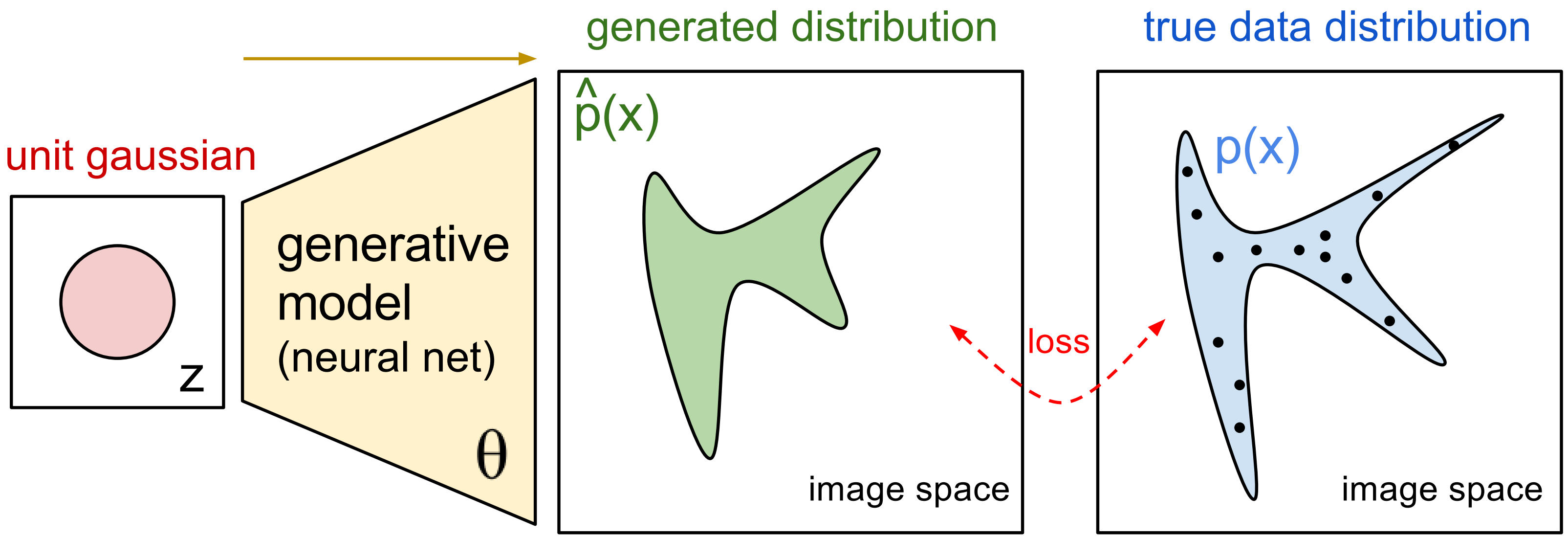
Introduction to Diffusion Models and its Application

Ruofeng Yang
Shanghai Jiao Tong University
wanshuiyin@sjtu.edu.cn

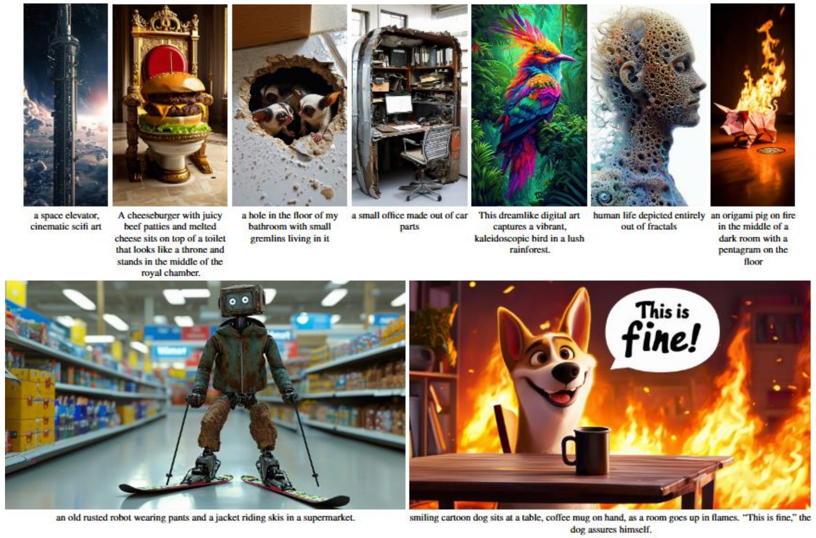
What is the Generative Model?

- Given a pure noise z, the generative models generate sample from the target distribution.



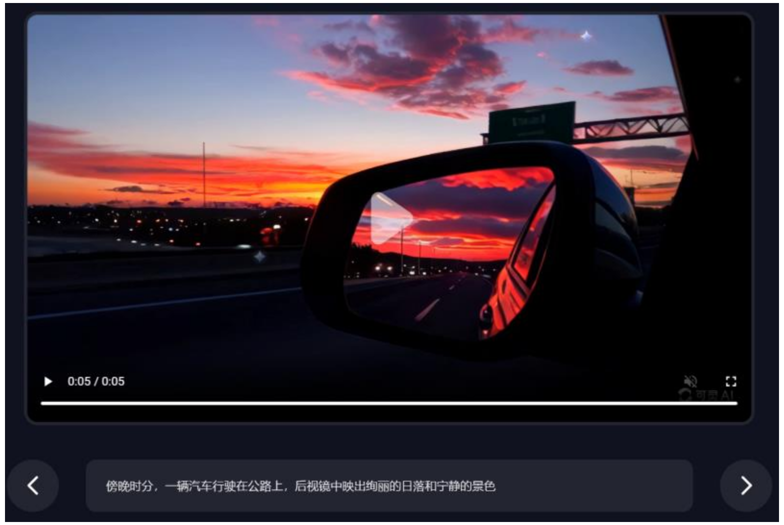
https://openai.com/index/generative-models/

Current Application

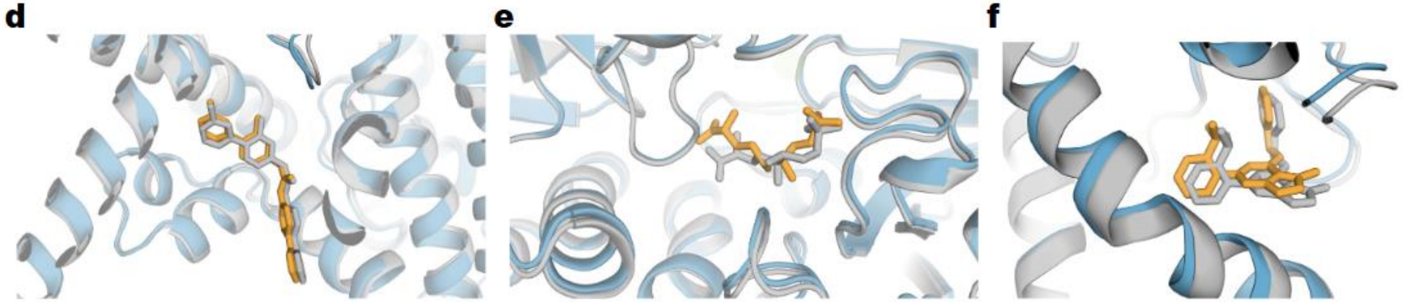


2D Generation: Stable Diffusion 3

3D Generation: Wonder3D

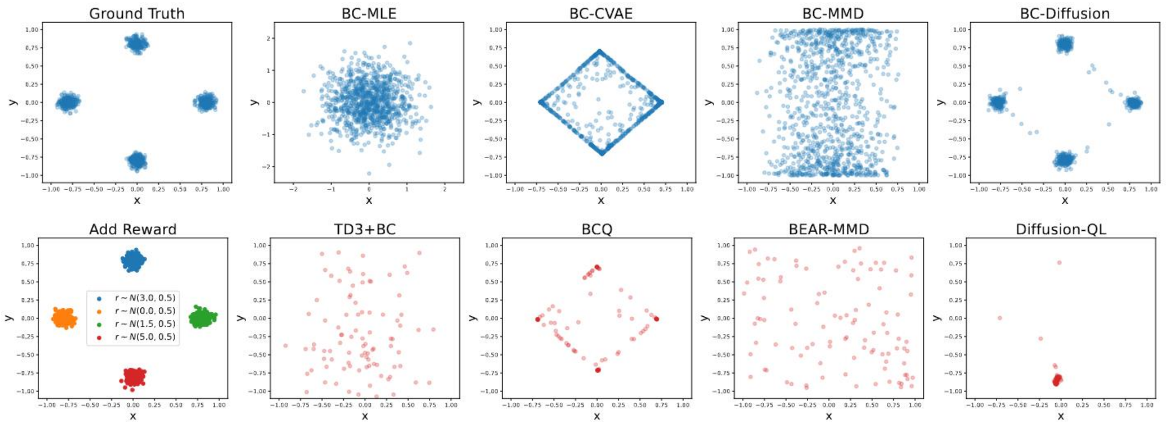


Video Generation: KLING



AlphaFold 3.

Current Application-Other Areas



Diffusion as the RL policy.

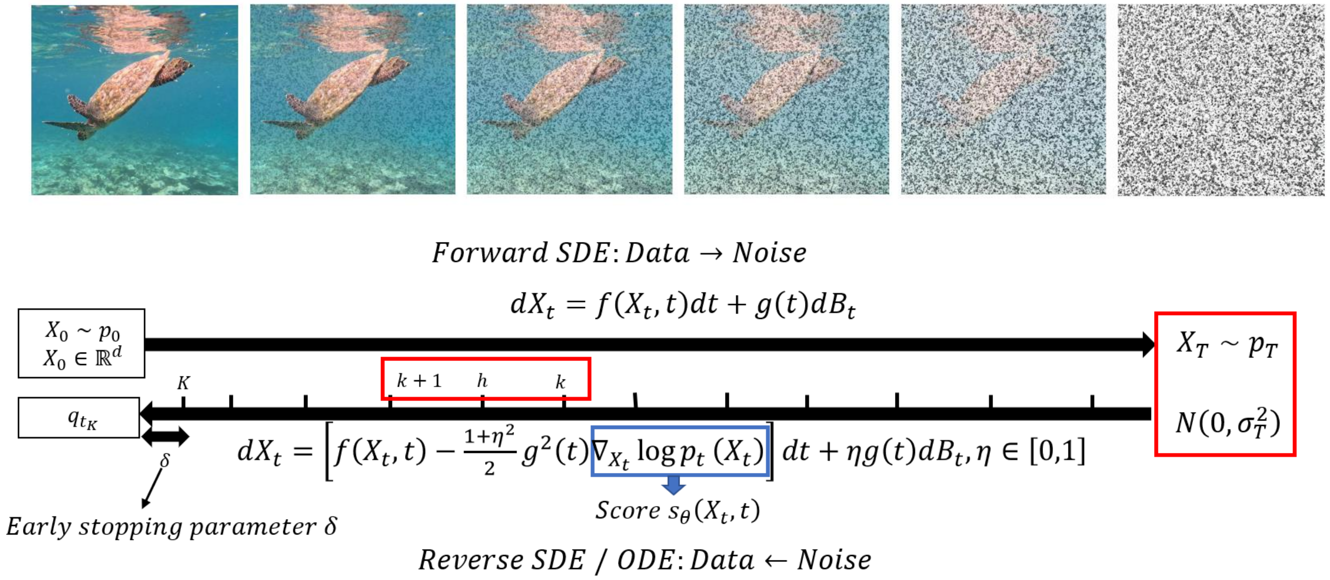
GPT-2 S	a hiring platform that ”includes a fun club meeting place,” says petitioner’s AQQFredericks. They’s the adjacent marijuana-hop. Others have allowed 3B Entertainment
GPT-2 M	misused, whether via Uber, a higher-order reality of quantified impulse or the No Mass Paralysis movement, but the most shamefully universal example is gridlock
SEDD S	As Jeff Romer recently wrote, “The economy has now reached a corner - 64% of household wealth and 80% of wealth goes to credit cards because of government austerity
SEDD M	Wyman worked as a computer science coach before going to work with the U.S. Secret Service in upstate New York in 2010. Without a license, the Secret Service will have to

(b) Generated Text (small models)

NLP Tasks: SEED

Basic of Diffusion

Diffusion Model: SDE Perspective



1. Three choices of the forward process:

(a) Variance Preserving SDE: $f(X_t, t) = -\frac{1}{2} X_t, g(X_t, t) = 1 \rightarrow X_t = e^{-\frac{t}{2}} X_0 + (1 - e^{-t})Z$

(b) Variance Exploding SDE: $f(X_t, t) = 0, g(X_t, t) = \sqrt{\frac{d\sigma_t^2}{d\,t}} \rightarrow X_t = X_0 + \sigma_t Z$

(c) Rectified Flow: $X_t = (1-t)X_0 + tZ, t \in [0,1]$

2. When $\eta = 1$, reverse SDE generative process; When $\eta = 0$, reverse PFODE process.

Score-Based Generative Modeling through Stochastic Differential Equations

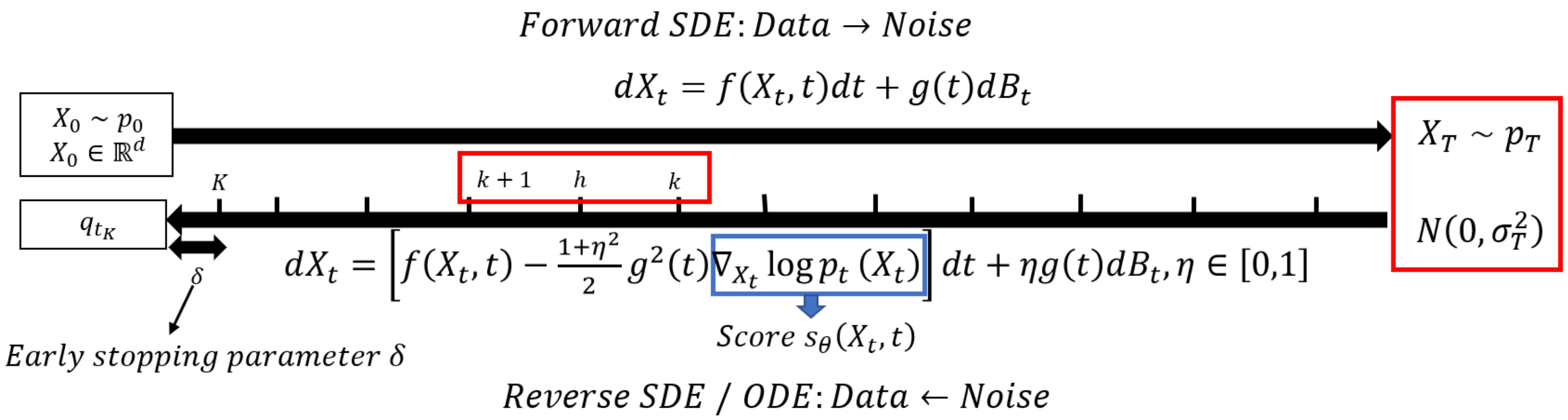
The Training Process

The score matching objective

$$L_{DM}(\theta) = \frac{1}{nK}\sum_{i=1}^{n}\sum_{k=1}^{K}\mathbb{E}_{X_{t_k,i}|x_i}\left[\left\|\nabla_{X_{t_k,i}}\log p_t\left(X_{t_k,i}|x_i\right) - s_\theta\left(X_{t_k,i}, t_k\right)\right\|_2^2\right].$$

The Generation Process

- After the approximated score function $s_\theta(X,t)$ is learned (in L_2 norm), we can use $s_\theta(X,t)$ to generate images by reverse SDE or PFODE.
- Choose a reverse beginning distribution $\mathcal{N}(0, \sigma_t^2 I)$.
- Choose a discretization scheme to run reverse process with approximated score.



Three Core Points in Diffusion Models

- The sample (discretization) complexity K of the generation process

  Including the results of VP, VE and RF-based models

- The theoretical understanding of the training process

  Including (a) the estimation error of the score, (b) the learning dynamic

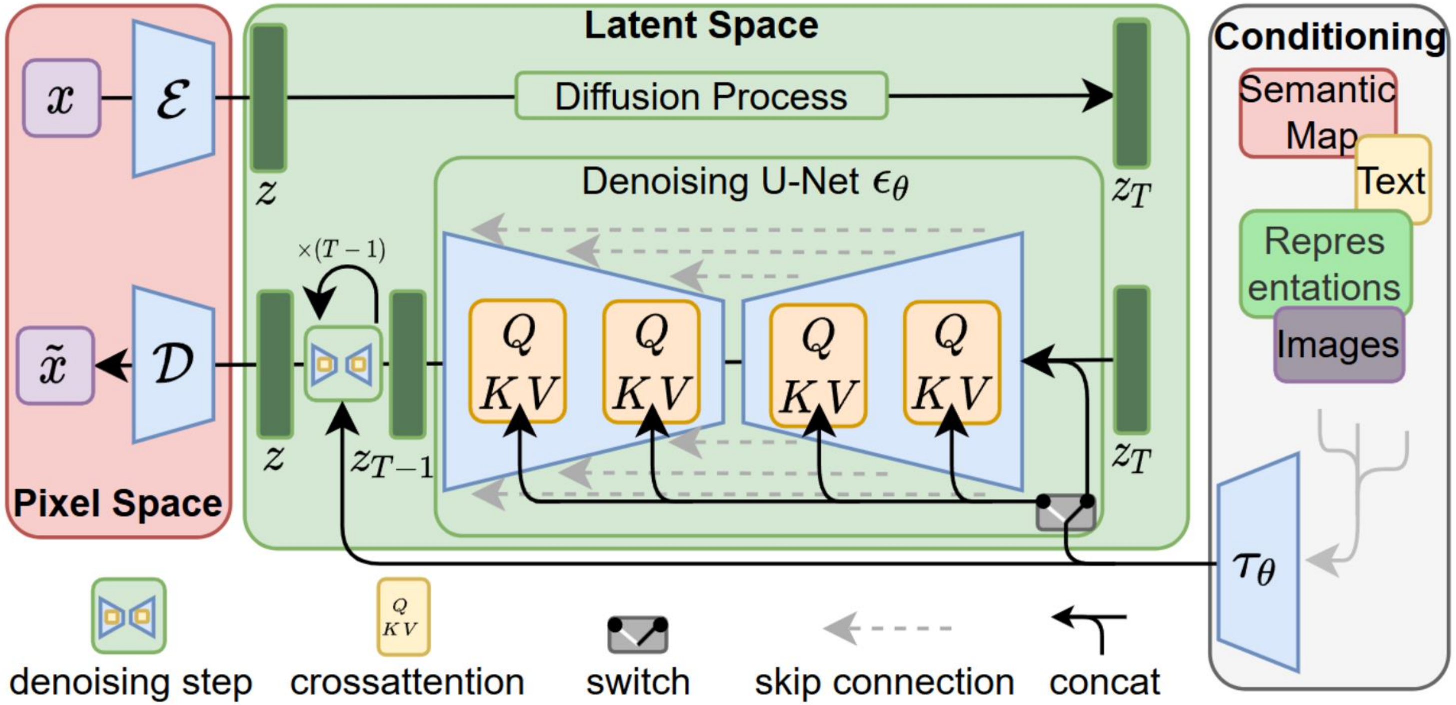
  (c) the generalization property

- The design of posting training process

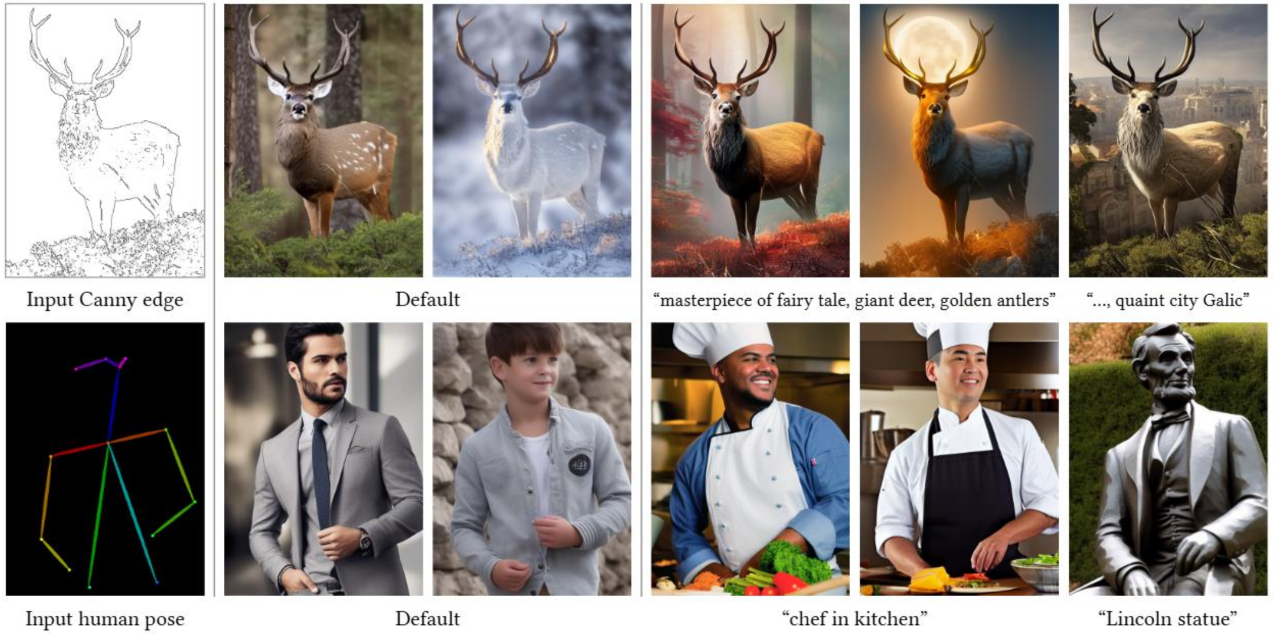
  Including (a) Guidance-based method and (b) Reinforcement Learning Fine-tuning (RLFT) method

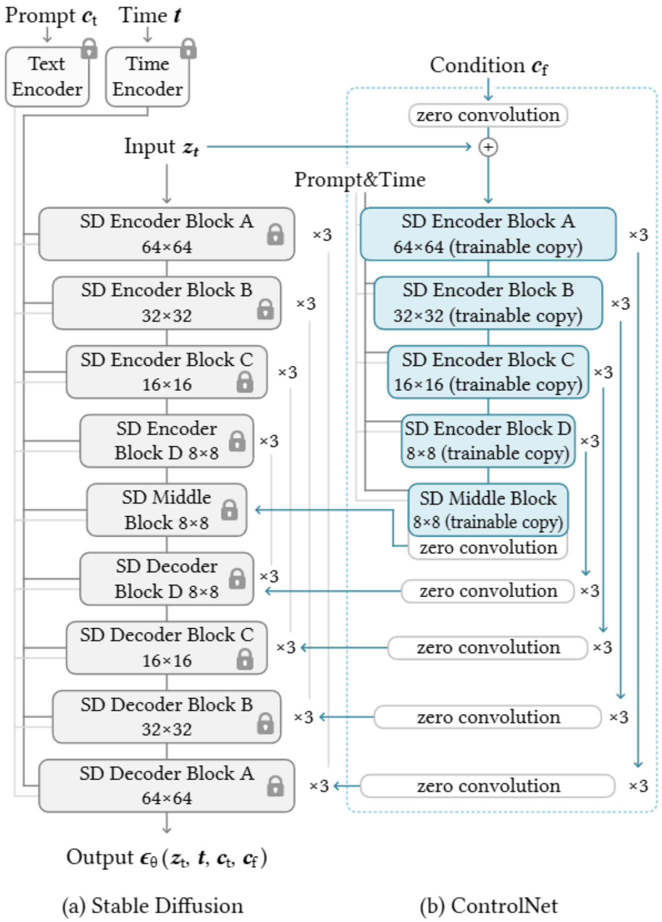
Diffusion in Application

How to Scale Up: Latent Diffusion Model



How to Control 1: ControlNet





[1] Adding Conditional Control to Text-to-Image Diffusion Models

How to Control 2: Classifier (or free) Guidance

The conditional score function:

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \nabla \log \left(\frac{p(\boldsymbol{x}_t)\, p(y \mid \boldsymbol{x}_t)}{p(y)} \right)$$
$$= \nabla \log p(\boldsymbol{x}_t) + \nabla \log p(y \mid \boldsymbol{x}_t) - \nabla \log p(y)$$
$$= \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{classifier gradient}}$$


CFG Scale : 1


CGG Scale : 3


CFG Scale : 7


CFG Scale : 10


CFG Scale : 15


CFG Scale : 20

The classifier guidance: Train an additional classifier

The classifier-free guidance (cfg): Train $s_\theta(x, y, t)$

$$\nabla_{\mathbf{x}_t} \log p(y \mid \mathbf{x}_t) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid y) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$$
$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right)$$

[1] Diffusion Models Beat GANs on Image Synthesis
[2] Classifier-Free Diffusion Guidance
[3] Tout savoir du CFG Scale. https://www.stablediffusion.blog/cfg-scale

How to Accelerate:

- The deterministic sampler:

  (a) Choose some important step to denoise (DDIM [1])

  (b) Using the form of reverse PFODE process (DPM-Solver Type Algorithm)

(a) DPM-Solver++ [32] (FID 18.59) (b) UniPC [58] (FID 12.24) (c) DPM-Solver-v3 (**Ours**) (FID 7.54)

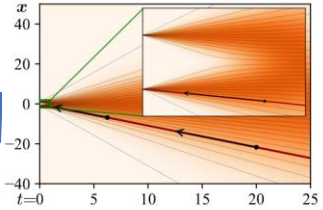
Figure 6: Random samples of Latent-Diffusion [43] on LSUN-Bedroom [55] with only NFE = 5.

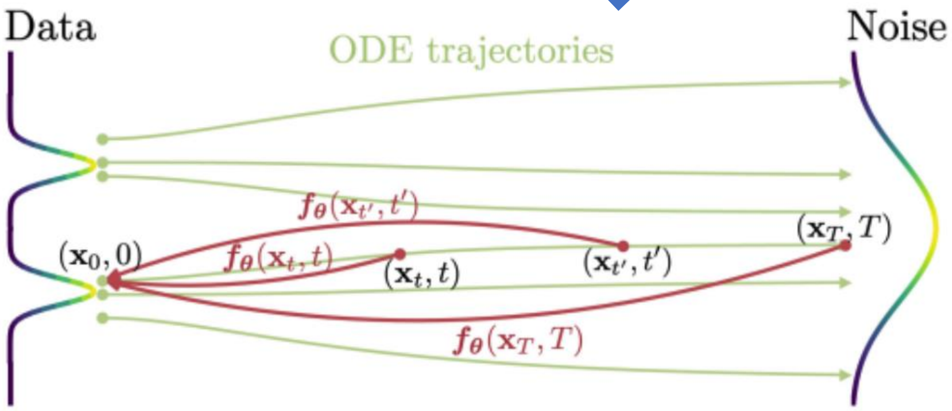
[1] Denoising Diffusion Implicit Models

[2] DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps (Following DPM-Solver-2 and 3)

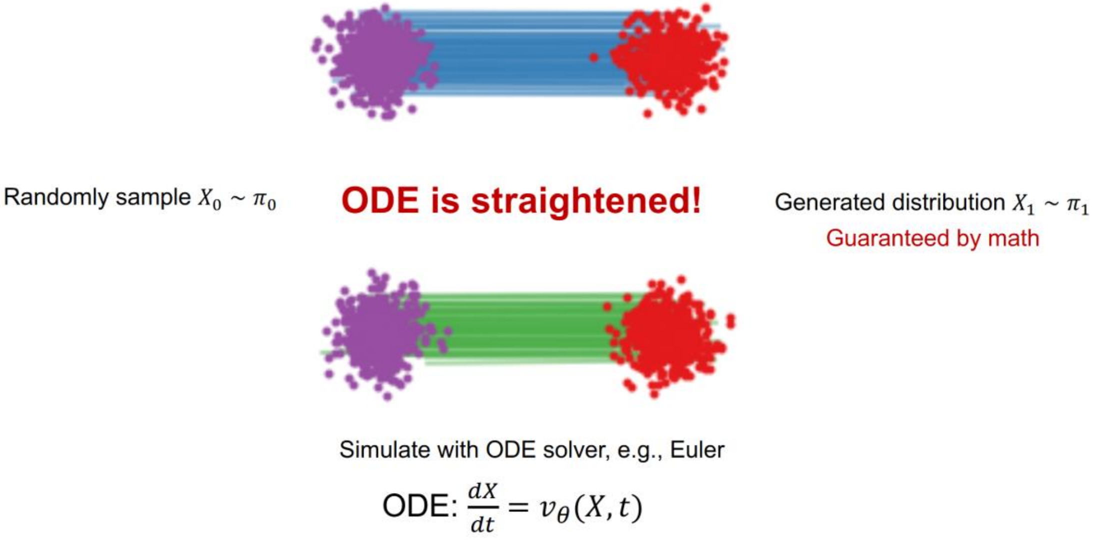
How to One-Step Sampling

- Consistency Models [1]
- InstaFlow [2]





Consistency Model



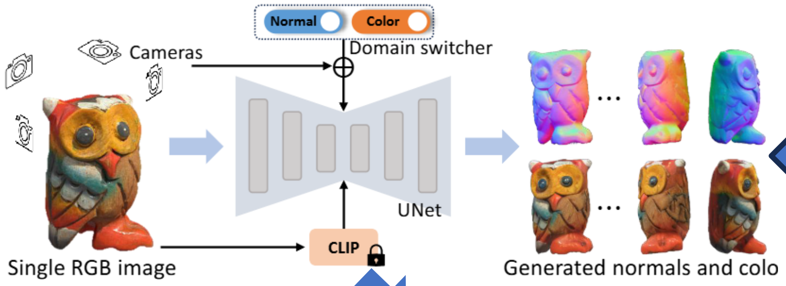
InstaFlow: $X_t = (1-t)X_0 + tZ, t \in [0,1]$

Question: Is the linear solution curve necessary?

[1] Consistency Model; Simplifying, Stabilizing and Scaling Continuous-Time Consistency Models
[2] Instaflow: One step is enough for high-quality diffusion-based text-to-image generation

How to 3D and Video (Supervised Fine-tuning)

- Here, we use 3D task as an example to show how to use 2D prior of SD.



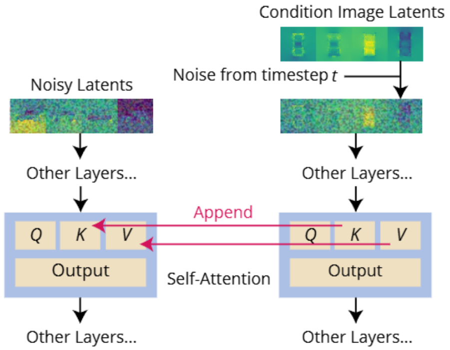
score-matching

loss

Ground-truth

Multi-view Images

Reference-Only Attention



- It is similar for video generation (additional spatial and temporal modules)

[1] Wonder3D: Single Image to 3D using Cross-Domain Diffusion

[2] Zero123++: a Single Image to Consistent Multi-view Diffusion Base Model

[3] VideoCrafter2: Overcoming Data Limitations for High-Quality Video Diffusion Models

[4] Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets

How to Alignment: RLHF in Diffusion

- View the denoised process as a MDP.

$$
\begin{aligned}
&\mathbf{s}_t \triangleq (\boldsymbol{c}, t, \boldsymbol{x}_{T-t}) \qquad P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t) \triangleq \left(\delta_{\boldsymbol{c}}, \delta_{t+1}, \delta_{\boldsymbol{x}_{T-1-t}}\right) \\
&\mathbf{a}_t \triangleq \boldsymbol{x}_{T-1-t} \qquad \pi(\mathbf{a}_t \mid \mathbf{s}_t) \triangleq p_\theta(\boldsymbol{x}_{T-1-t} \mid \boldsymbol{c}, t, \boldsymbol{x}_{T-t}) \\
&\rho_0(\mathbf{s}_0) \triangleq (p(\boldsymbol{c}), \delta_0, \mathcal{N}(\mathbf{0}, \mathbf{I})) \\
&r(\mathbf{s}_t, \mathbf{a}_t) \triangleq r((\boldsymbol{c}, t, \boldsymbol{x}_{T-t}), \boldsymbol{x}_{T-t-1})
\end{aligned}
$$

- Then, PPO or DPO

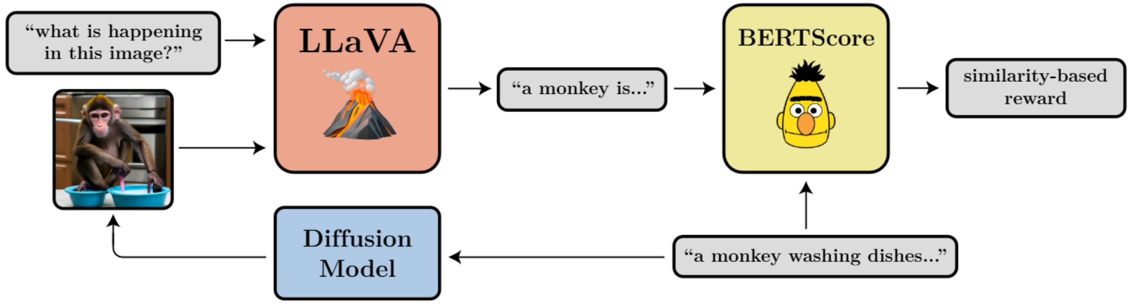


Figure 1. **An illustration of our annotation UI**. Annotators mark points on the image to indicate artifact/implausibility regions (red points) or misaligned regions (blue points) w.r.t the text prompt. Then, they click on the words to mark the misaligned keywords (underlined and shaded) and choose the scores for plausibility, text-image alignment, aesthetics, and overall quality (underlined).

RichHF-18K dataset [1]

[1] Rich Human Feedback for Text-to-Image Generation, CVPR 24 best paper

[2] Training Diffusion Models with Reinforcement Learning

How to RL 1: The Diffusion Policy

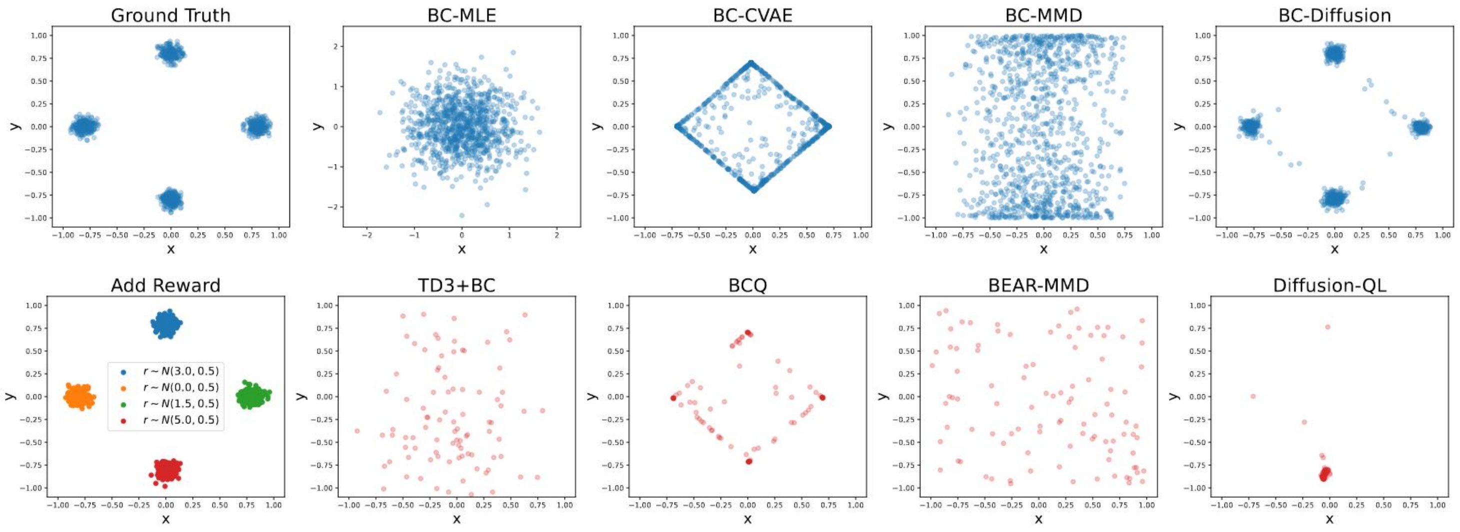
- View the conditional diffusion as policy: condition on state s to generate a.

$$\pi = \arg\min_{\pi_\theta} \mathcal{L}(\theta) = \boxed{\mathcal{L}_d(\theta)} + \boxed{\mathcal{L}_q(\theta)} = \mathcal{L}_d(\theta) - \alpha \cdot \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D},\boldsymbol{a}^0\sim\pi_\theta}\left[Q_\phi(\boldsymbol{s},\boldsymbol{a}^0)\right].$$

Diffusion loss

(BC term)

Max Q function

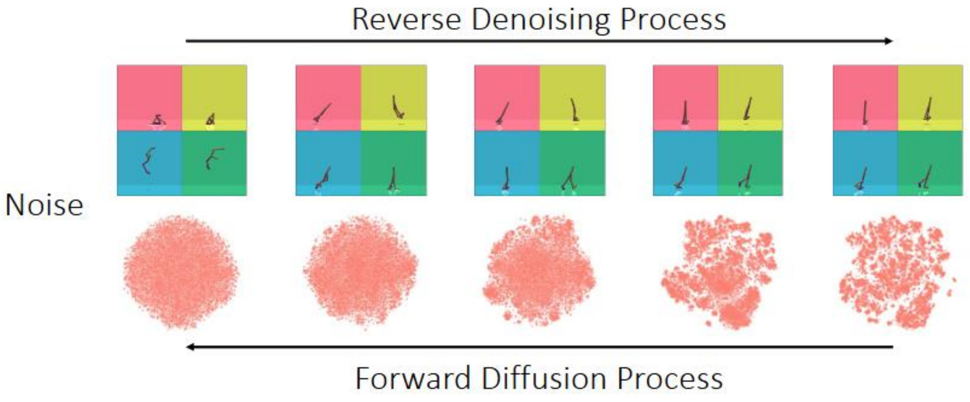


[1] Diffusion Policies as an Expressive Policy Class for Offline Reinforcement Learning

[2] Consistency Models as a Rich and Efficient Policy Class for Reinforcement Learning

How to RL 2: The Diffusion Augmentation

[1]



Algorithm 1 SYNTHER for online replay-based algorithms. Our additions are highlighted in **blue**.

Input: real data ratio $r \in [0, 1]$
Initialize: $\mathcal{D}_{\text{real}} = \emptyset$ real replay buffer, π agent, $\mathcal{D}_{\text{synthetic}} = \emptyset$ synthetic replay buffer, M diffusion model
for $t = 1, \ldots, T$ **do**
  Collect data with π in the environment and add them to $\mathcal{D}_{\text{real}}$
  Update diffusion model M with samples from $\mathcal{D}_{\text{real}}$
  Generate samples from M and add them to $\mathcal{D}_{\text{synthetic}}$
  Train π on samples from $\mathcal{D}_{\text{real}} \cup \mathcal{D}_{\text{synthetic}}$ mixed with ratio r
end for

[2]

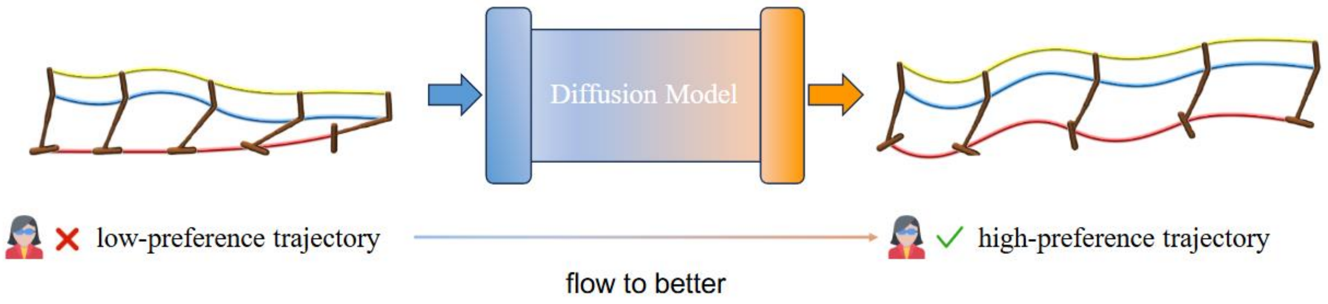


Figure 1: Illustration of the key idea of our method. Given a low-preference trajectory (left), the FTB model generates a higher-preference trajectory (right).

[1] Synthetic Experience Replay

[2] Flow to Better: Offline Preference-based Reinforcement Learning via Preferred Trajectory Generation

How to Start: Some tutorials and Code

- DDPM Demo: https://github.com/lucidrains/denoising-diffusion-pytorch

- Score SDE Demo: https://colab.research.google.com/drive/17lTrPLTt_0EDXa4hkbHmbAFQEkpRDZnh?usp=sharing

- Diffusers: common diffusion pipelines

- Stable Diffusion (base on ldm): https://github.com/Stability-AI/stablediffusion