Diffusion and Consistency models in Latent Spaces

Jian Li

IIIS, Tsinghua

2023

- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Introduction to diffusion process

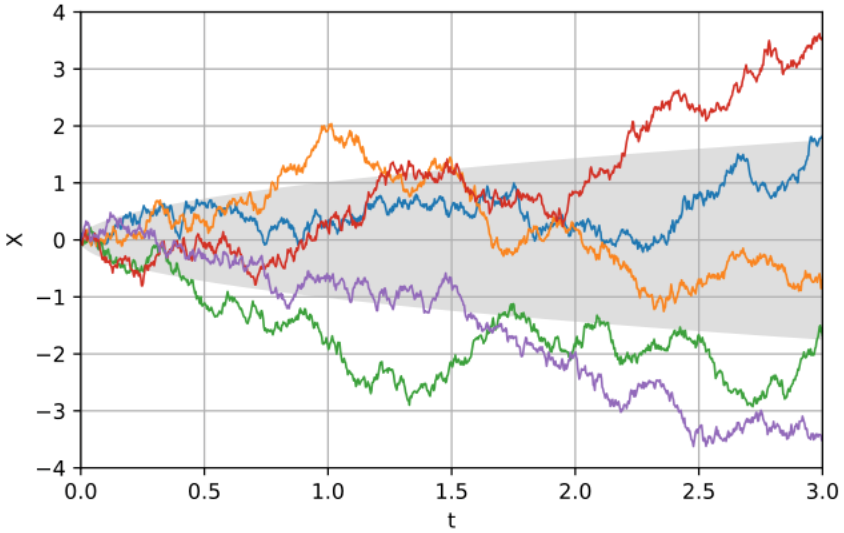
(Forward) Diffusion process (adding noise)

SDE: $\mathrm{d}\boldsymbol{x}_t = f(t)\boldsymbol{x}_t\,\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t, \quad \boldsymbol{x}_0 \sim p_{data}\left(\boldsymbol{x}_0\right)$

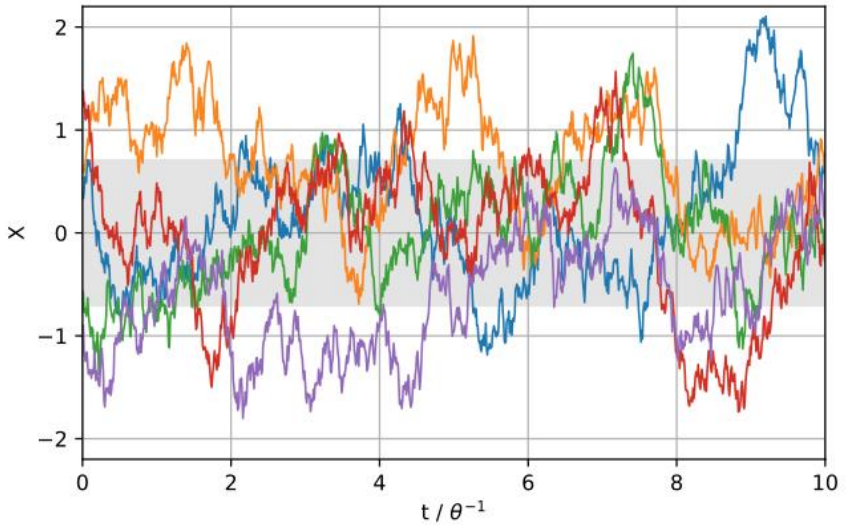
Drift term

Diffusion term

Brownian motion (Wiener process)



Standard 1-d Brownian motion
f=0, g=1



Ornstein–Uhlenbeck Process
f=-1, g=1

Introduction to diffusion process

Forward process (adding noise)

SDE: $\mathrm{d}\boldsymbol{x}_t = f(t)\boldsymbol{x}_t\,\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t, \quad \boldsymbol{x}_0 \sim p_{data}\left(\boldsymbol{x}_0\right)$

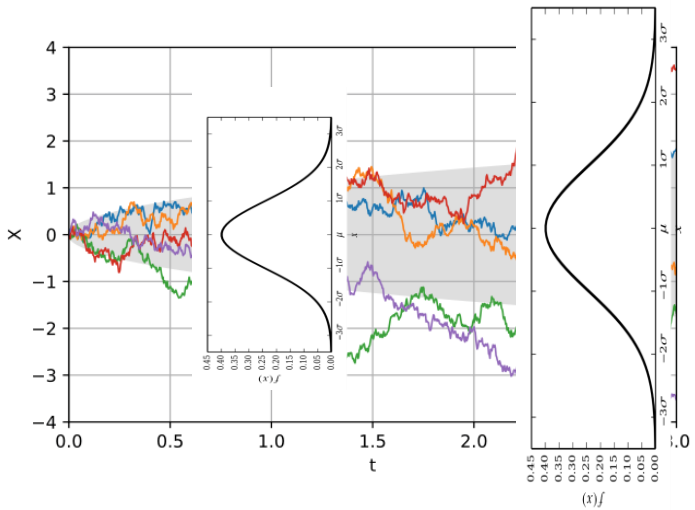
Reverse process (denoising)

Anderson's theorem

Reverse-time Brownian motion

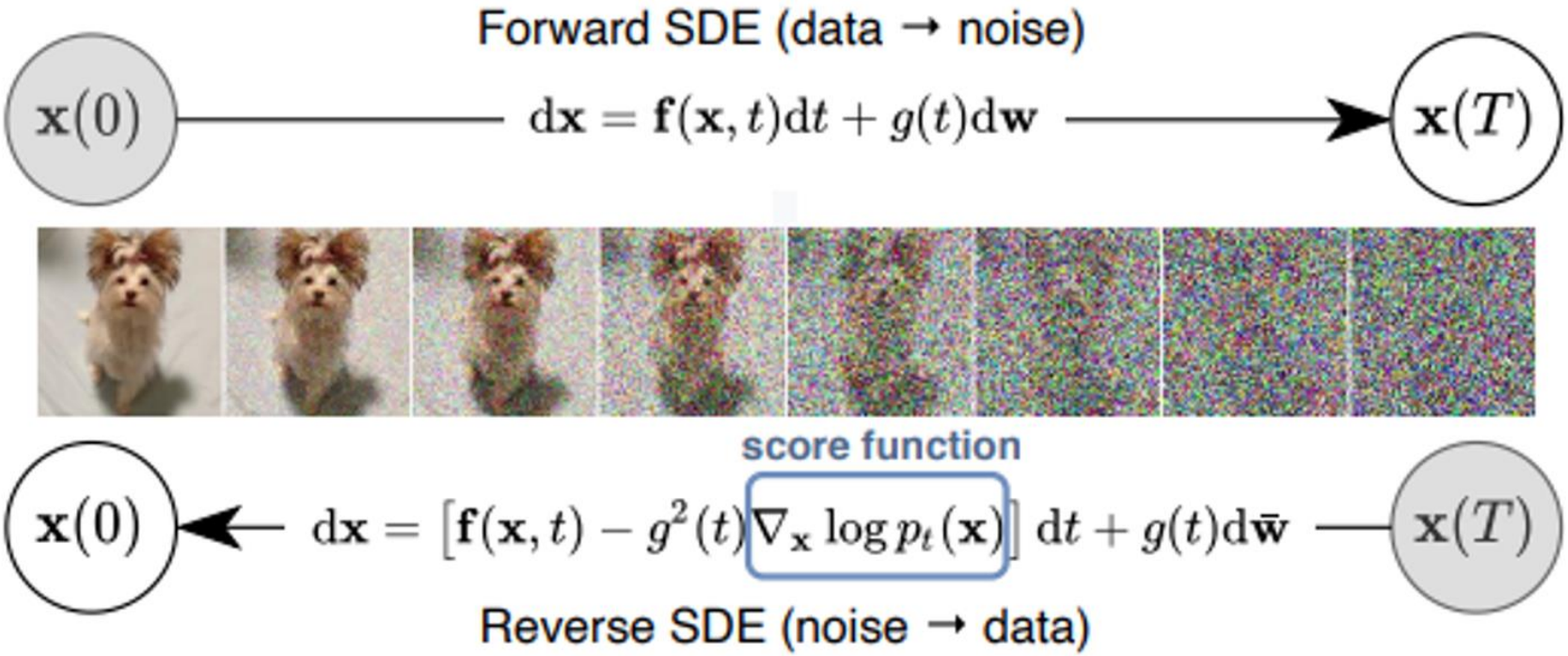
$$\mathrm{d}\boldsymbol{x}_t = \left[f(t)\boldsymbol{x}_t - g^2(t)\nabla_{\boldsymbol{x}} \log q_t\left(\boldsymbol{x}_t\right)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim q_T\left(\boldsymbol{x}_T\right)$$

Marginal probability distribution of x_t at time t



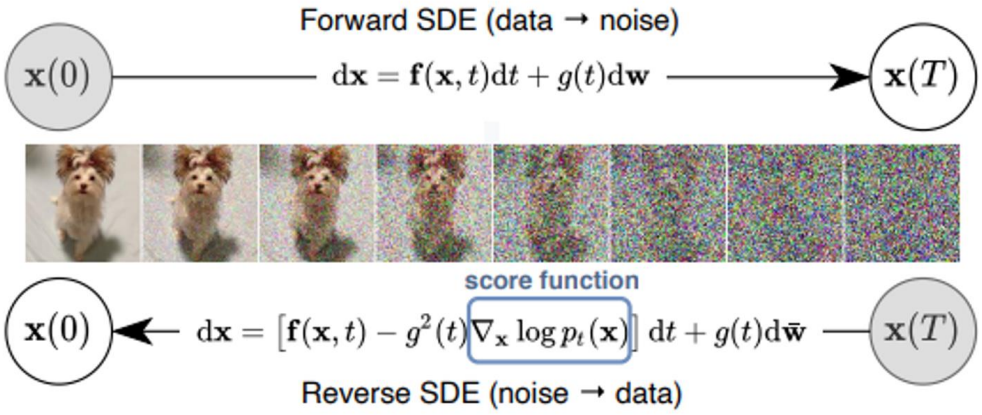
- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Introduction to diffusion models



Picture from Song et al.

Score-based diffusion models



However, we do not know the score function. We only have training images.

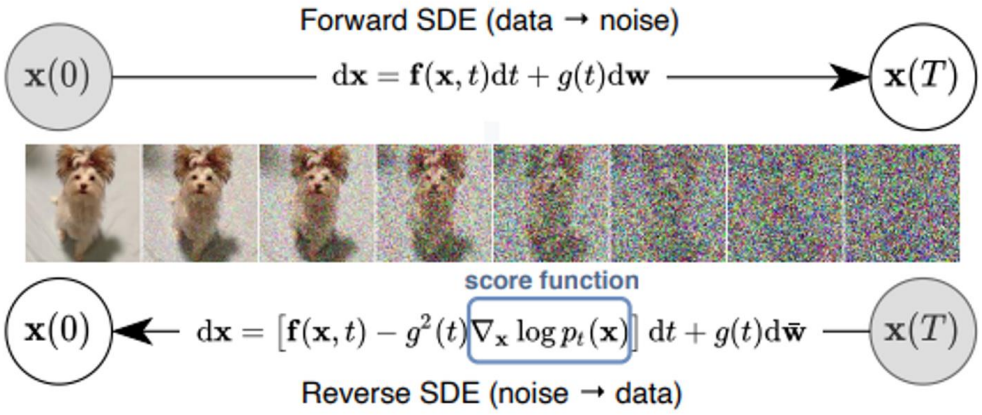
Learn the score function from the training data!
Score matching objective [Song & Ermon (2019)]:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sigma_i^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{p_{\sigma_i}(\tilde{\mathbf{x}}|\mathbf{x})} \left[\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma_i) - \nabla_{\tilde{\mathbf{x}}} \log p_{\sigma_i}(\tilde{\mathbf{x}} \mid \mathbf{x}) \|_2^2 \right].$$

A deep neural network

Forward probability
we know how to compute

Score-based diffusion models



However, we do not know the score function. We only have training images.

Learn the score function from the training data!
Score-matching objective [Song & Ermon (2019)]:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sigma_i^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{p_{\sigma_i}(\tilde{\mathbf{x}}|\mathbf{x})} \left[\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma_i) - \nabla_{\tilde{\mathbf{x}}} \log p_{\sigma_i}(\tilde{\mathbf{x}} \mid \mathbf{x}) \|_2^2 \right].$$

A deep neural network

Forward probability
we know how to compute

Score-based diffusion models

- It is possible to further simply the loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{t\in[0,T],x_t\sim q_t}\left[w(t)||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t) + \sigma(t)\nabla \log q_t(\boldsymbol{x}_t))||^2\right]$$
$$= \mathbb{E}_{t\in[0,T],\boldsymbol{x}_0\sim q_0,\boldsymbol{\epsilon}}\left[w(t)||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t) - \boldsymbol{\epsilon}||^2\right]$$

A deep neural network

where $\boldsymbol{\epsilon} \sim N(0,I)$ and $\boldsymbol{x}_t = \alpha(t)\boldsymbol{x}_0 + \sigma(t)\boldsymbol{\epsilon}$

So $\boldsymbol{\epsilon_\theta}$**(.,.)** is also called the **noise prediction model**

- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Prob-Flow ODE

Reverse process

Anderson's theorem

Reverse-time Brownian motion

$$\mathrm{d}\boldsymbol{x}_t = \left[f(t)\boldsymbol{x}_t - g^2(t)\nabla_{\boldsymbol{x}} \log q_t\left(\boldsymbol{x}_t\right)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim q_T\left(\boldsymbol{x}_T\right)$$

The above SDE defines the same marginal distribution $q_t(\boldsymbol{x}_t)$ as the following **Probability-Flow ODE** [Song et al. 2020]

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = f(t)\boldsymbol{x}_t - \frac{1}{2}g^2(t)\nabla_{\boldsymbol{x}} \log q_t\left(\boldsymbol{x}_t\right), \quad \boldsymbol{x}_T \sim q_T\left(\boldsymbol{x}_T\right)$$

Since the noise prediction model $\boldsymbol{\epsilon_\theta}(.,.)$ tries to fit the score function, the following **empirical PF-ODE** is an approximation of the above PF-ODE

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = f(t)\boldsymbol{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t\right), \quad \boldsymbol{x}_T \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\boldsymbol{I}\right)$$

Probability Flow ODE

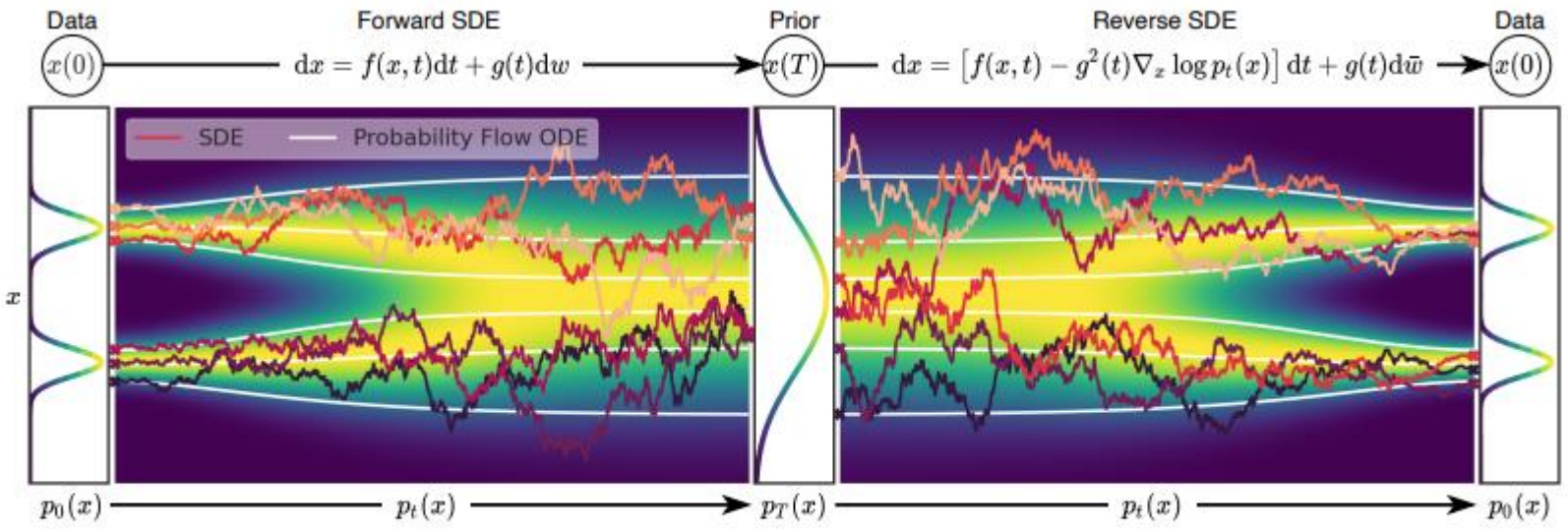


Figure 2: **Overview of score-based generative modeling through SDEs.** We can map data to a noise distribution (the prior) with an SDE (Section 3.1), and reverse this SDE for generative modeling (Section 3.2). We can also reverse the associated probability flow ODE (Section 4.3), which yields a deterministic process that samples from the same distribution as the SDE. Both the reverse-time SDE and probability flow ODE can be obtained by estimating the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ (Section 3.3).

prob flow ODE: $\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x},t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t$

Sample from PF-ODE

- Empirical PF-ODE

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = f(t)\boldsymbol{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t\right), \quad \boldsymbol{x}_T \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\boldsymbol{I}\right)$$

- ODE trajectories are smoother than SDE trajectories
- To accelerate sampling, need to build better ODE solvers

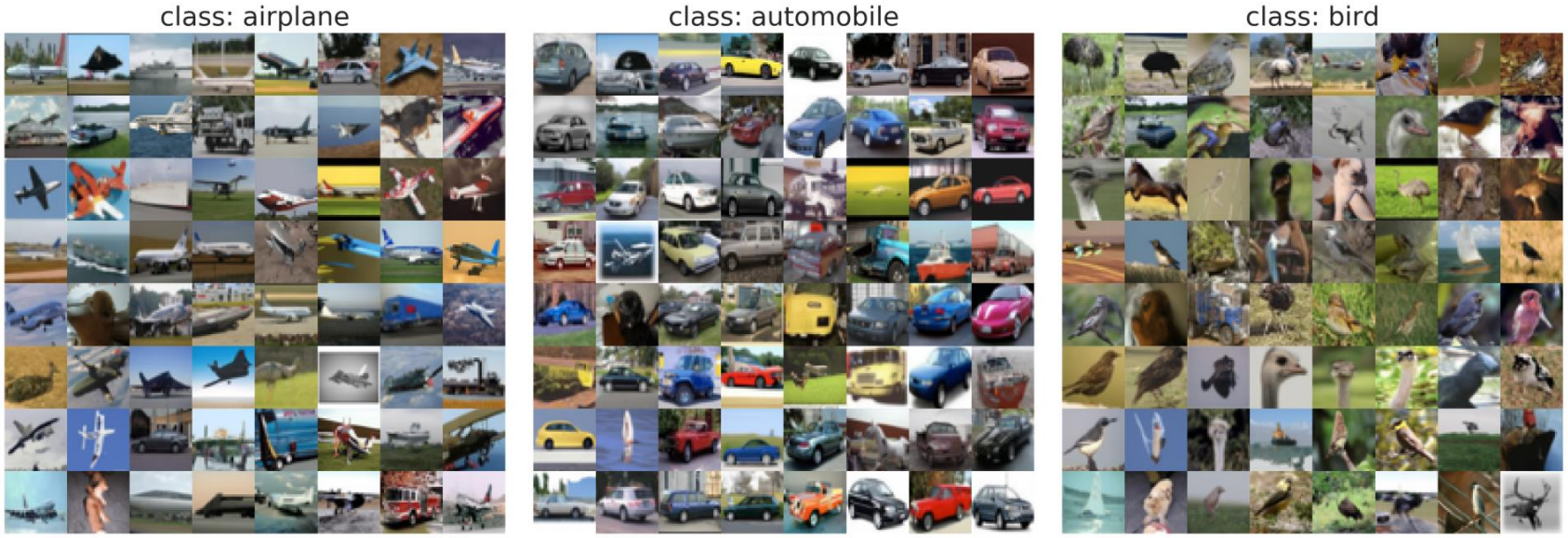
These ODE solvers can approximate the integration

$$\int_{t_{n+1}}^{t_n} \left(f(t)\boldsymbol{z}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{z}_t, \boldsymbol{c}, t\right) \right) \mathrm{d}t \approx \Psi(\boldsymbol{z}_{t_{n+1}}, t_{n+1}, t_n, \boldsymbol{c})$$

- Several diffusion models are based on the above idea: DDPM, DDIM, DPM-Solver, DPM-Solver++

- Current best ODE solvers (DPM-Solver, DPM-Solver++) requires 10-20 steps [Lu et al. 2022]

Generated Images



Cifar 10

CelebA-HQ

Song et al. ICLR 2021

- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Stable Diffusion

'A painting of the last supper by Picasso.'

High-Resolution Image Synthesis with Latent Diffusion Models

Stable diffusion

Idea: the reconstructions are confined to the image manifold (enforcing local realism), rather than on pixel-space **(Diffusion in latent space)**

Use an autoencoder in VQGAN (trained by combination of a perceptual loss and a patch-based adversarial objective)

Encoder E encodes image x into a latent representation z = E(x)

The encoder downsamples the image by a factor f

Decoder D reconstructs the image from the latent, giving x˜ = D(z) = D(E(x))

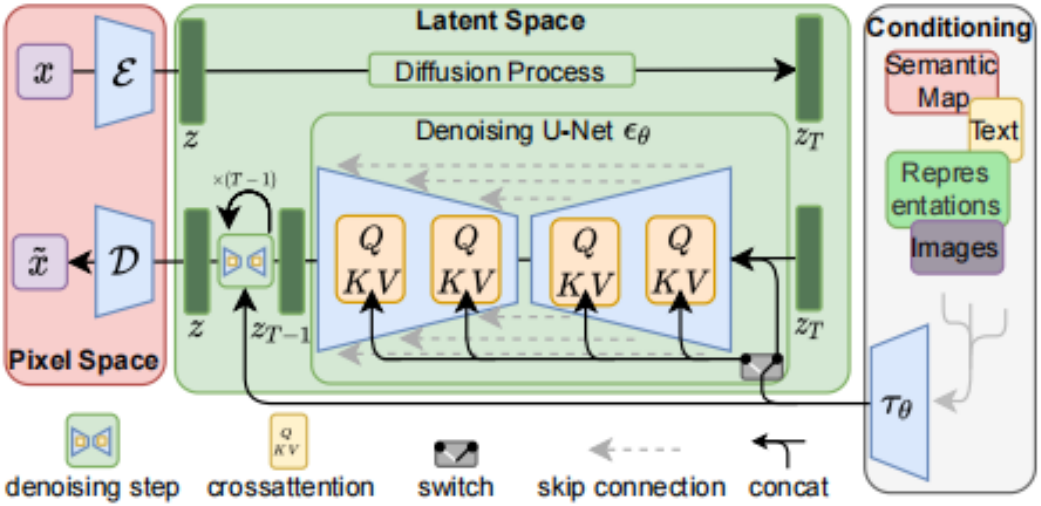


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Stable diffusion

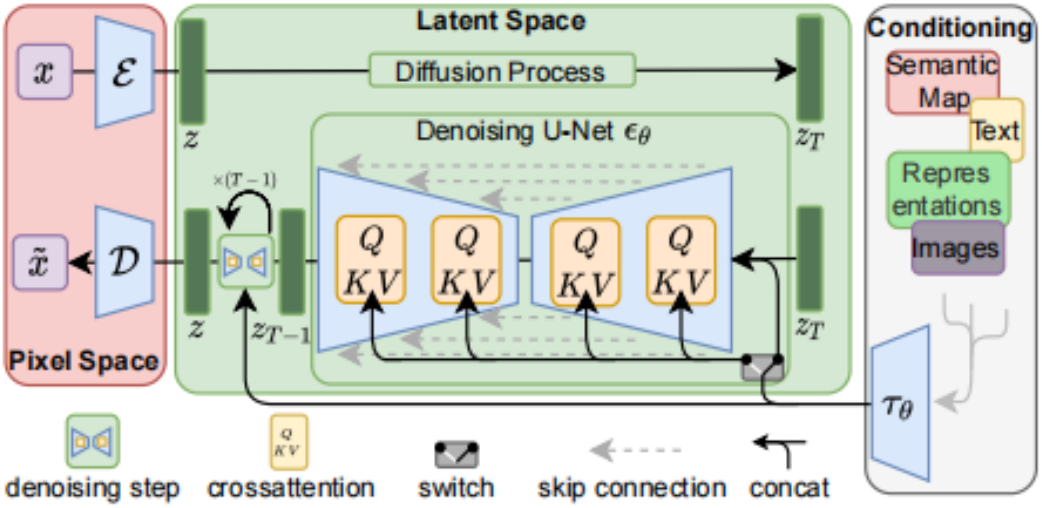


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Loss from variational lower bound on p(x), which mirrors denoising score-matching

Loss of Diffusion model: $L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right]$

denoising autoencoders $\epsilon_\theta(x_t, t)$

Loss of Diffusion model in latent space: $L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\right].$

The neural backbone of the model $\epsilon_\theta(z_t, t)$ is realized as a time-conditional UNet

Stable Diffusion: Text-to-Image

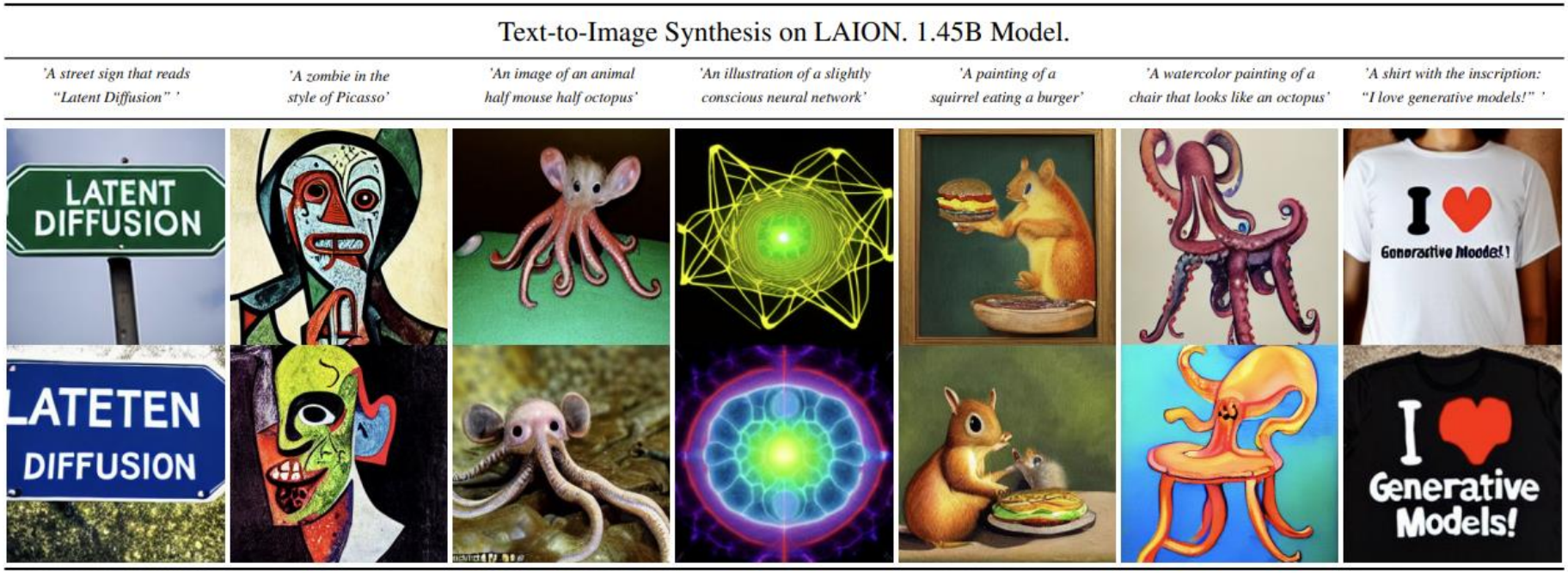


Figure 5. Samples for user-defined text prompts from our model for text-to-image synthesis, *LDM-8 (KL)*, which was trained on the LAION [78] database. Samples generated with 200 DDIM steps and $\eta = 1.0$. We use unconditional guidance [32] with $s = 10.0$.

- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Consistency Models

Goal: Accelerate inference

How: Learn a consistency function $\boldsymbol{f} : (\mathbf{x}_t, t) \mapsto \mathbf{x}_\epsilon$

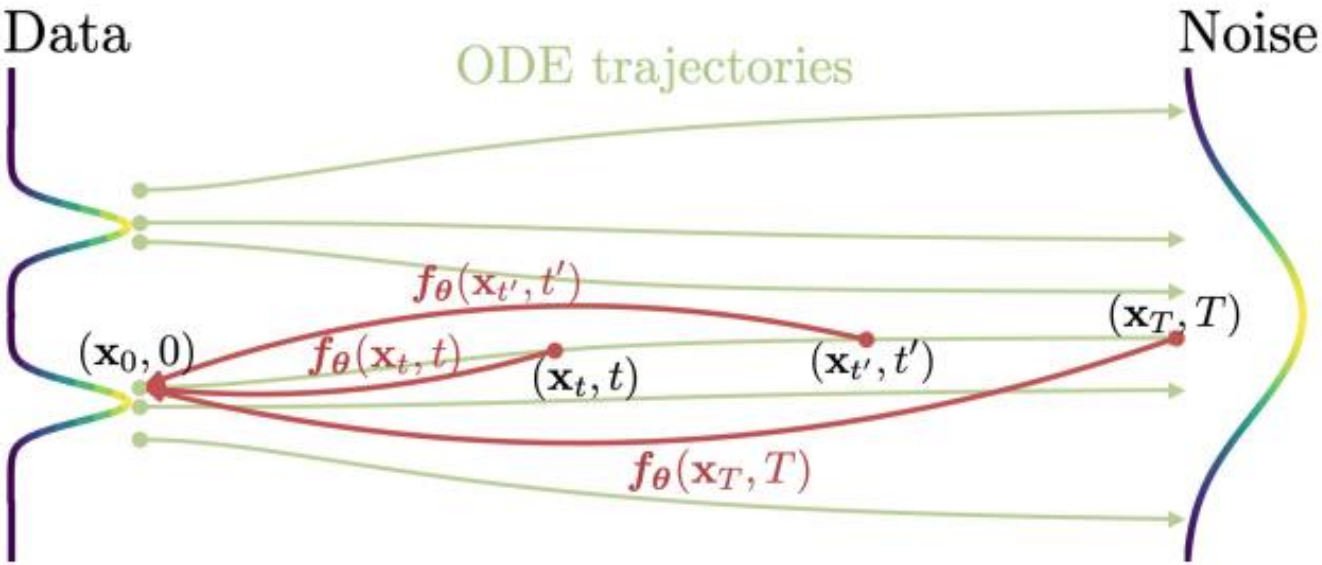


Figure 2: Consistency models are trained to map points on any trajectory of the PF ODE to the trajectory's origin.

Picture from Song et al.

Consistency Models

Consistency Property:

$$\boldsymbol{f}(\mathbf{x}_t, t) = \boldsymbol{f}(\mathbf{x}_{t'}, t') \text{ for all } t, t' \in [\epsilon, T]$$

How to ensure such property?

Define the following consistency loss:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \Phi) = \mathbb{E}_{\boldsymbol{x},t}\left[d\left(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t_{n+1}}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\boldsymbol{x}}^{\phi}_{t_n}, t_n)\right)\right]$$

Here: $\boldsymbol{\theta}^- \leftarrow \mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta}$

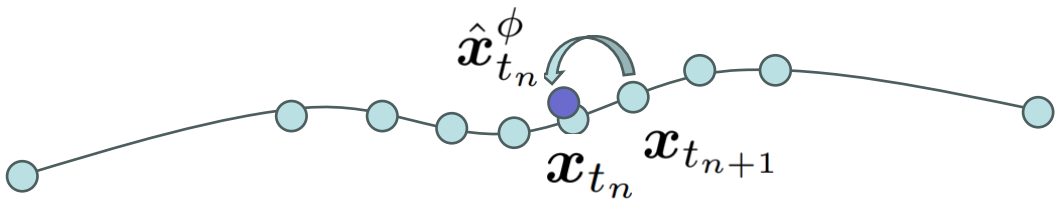
$$\hat{\boldsymbol{x}}^{\phi}_{t_n} \leftarrow \boldsymbol{x}_{t_{n+1}} + (t_n - t_{n+1})\Phi(\boldsymbol{x}_{t_{n+1}}, t_{n+1}; \phi)$$

Explain in next page

Consistency Models

$$\hat{\boldsymbol{x}}_{t_n}^{\phi} \leftarrow \boldsymbol{x}_{t_{n+1}} + (t_n - t_{n+1})\Phi(\boldsymbol{x}_{t_{n+1}}, t_{n+1}; \phi)$$

- How do we estimate x_{t_n} from $x_{t_{n+1}}$



Previous ODE solvers requires many steps to generate high quality images

Consistency Models

We can distill a consistency model from an existing diffusion models

Algorithm 2 Consistency Distillation (CD)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, ODE solver $\Phi(\cdot, \cdot; \boldsymbol{\phi})$, $d(\cdot, \cdot)$, $\lambda(\cdot)$, and μ

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

Sample $\mathbf{x} \sim \mathcal{D}$ and $n \sim \mathcal{U}[\![1, N-1]\!]$

Sample $\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}; t_{n+1}^2 \boldsymbol{I})$

$\hat{\mathbf{x}}_{t_n}^{\boldsymbol{\phi}} \leftarrow \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) \Phi(\mathbf{x}_{t_{n+1}}, t_{n+1}; \boldsymbol{\phi})$

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \boldsymbol{\phi}) \leftarrow$

$\lambda(t_n) d(\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\mathbf{x}}_{t_n}^{\boldsymbol{\phi}}, t_n))$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \boldsymbol{\phi})$

$\boldsymbol{\theta}^- \leftarrow \operatorname{stopgrad}(\mu \boldsymbol{\theta}^- + (1 - \mu) \boldsymbol{\theta})$

until convergence

Image generated by CM

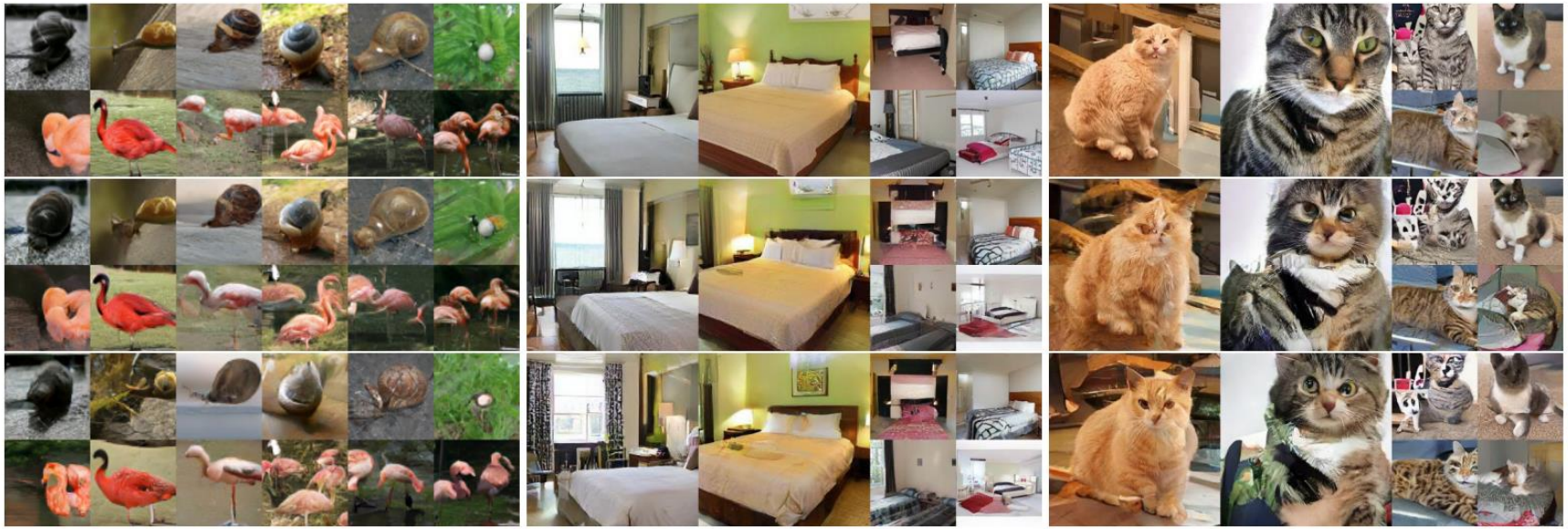


Figure 5: Samples generated by EDM (*top*), CT + single-step generation (*middle*), and CT + 2-step generation (*Bottom*). All corresponding images are generated from the same initial noise.

Picture from Song et al. 2023

- Intro to Diffusion Process
- Intro to Generative Diffusion Models
- Probability Flow ODE
- Latent Diffusion Models (Stable Diffusion)
- Consistency Models
- Latent Consistency Models

Latent Consistency Models

- Goal:
  - Synthesizing high quality images, like Stable Diffusion
  - Conditional generation: text-to-image
    - Incorporate classifier free guidance
  - Very fast inference: 2-4 steps, even 1 step
  - Allow easy finetuning on downstream customized dataset

Latent Consistency Models: Synthesizing high quality images with few-step inference
Joint work with Simian Luo, Yiqin Tan, Longbo Huang, Hang Zhao

Incorporate Classifier-Free Guidance

- Classifier-Free Guidance (CFG) [Ho & Salimans, 2022] is an important technique for generating high quality images
- Given a CFG scale ω, the original noise prediction is replaced by a linear combination of conditional and unconditional noise prediction

$$\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, \omega, \boldsymbol{c}, t) = (1+\omega)\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, \boldsymbol{c}, t) - \omega\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}, \varnothing, t)$$

Latent Consistency Distillation

Algorithm 1 Latent Consistency Distillation (LCD)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, EMA rate μ, noise schedule $\alpha(t), \sigma(t)$, guidance scale $[w_{\text{min}}, w_{\text{max}}]$, skipping interval k, and encoder $E(\cdot)$

Encoding training data into latent space: $\mathcal{D}_z = \{(\boldsymbol{z}, \boldsymbol{c}) | \boldsymbol{z} = E(\boldsymbol{x}), (\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{D}\}$

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

Sample $(\boldsymbol{z}, \boldsymbol{c}) \sim \mathcal{D}_z$, $n \sim \mathcal{U}[1, N-k]$ and $\omega \sim [\omega_{\text{min}}, \omega_{\text{max}}]$

Sample $\boldsymbol{z}_{t_{n+k}} \sim \mathcal{N}(\alpha(t_{n+k})\boldsymbol{z}; \sigma^2(t_{n+k})\mathbf{I})$

$\hat{\boldsymbol{z}}^{\Psi,\omega}_{t_n} \leftarrow \boldsymbol{z}_{t_{n+k}} + (1+\omega)\Psi(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, t_n, \boldsymbol{c}) - \omega\Psi(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, t_n, \varnothing)$

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \Psi) \leftarrow d(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}_{t_{n+k}}, \omega, \boldsymbol{c}, t_{n+k}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\boldsymbol{z}}^{\Psi,\omega}_{t_n}, \omega, \boldsymbol{c}, t_n))$

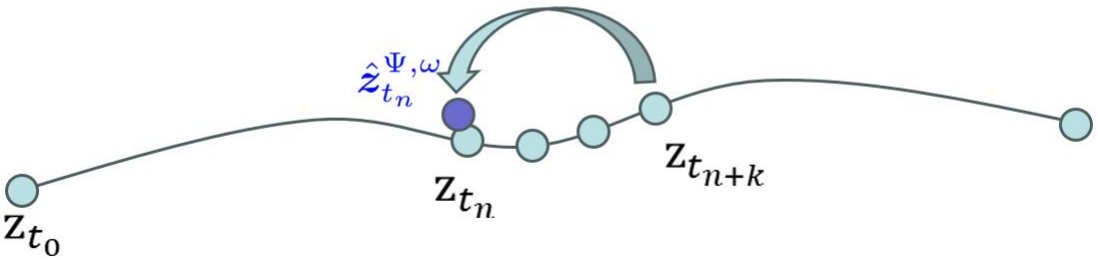
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$

$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta})$

until convergence

CFG and more advanced ODE-solver

Use existing ODE solver Ψ to approximate z_{t_n} (e.g. DDIM, DPM-solver, DPM-solver++)



$$\Psi_{\text{DDIM}}(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, t_n, \boldsymbol{c}) = \underbrace{\frac{\alpha_{t_n}}{\alpha_{t_{n+k}}}\boldsymbol{z}_{t_{n+k}} - \sigma_{t_n}\left(\frac{\sigma_{t_{n+k}}\cdot\alpha_{t_n}}{\alpha_{t_{n+k}}\cdot\sigma_{t_n}} - 1\right)\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_{t_{n+k}}, \boldsymbol{c}, t_{n+k})}_{\text{DDIM Estimated } \boldsymbol{z}_{t_n}} - \boldsymbol{z}_{t_{n+k}}$$

Inference/Sampling

Algorithm 3 Multistep Latent Consistency Sampling

Input: Latent Consistency Model $\boldsymbol{f_\theta}(\cdot,\cdot,\cdot,\cdot)$, Sequence of timesteps $\tau_1 > \tau_2 > \cdots > \tau_{N-1}$, Text condition $\boldsymbol{c}$, Classifier-Free Guidance Scale ω, Noise schedule $\alpha(t), \sigma(t)$, Decoder $D(\cdot)$

Sample initial noise $\hat{\boldsymbol{z}}_T \sim \mathcal{N}(\mathbf{0}; \boldsymbol{I})$

$\boldsymbol{z} \leftarrow \boldsymbol{f_\theta}(\hat{\boldsymbol{z}}_T, \omega, \boldsymbol{c}, T)$

for $n = 1$ to $N-1$ **do**

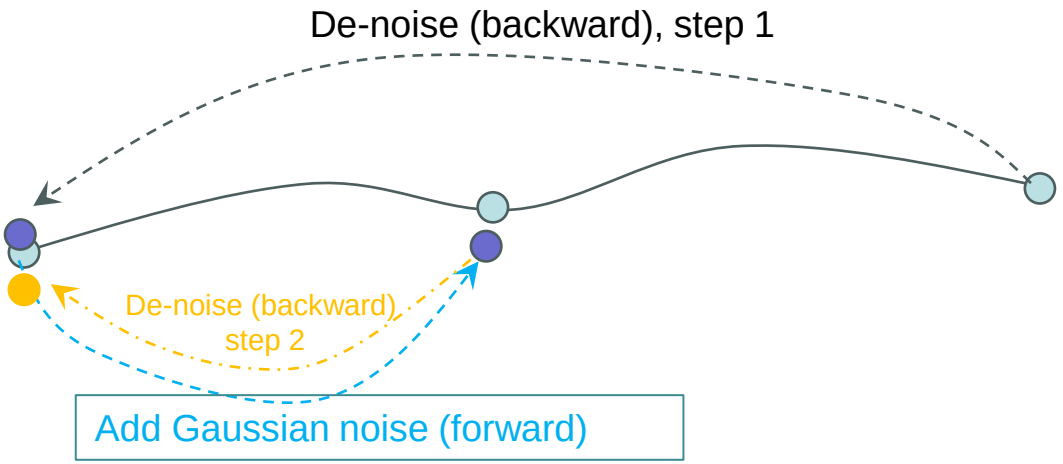
  $\hat{\boldsymbol{z}}_{\tau_n} \sim \mathcal{N}(\alpha(\tau_n)\boldsymbol{z}; \sigma^2(\tau_n)\mathbf{I})$ Add noise (forward)

  $\boldsymbol{z} \leftarrow \boldsymbol{f_\theta}(\hat{\boldsymbol{z}}_{\tau_n}, \omega, \boldsymbol{c}, \tau_n)$

end for

$\boldsymbol{x} \leftarrow D(\boldsymbol{z})$

Output: $\boldsymbol{x}$

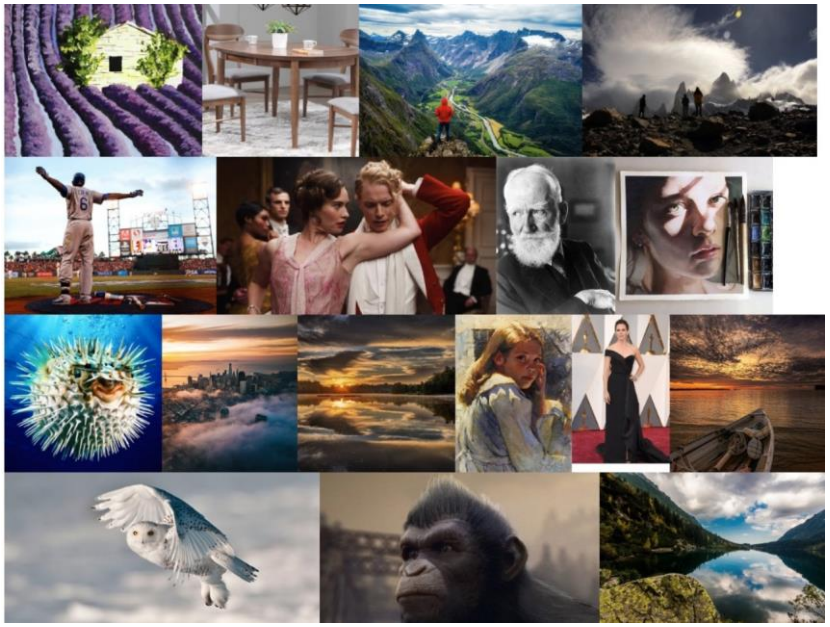


Two-step sampling

Experiments

- LAION-5B dataset: LAION-Aesthetics-6+ (12M) and LAION-Aesthetics-6.5+ (650K)
- Training: high-quality 768×768 2~4-step LCM takes only 32 A100 GPU hours for training

An orange cat is looking at its reflection in the mirror.



Experiments

Model (512 × 512) Reso	FID ↓				CLIP Score ↑			
	1 Step	2 Steps	4 Steps	8 Steps	1 Steps	2 Steps	4 Steps	8 Steps
DDIM (Song et al., 2020a)	183.29	81.05	22.38	13.83	6.03	14.13	25.89	29.29
DPM (Lu et al., 2022a)	185.78	72.81	18.53	12.24	6.35	15.10	26.64	29.54
DPM++ (Lu et al., 2022b)	185.78	72.81	18.43	12.20	6.35	15.10	26.64	**29.55**
Guided-Distill (Meng et al., 2023)	108.21	33.25	15.12	13.89	12.08	22.71	27.25	28.17
LCM (Ours)	**35.36**	**13.31**	**11.10**	**11.84**	**24.14**	**27.83**	**28.69**	28.84

Table 1: Quantitative results with $\omega = 8$ at 512×512 resolution. LCM significantly surpasses baselines in the 1-4 step region on LAION-Aesthetic-6+ dataset. For LCM, DDIM-Solver is used with a skipping step of $k = 20$.

Model (768 × 768) Reso	FID ↓				CLIP Score ↑			
	1 Step	2 Steps	4 Steps	8 Steps	1 Steps	2 Steps	4 Steps	8 Steps
DDIM (Song et al., 2020a)	186.83	77.26	24.28	15.66	6.93	16.32	26.48	29.49
DPM (Lu et al., 2022a)	188.92	67.14	20.11	**14.08**	7.40	17.11	27.25	29.80
DPM++ (Lu et al., 2022b)	188.91	67.14	20.08	14.11	7.41	17.11	27.26	**29.84**
Guided-Distill (Meng et al., 2023)	120.28	30.70	16.70	14.12	12.88	24.88	28.45	29.16
LCM (Ours)	**34.22**	**16.32**	**13.53**	14.97	**25.32**	**27.92**	**28.60**	28.49

Table 2: Quantitative results with $\omega = 8$ at 768×768 resolution. LCM significantly surpasses the baselines in the 1-4 step region on LAION-Aesthetic-6.5+ dataset. For LCM, DDIM-Solver is used with a skipping step of $k = 20$.

High resolution images generate by LCM

4 steps

High resolution images generate by LCM

4 steps

High resolution images generate by LCM

4 steps

High resolution images generate by LCM

2 steps

High resolution images generate by LCM

2 steps

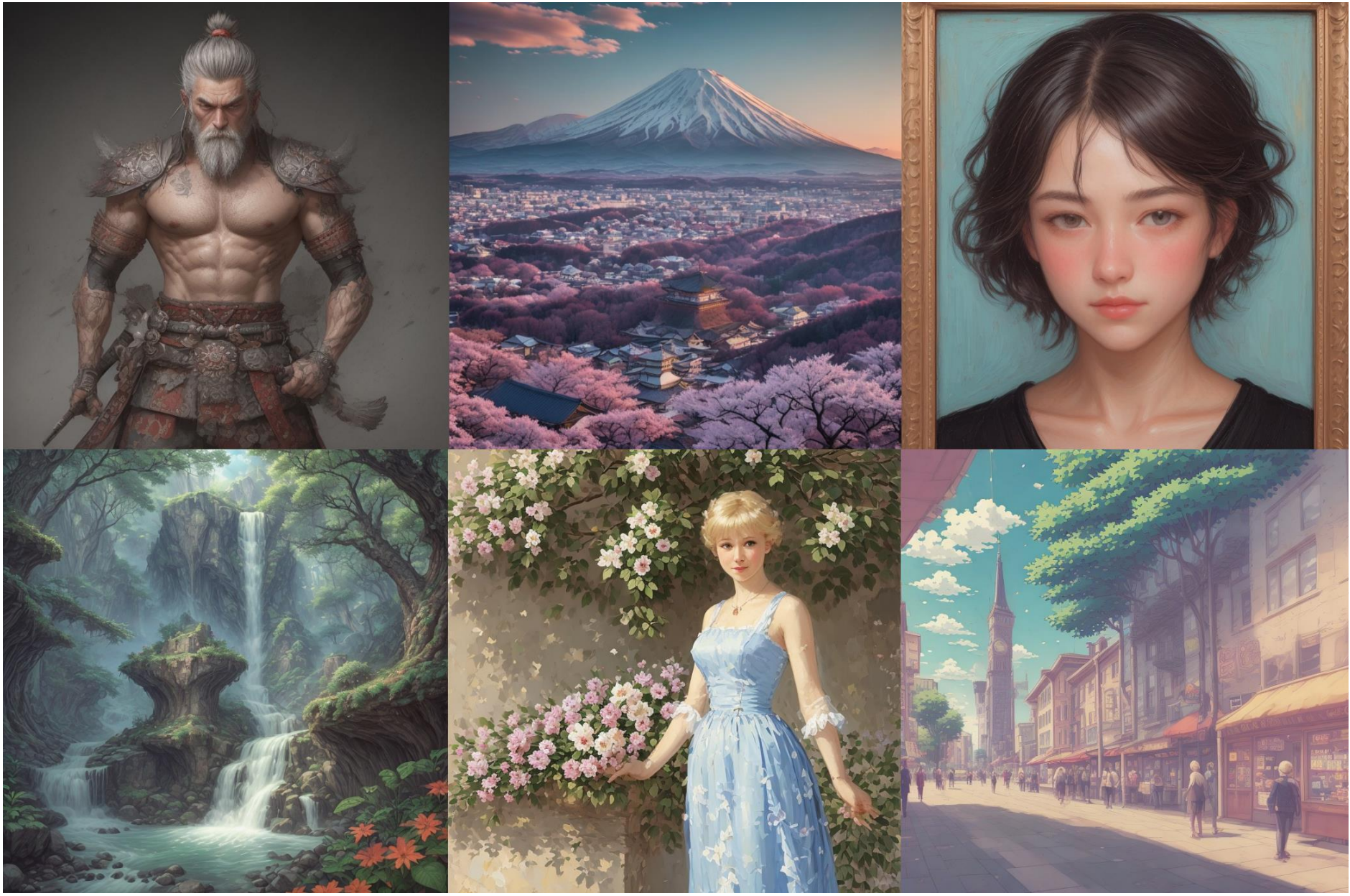
High resolution images generate by LCM

1 step

High resolution images generate by LCM

1 step

2 steps



2 steps

Finetuning on customized dataset

Algorithm 4 Latent Consistency Fine-tuning (LCF)

Input: customized dataset $\mathcal{D}^{(s)}$, pre-trained LCM parameter $\boldsymbol{\theta}$, learning rate η, distance metric $d(\cdot,\cdot)$, EMA rate μ, noise schedule $\alpha(t), \sigma(t)$, guidance scale $[w_{\text{min}}, w_{\text{max}}]$, skipping interval k, and encoder $E(\cdot)$

Encode training data into the latent space: $\mathcal{D}_z^{(s)} = \{(\boldsymbol{z}, \boldsymbol{c}) | \boldsymbol{z} = E(\boldsymbol{x}), (\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{D}^{(s)}\}$

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

Sample $(\boldsymbol{z}, \boldsymbol{c}) \sim \mathcal{D}_z^{(s)}$, $n \sim \mathcal{U}[1, N-k]$ and $w \sim [w_{\text{min}}, w_{\text{max}}]$

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

$\boldsymbol{z}_{t_{n+k}} \leftarrow \alpha(t_{n+k})\boldsymbol{z} + \sigma(t_{n+k})\boldsymbol{\epsilon} \quad , \quad \boldsymbol{z}_{t_n} \leftarrow \alpha(t_n)\boldsymbol{z} + \sigma(t_n)\boldsymbol{\epsilon}$

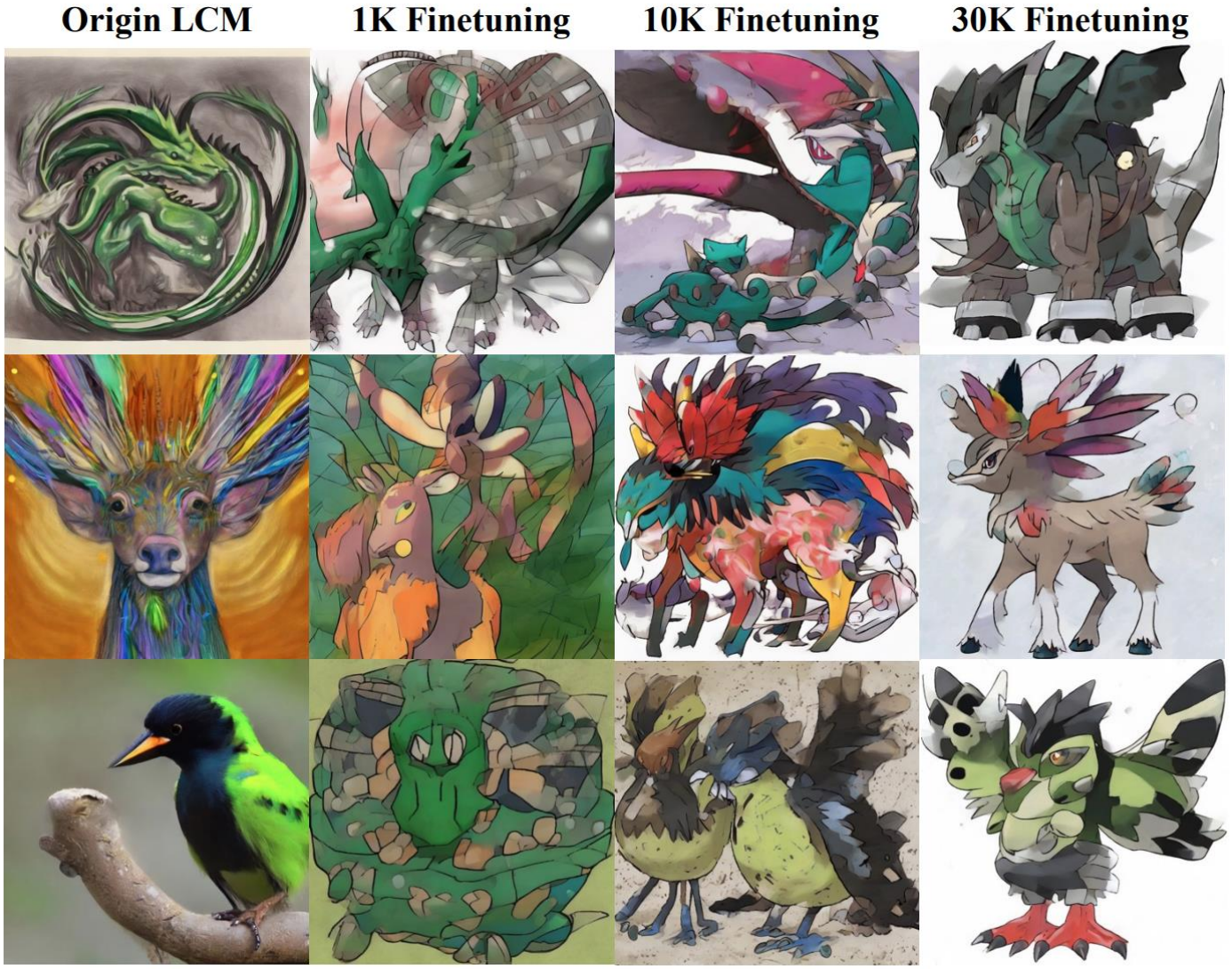
$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-) \leftarrow d(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, \boldsymbol{c}, w), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\boldsymbol{z}_{t_n}, t_n, \boldsymbol{c}, w))$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$

$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta})$

until convergence

Finetuning on customized dataset



Pokemon dataset

Thanks

Jian Li 李建
lapordge@gmail.com
Wechat id: lapordge

Latent Consistency Distillation

Algorithm 1 Latent Consistency Distillation (LCD)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, EMA rate μ, noise schedule $\alpha(t), \sigma(t)$, guidance scale $[w_{\text{min}}, w_{\text{max}}]$, skipping interval k, and encoder $E(\cdot)$

Encoding training data into latent space: $\mathcal{D}_z = \{(\boldsymbol{z}, \boldsymbol{c}) | \boldsymbol{z} = E(\boldsymbol{x}), (\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{D}\}$

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

Sample $(\boldsymbol{z}, \boldsymbol{c}) \sim \mathcal{D}_z$, $n \sim \mathcal{U}[1, N-k]$ and $\omega \sim [\omega_{\text{min}}, \omega_{\text{max}}]$

Sample $\boldsymbol{z}_{t_{n+k}} \sim \mathcal{N}(\alpha(t_{n+k})\boldsymbol{z}; \sigma^2(t_{n+k})\mathbf{I})$

$\hat{\boldsymbol{z}}^{\Psi,\omega}_{t_n} \leftarrow \boldsymbol{z}_{t_{n+k}} + (1+\omega)\Psi(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, t_n, \boldsymbol{c}) - \omega\Psi(\boldsymbol{z}_{t_{n+k}}, t_{n+k}, t_n, \varnothing)$

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \Psi) \leftarrow d(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}_{t_{n+k}}, \omega, \boldsymbol{c}, t_{n+k}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\boldsymbol{z}}^{\Psi,\omega}_{t_n}, \omega, \boldsymbol{c}, t_n))$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$

$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta})$

until convergence

Use existing ODE solver Ψ to approximate z_{t_n} (e.g. DDIM, DPM-solver, DPM-solver++)