Diffusion Models for Reinforcement Learning

Weinan Zhang

Shanghai Jiao Tong University

http://wnzhang.net

May 2024

Survey Paper. https://arxiv.org/abs/2311.01223
Github Repo. https://github.com/apexrl/Diff4RLSurvey

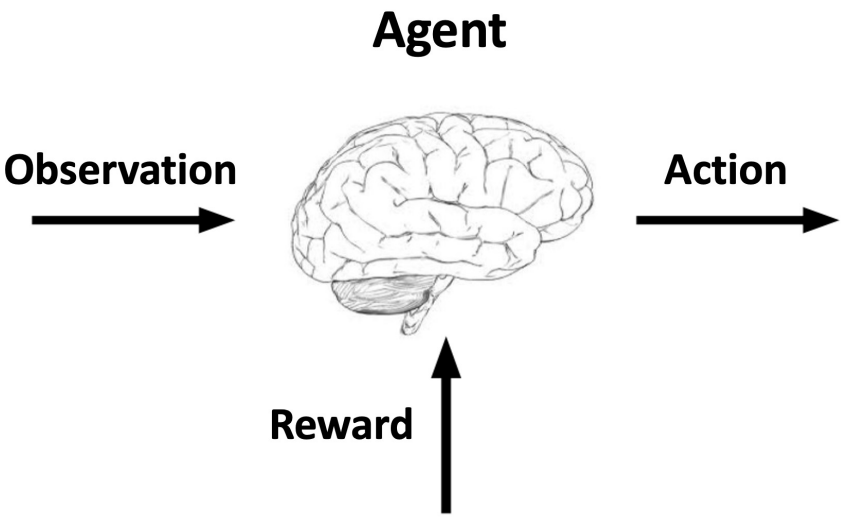
Content

1. Reinforcement Learning and its Challenges
2. Foundations of Diffusion Models
3. Roles of Diffusion Models in RL
   - Planners
   - Policies
   - Data Synthesizer
   - Others
4. Applications in RL and Related Tasks
   - Reinforcement Learning
   - Imitation Learning
   - Data Augmentation
5. Future Developments

Reinforcement Learning and its Challenges

Reinforcement Learning (RL)

A computational method that learn to achieve objectives through **interactions**

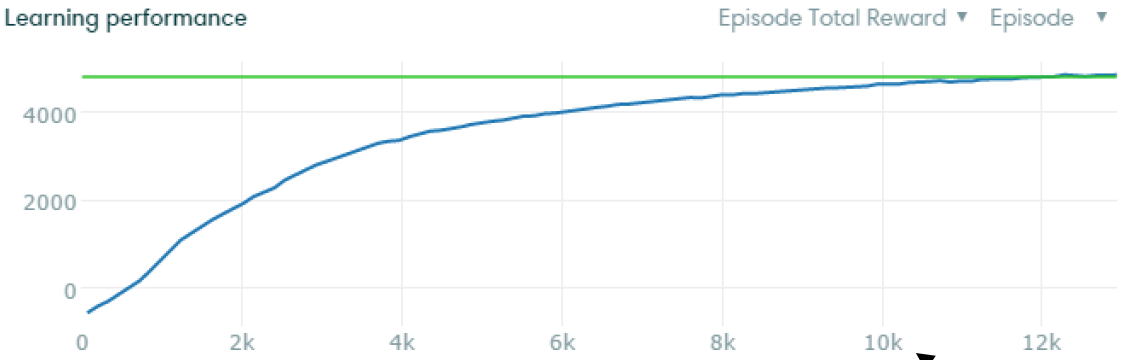


Three Aspects:

- **Perceive**: perceive the state of the environment to some extent
- **Act**: can take actions to influence states or achieve goals
- **Objective**: maximize cumulative rewards as time passes

Low Sample Efficiency of RL

Mujoco HalfCheetah



TRPO + GAE

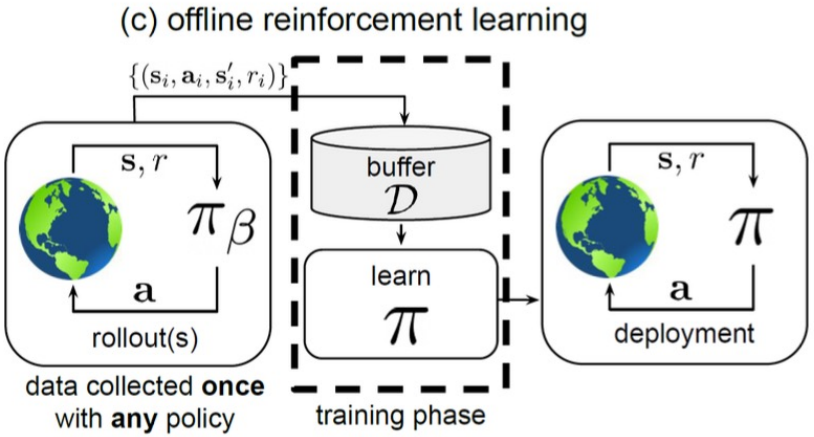
10,000,000 steps
(10,000 episodes)
(~1.5 days real time)

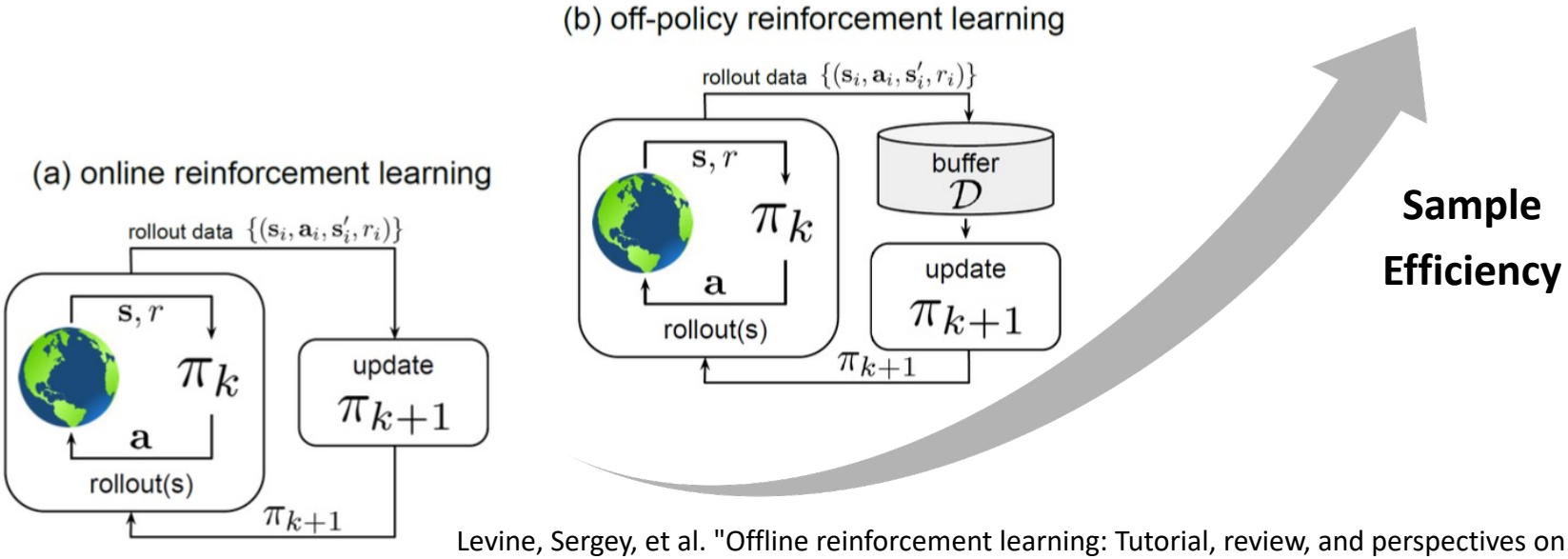
- On-policy RL algorithms requires many interactions to learn a good policy, which is quite sample inefficient considering the time and potential cost
- Precludes using RL in complex simulated tasks and real-world applications

Figure: CS285. Lecture 23

Experience Reuse can Improve Sample Efficiency

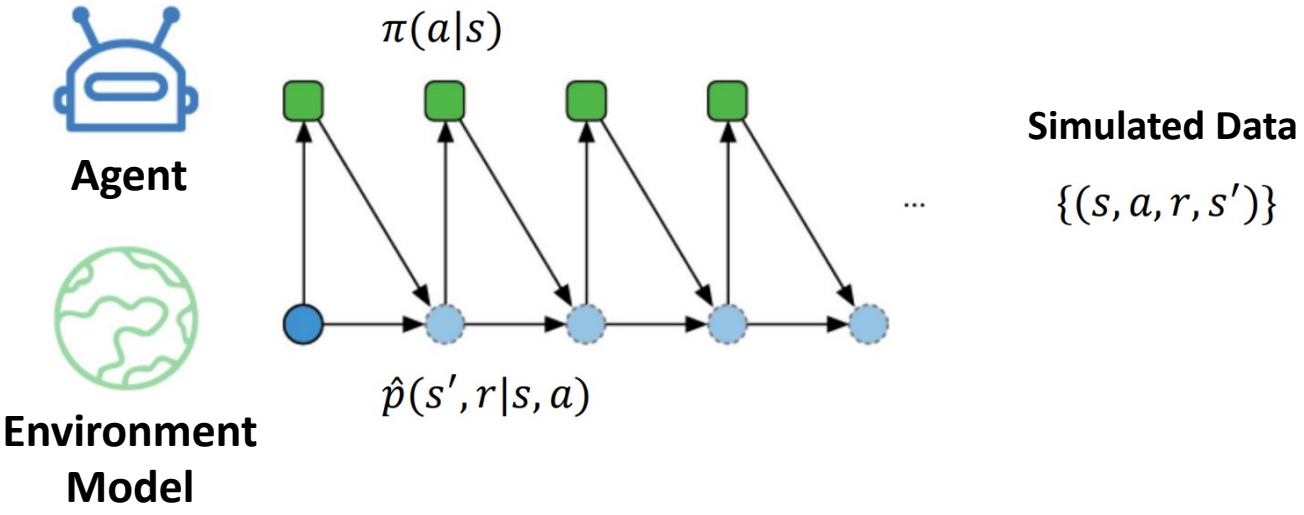
- Though both off-policy RL and offline RL evaluate the policy using the data sampled from a replay buffer, they are different
- The key difference is whether the agent can interact with the environment while learning





Levine, Sergey, et al. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems." *arXiv 2005.01643* (2020).

Model-based RL (MBRL)



- Model-based RL is another attempt to improve sample efficiency in RL
- There are two typical usage of the learned model:
  - Planning with model simulation (MPC, MuZero, …)
  - Data augmentation with model simulation (MBPO, BMPO, …)

Sample-efficient RL: Are We There Yet?

- Restricted Expressiveness in Offline Learning

Directly applying off-policy RL to offline settings leads to extrapolation error problem in Bellman updates:

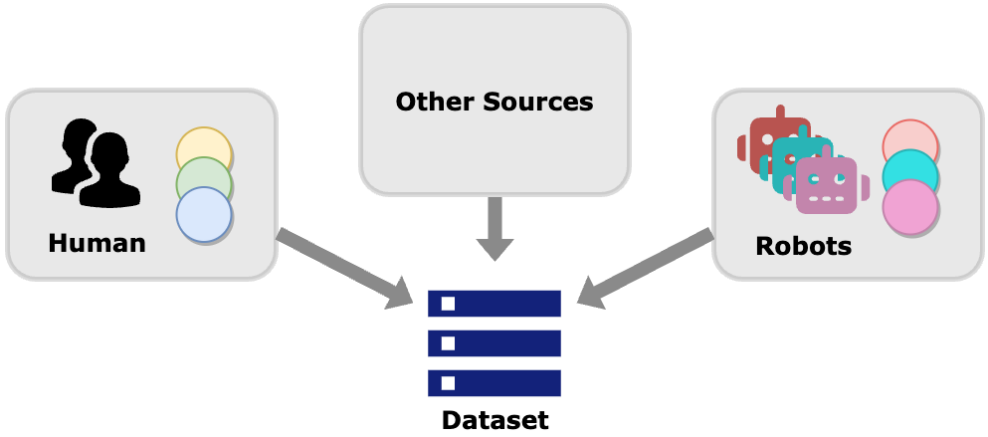
$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}' \sim \mathcal{D}} \left[\left((r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}' \sim \hat{\pi}^k(\mathbf{a}'|\mathbf{s}')} [\hat{Q}^k(\mathbf{s}', \mathbf{a}')]) - Q(\mathbf{s},\mathbf{a}) \right)^2 \right]$$

There are two (model-free) ways to address the problem:

- penalizing the value predictions on out-of-distribution actions
  - **can be over-conservative & requires careful tuning**
- regularizing the policy to not deviate too much from the behavior policy (data-collect policy)

Sample-efficient RL: Are We There Yet?

■ Restricted Expressiveness in Offline Learning



In offline RL, offline datasets are often collected by a mixture of policies, the behavior policy may exhibit:

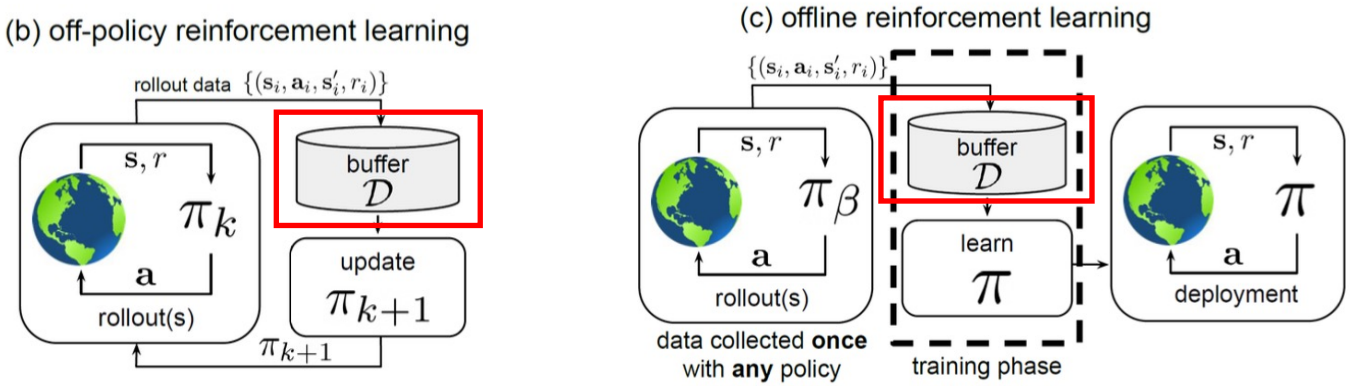
- strong multi-modalities,
- skewness,
- dependencies between different action dimensions,

which cannot be well modeled by diagonal Gaussian policies.

We need a policy class that can model very complex distributions!

Sample-efficient RL: Are We There Yet?

■ Data Scarcity in Experience Replay

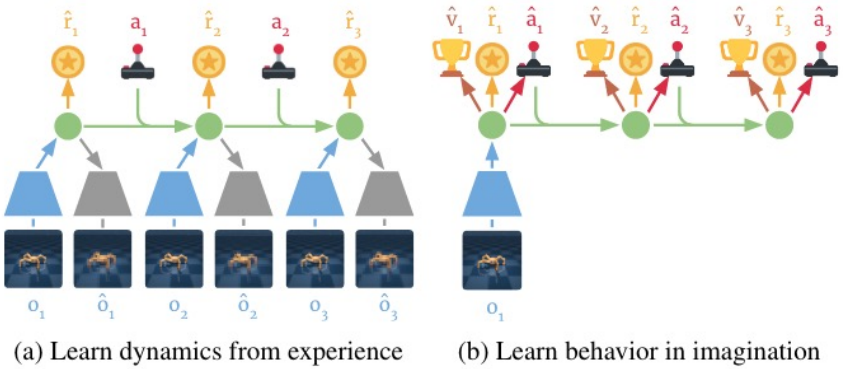


- Both off-policy and offline RL learns from replay buffers with limited data (given limited environment interactions)
- Perform data augmentation on replay buffers?
  - **random augmentations** (crop, rotate, ...)
    - lack of fidelity
  - **distribution modeling** (GAN, VAE)
    - limited to simple environments

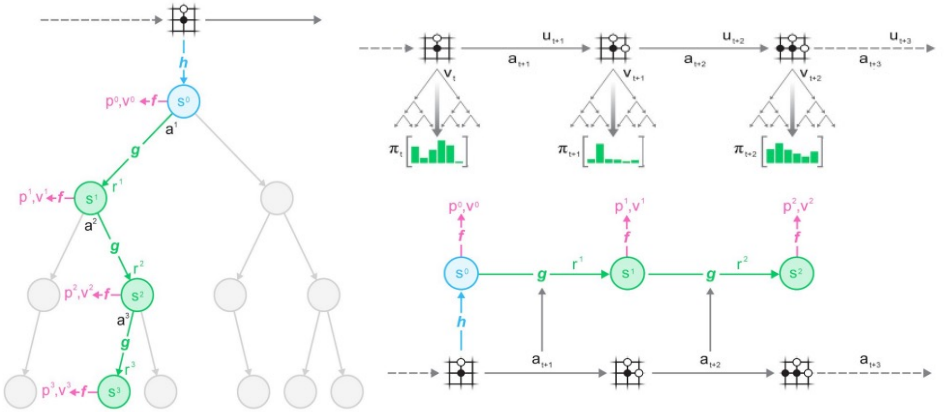
We need augmentation methods that can synthesize realistic and diverse data in complex tasks!

Sample-efficient RL: Are We There Yet?

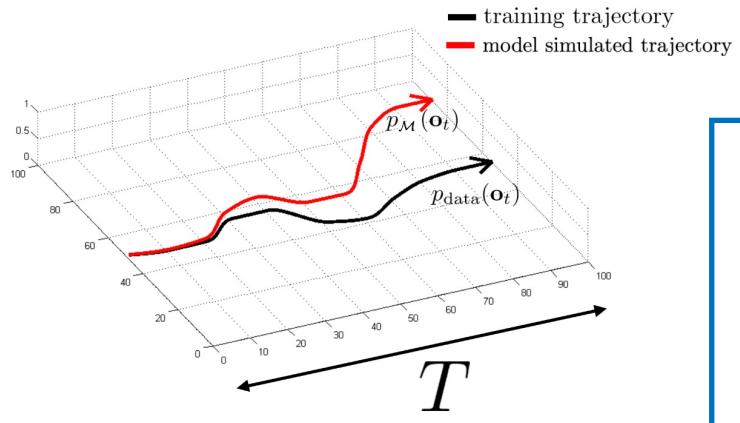
- Compounding Error in Model-based Planning



Dreamer



MuZero



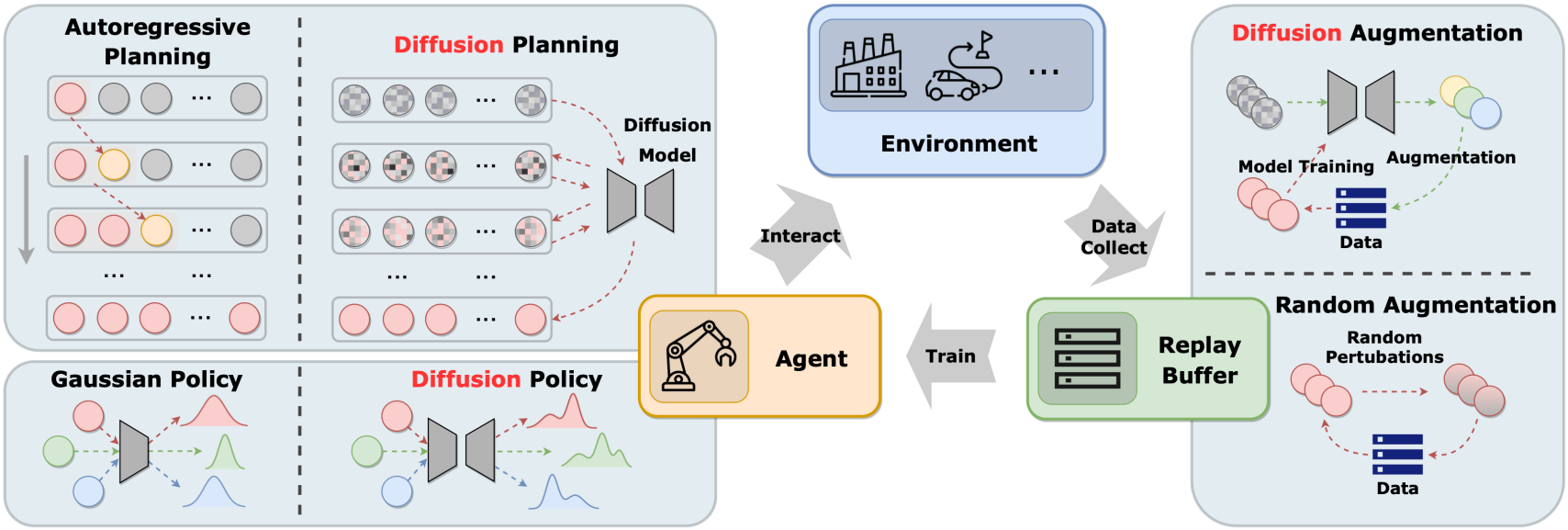
Auto-regressive planning/rollout in a learned dynamic model can cause **compounding errors**

We need to go beyond single-step prediction and auto-regressive planning to match the distributions of model-simulated and training trajectories!

Diffusion Models for Sample-efficient RL

Summary of Challenges in sample-efficient RL:

- Restricted Expressiveness in Offline Learning
- Data Scarcity in Experience Replay
- Compounding Error in Model-based Planning



Use **diffusion planning**, **diffusion policy** and **diffusion augmentation**!

Foundations of Diffusion Models

Foundations of Diffusion Models

Prominent Diffusion Formulations

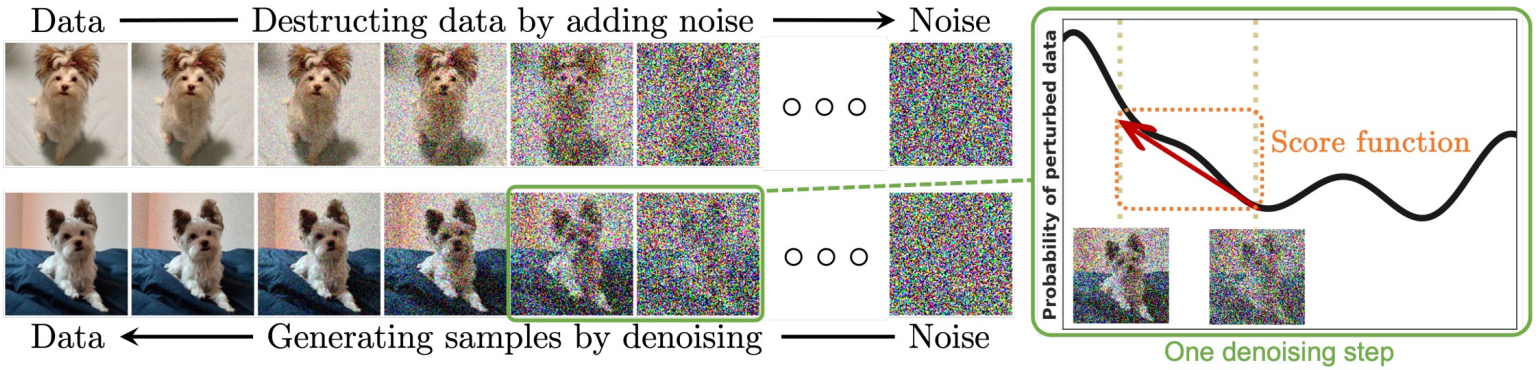
- Denoising Diffusion Probabilistic Models (DDPM)
- Score-based Diffusion Models

Guided Sampling Methods

- Classifier Guidance
- Classifier-free Guidance

Diffusion Models

- An efficient generative method with superior sampling quality and training stability
- Generate samples from pure noises with multi-step denoising

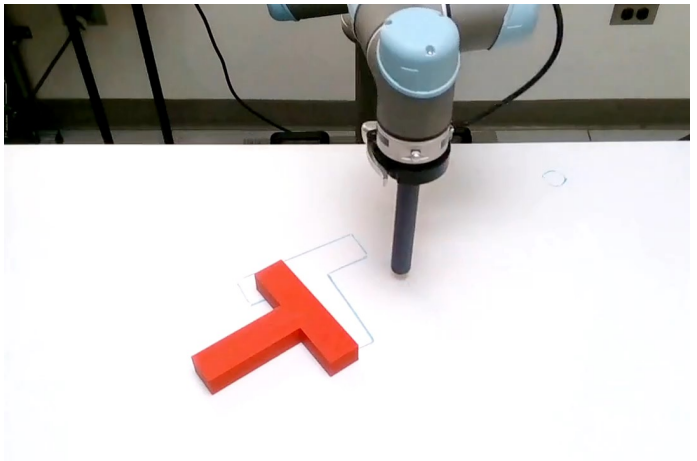


Yang, Ling, et al. "Diffusion models: A comprehensive survey of methods and applications." ACM Computing Surveys 56.4 (2023): 1-39.

Diffusion Models

Text-to-image Generation

Text-to-video Generation



Robotics Manipulation

Denoising Diffusion Probabilistic Models

Sampling & Diffusion process:

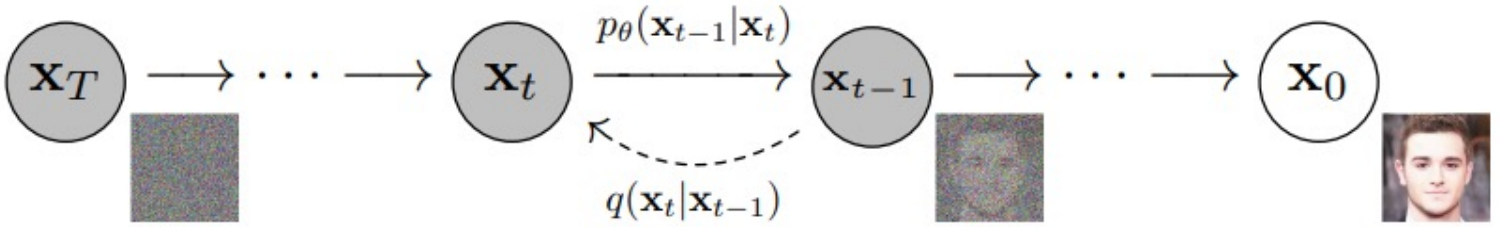
$$p_\theta(x_0) = \int p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)\, dx_{1:T}\,, p(x_T) = \mathcal{N}(0, I)$$

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

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\, \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$$

$$\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

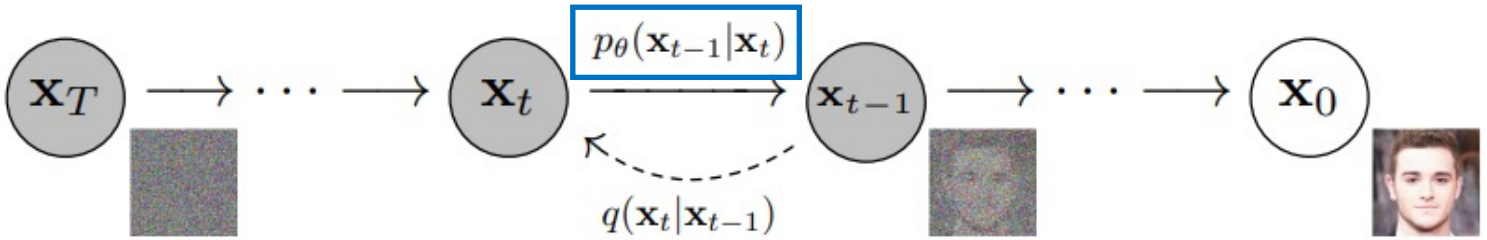
reverse process or sampling process



forward process or diffusion process

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

Denoising Diffusion Probabilistic Models



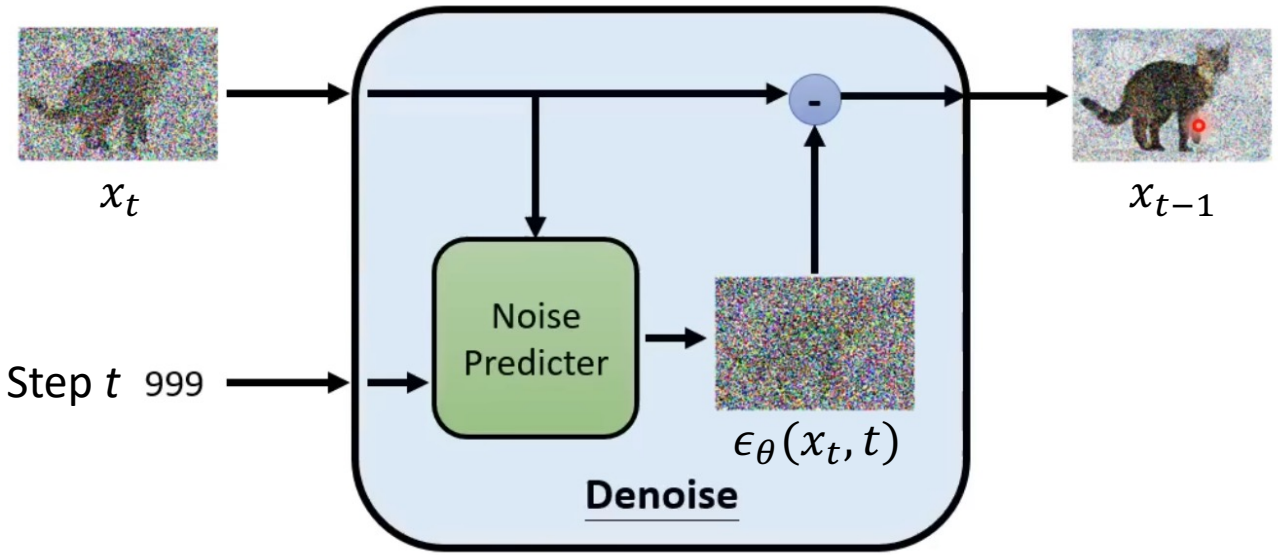
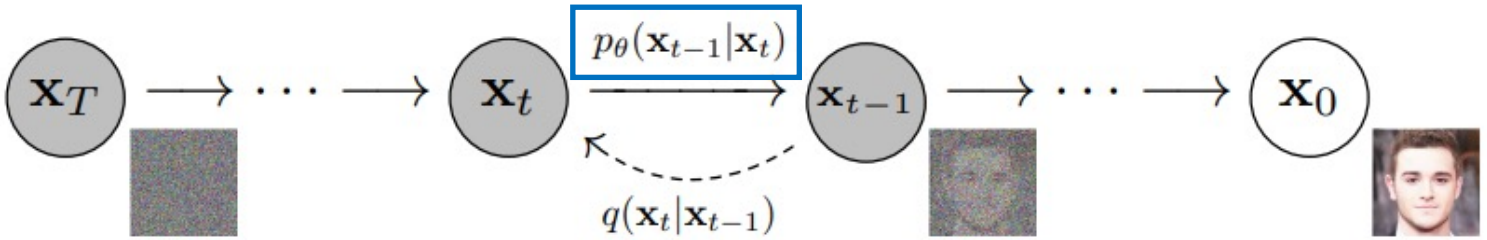


Image from Prof. Hung-yi Lee

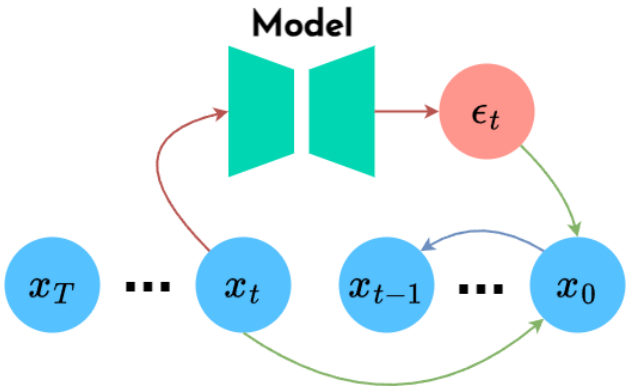
Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

Denoising Diffusion Probabilistic Models



Use learned model in sampling:

If the diffusion model is designed to predict the noise, the sampling process is alternating between **recovering the (approximated) clean sample** and **jump back to the previous sample**.



Use a neural network $\epsilon_\theta(x_t, t)$ to predict the noise at step t:

$$\mathbb{E}_{t\sim\mathcal{U}[\![1,T]\!],\mathbf{x}_0\sim q(\mathbf{x}_0),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\lambda(t)\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right\|^2\right]$$

$$\mathbf{x}_{t-1}=\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right)+\sigma_t\mathbf{z}$$

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

Guided Sampling Methods

- Instead of sampling from the dataset distribution $p(x)$, *guided sampling* samples from $p(x|y)$.
- The conditional reverse process can be written as

$$p_{\theta,\phi}(x^{t-1}|x^t, y) = Z p_\theta(x^{t-1}|x^t) p_\phi(y|x^{t-1})$$

where Z is the normalization factor.

- The reverse process can be approximated by another Gaussian distribution with a different mean:

$$p_{\theta,\phi}(x^{t-1}|x^t, y) = \mathcal{N}(\mu^t + w\Sigma^t g, \Sigma^t)$$

where $g = \nabla_{x^t} \log p_\phi(y|x^t)|_{x^t=\mu^t}$ and w is the guidance scale.

Dhariwal, Prafulla, and Alexander Nichol. “Diffusion models beat gans on image synthesis.” NeurIPS 2021.

Guided Sampling Methods

$$p_{\theta,\phi}(x^{t-1}|x^t, y) = \mathcal{N}(\mu^t + w\Sigma^t g, \Sigma^t) \qquad g = \nabla_{x^t} \log p_\phi(y|x^t)|_{x^t=\mu}$$

Classifier guided sampling

- Use the gradient from an extra classifier $p_\phi(y|x^t)$

Classifier-free guided sampling

- The noise prediction target is a scaled score function of $p(x^t)$:

$$\epsilon(x^t, t) = -\sigma_t \nabla_{x^t} \log p(x^t)$$

variance schedule $\beta^{1:T}$

$\sigma^t = \sqrt{\beta_t}$

$\alpha^t = 1 - \beta^t$

- From Bayes theorem, we have

$$\nabla_{x^t} \log p(y|x^t) = -1/\sigma_t(\epsilon(x^t, y, t) - \epsilon(x^t, t))$$

- The noise in classifier-free guided sampling becomes

$$\hat{\epsilon}_w(x^t, y, t) = w\epsilon_\theta(x^t, y, t) + (1-w)\epsilon_\theta(x^t, t)$$

simply input a null token ∅

Ho, Jonathan, and Tim Salimans. "Classifier-free diffusion guidance." arXiv preprint arXiv:2207.12598 (2022).

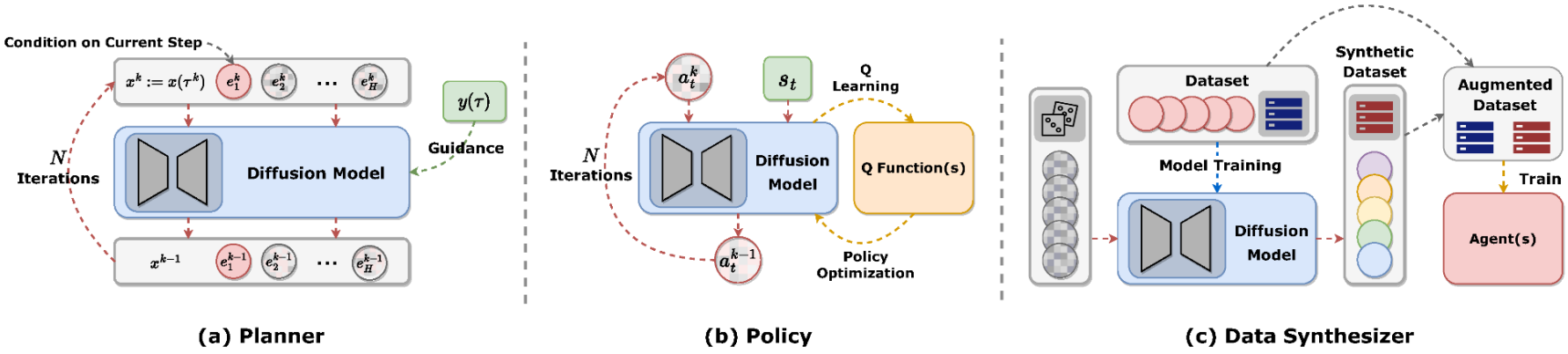
Roles of Diffusion Models

in Reinforcement Learning

Roles of Diffusion Models in RL

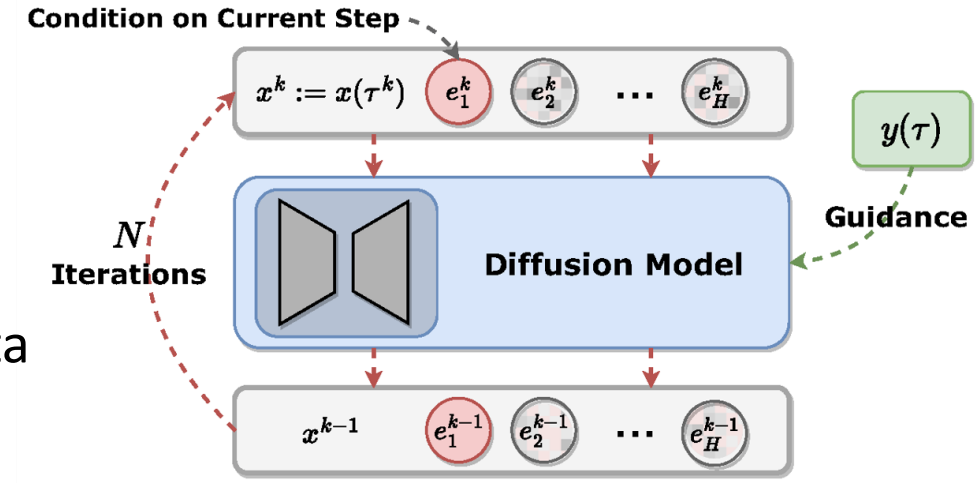
Current works mainly fall into three categories

- Planner
- Policy
- Data Synthesizer



Diffusion Planner

- Planning: making decisions imaginarily
- Traditional planners
  - Autoregressive, suffering from compounding error
  - Learn from Markovian data
- Diffusion model planners
  - Non-autoregressive
  - Multi-modal
  - Can learn from non-Markovian data by adding temporal information

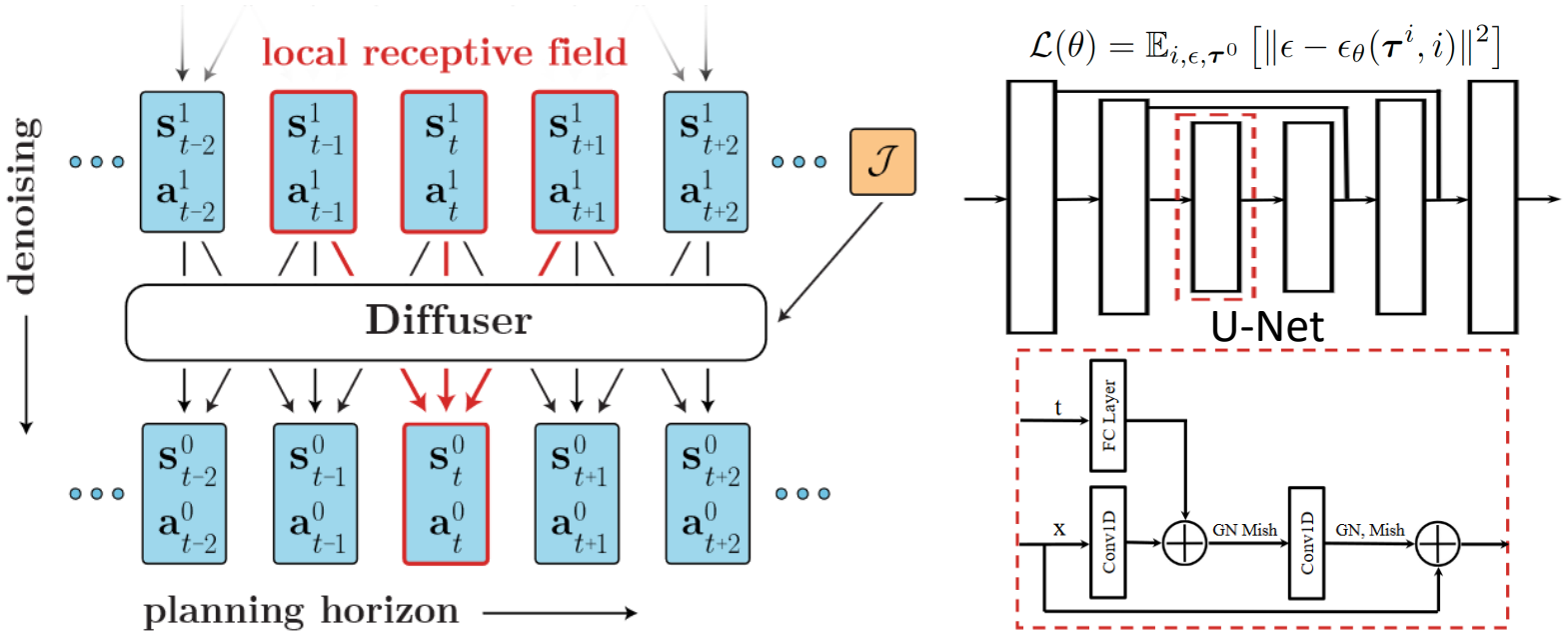


Notations of Diffusion Planners

- Clip of a trajectory $\tau = (s_1, a_1, r_1, \dots, s_H, a_H, r_H)$.
- Diffusion models work on clips $x(\tau) = (e_1, e_2, \dots, e_H)$.
  - Basic elements e vary from tasks. They can be (s, a, r), (s, a), s, or a
  - Generating **multiple** steps -> **planner**
  - Generating a **single**-step action -> **policy**
- Guidance $y(\tau)$ contains desired properties of τ
  - Accumulative discounted reward until the terminal (i.e., return)
  - Whether the task is complete
  - Safe constraints
- Either classifier guidance or classifier-free guidance can be used

Why Non-autoregressive Works

- Taking U-Net as an example
  - **Local consistency:** e_t^k takes information in a local receptive area $e_{t-l:t+l}^{k+1}$
  - **Global consistency:** such information spreads along the trajectory during the multi-step reverse process



Janner, Michael, et al. “Planning with Diffusion for Flexible Behavior Synthesis.” ICML 2022.

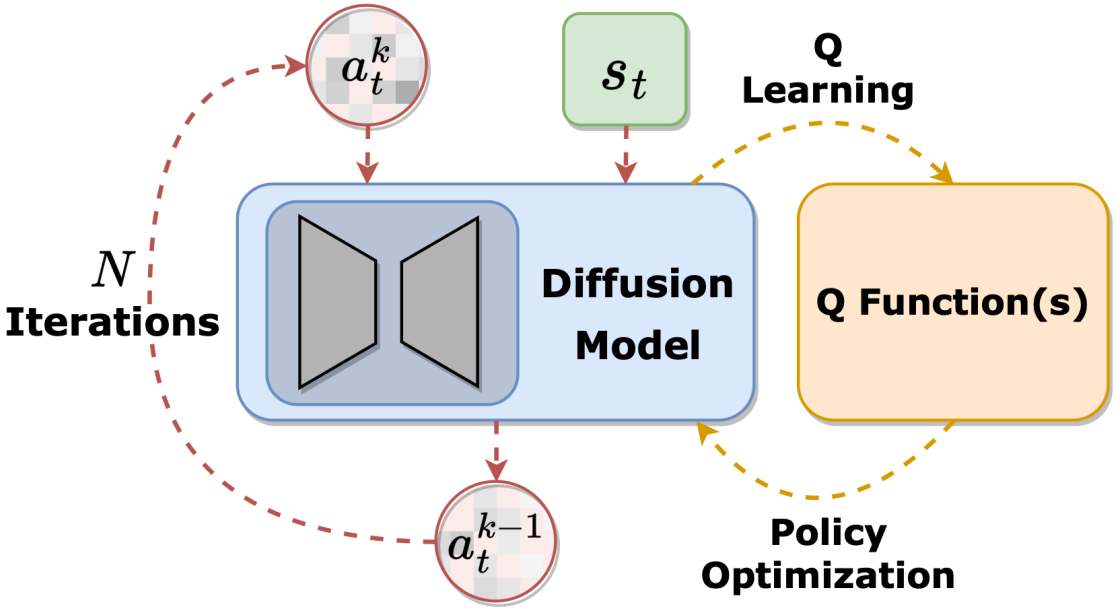
Why Non-autoregressive Works

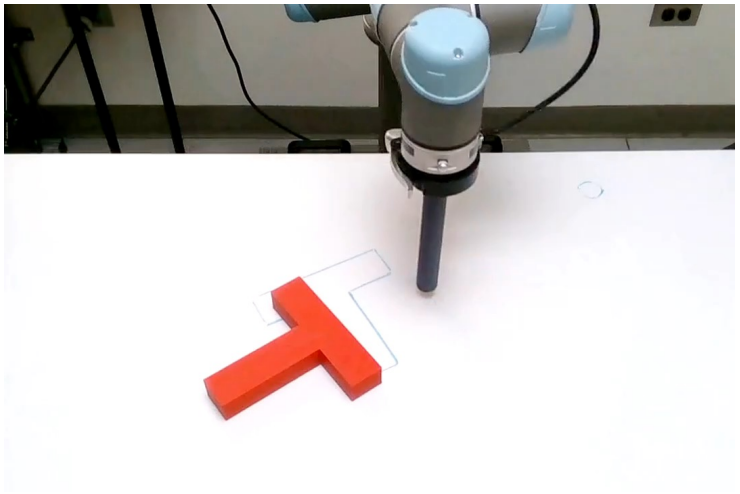
Non-autoregressive generation seems anti-causal

- Why the current step can be generated conditioned on future information?
- In fact, RL itself includes a hypothesis that the current decision will lead to optimal future outcomes.
- Think about whether the route planning of human is autoregressive? Not really.

Diffusion Policy

- Use diffusion model as the parametrized policy class to output single-step actions
- Can model arbitrary distributions
- Train diffusion model on the dataset distribution



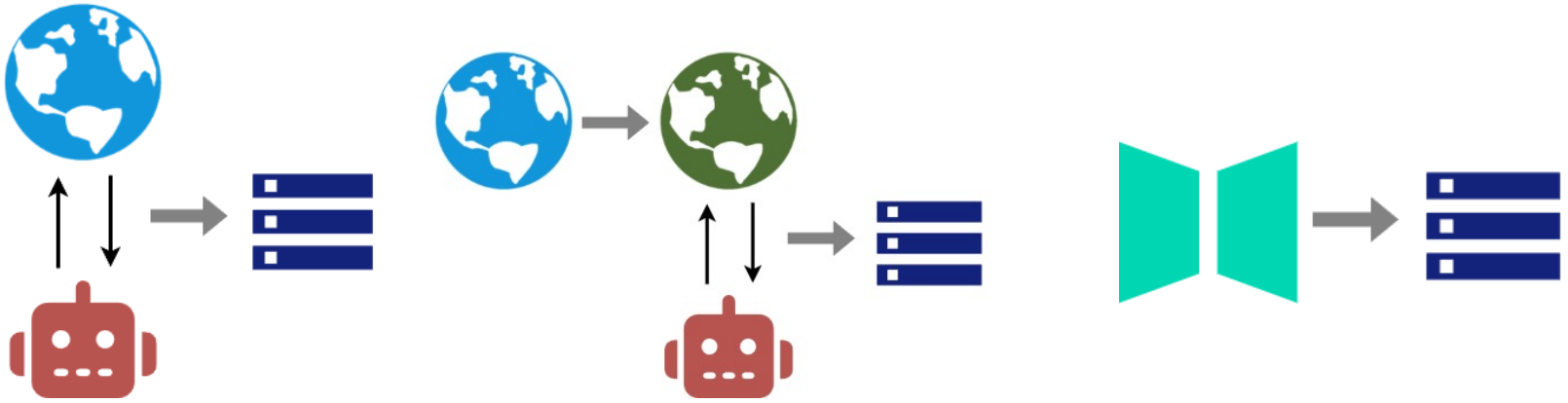


Optimize the Diffusion Policy

- When learning from mixed-quality datasets, the diffusion loss is pure behavior cloning, and additional policy optimization is needed
- How to optimize the generated actions?
  - Guided sampling
  - Q-learning
  - Weighted regression
  - Policy extraction (*e.g.*, combine DMs with IQL for multi-modal policy building)
  - Sample multiple candidates then reweight / select

Diffusion Data Synthesizer

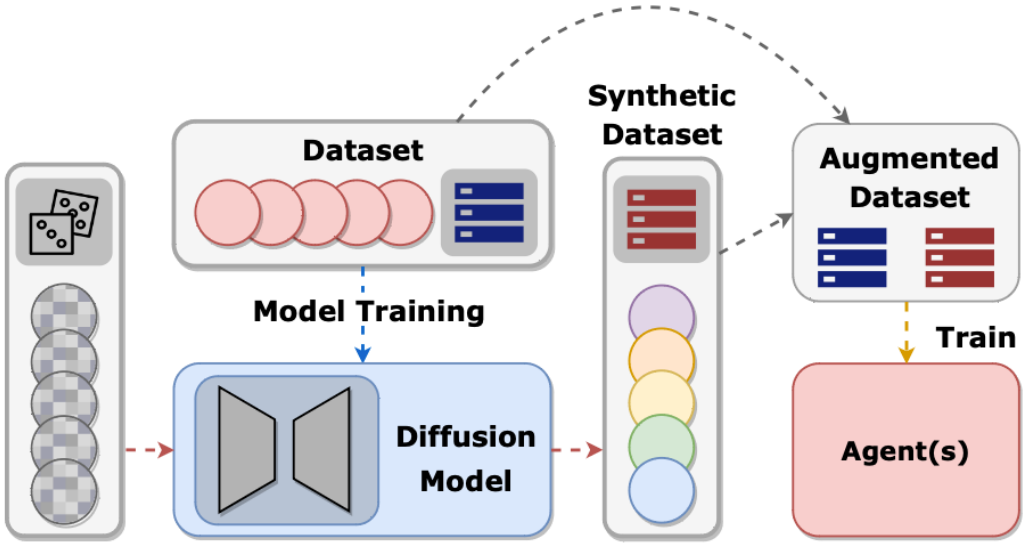
Synthesizing a large amount of high-quality data is the key to scaling up our agents.



Real world	Simulator	Generative models
Limited data	Infinite data but time ineffective	Infinite and fast

How to Synthesize

- Leverage diffusion models to learn from the original dataset
- Synthesize new samples to augment the dataset



$$\mathcal{D}_{\text{syn}} = \{\tau \sim \rho_\theta(\tau)\}$$

$$\rightarrow \quad \mathcal{D} = \mathcal{D}_{\text{real}} \cup \mathcal{D}_{\text{syn}}$$

Other Roles

- **Value functions:** DVF uses diffusion models as value functions by learning the discounted state occupancy
- **Latent representations:** LDCQ applies diffusion modeling on latent embeddings of trajectories
- **Dynamics models:** PolyGRAD builds a diffusion dynamics model and allow an online RL policy to collect synthetic trajectories on it
- **Reward functions:** A relative reward function can be extracted from two diffusion models (base and expert models)

Mazoure, Bogdan, et al. "Value function estimation using conditional diffusion models for control." *arXiv preprint arXiv:2306.07290* (2023).
Venkatraman, Siddarth, et al. "Reasoning with Latent Diffusion in Offline Reinforcement Learning." The Twelfth International Conference on Learning Representations. 2023.
Rigter, Marc, Jun Yamada, and Ingmar Posner. "World models via policy-guided trajectory diffusion." *arXiv preprint arXiv:2312.08533* (2023).
Nuti, Felipe, Tim Franzmeyer, and João F. Henriques. "Extracting reward functions from diffusion models." Advances in Neural Information Processing Systems 36 (2024).

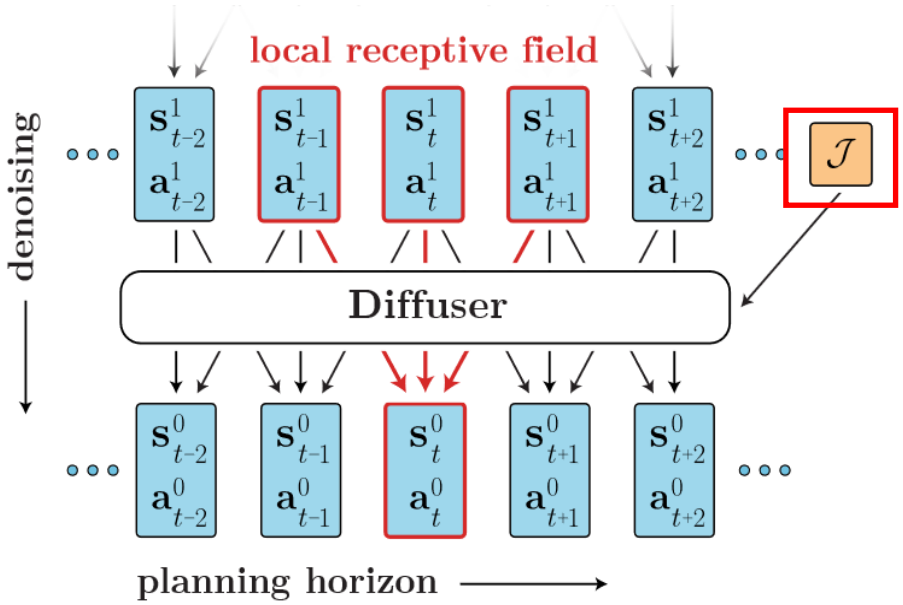
Applications of DMs for RL

Elaborate on specific methods according to problem settings

Planner

Offline Reinforcement Learning

- **Diffuser** generates optimal trajectories through iteratively denoising process with classifier-guided sampling.



$$p_\theta(\boldsymbol{\tau}^{i-1} \mid \boldsymbol{\tau}^i, \mathcal{O}_{1:T}) \approx \mathcal{N}(\boldsymbol{\tau}^{i-1}; \mu + \Sigma g, \Sigma)$$

$$g = \nabla_{\boldsymbol{\tau}} \log p(\mathcal{O}_{1:T} \mid \boldsymbol{\tau})|_{\boldsymbol{\tau}=\mu}$$

$$= \sum_{t=0}^{T} \nabla_{\mathbf{s}_t, \mathbf{a}_t} r(\mathbf{s}_t, \mathbf{a}_t)|_{(\mathbf{s}_t, \mathbf{a}_t)=\mu_t} = \nabla \mathcal{J}(\mu).$$

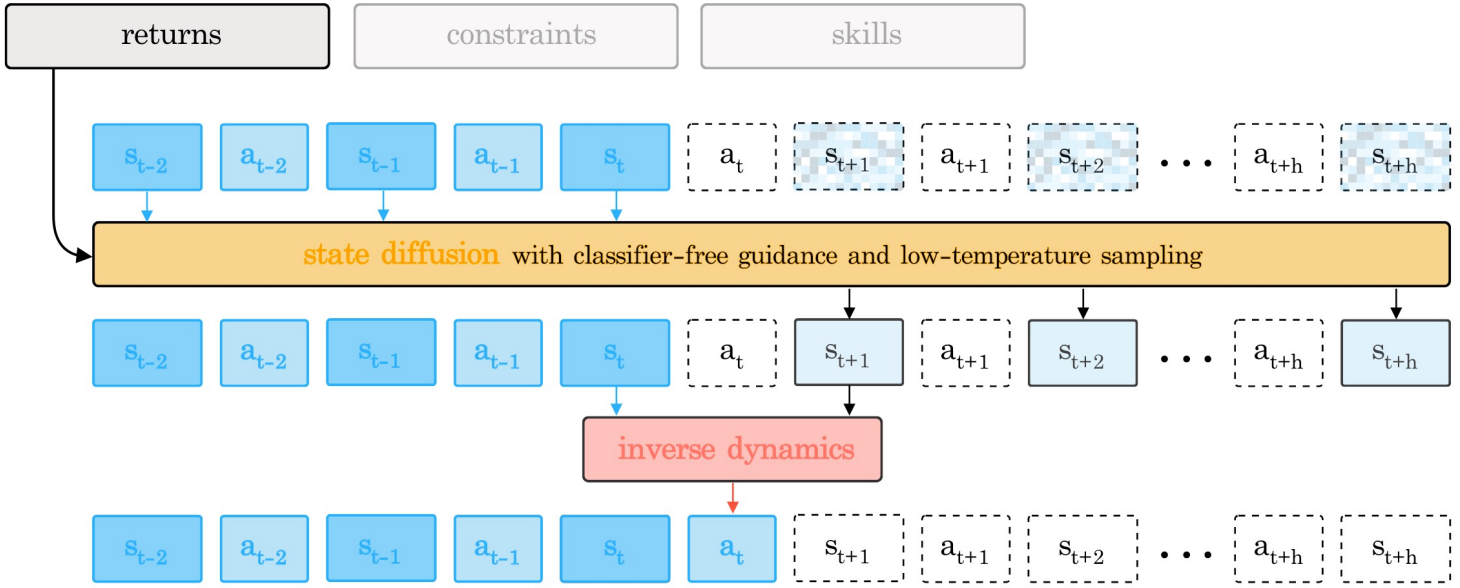
- Diffuser separately trains a classifier model and use its gradient to guide the sampling towards high-return trajectories.

Janner, Michael, et al. “Planning with Diffusion for Flexible Behavior Synthesis.” ICML 2022.

Planner

Offline Reinforcement Learning

- **Decision Diffuser** adopts a classifier-free guided diffusion model which only generates state sequences.



- Action sequences are less smooth and harder to model. Instead, each action is predicted by a separate inverse dynamics model.

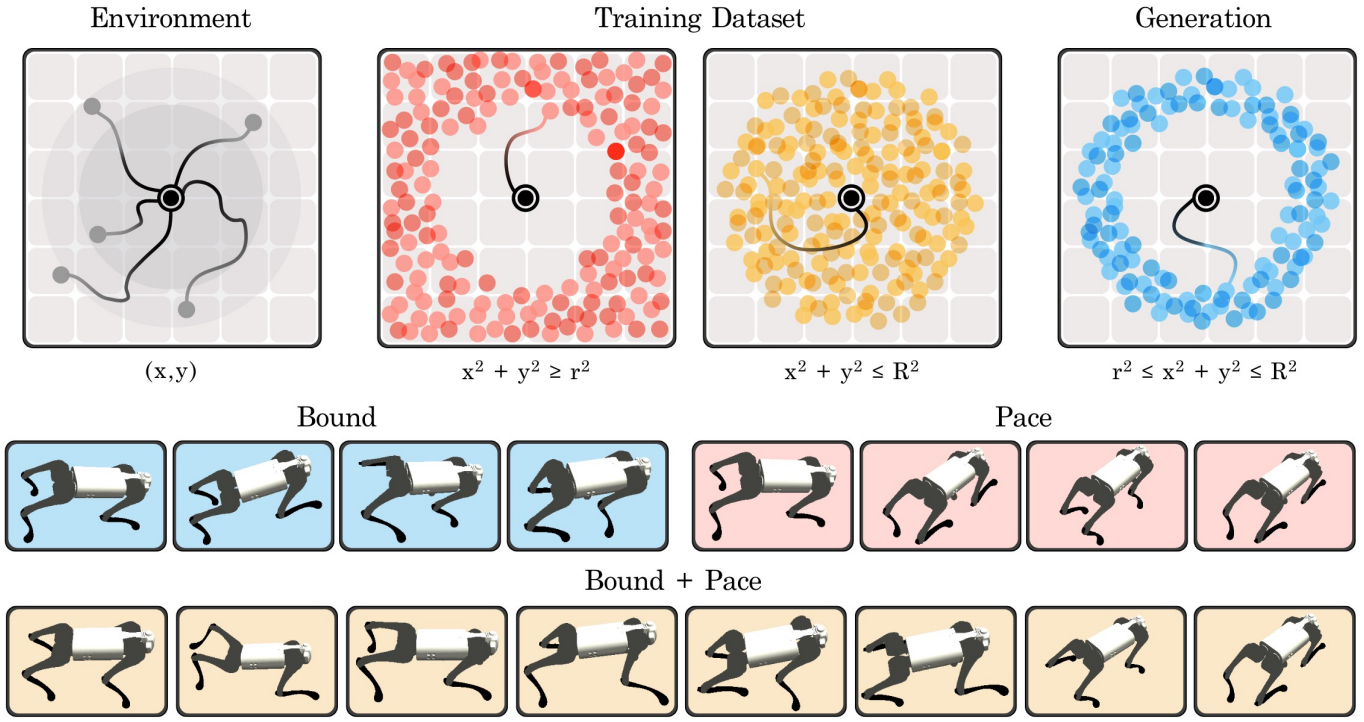
Ajay, Anurag, et al. "Is Conditional Generative Modeling all you need for Decision Making?." ICLR 2022.

Planner

Offline Reinforcement Learning

- With classifier-free guidance, Decision Diffuser allows conditioning beyond returns, and composing different skills or constraints:

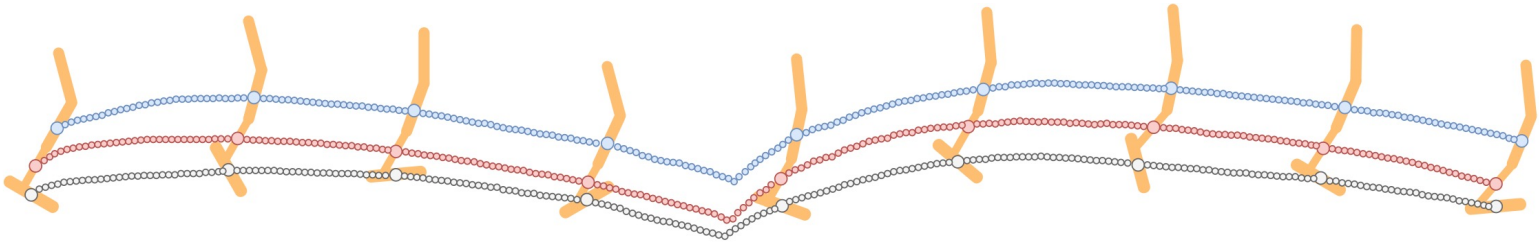
$$\hat{\epsilon} := \epsilon_\theta(\boldsymbol{x}_k(\tau), \varnothing, k) + \omega \sum_{i=1}^{n} (\epsilon_\theta(\boldsymbol{x}_k(\tau), \boldsymbol{y}^i(\tau), k) - \epsilon_\theta(\boldsymbol{x}_k(\tau), \varnothing, k))$$

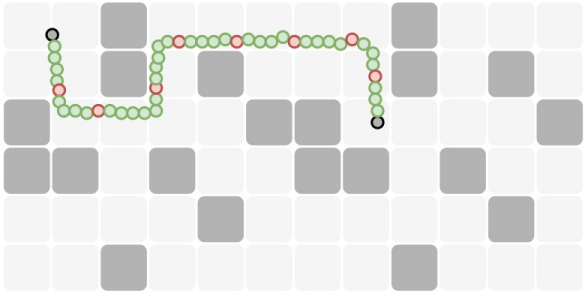


Planner

Offline Reinforcement Learning

- Besides end-to-end diffusion planning, HDMI proposes a hierarchical diffusion framework.
- Subgoals are first extracted from the original dataset.
  - A goal diffusion model is used to generate subgoal sequences.
  - The generated subgoals are then used as conditions in the trajectory inpainting diffusion model.





Li, Wenhao, et al. "Hierarchical diffusion for offline decision making." International Conference on Machine Learning. PMLR, 2023.

Policy

Offline Reinforcement Learning

- Diffusion models are used as a more expressive policy class to boost existing offline RL algorithms.
- Diffusion-QL introduces the diffusion policy to Q learning framework, using the diffusion loss as an additional policy regularization term:

Given Q, optimize π

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

$$\mathcal{L}_d(\theta) = \mathbb{E}_{i\sim\mathcal{U},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I}),(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}}\left[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_i}\boldsymbol{a} + \sqrt{1-\bar{\alpha}_i}\boldsymbol{\epsilon}, \boldsymbol{s}, i)||^2\right]$$

Given π, optimize Q

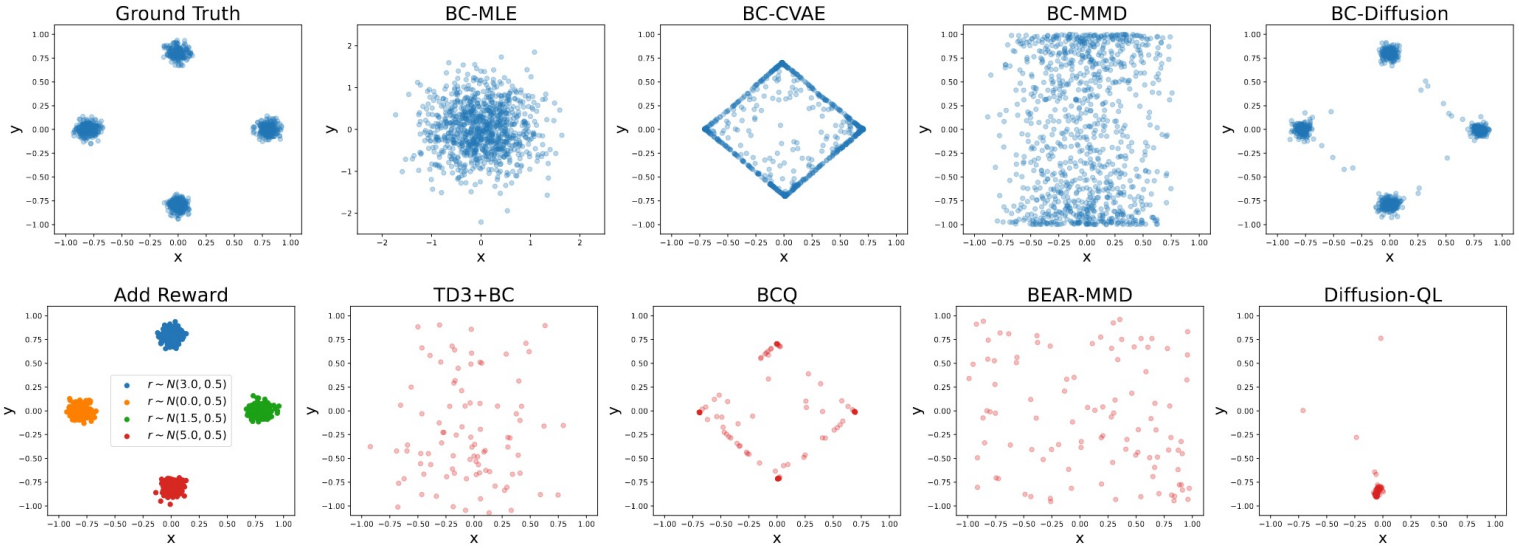
$$\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1})\sim\mathcal{D},\boldsymbol{a}^0_{t+1}\sim\pi_{\theta'}}\left[\left|\left|\left(r(\boldsymbol{s}_t,\boldsymbol{a}_t) + \gamma \min_{i=1,2} Q_{\phi'_i}(\boldsymbol{s}_{t+1},\boldsymbol{a}^0_{t+1})\right) - Q_{\phi_i}(\boldsymbol{s}_t,\boldsymbol{a}_t)\right|\right|^2\right]$$

Wang, Zhendong, Jonathan J. Hunt, and Mingyuan Zhou. "Diffusion Policies as an Expressive Policy Class for Offline Reinforcement Learning." The Eleventh International Conference on Learning Representations. 2022.

Policy

Offline Reinforcement Learning

- Use diffusion model as the policy class can better capture the multi-modal behavior dataset.

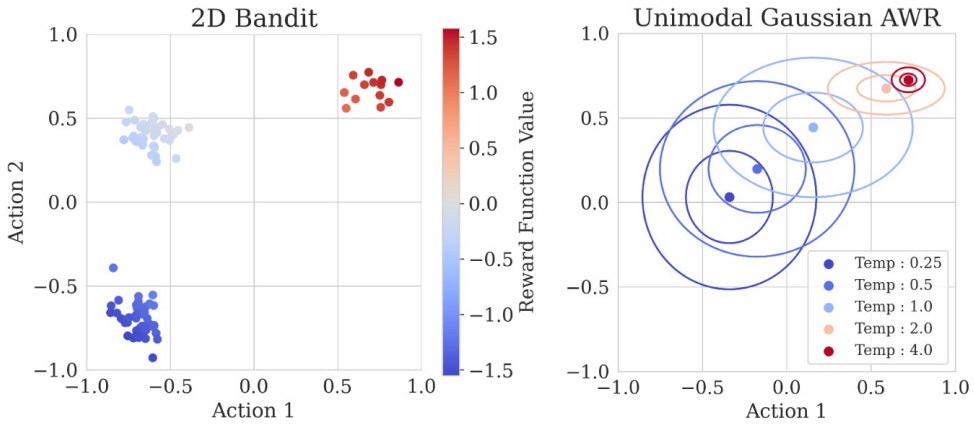


Gym Tasks	BC	AWAC	Diffuser	MoRel	Onestep RL	TD3+BC	DT	CQL	IQL	Diffusion-QL
halfcheetah-medium-v2	42.6	43.5	44.2	42.1	48.4	48.3	42.6	44.0	47.4	**51.1** ± 0.5
hopper-medium-v2	52.9	57.0	58.5	**95.4**	59.6	59.3	67.6	58.5	66.3	90.5 ± 4.6
walker2d-medium-v2	75.3	72.4	79.7	77.8	81.8	83.7	74.0	72.5	78.3	**87.0** ± 0.9
halfcheetah-medium-replay-v2	36.6	40.5	42.2	40.2	38.1	44.6	36.6	45.5	44.2	**47.8** ± 0.3
hopper-medium-replay-v2	18.1	37.2	96.8	93.6	97.5	60.9	82.7	95.0	94.7	**101.3** ± 0.6
walker2d-medium-replay-v2	26.0	27.0	61.2	49.8	49.5	81.8	66.6	77.2	73.9	**95.5** ± 1.5
halfcheetah-medium-expert-v2	55.2	42.8	79.8	53.3	93.4	90.7	86.8	91.6	86.7	**96.8** ± 0.3
hopper-medium-expert-v2	52.5	55.8	107.2	108.7	103.3	98.0	107.6	105.4	91.5	**111.1** ± 1.3
walker2d-medium-expert-v2	107.5	74.5	108.4	95.6	**113.0**	110.1	108.1	108.8	109.6	110.1 ± 0.3
Average	51.9	50.1	75.3	72.9	76.1	75.3	74.7	77.6	77.0	**88.0**

Policy

Offline Reinforcement Learning

- Implicit Q-Learning (IQL) first approximates the optimal Q function without querying policy models, while the policy are later extracted via advantage weighted regression (AWR).
- The learned optimal Q function can be multi-modal, which makes a simple Gaussian policy hard to fit corresponding optimal policy.



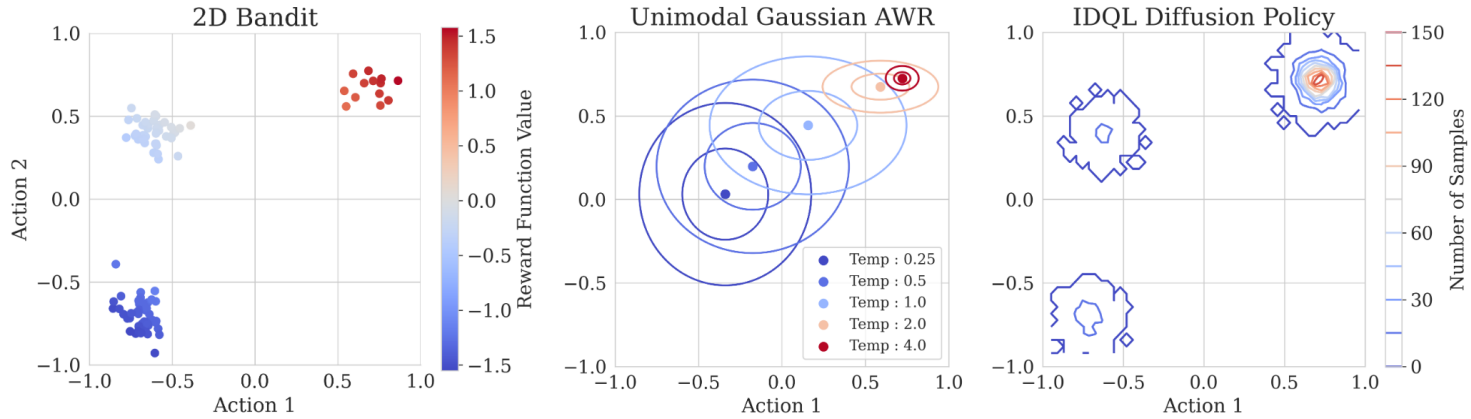
The unimodal actor incorrectly approximates the implicit policy for all temperatures and requires tuning to capture the maximum action.

Hansen-Estruch, Philippe, et al. ”IDQL: Implicit q-learning as an actor-critic method with diffusion policies." arXiv preprint arXiv:2304.10573 (2023).

Policy

Offline Reinforcement Learning

- IDQL uses diffusion policy in policy extraction, which captures the implicit policy accurately.



Dataset	%BC	DT	TD3	CQL	IQL	EQL	SfBC	DQL	IDQL
locomotion-v2 total	666.2	672.6	677.4	698.5	692.4	**725.3**	680.4	**791.2**	**738.8**
antmaze-v0 total	134.2	112.2	163.8	303.6	378.0	386	**445.2**	418.5	**474.6**
total	800.4	784.8	841.2	1002.1	1070.4	1111.3	1125.6	**1209.7**	**1213.8**
training time	10m	960m	20m	80m	20m	20m	785m	240m	60m

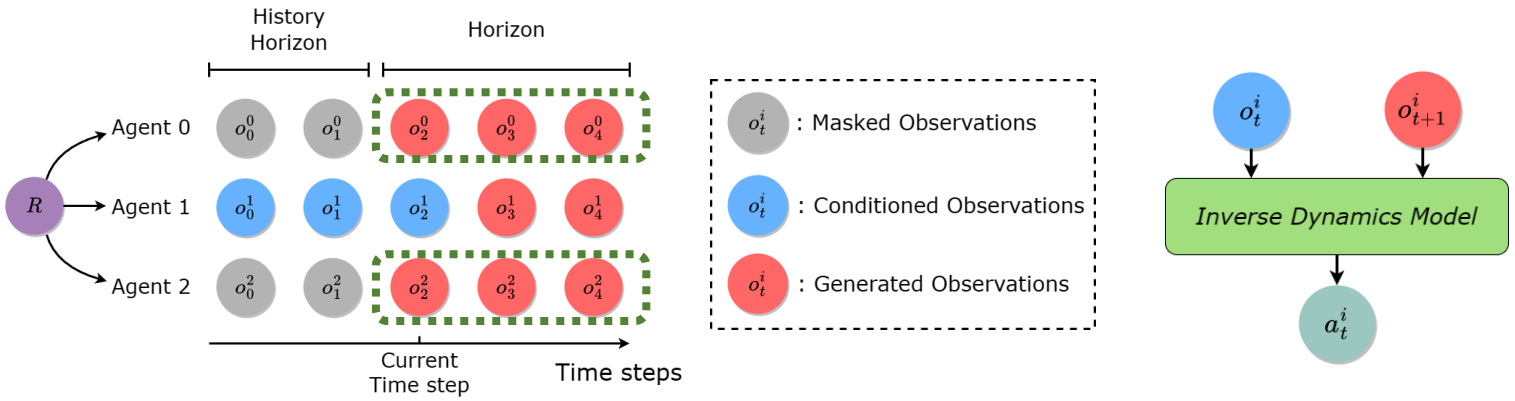
Table 2: **Full comparison offline RL.** We compare IDQL-A to other prior offline RL methods. IDQL outperforms all methods in total score and receives the strongest antmaze results.

Hansen-Estruch, Philippe, et al. ”IDQL: Implicit q-learning as an actor-critic method with diffusion policies." arXiv preprint arXiv:2304.10573 (2023).

Planner

Multi-agent Offline RL

- MADiff uses attention-based diffusion model to plan joint trajectories of multiple agents, explicitly considering the coordination among agents.
- In CTDE scenarios, each agent also uses the same model to plan for not only themselves, but also other agents, which can be seen as teammate modeling.

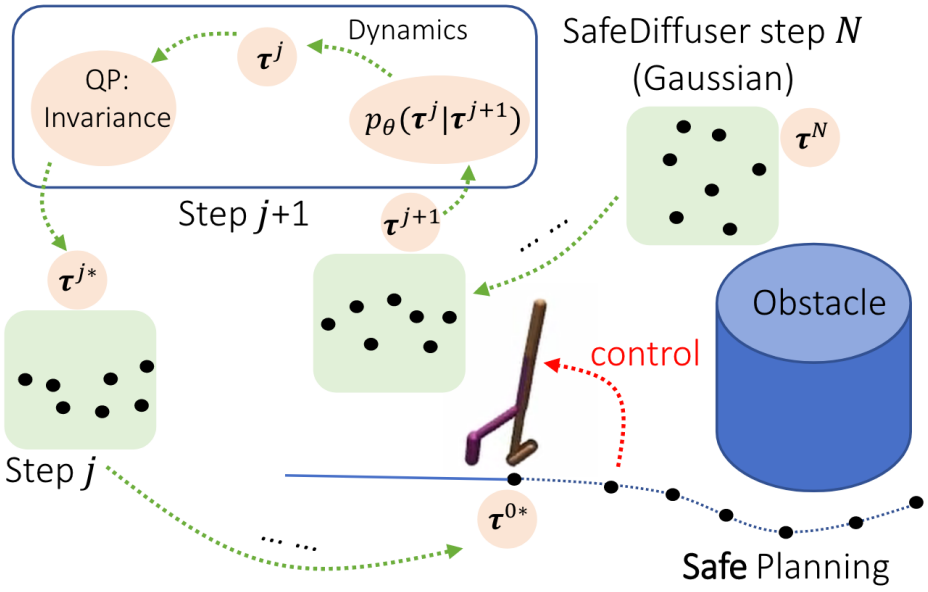


Zhu, Zhengbang, et al. "Madiff: Offline multi-agent learning with diffusion models." arXiv preprint arXiv:2305.17330(2023).

Planner

Safe Offline RL

- Besides maximizing cumulative rewards, Safe RL requires agents to satisfy extra safety constraints.
- For diffusion agents, the constraints need to be injected in each denoising step.
- Safe Diffuser adds a QP solver in each denoising step to steer the original diffusion process.



Xiao, Wei, et al. "Safediffuser: Safe planning with diffusion probabilistic models." arXiv preprint arXiv:2306.00148 (2023).

Planner

Safe Offline RL

- To trade-off between reward-maximization and constraint-satisfaction in each denoising step, TREBI learns a reward return model and cost return model as classifier guidance.
- They use the cost model to evaluate the cost value of trajectory in previous step, which is used to decide whether to add cost minimization guidance in the current step.

for $i = N, ..., 1$ **do**
  if $C(\tau^i) \leq z_t$ **then**
    $g = \alpha \nabla R(\tau^i)$;
  else
    $g = \alpha(\nabla R(\tau^i) - n \cdot \nabla C(\tau^i))$;
  end if
  $\mu \leftarrow \mu_\theta(\tau^i)$;
  $\tau^{i-1} \sim \mathcal{N}(\mu + \Sigma^i g, \Sigma^i)$;
  $\tau^{i-1} \leftarrow s$;

cost minimization guidance

Lin, Qian, et al. “Safe offline reinforcement learning with real-time budget constraints.” ICML 2023.

Online Reinforcement Learning

- Value estimations in online RL are noisier and change with the current policy, which poses additional challenges on training a multistep diffusion model.

- Recently, there have been some works showing that diffusion models can also boost online RL training.

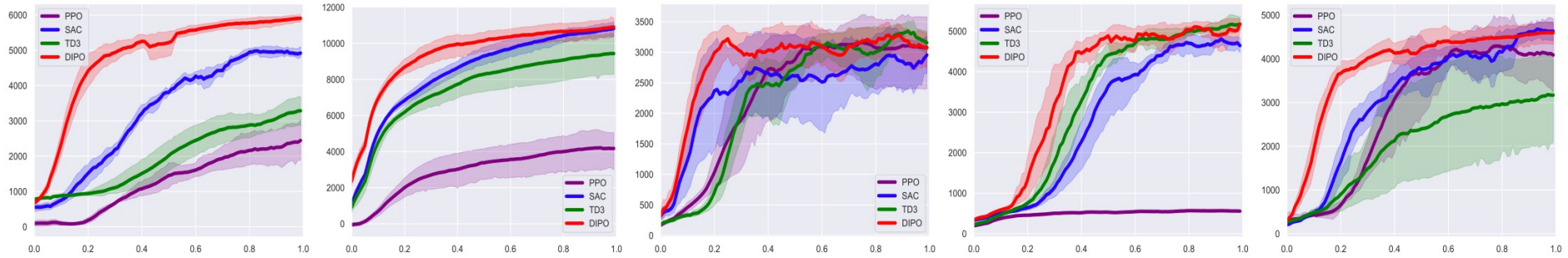
Policy

Online Reinforcement Learning

- DIPO adopts an action relabeling strategy to perform diffusion policy improvement, bypassing the potentially unstable value-guided training.
- During the training phase, actions in the replay buffer are replaced by Q value gradient ascent:

$$\mathbf{a}_t \leftarrow \mathbf{a}_t + \eta \nabla_{\mathbf{a}} Q_\psi(\mathbf{s}_t, \mathbf{a})|_{\mathbf{a}=\mathbf{a}_t}$$

- After relabeling, the diffusion policy only need to perform pure supervised learning: predict the relabeled action given the state.



(a) Ant-v3 (b) HalfCheetah-v3 (c) Hopper-v3 (d) Humanoid-v3 (e) Walker2d-v3

Yang, Long, et al. "Policy representation via diffusion probability model for reinforcement learning." arXiv preprint arXiv:2305.13122 (2023).

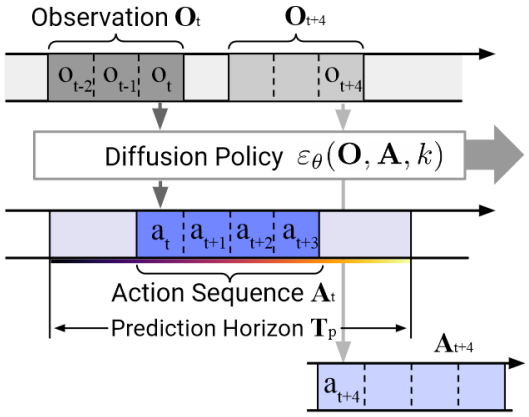
Imitation Learning

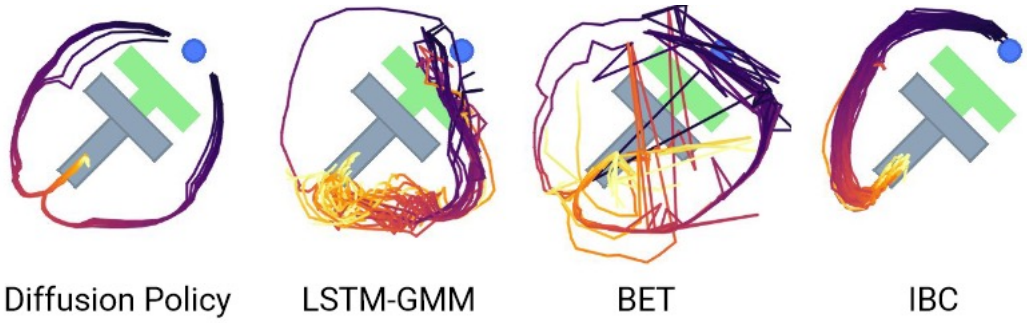
- Imitation learning aims to reproduce dataset behavior by extracting knowledge from expert demonstrations.
- Basically, demonstrations are a set of trajectories containing observations and actions, without reward signals.
- Diffusion models are more expressive, thus can solve complex tasks of imitation learning
  - Diverse trajectories from complex environments
  - Multimodal input (image, raw state, text, etc.)
  - Multiple tasks
  - Skill discovering
  - ……

Planner

Imitation Learning

- Diffusion Policy utilizes diffusion models to plan on action sequences conditioned on history observations.
- Diffusion models can flexibly mimic multimodal behavior, whereas MLP-based models fail.



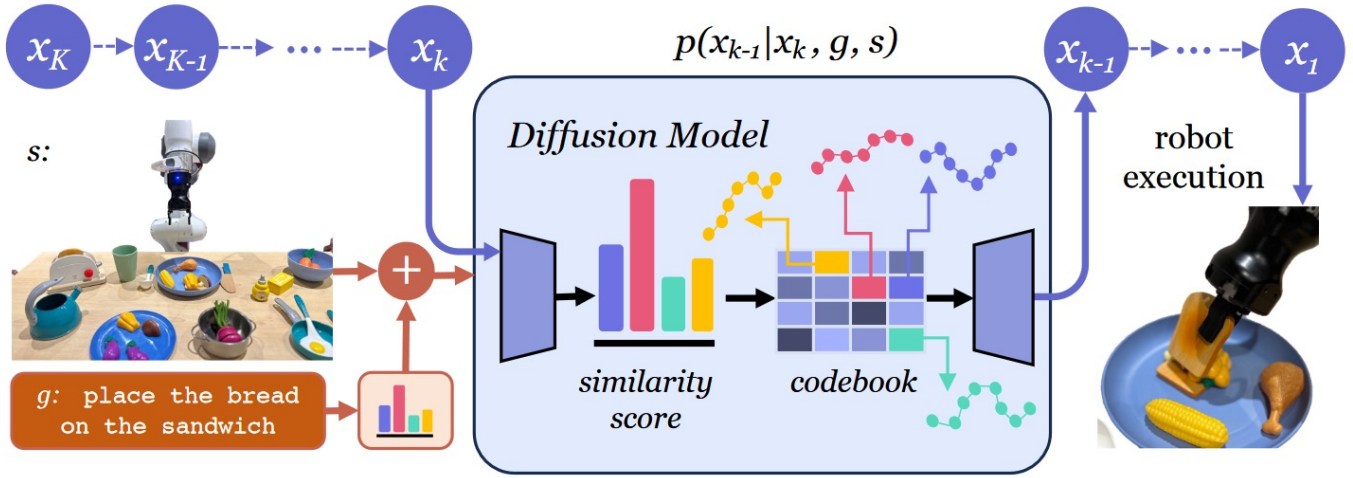


Chi, Cheng, et al. "Diffusion policy: Visuomotor policy learning via action diffusion." RSS. 2023.

Planner

Imitation Learning

- PlayFusion focuses on the ability of taking multimodal input & skill discovery
- PlayFusion trains a discrete diffusion model on a text-annotated play dataset
  - States and language instructions are encoded to the latent space, as conditions of diffusion models.
  - A discrete codebook is maintained to represent different skills.
  - Diffusion models generate the latent action, and the closest code is retrieved to reconstruct the real action.



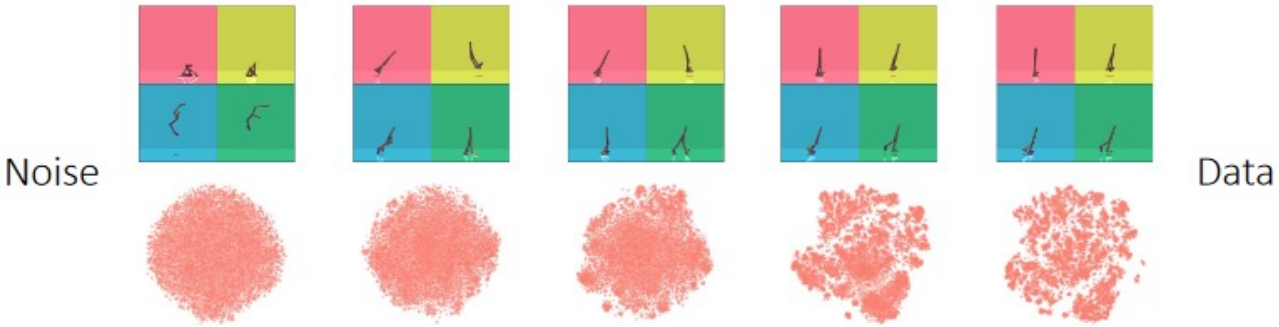
Chen, Lili, Shikhar Bahl, and Deepak Pathak. "Playfusion: Skill acquisition via diffusion from language-annotated play." Conference on Robot Learning. PMLR, 2023.

Synthesizer

Data Augmentation

- Synthesize new images for vision-based RL.

- Synthesize new (s, a, r, s') for state-based RL.

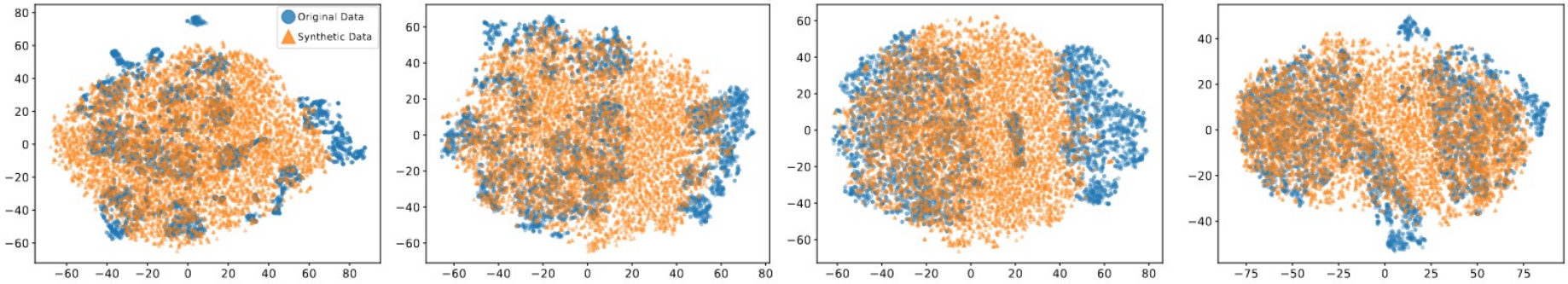


Chen, Zoey, et al. "Genaug: Retargeting behaviors to unseen situations via generative augmentation." arXiv preprint arXiv:2302.06671 (2023).
Yu, Tianhe, et al. "Scaling robot learning with semantically imagined experience." RSS. 2023.
Lu, Cong, et al. "Synthetic experience replay." Advances in Neural Information Processing Systems 36 (2024).

Synthesizer

Data Augmentation

- A more effective way is to leverage diffusion models to learn from multi-modal multi-task rollout data.
- This way outperforms other methods and demonstrates high fidelity of the synthesized data.



(a) Coffee-push (b) Disassemble (c) hand-insert (d) box-close

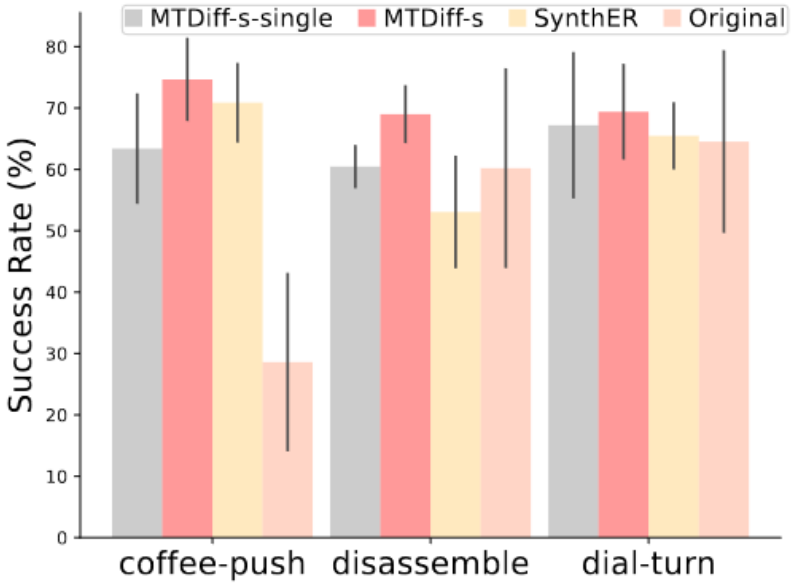
Experiments on 4 selected tasks demonstrates that the synthetic data **overlap** and **expand** the original data distribution while also **keeping consistency** with the underlying MDP.

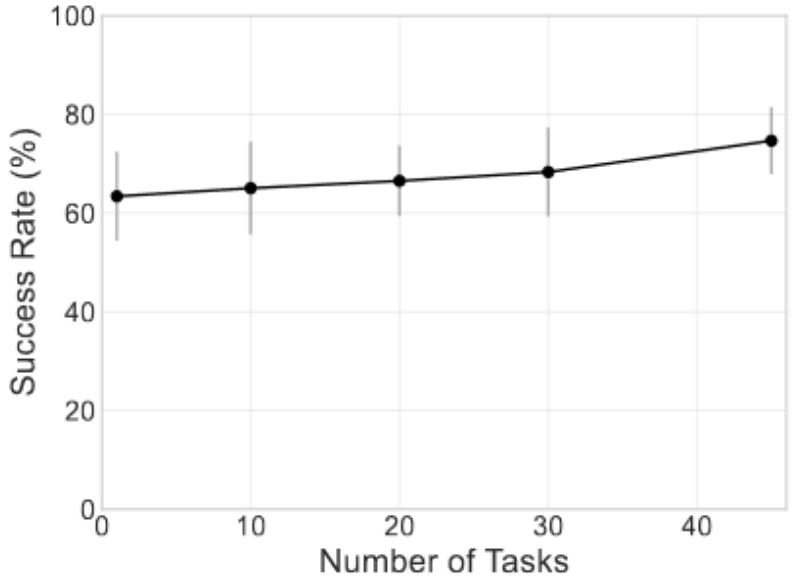
He, Haoran, et al. “Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning.” NeurIPS 2023.

Synthesizer

Data Augmentation

- **MTDiff-s** achieves superior policy improvement compared with previous SOTA augmentation methods;
- **MTDiff-s** can perform implicit data sharing by integrating other tasks' knowledge into data synthesis of the current task.



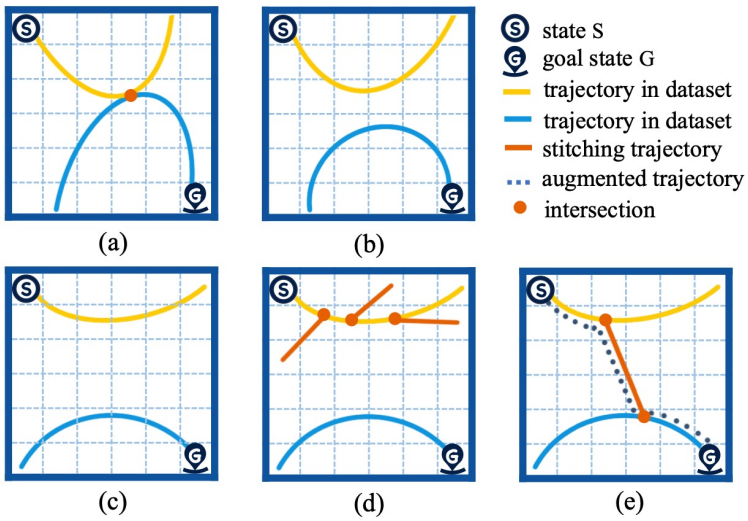


He, Haoran, et al. "Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning." NeurIPS 2023.

Synthesizer

Data Augmentation

- Performance of offline RL algorithms are constrained by the qualities of the training dataset.
- In some cases, offline dataset only contains very limited optimal trajectories, which requires extensive **“trajectory stitching”.**



- Sequence modeling algorithms such as decision transformer or diffuser are known for **lack of stitching abilities.**
- Q-learning algorithms rely on the generalization ability of function approximations to do **implicit stitching.**

Li, Guanghe, et al. “DiffStitch: Boosting Offline Reinforcement Learning with Diffusion-based Trajectory Stitching.” ICML 2024.

Synthesizer

Data Augmentation

- DiffStitch proposes to use diffusion trajectory model to perform explicit generative stitching.
- The generated sub-trajectories connects different suboptimal regions, resulting in globally optimal trajectories.

Dataset	Environment	IQL				TD3+BC				DT	
		Original	SER	TATU	**DStitch**	Original	SER	TATU	**DStitch**	Original	**DStitch**
Med-Expert	HalfCheetah	92.7±2.5	88.9±2.1	**94.4**±0.6	**94.4**±1.4	94.3±0.9	86.5±8.8	89.3±3.9	**96**±0.5	90.8±1.4	**92.6**±0.1
Med-Expert	Hopper	98.7±10	110.4±1.6	93.4±17.8	**110.9**±2.9	94.8±13	104±10.8	99±14.9	**107.1**±7	**109.9**±1.9	109.4±1.9
Med-Expert	Walker2d	**112.4**±0.8	111.7±1.1	110.7±0.6	111.6±0.1	109.9±0.8	110.5±0.3	**110.7**±0.7	110.2±0.3	108.1±0.3	**108.6**±0.4
Medium	HalfCheetah	48.5±0.3	49.3±0.1	48.2±0.1	**49.4**±0.14	48.4±0.1	48.4±0.4	48.1±0.2	**50.4**±0.5	40.4±2.2	**44.2**±0.3
Medium	Hopper	65.9±5.7	66.6±2.4	60.3±3.6	**71**±4.2	58±2.8	56.4±1.3	58.3±4.8	**60.3**±4.9	**61.5**±2.2	60.5±4.3
Medium	Walker2d	81.1±1.8	**85.9**±1.6	76.6±10.7	83.2±2.2	81.4±2.3	**84.9**±0.3	75.8±3.5	83.4±1.7	70.5±1.6	**72**±4.9
Med-Replay	HalfCheetah	44.1±0.5	**46.6**±0.1	44.2±0.1	44.7±0.1	44.5±0.2	**45.2**±0.1	44.5±0.3	44.7±0.3	39.8±1.6	**41**±0.4
Med-Replay	Hopper	91.4±8.1	**102.4**±0.5	79.6±7.6	102.1±0.4	64.8±19.2	56.8±9.9	64.1±10.5	**79.6**±13.5	83.6±3.9	**96.1**±2
Med-Replay	Walker2d	80.7±7	85.7±3.6	75±12.1	**86.6**±2.8	82.4±5	89.1±0.5	62.1±10.4	**89.7**±4.2	53.3±11.2	**60.2**±1.8
Locomotion Average		79.5	83.1	75.8	**83.8**	75.4	75.8	72.4	**80.2**	73.1	**76.1**
human	pen	79.1±28.5	88.9±22.6	**96.8**±8.6	87.4±8.6	-3.3±0.4	-2.8±1.5	7.3±14.1	**29.8**±6.9	62.8±2.1	**70**±6.8
cloned	pen	45.8±29.9	52.5±27.9	45.3±23.4	**64**±29.6	-3.1±0.4	-2.8±0.1	-3±0.3	**11.3**±6.2	57.7±3.3	**59.7**±9.6
Pen Average		62.5	70.7	71.1	**75.7**	-3.2	-2.8	2.2	**20.6**	60.3	**64.9**

Li, Guanghe, et al. “DiffStitch: Boosting Offline Reinforcement Learning with Diffusion-based Trajectory Stitching.” ICML 2024.

Summary

1. DMs for RL: towards sample-efficient methods & highly expressive non-autoregressive modeling of RL
2. Roles of Diffusion Models in RL
   - Planners: generate future trajectories and select one-step action
   - Policies: generate one-step action
   - Data synthesizer: generate more training data
3. Applications in RL and Related Tasks
   - Reinforcement Learning
   - Imitation Learning
   - Data Augmentation
4. Limitations
   - Some difficulties for online RL
   - Computational efficiency

Future Developments of DMs for RL

- Generative Simulation
  - Use diffusion model to augment the online environment
  - generate configurations, dynamics, reward functions, opponents, ...
- Retrieval-augmented Generation
  - Accelerates sampling as well
  - Updating retrieval dataset as online adaptation
- Composing different skills
  - Diffusion models can share knowledge across skills and combine them up

Recent Publications of Our Group

- Zhengbang Zhu et al. Diffusion models for reinforcement learning: A survey. Arxiv 2023. https://arxiv.org/abs/2311.01223
- Haoran He et al. Diffusion Model is an Effective Planner and Data Synthesizer for Multi-Task Reinforcement Learning. NeurIPS 2023. https://arxiv.org/abs/2305.18459
- Zhengbang Zhu et al. Madiff: Offline multi-agent learning with diffusion models. Arxiv 2023. https://arxiv.org/abs/2305.17330
- Guanghe Li, et al. DiffStitch: Boosting Offline Reinforcement Learning with Diffusion-based Trajectory Stitching. ICML 2024. https://arxiv.org/abs/2402.02439
- Yixiang Shan, et al. Contrastive Diffuser: Planning Towards High Return States via Contrastive Learning. Arxiv 2024. https://arxiv.org/abs/2402.02772
- Haoran He et al. Large-Scale Actionless Video Pre-Training via Discrete Diffusion for Efficient Policy Learning. Arxiv 2024. https://arxiv.org/abs/2402.14407