impressive Panorama Generation Diffusion Models



Conditional Diffusion Models: Include Condition as Input to Reverse Process

- Reverse Process:

$$p_\theta(x_{0:T}|c) = p(x_T) = \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t, c),\ \ p_\theta(x_{t-1}|x_t, c) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t, c), \Sigma(x_t, t, c)).$$

- Variational Upper Bound:

$$L_\theta(x_0|c) = \mathbb{E}_q\lceil L_T(x_0) + \sum_{t>1} D_{\mathrm{KL}}\left(q(x_{t-1}|x_t, x_0)||p(x_{t-1}|x_t, c)\right) - \log p_\theta(x_0|x_1, c)\rceil.$$

- 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 sequence of vector embeddings - cross-attention.

Classifier-Guided Conditional Diffusion Models: 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**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ 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

- For class-conditional modeling of $p(x_t|c)$, train an extra classifier $p(c|x_t)$.
- Mix its gradient with the diffusion/score model during sampling.

Classifier-Guided Conditional Diffusion Models: 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**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ 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

- Sample with a modified score: $\nabla_{x_t} [\log p(x_t|c) + \omega \log p(c|x_t)]$.
- Approximate samples from the distribution $\tilde{p}(x_t|c) \propto p(x_t|c)p(c|x_t)^\omega$.

Classifier-Free Conditional Diffusion Models: Guidance by Bayes' Rule on Conditional Diffusion Models

Lecture #14

- Instead of training an additional classifier, get an "implicit classifier" by jointly training a conditional and unconditional diffusion model:

$$p(c|x_t) \propto \frac{p(x_t|c)}{p(x_t)}.$$

$p(x_t|c)$ ← Conditional Diffusion Model

$p(x_t)$ ← Unconditional Diffusion Model

- In practice, compute $p(x_t|c)$ and $p(x_t)$ by randomly dropping the condition of the diffusion model at certain chance.

- The modified score with this implicit classifier included is:

$$\begin{aligned}\nabla_{x_t}\left[\log p(x_t|c) + \omega \log p(c|x_t)\right] &= \nabla_{x_t}\left[\log p(x_t|c) + \omega(\log p(x_t|c) - \log p(x_t))\right] \\ &= \nabla_{x_t}\left[(1+\omega)\log p(x_t|c) - \omega \log p(x_t)\right].\end{aligned}$$

Classifier-Free Conditional Diffusion Models: Trade-Off for Sample Quality and Sample Diversity

Non-Guidance

$\omega = 1$

$\omega = 3$

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

Introduction to Deep Generative Modeling

Lecture #14

HY-673 – Computer Science Dep., University of Crete

Professors: Yannis Pantazis, Yannis Stylianou

Teaching Assistant: Michail Raptakis