Advancements in Offline Reinforcement Learning through Generative Models

Wenjia Wang

The Hong Kong University of Science and Technology (Guangzhou)

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

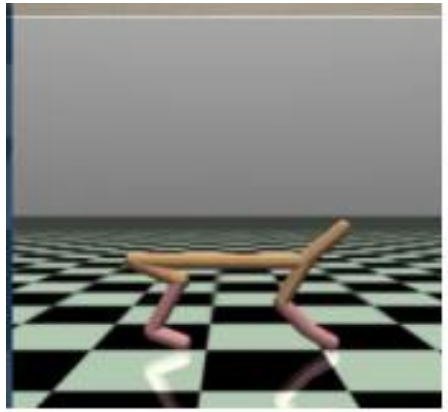
- Fang, L., Liu, R., Zhang, J., Wang, W., & Jing, B. Y. (2025). Diffusion Actor-Critic: Formulating Constrained Policy Iteration as Diffusion Noise Regression for Offline Reinforcement Learning. *ICLR,* 2025.
- Zhang, J., Fang, L., Shi, K., Wang, W., & Jing, B. Y. (2024). Q-Distribution guided Q-learning for offline reinforcement learning: Uncertainty penalized Q-value via consistency model. *NeurIPS*, 2024.
- Zhang, J., Zhang, C., Wang, W., & Jing, B. Y. (2023). Constrained Policy Optimization with Explicit Behavior Density For Offline Reinforcement Learning. *NeurIPS,* 2023.

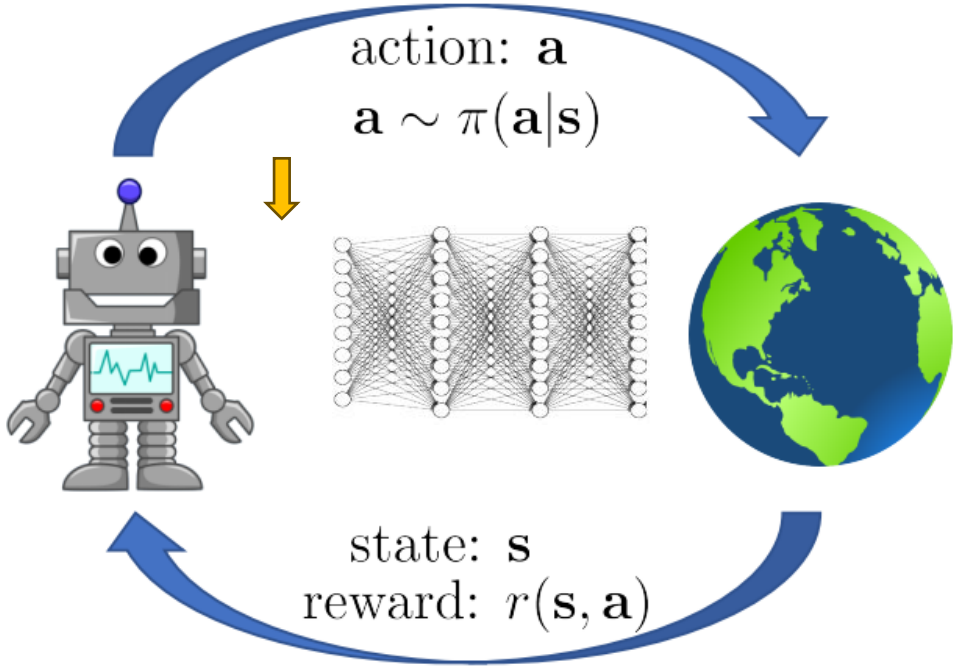
Content

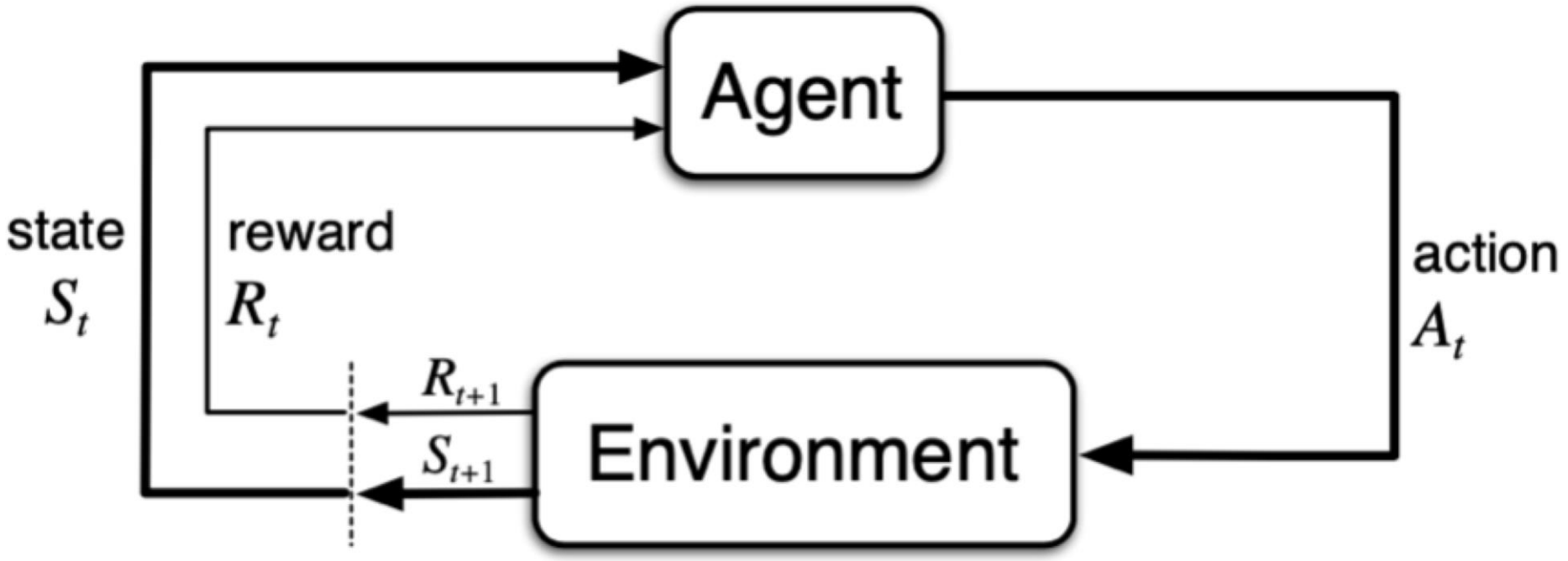
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Background: What is RL?

- RL = decision under uncertainty
- RL models the natural learning-based control process.
- The agent progressively improves its behavioral skills (policy) through iterative interactions with the environment and feedback in the form of rewards.

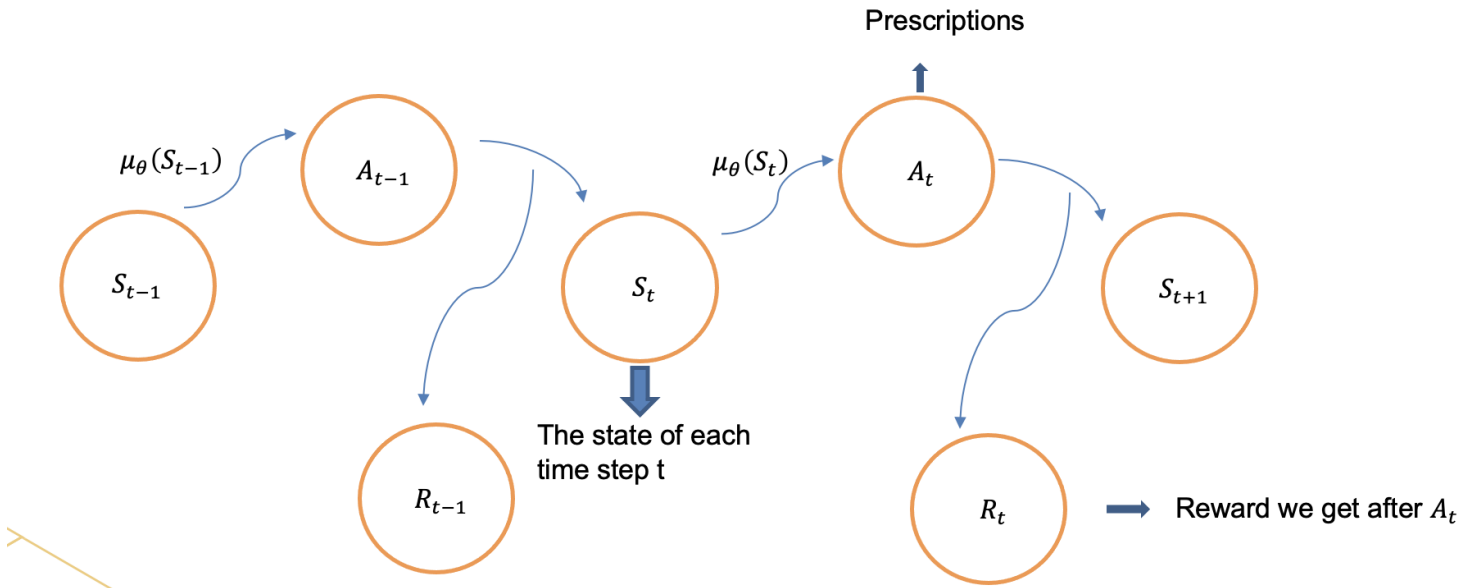






The agent-environment interaction in a MDP

Example: Dynamic treatment recommendation



.

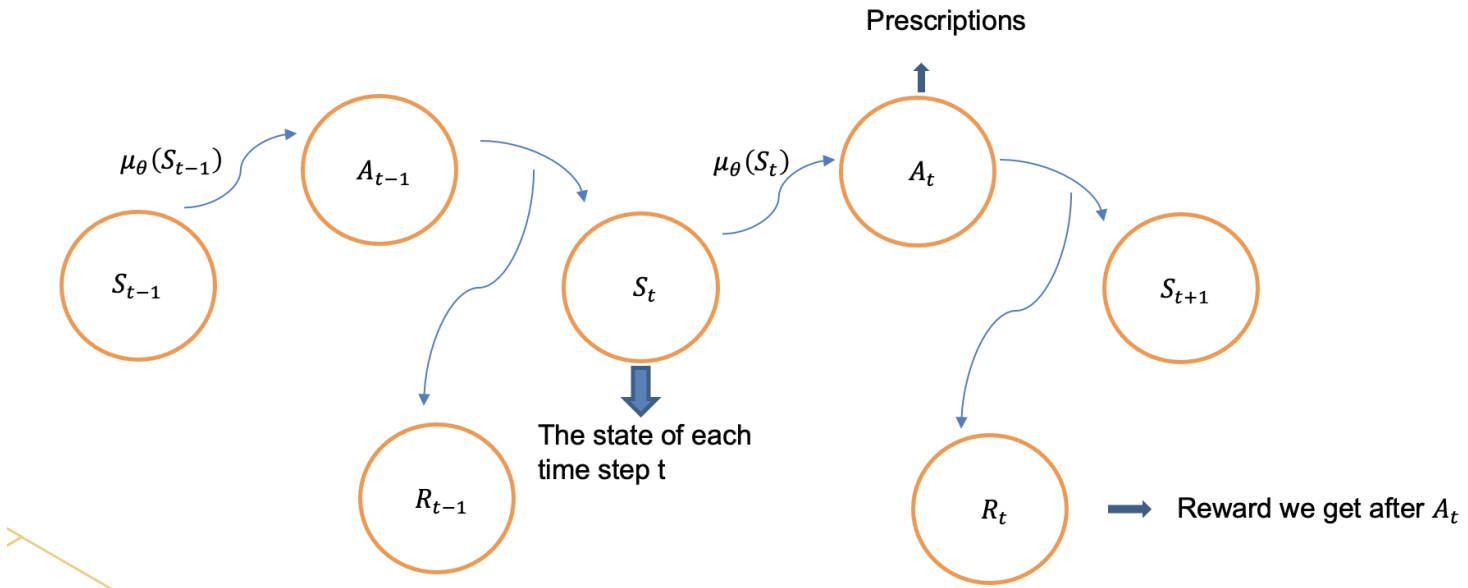
At time t, doctor observes patient's current state S_t ,

chooses medication set A_t,

receives immediate reward R_t ,

takes us to new state S_{t+1}

Example: Buy & Sell in Stock Market



.

At time t, you observe past stock history S_t ,

chooses an action A_t (buy or sell),

receives immediate reward R_t ,

takes us to new state S_{t+1}

Characters of RL

- RL models consist of two main components: an **agent** and an **environment**.
- The agent is the model itself which makes decisions about which actions it should take; these actions could take the form of a car (the agent) determining which way to turn (an action).
- The environment is the world within which the agent lives; for our car example, this would be the road on which the car drives and the other surrounding cars.

Characters of RL

- The environment also provides a **reward** to the agent depending on how "good" of a situation the agent is experiencing.
  - For example, if the car makes a turn and does not crash, then the environment may reward the car.
- The main goal of the RL agent is to maximize the expected cumulative reward it obtains from interacting in its environment -- which is called the **return**.

An RL Policy

- The agent of a RL model takes in the current **state** at a given time s_t and makes an **action** a_t based the agent's **policy** (i.e. $a_t = \mu(s_t)$, where $\mu(\cdot)$ is the policy).
- If these actions are made in a stochastic manner, the policy is usually represented as $a_t \sim \pi_\theta(\cdot\,|s_t)$, where θ are the parameters of the policy.
- Using these policies, over time agents take certain **trajectories** through the model, such that $\tau = (s_0, a_0, s_1, a_1, \dots,)$.

An RL Policy

- Using these τ, rewards can be calculated from a reward function (determined by the environment) such that $r_t = R(s_t, a_t, s_{t+1})$.
- As such, the reward of a given trajectory can be represented by $R(\tau)$, and the expected return of the policy can be determined by $J(\pi_\theta) = E_{\tau \sim \pi_\theta}[R(\tau)]$.
- The overall goal is for the agent to find $\pi^* = argmax_\pi J(\pi)$.

Characteristics of RL

- Rewards are not labelled.
  - After taking an action, we get feedback.
  - We don’t get feedback on the actions we didn’t take. In a supervised setting we know if a prediction was right or wrong. In RL, we know if we obtained a reward but we don’t know if this was the best reward we could have obtained.
  - This makes RL a missing data/causal inference problem.

Characteristics of RL

- Temporal credit assignment.
  - In RL, the goal is to find a policy π that maximizes a discounted sum of rewards as opposed to finding a policy to produce maximal reward.
  - An action is good if it produces good immediate and “future” reward.
  - It may be better to sacrifice immediate reward to gain more long-term reward

Characteristics of RL

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- It can be advantageous to explore
  - In RL, there is a tradeoff between exploration and exploitation.
  - Exploration will increase our knowledge and exploitation is to leverage our knowledge.
  - Choosing a non-optimal action may lead to better long-term policies by allowing us to observe more of the state-action space.
  - This is different than the supervised setting, where if we were to switch our prediction from a non-error prediction to a low-error prediction we would incur loss and gain nothing.

Characteristics of RL

- Partially observable states
  - Sensors do not record the entire state of the environment or more accurately, sensors may not record enough of the environment so that one must not only use present sensor observations but past sensor observations to form a good policy.
  - In this setting, some actions may be used only to improve observation of environment. One can understand this as providing diagnostic information but may not alter the state.

Markov Decision Process (MDP)

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- RL operates within a framework called a Markov Decision Process
- MDP's: General formulation for decision making under uncertainty

  Defined by: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{T}, \gamma)$

  $\mathcal{S}$: set of possible states [start state = s_0, optional terminal / absorbing state]

  $\mathcal{A}$: set of possible actions

  $\mathcal{R}(s, a, s')$: distribution of reward given (state, action, next state) tuple

  $\mathbb{T}(s, a, s')$: transition probability distribution, also written as $p(s'|s, a)$

  γ : discount factor

$$\ldots s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}, r_{t+2}, s_{t+2}, \ldots$$

- Life is trajectory:
- **Markov property**: Current state completely characterizes state of the world
- **Assumption**: Most recent observation is sufficient statistic of history

$$p\left(S_{t+1} = s' | S_t = s_t, A_t = a_t, S_{t-1} = s_{t-1}, \ldots S_0 = s_0\right) = p\left(S_{t+1} = s' | S_t = s_t, A_t = a_t\right)$$

Value and optimal policy

The *value* of state s_t under policy π is defined by

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \cdots$$
$$= \sum_{i=0}^{\infty} \gamma^i r_{t+i},$$

where r_{t+i} is generated by following the action selection determined by π.

Our goal is to learn the *optimal policy* π^*, defined by

$$\pi^* = \arg\max_{\pi} V^{\pi}.$$

Value Function

- Following policy π that produces sample trajectories s_0, a_0, r_0, s_1, a_1, …
- How good is a state?
  - State-value function $V^{\pi}(s_t)$ of a policy π at state s_t = the expected future return of π starting from s_t :

$$V^{\pi}(s_t) = \mathbb{E}_{\pi,P,R}[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k})|s_t].$$

- How good is a state-action pair?
  - Action-value function or the Q-function $Q^{\pi}(s_t, a_t)$ = expected future return after performing action a_t :

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\pi,P,R}[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k})|s_t, a_t].$$

Recursive definition of value: Bellman equations

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Extracting optimal value / policy from Q-values:

$$V^*(s) = \max_a Q^*(s,a) \qquad \pi^*(s) = \arg\max_a Q^*(s,a)$$

- Bellman Equations:

$$V^*(s) = \max_a \sum_{s'} p\left(s'|s,a\right)\left[r(s,a) + \gamma V^*\left(s'\right)\right]$$

$$Q^*(s,a) = \sum_{s'} p\left(s'|s,a\right)\left[r(s,a) + \gamma V^*\left(s'\right)\right]$$

Reinforcement Learning

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Value-based RL
  - (Deep) Q-Learning, approximating $Q^*(s, a)$ with a deep Q-network
- Policy-based RL
  - Directly approximate optimal policy π^* with a parametrized policy π^*_θ
- Model-based RL
  - Approximate transition function $T(s', a, s)$ and reward function $\mathcal{R}(s, a)$
  - Plan by looking ahead in the (approx.) future!

Actor-Critic

- Learn both policy and Q function
  - Use the "actor" to sample trajectories
  - Use the Q function to "evaluate" or "critic" the policy

- REINFORCE: $\nabla_\theta J(\pi_\theta) = \mathbb{E}_{a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s)\mathcal{R}(s,a)\right]$

- Actor-critic: $\nabla_\theta J(\pi_\theta) = \mathbb{E}_{a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s)Q^{\pi_\theta}(s,a)\right]$

- Q function is unknown too! Update using $\mathcal{R}(s,a)$

Approximate Dynamic Programming

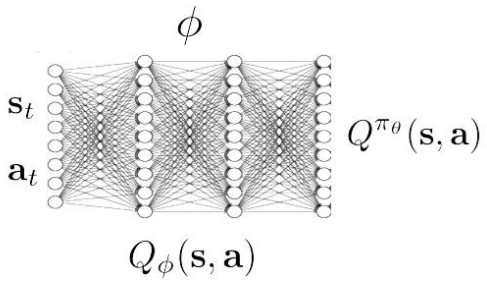
- For the off-policy algorithm, the average long-term return can be optimized by recovering the Q value function or policy value, and the optimal Q value function satisfies the Bellman equation:

Bellman equation:

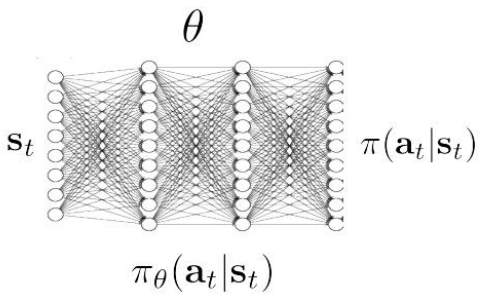
$$Q(s,a) = (\mathcal{B}Q)(s,a)$$

Bellman operator:

$$(\mathcal{B}Q)(s,a) := r(s,a) + \gamma \mathbb{E}_{s'' \sim T(\cdot|s,a)}[\max_{a'} Q(s',a')].$$



Solved by



- **Value Iteration(VI)**

  (1) Initialization: $Q_0 : \|Q_0\|_\infty \in (0, \frac{1}{1-\gamma})$.

  (2) Iterate until convergence: $Q^{t+1} = \mathcal{B}Q^t$.

  (3) Policy for each iteration: $\pi^t(s) = \arg\max_a Q^t(s,a)$.

- **Policy Iteration(PI)**

  (1) Initialization: $\pi_0 : \mathcal{S} \mapsto \mathcal{A}$.

  (2) Policy evaluation: $Q^{\pi^t}(s,a) = (\mathcal{B}Q^{\pi^t})(s,a)$.

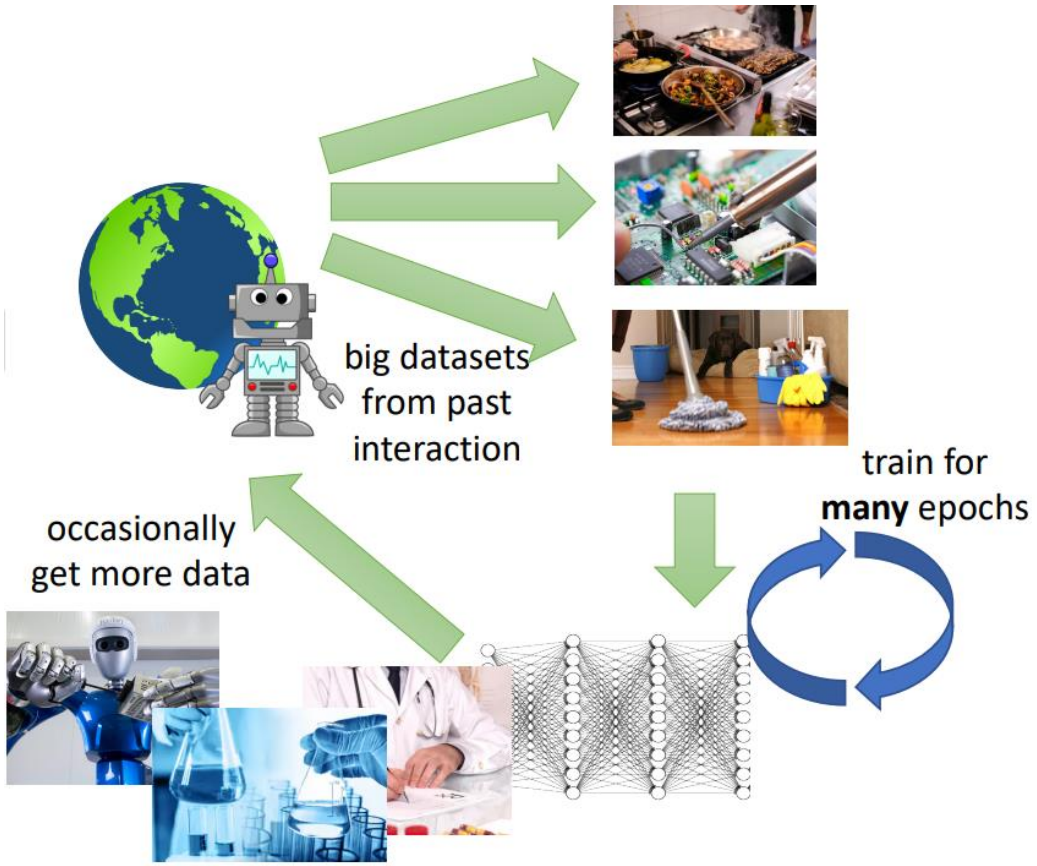
  (3) Policy improvement: $\pi^{t+1}(s) = \arg\max_a Q^t(s,a)$.

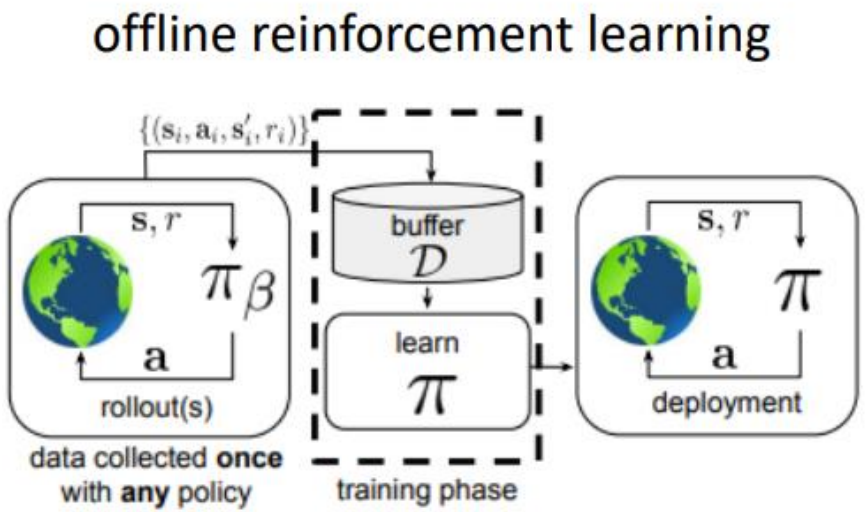
Content

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Why offline RL?

- Traditional RL (online RL) requires environmental interactions, which is very consuming
- Not feasible in certain domains, e.g., autonomous driving, medical diagnosis.

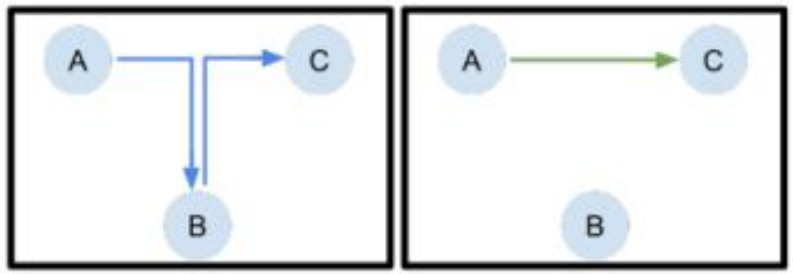


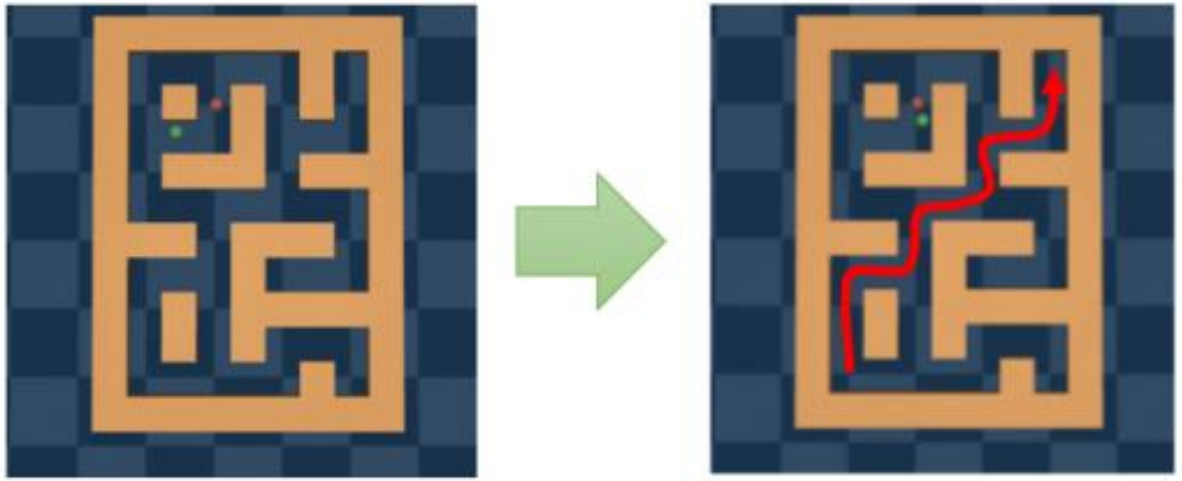


Offline RL

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- One possible reason that offline RL can find a suboptimal policy from a medium or even worse quality dataset is stitching.
- In the maze, the goal is to walk from point A to point C. The behavior policy has experience walking from A to B and from B to C.
- The learning policy can learn walking from A to C by stitching the suboptimal data (Levine et al., 2020).



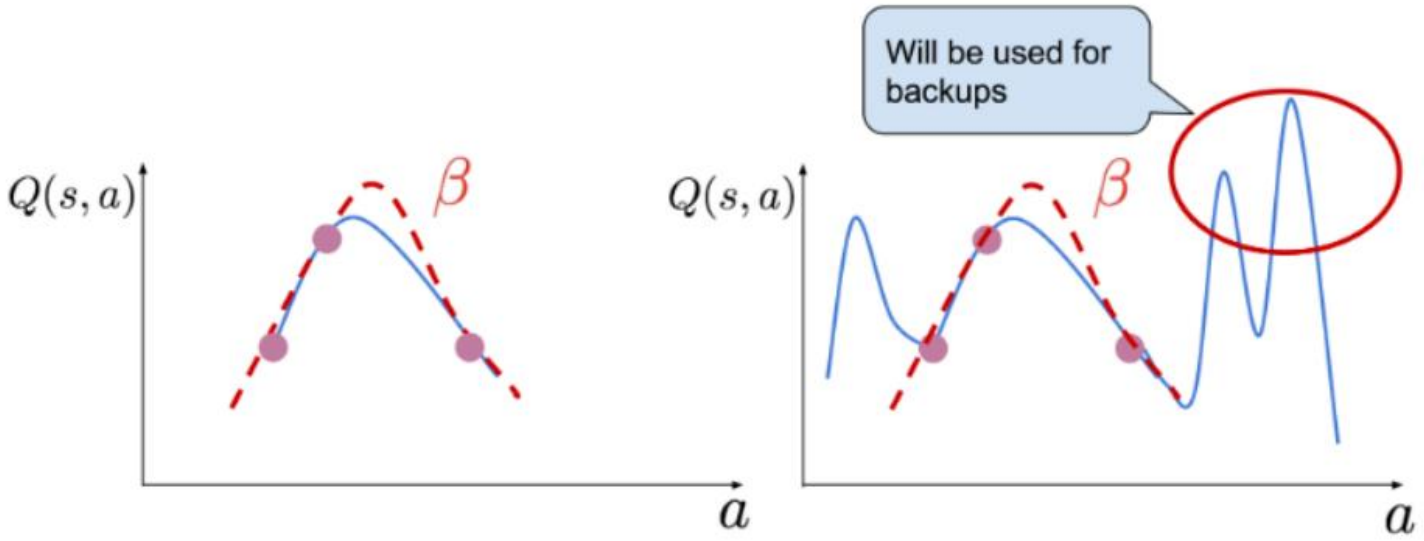


Challenges in Offline RL: "distribution shift"

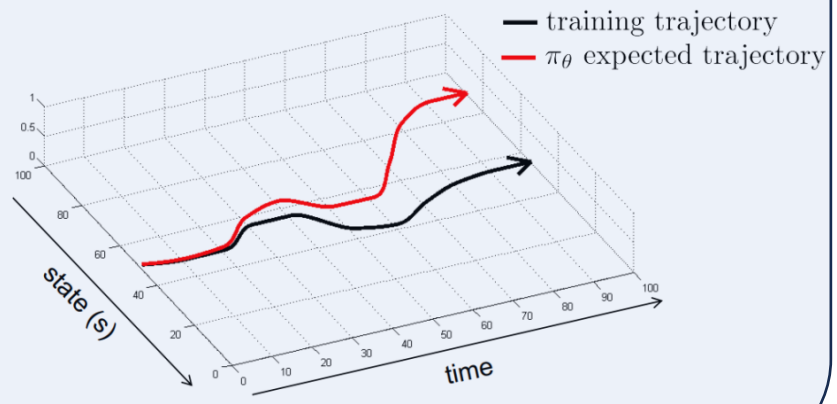
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- In offline RL, Q-value function is solved by:

$$\hat{Q} := \arg\min_{\hat{Q}} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\beta(\cdot|s)}\left(Q(s,a) - \left(r(s,a) + \gamma \mathbb{E}_{s'\sim T_{\mathcal{D}}(\cdot|s,a),a'\sim\pi(\cdot|s')}\left[\max_{a'} Q(s',a')\right]\right)\right)^2$$



Bootstrap error can accumulate

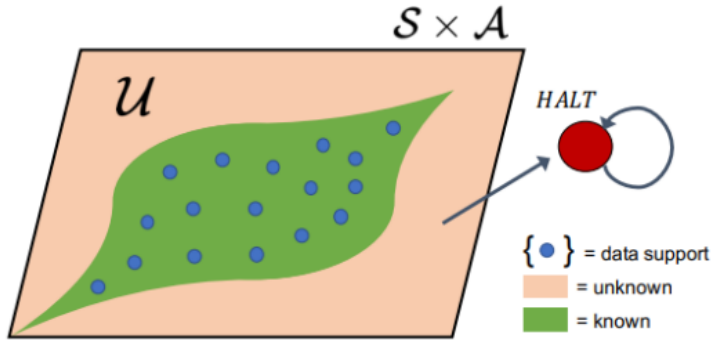


How to avoid "distribution shift" ?

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Under the Approximate Dynamic Programming

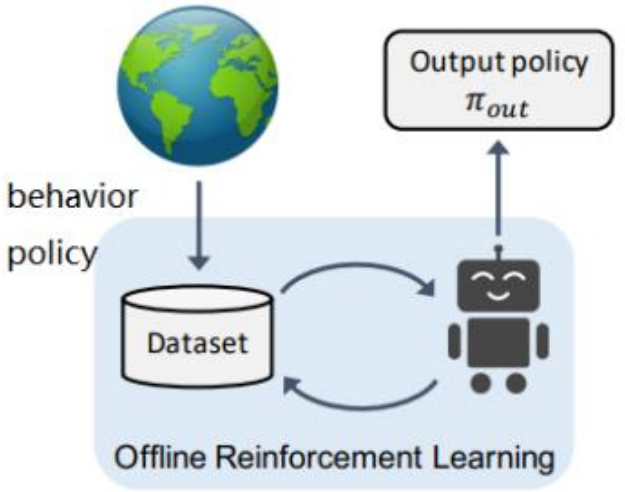
Policy control:
π only covers the green area

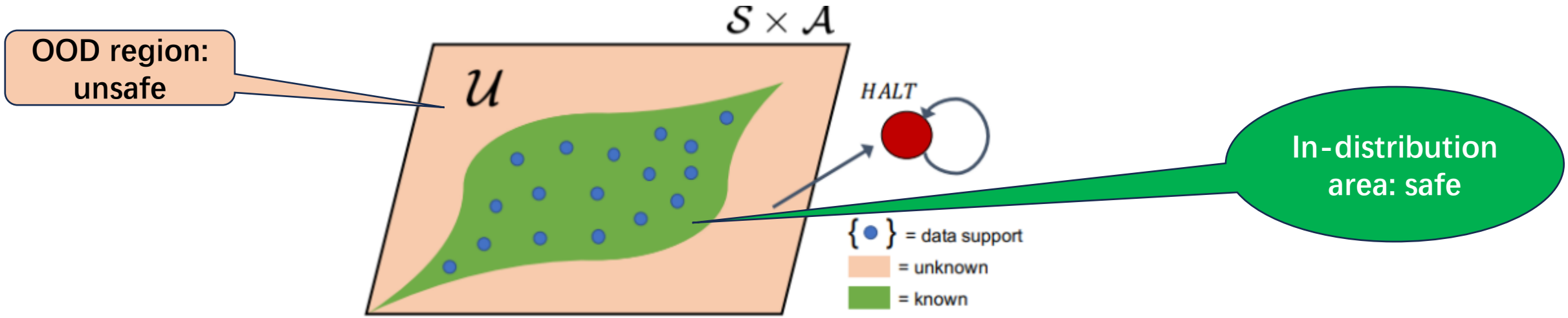


$$\hat{Q} := \arg\min_{\hat{Q}} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\beta(\cdot|s)}\left(Q(s,a) - \left(r(s,a) + \gamma \mathbb{E}_{s'\sim T_{\mathcal{D}}(\cdot|s,a),a'\sim\pi(\cdot|s')}\left[\max_{a'} Q(s',a')\right]\right)\right)^2$$

Q value constraint:
make Q lower in the pink area

Offline RL: Constraint MDP



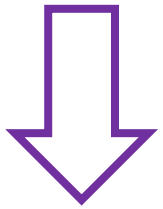


Control learning policy

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- To make offline RL work, “distribution shift” problem should be solved.

- Policy control prevents learning policy from visiting OOD actions that cause “distribution shift” between behavior and learning policies.



- Optimal policy control is distribution support consistency.

Policy control method

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Distribution matching control: $D(\pi, \pi_\beta) < \epsilon$

Kullback-Leibler(KL)-divergence.(Wu et al., 2019a,b; Nair et al., 2020; Jaques et al., 2019; Peng et al., 2019)

$$D(\pi, \pi_\beta) = \mathbb{E}_{a\sim\pi(\cdot|s)}[\log \pi(s,a) - \log \pi_\beta(s,a)].$$

Jensen–Shannon divergence(JSD) (Yang et al., 2022b)

$$D(\pi, \pi_\beta) = -\log(4) + KL(\pi_\beta \parallel \frac{\pi_\beta + \pi}{2}) + KL(\pi_\parallel \frac{\pi_\beta + \pi}{2}).$$

Wasserstein distance.(Wu et al., 2019a; Yang et al., 2022b)

$$D(\pi, \pi_\beta) = -\sup_{g:\|g\|_L\leq 1} + \mathbb{E}_{a\sim\pi_\beta(\cdot|s)}(g(a)) - \mathbb{E}_{a'\sim\pi(\cdot|s)}(g(a')).$$

Policy control method

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

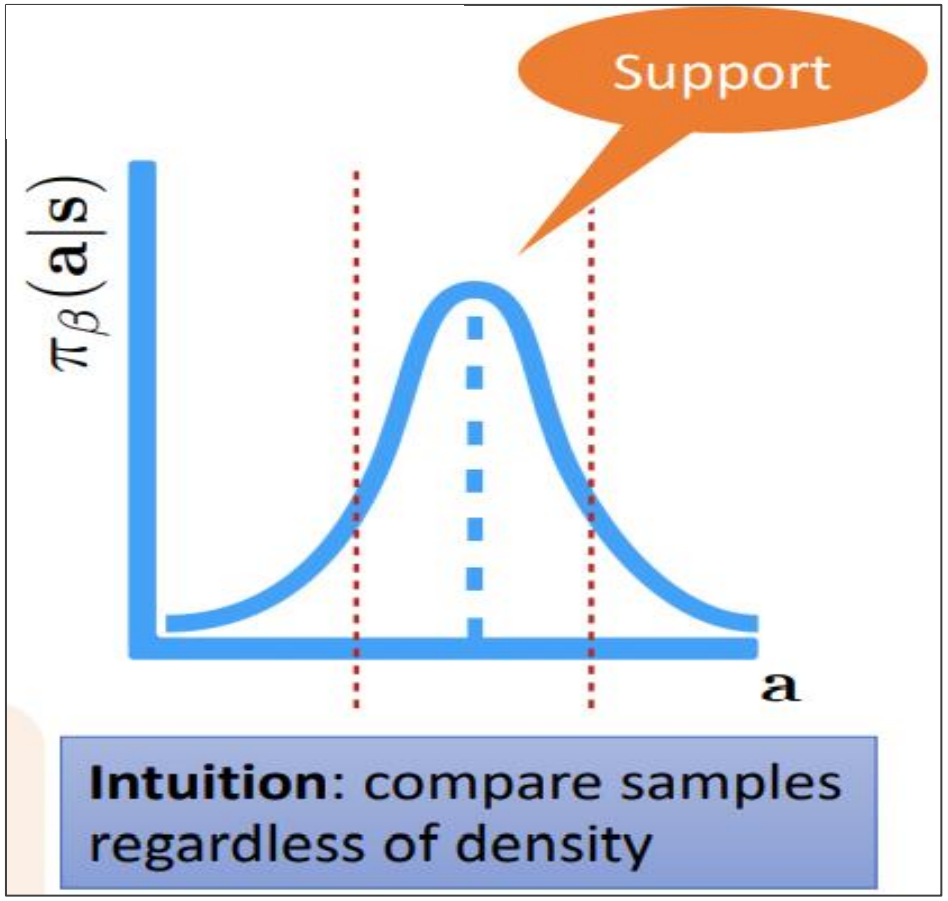
- Distribution support consistence control: $D(\pi, \pi_\beta) < \epsilon$

Maximum Mean discrepancy (MMD) distance.(Kumar et al., 2019)

$$D(\pi, \pi_\beta) = MMD^2(\pi(\cdot|s), \pi_\beta(\cdot|s)) = \mathbb{E}_{a\sim\pi(\cdot|s), a'\sim\pi(\cdot|s)}[k(a, a')] - 2\mathbb{E}_{a\sim\pi(\cdot|s), a'\sim\pi_\beta(\cdot|s)}[k(a, a')] + \mathbb{E}_{a\sim\pi_\beta(\cdot|s), a\sim\pi_\beta(\cdot|s)}[k(a, a')].$$

Estimate the behavior policy density function(SPOT (Wu et al., 2022))

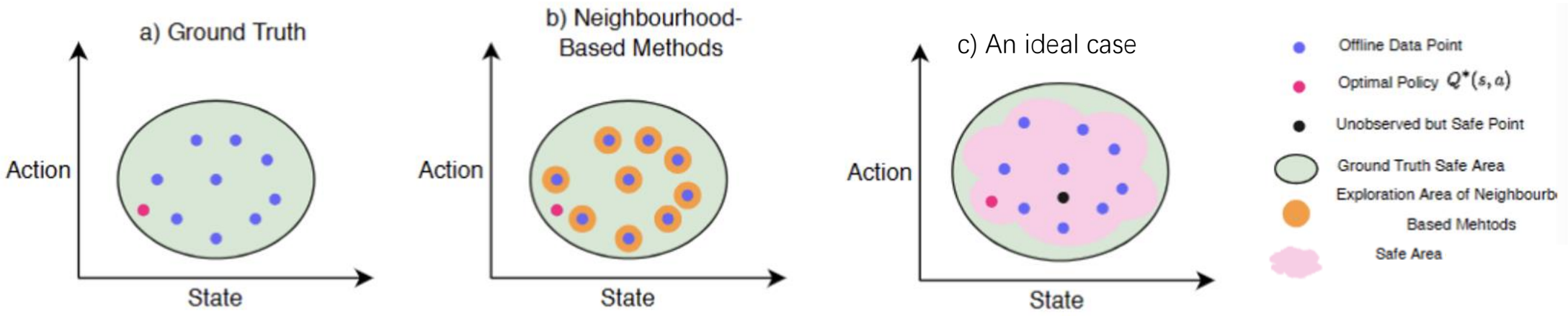
$$-\log \pi_\beta(\pi_\phi(a|s)) < \epsilon$$



Open question

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Most policy control methods are neighborhood-based
- Too conservative (as they use distribution matching)



Content

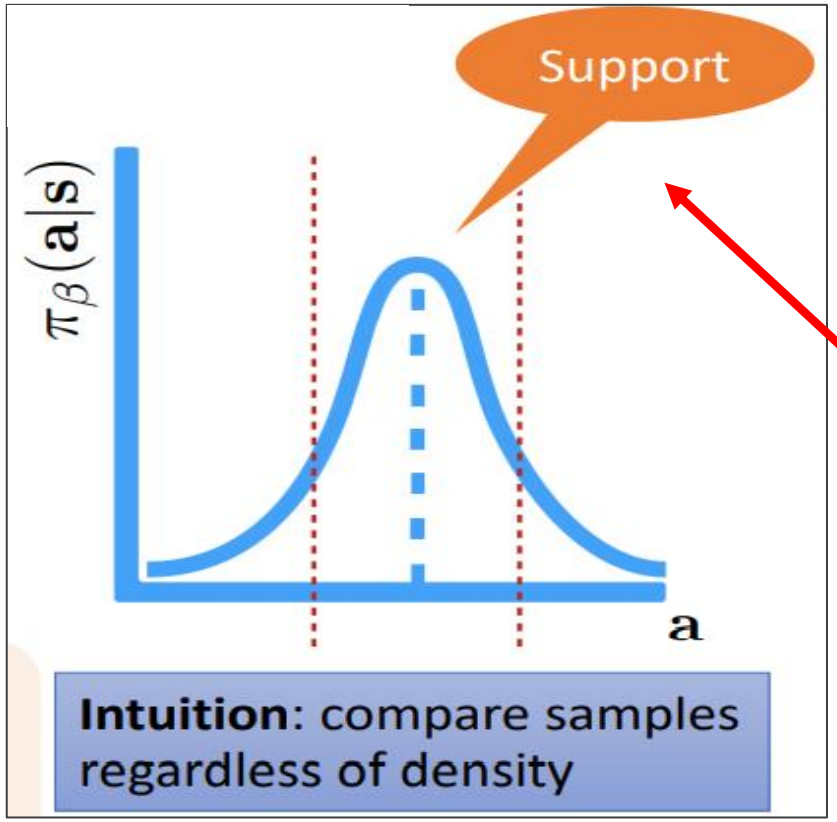
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Background
- Offline Reinforcement Learning
- Constrained Policy Optimization with Explicit Behavior Density (CPED)
- Q-Distribution guided Q-learning (QDQ)
- Diffusion Actor-Critic (DAC)
- Conclusion

CPED: Motivation

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Our proposed CEPD does support constraint very intuitively by solving the difficult problem of estimating the density of behavior policies and obtaining the density function of behavior policies directly.



Explicit density of π_β

Gan +Flow

CPED

CPED: Motivation

- Estimate the behavior policy using offline dataset in inverse RL is equivalent to train the offline dataset with GAN.

Proposition 3.1. *For offline dataset $\mathcal{D}$ generated by behavior policy π_β, the learned likelihood function L^{π_β}, using GAN with hybrid loss in Eq. 4 is equivalent to that trained by MaxEnt IRL. If the generator of GAN can give a specific likelihood function $p_\theta^G(\tau)$, then*

$$L_\theta^{\pi_\beta}(\tau) = CZ^{-1}\exp(-c_\theta(\tau)) \propto p_\theta^G(\tau) \tag{5}$$

where C is a constant related to $\mathcal{D}$.

GAN with generator which can provide a specific density function is a valid estimator for learning the distribution of the behavior policy

Method

- GAN with normalizing flow model gives a specific density function it learns.

Assuming that $f = f_L \circ f_{L-1} \ldots f_2 \circ f_1$ is a set of composed nonlinear transformations from the prior noise to D($\mathbb{R}^A \rightarrow \mathbb{R}^A$, A is the dimension of action space $\mathscr{A}$) and $p_H(Z)$ is a prior noise distribution with known probability density. Based on the normalizing flow model, we can estimate the probability measure on D as follows:

$$p_\theta(\tau) = p_H(f(\tau))|det\frac{\partial f(\tau)}{\partial \tau}|$$

, $\tau \in D$ is a trajectory.

Directly training a Flow-GAN by original loss function in GAN may lead to poor log-likelihoods.

CPED adopt a hybrid loss function containing a MLE and a GAN:

$$\min_\theta \max_\phi \mathcal{L}_\mathcal{G}(G_\theta, D_\phi) - \lambda \mathbb{E}_{\tau \sim P_{data}}[\log(p_\theta(\tau))]$$

Contributions

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- **Keeping the action support** of the learning policy close to that of the behavior policy is the key idea in policy control methods. In our proposed CPED, by **introducing an explicit density function**, we can effectively achieve this objective.
- The combination of **GAN and Flow based density estimation is more suitable for estimating behavior policies** in RL, both in theory and experiments.
- We provide a theoretical guarantee: the CPED can access the optimal function, and extensive experiments show that CPED substantially outperforms SOTA methods.
- CPED is not only intended to demonstrate its superiority through experimental results but also aims to **advance the completeness of policy control methods**.
- The CPED has **significant potential**, considering its representation of the policy density learning process in theory and its promising performance in experiments.
- Our work serves as a catalyst, driving the development of the idea of estimating behavior policies to address offline RL problems.

Algorithms

- The objective of Flow-GAN is the hybrid loss:

$$\min_{\theta}\max_{\phi}\mathcal{L}_{\mathcal{G}}(G_\theta, D_\phi) - \lambda\mathbb{E}_{(s,a)\sim D}[\log(L_\theta^{\pi_\beta}(s,a))].$$

⇒ density estimator of behavior policy: $L_\theta^{\pi_\beta}(\cdot)$.

- The critic training follows from Bellman equation:

$$\mathbb{E}_{s'\sim T(s'|s,a),a'\sim\pi(a|s)}[Q(s,a) - (r(s,a) + \gamma\mathbb{E}_{s''\sim T(s'|s,a)}[\max_{a'} Q(s',a')])]^2.$$

- The actor learning is processed in a bounded safe region (support of behavior policy):

$$\max_{\psi}\ \mathbb{E}_{s\sim\mathbb{P}_{\mathcal{D}},a\sim\pi_\psi(\cdot|s),(s,a)\in\widetilde{\mathcal{S}}\times\widetilde{\mathcal{A}}}[Q_\eta(s,a)].$$

where $\widetilde{\mathcal{S}}\times\widetilde{\mathcal{A}} = \{(s,a)\in\mathcal{S}\times\mathcal{A} : -\log L_\theta^{\pi_\beta}(s,a) < \epsilon\}$.

Algorithm

Algorithm 1 The CPED algorithm

Input: dataset $\mathcal{D}$, target network update rate κ, Lagrange multiplier α, training ratio of the generator and discrimination r, mini-batch size N, hyperparameters λ, c.
Initialize generator G_θ, discriminator D_ϕ, loss function in GAN $\mathcal{L}_\mathcal{G}(\cdot)$, behavior policy likelihood $L_\theta^{\pi_\beta}(\cdot)$, Q networks $\{Q_{\eta_1}, Q_{\eta_2}\}$, actor π_ψ, target networks $\{Q_{\eta_1'}, Q_{\eta_2'}\}$, target actor $\pi_{\psi'}$, with $\psi' \leftarrow \psi, \eta_i' \leftarrow \eta_i, i = 1, 2$.
for $i = 1$ **to** M **do**
  Sample mini-batch of transitions $(s, a, r, s') \sim \mathcal{D}$
  Initial Training Flow-GAN:
  $\phi \leftarrow \text{argmax}_\phi \, \mathcal{L}_\mathcal{G}(G_\theta, D_\phi)$
  $\theta \leftarrow \text{argmin}_\theta \, \text{max}_\phi \, \mathcal{L}_\mathcal{G}(G_\theta, D_\phi) - [\lambda \mathbb{E}_{(s,a)\sim\mathcal{D}} \log(L_\theta^{\pi_\beta}(s, a))]$
end for
for $i = 1$ **to** N **do**
  Sample mini-batch of transitions $(s, a, r, s') \sim \mathcal{D}$
  Updating Flow-GAN:
  $\phi \leftarrow \text{argmax}_\phi \, \mathcal{L}_\mathcal{G}(G_\theta, D_\phi)$
  $\theta \leftarrow \text{argmin}_\theta \, \text{max}_\phi \, \mathcal{L}_\mathcal{G}(G_\theta, D_\phi) - [\lambda \mathbb{E}_{(s,a)\sim\mathcal{D}} \log(L_\theta^{\pi_\beta}(s, a))]$
  Updating Q-function:
  Get action for the next state, $\{\tilde{a} \sim \pi_{\psi'}(\cdot|s') + \varepsilon, \varepsilon \sim clip(\mathcal{N}(0, \sigma), -c, c)\}$
  Let $y(s, a) := \min(Q_{\eta_1'}(s', \tilde{a}), Q_{\eta_2'}(s', \tilde{a}))$
  $\eta_i \leftarrow \text{argmin}_{\eta_i}(Q_{\eta_i}(s, a) - (r + \gamma y(s, a)))^2, i = 1, 2$
  Updating Actor:
  Update ψ according to Eq.7, by using dual gradient descent with Lagrange multiplier α
  Update Target Networks:
  $\psi' \leftarrow \kappa\psi + (1 - \kappa)\psi'; \eta_i' \leftarrow \kappa\eta_i + (1 - \kappa)\eta_i', i = 1, 2$
end for

Theoretical Findings

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Convergence of GAN with Hybrid Loss and using flow model as generator:

Theorem 4.1(informal)

$$d_{\mathcal{F}_d}(\mu^*, \mu_n) = O_{\mathbb{P}}(n^{-1/2}), \mathrm{KL}(\mu^* || \mu_n) = O_{\mathbb{P}}(n^{-1/2}),$$

where μ^* is the true probability measure, μ_n is is an accurate estimator of the underlying true distribution.

i.e., Integral Probability Metric (IPM) and K-L divergence achieve a fast convergence rate $O_p(n^{-1/2})$

- Hence, hybrid loss provides: (1) good generator; (2) accurate density estimate, which can be used to control the safe region.

Theoretical Findings

- CPED achieves optimal value function with a linear convergence rate as the iteration approaches infinity.
- Despite under offline settings, this rate is consistent with the standard rate in online RL (Agarwal et al. 2019).
- When combined with Theorems 4.2 and 4.3, we can confidently conclude that CPED's effectiveness is theoretically guaranteed.

Theorem 4.2 Let $\widehat{\pi}^{\Delta}(\cdot|s)$ be the learning policy using CPED, which is defined on the support of π_β. Suppose $\widehat{\pi}^{\Delta}(\cdot|s) > 0$ on A. Then with probability tending to one,

$$Q^*(s,a) = Q(s, \widehat{\pi}^{\Delta}(\cdot|s)).$$

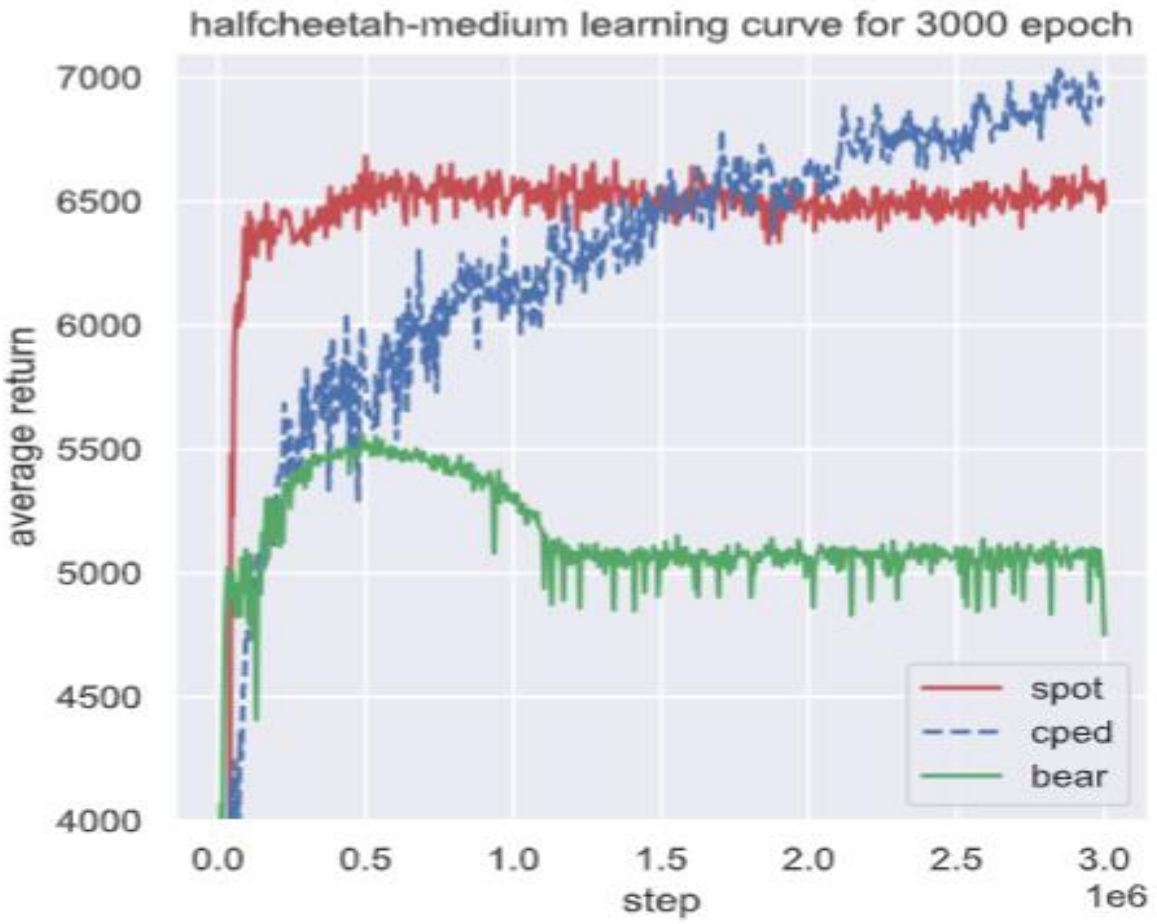
Theorem 4.3 With probability tending to one,

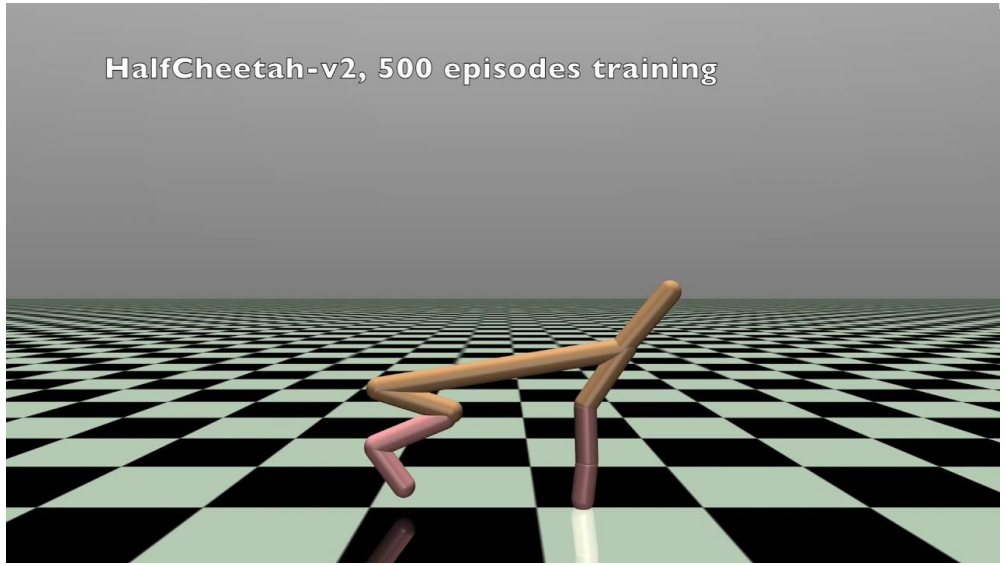
$$\|V^{\widehat{\pi}^{\Delta}_{k+1}} - V^*\|_\infty \leq \gamma^{k+1} \|V^{\widehat{\pi}^{\Delta}_0} - V^*\|_\infty.$$

Experiments

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

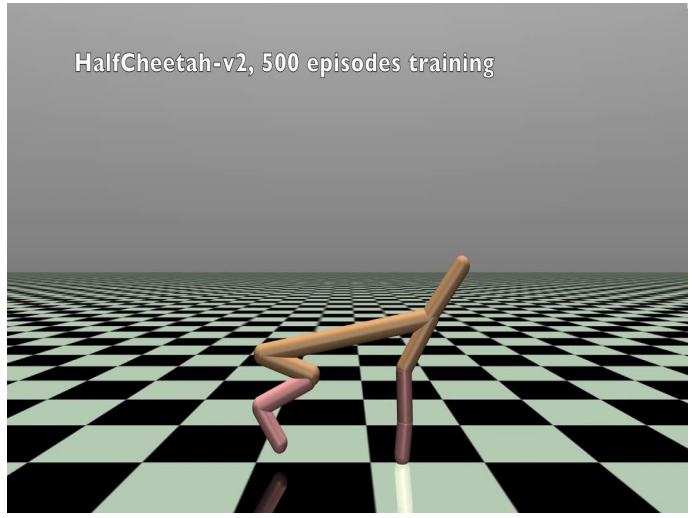
- Gym-MuJoCo task: half-cheetah

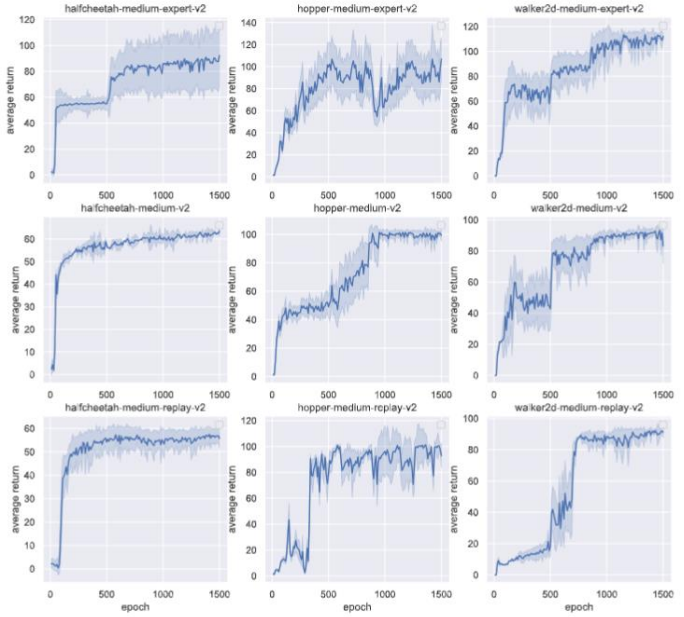




CPED is a support consistency methods so it can avoid being too conservative.

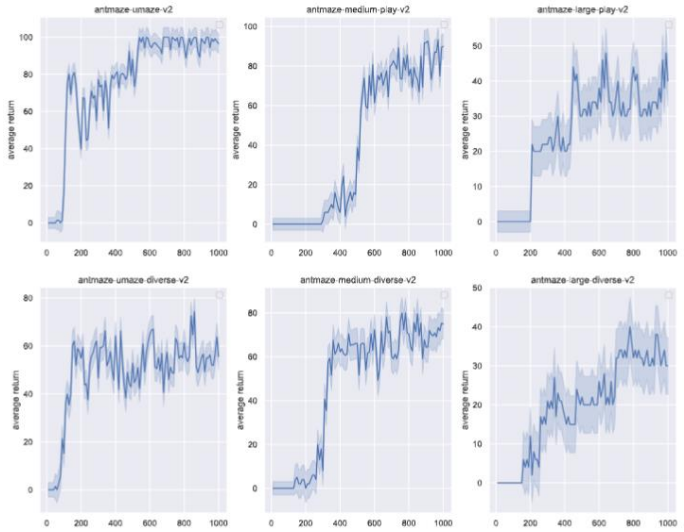
Experiments: Gym-MuJoCo tasks





Dataset	BC	AWAC	DT	Onestep	TD3+BC	CQL	IQL	SPOT	CPED(Ours)
hc-med	42.6	43.5	42.6	48.4	48.3	44.0	47.4	58.4	**61.8**$_{\pm 1.6}$
ho-med	52.9	57.0	67.6	59.6	59.3	58.5	66.2	86.0	**100.1**$_{\pm 2.8}$
wa-med	75.3	72.4	74.0	81.8	83.7	72.5	78.3	86.4	**90.2**$_{\pm 1.7}$
hc-med-r	36.6	40.5	36.6	38.1	44.6	45.5	44.2	52.2	**55.8**$_{\pm 2.9}$
ho-med-r	18.1	37.2	82.7	97.5	60.9	95.0	94.7	**100.2**	98.1$_{\pm 2.1}$
wa-med-r	26.0	27.0	66.6	49.5	81.8	77.2	73.8	91.6	**91.9**$_{\pm 0.9}$
hc-med-e	55.2	42.8	86.8	**93.4**	90.7	91.6	86.7	86.9	85.4$_{\pm 10.9}$
ho-med-e	52.5	55.8	**107.6**	103.3	98.0	105.4	91.5	99.3	95.3$_{\pm 13.5}$
wa-med-e	107.5	74.5	108.1	113.0	110.1	108.8	109.6	112.0	**113.04**$_{\pm 1.4}$
Total	466.7	450.7	672.6	684.6	677.4	698.5	692.4	773.0	**791.7**$_{\pm 37.8}$

Experiments: AntMaze tasks



Dataset	BCQ	BEAR	BC	DT	TD3+BC	PLAS	CQL	IQL	SPOT	CPED(Ours)
umaze	78.9	73.0	49.2	54.2	73.0	62.0	82.6	89.6	93.5	**96.8**$_{\pm 2.6}$
umaze-d	55.0	61.0	41.8	41.2	47.0	45.4	10.2	**65.6**	40.7	55.6$_{\pm 2.2}$
med-p	0.0	0.0	0.4	0.0	0.0	31.4	59.0	76.4	74.7	**85.1**$_{\pm 3.4}$
med-d	0.0	8.0	0.2	0.0	0.2	20.6	46.6	72.8	**79.1**	72.1$_{\pm 2.9}$
large-p	6.7	0.0	0.0	0.0	0.0	2.2	16.4	**42.0**	35.3	34.9$_{\pm 5.3}$
large-d	2.2	0.0	0.0	0.0	0.0	3.0	3.2	**46.0**	36.3	32.3$_{\pm 7.4}$
Total	142.8	142.0	91.6	95.4	120.2	164.6	218.0	**392.4**	359.6	376.8$_{\pm 23.8}$

Content

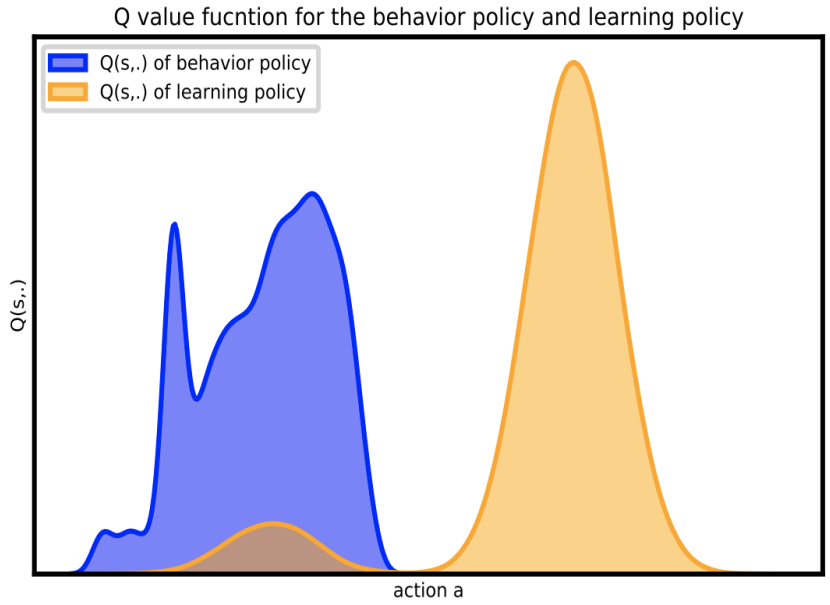
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Background
- Offline Reinforcement Learning
- Constrained Policy Optimization with Explicit Behavior Density (CPED)
- Q-Distribution guided Q-learning (QDQ)
- Diffusion Actor-Critic (DAC)
- Conclusion

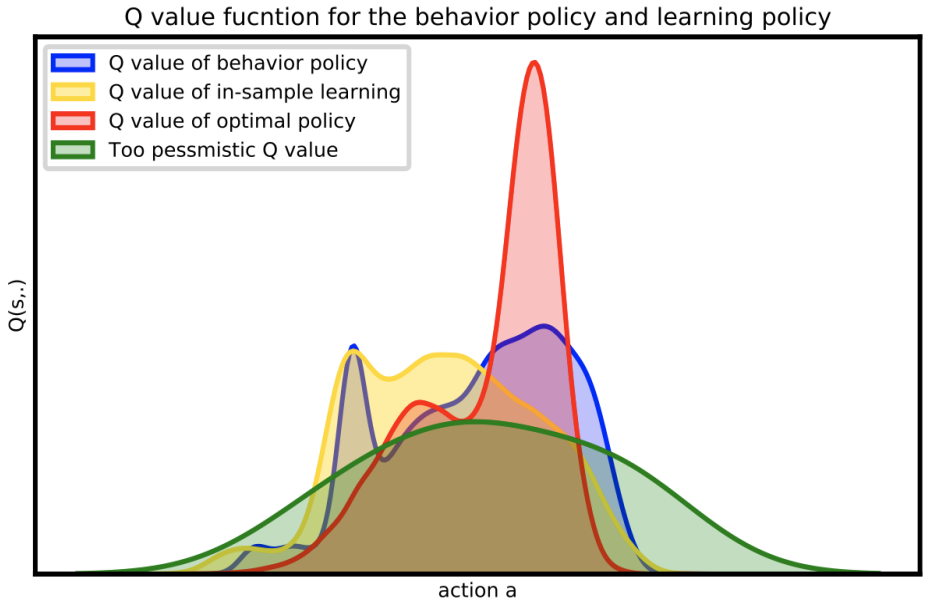
QDQ: Motivation

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Sometimes it is hard to explicitly estimate the behavior policy.
- A pessimistic approach: Add penalizations on Q-function.



(a)



(b)

QDQ: Motivation

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

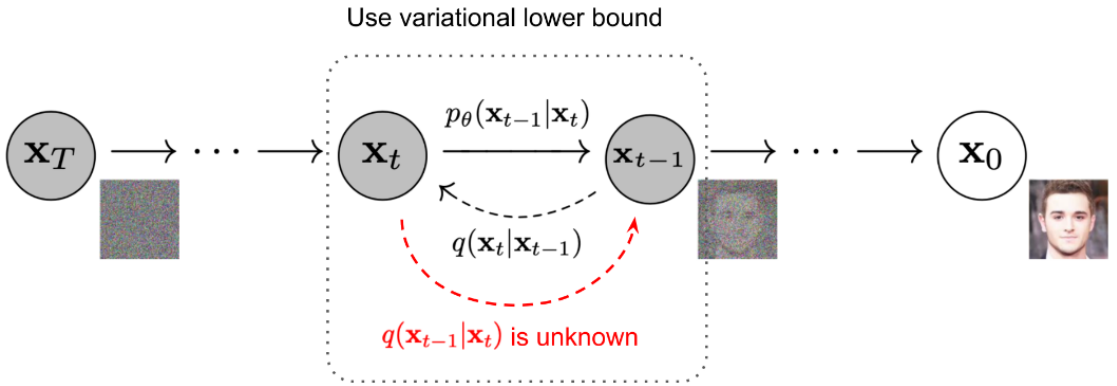
- One common method to identify whether the Q-value function is updated by OOD actions is estimating the uncertainty of the Q-value.
- However, estimating the uncertainty of the Q function poses a significant challenge, particularly in the context of a deep neural network Q estimator.
- We utilize the consistency model, which is a faster generator than the original diffusion model.

Denoising diffusion probabilistic models (DDPM)

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- *Forward diffusion process*: add small amount of Gaussian noise to the sample in T steps

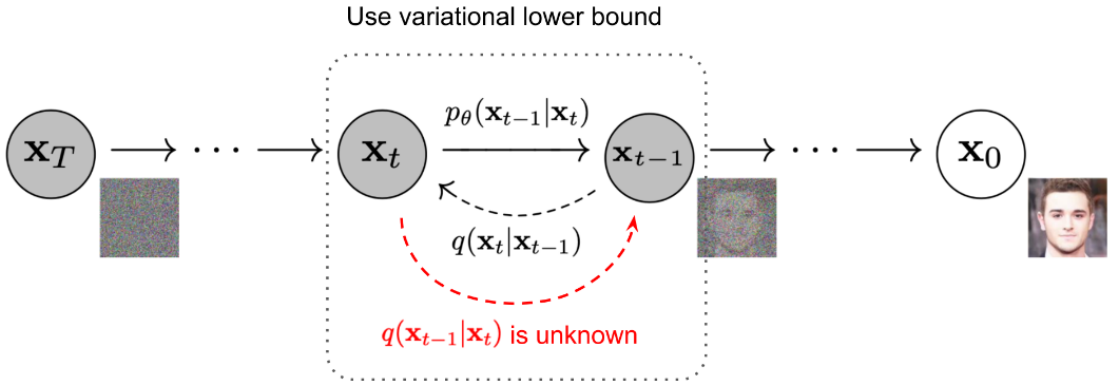
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$



Reverse diffusion process

- Learn a model p_θ to approximate the reverse diffusion process:

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



Reverse diffusion process

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Algorithm 1 Training

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

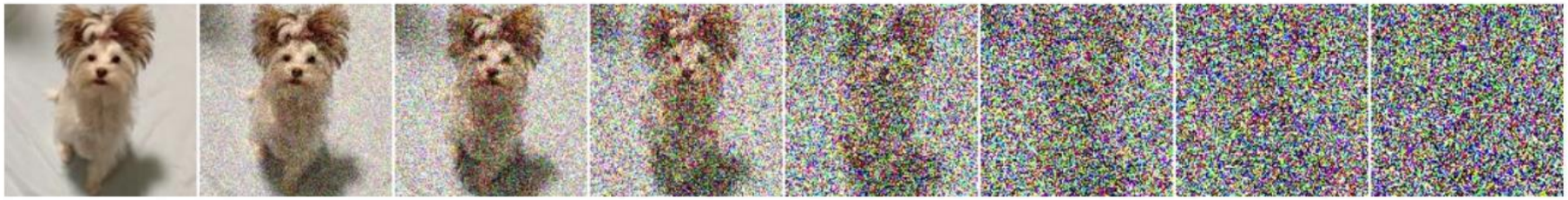
Algorithm 2 Sampling

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

Diffusion models

Forward SDE (data → noise)

$\mathbf{x}(0)$ — $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$ → $\mathbf{x}(T)$



score function

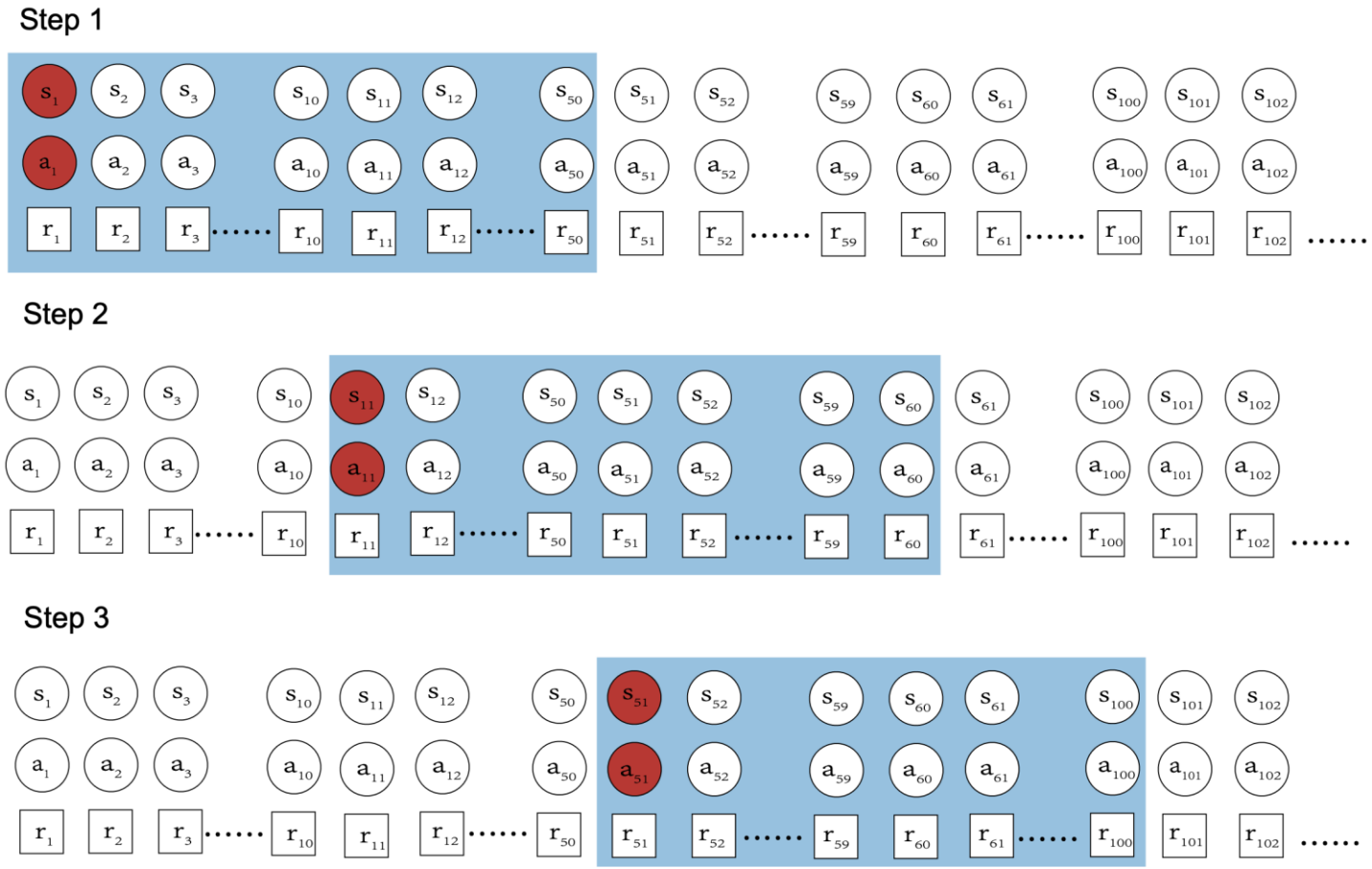
$\mathbf{x}(0)$ ← $\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$ — $\mathbf{x}(T)$

Reverse SDE (noise → data)

QDQ: several key ideas

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Trajectory-level truncated Q-value for generating raw data



QDQ: several key ideas

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- The uncertainty-aware learning objective for Q-value function

$$\mathcal{L}_{adv}(Q) = \min_{\theta}\{\alpha\mathcal{L}(Q)_H + (1-\alpha)\mathcal{L}(Q)_L\}. \tag{7}$$

In Eq.7, $\mathcal{L}(Q)_H$ represent the classic Bellman residual as defined in Eq.2 that is used in online RL and it encourages the optimistic optimization of the Q-value, while $\mathcal{L}(Q)_L$ is a pessimistic Bellman residual based on the uncertainty penalized Q target $Q_L(s', a')$, defined as

$$Q_L(s', a') = \frac{1}{\mathcal{H}_Q(a'|s')}Q(s', a')1_{(a' \in \mathcal{U}(Q))} + \beta Q(s', a')1_{(a' \notin \mathcal{U}(Q))}. \tag{8}$$

- Improve the learning policy.

$$\mathcal{L}_{\phi}(\pi) = \max_{\phi} \left[\mathbb{E}_{s\sim\mathbb{P}_{\mathcal{D}}(s), a\sim\pi_{\phi}(\cdot|s)}[Q_{\theta}(s,a)] + \gamma\mathbb{E}_{a\sim\mathcal{D}}[\log\pi_{\phi}(a)]\right].$$

Algorithm

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Algorithm 1 Q-Distribution guided Q-learning (QDQ)

Initialize: target network update rate κ, uncertainty-aware learning hyperparameter α, β,,policy training hyperparameters γ. Consistency model f_η, Q networks $\{Q_{\theta_1}, Q_{\theta_2}\}$, actor π_ϕ, target networks $\{Q_{\theta'_1}, Q_{\theta'_2}\}$, target actor $\pi_{\phi'}$.

Q-distribution learning:

Calculate Q dataset $\mathcal{D}_Q = \{Q_\mathcal{T}^{\pi_\beta}(s,a)\}$ scanning each trajectory $\tau \in \mathcal{D}$ by Eq.5.

for each gradient step **do**

  Sample minibatch of $q_\mathcal{T}^{\pi_\beta}(s,a) \sim \mathcal{D}_Q$

  Update η minimizing consistency distillation loss in Eq.(7) [19]

end for

for each gradient step **do**

  Sample mini-batch of transitions $(s,a,r,s') \sim \mathcal{D}$

  Updating Q-function:

  Update $\theta = (\theta_1, \theta_2)$ minimizing $\mathcal{L}_{adv}(Q)$ in Eq.7

  Updating policy:

  Update ϕ minimizing $\mathcal{L}_\phi(\pi)$ in Eq.9

  Update Target Networks:

  $\phi' \leftarrow \kappa\phi + (1-\kappa)\phi'; \theta'_i \leftarrow \kappa\theta_i + (1-\kappa)\theta'_i, i = 1,2$

end for

Theoretical guarantee

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- The optimal Q-value, Q^{Δ}, derived by the QDQ algorithm can closely approximate the optimal Q-value function, Q^{*}, benefiting from the balanced approach of the QDQ algorithm that avoids excessive pessimism.

Theorem 4.4 (Informal). *Under mild conditions, with probability* $1-\eta$ *we have*

$$\left\|Q^{\Delta}-Q^{*}\right\|_{\infty} \leq \epsilon, \tag{12}$$

where Q^{Δ} *is learned by the uncertainty-aware loss Eq.7,* ϵ *is error rate related to the difference between the classical Bellman operator* $\mathcal{B}Q$ *and* $\mathcal{F}Q$.

Experiments: Gym-MuJoCo tasks and AntMaze tasks

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Dataset	BC	AWAC	DT	TD3+BC	CQL	IQL	UWAC	MCQ	QDQ(Ours)
ha-med	42.6	43.5	42.6	48.3	44.0	47.4	42.2	64.3	**74.1**±1.7
ho-med	52.9	57.0	67.6	59.3	58.5	66.2	50.9	78.4	**87.0**±4.3
wa-med	75.3	72.4	74.0	83.7	72.5	78.3	75.4	**91.0**	86.9±0.08
ha-med-r	36.6	40.5	36.6	44.6	45.5	44.2	35.9	56.8	**63.7**±2.9
ho-med-r	18.1	37.2	82.7	60.9	95.0	94.7	25.3	101.6	**102.4**±0.28
wa-med-r	26.0	27.0	66.6	81.8	77.2	73.8	23.6	91.3	**93.2**±1.1
ha-med-e	55.2	42.8	86.8	90.7	91.6	86.7	42.7	87.5	**89.4**±3.5
ho-med-e	52.5	55.8	107.6	98.0	105.4	91.5	44.9	112.3	**113.5**±3.5
wa-med-e	107.5	74.5	108.1	110.1	108.8	109.6	96.5	114.2	**115.9**±0.2
Total	466.7	450.7	672.6	684.6	677.4	698.5	437.4	797.4	**826.1**±17.56

Dataset	BC	TD3+BC	DT	Onestep RL	AWAC	CQL	IQL	QDQ(Ours)
umaze	54.6	78.6	59.2	64.3	56.7	74.0	87.5	**90.9**±3.1
umaze-diverse	45.6	71.4	53.0	60.7	49.3	**84.0**	62.2	67.8±2.5
medium-play	0.0	10.6	0.0	0.3	0.0	61.2	71.2	**78.5**±2.2
medium-diverse	0.0	3.0	0.0	0.0	0.7	53.7	70.0	**80.8**±2.6
large-play	0.0	0.2	0.0	0.0	0.0	15.8	**39.6**	28.9±3.4
large-diverse	0.0	0.0	0.0	0.0	1.0	14.9	**47.5**	31.2±4.5
Total	100.2	163.8	112.2	125.3	142.4	229.8	**378**	**378.1**±18.3

Content

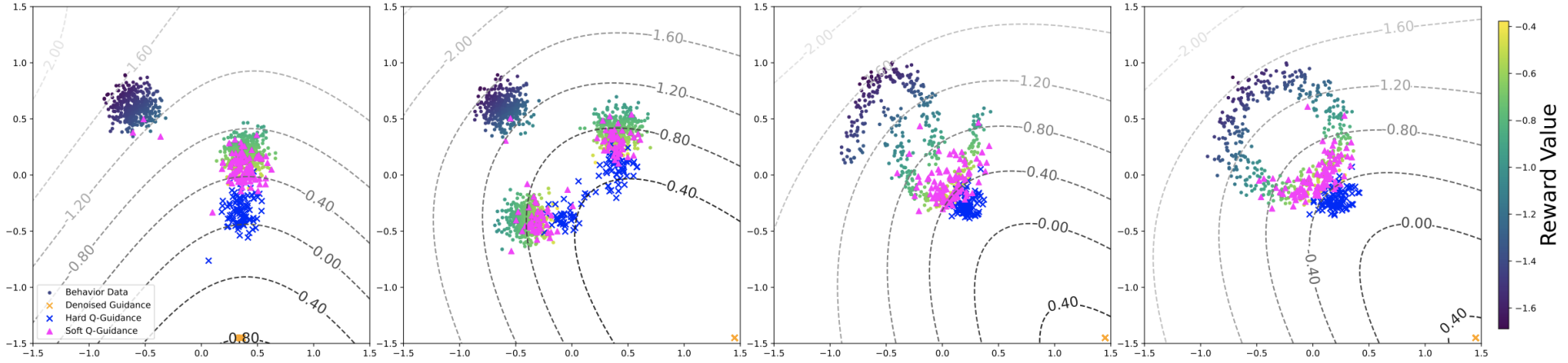
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Background
- Offline Reinforcement Learning
- Constrained Policy Optimization with Explicit Behavior Density (CPED)
- Q-Distribution guided Q-learning (QDQ)
- Diffusion Actor-Critic (DAC)
- Conclusion

DAC: Motivation

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- DAC takes another approach: it directly formulates the Kullback-Leibler (KL) constraint policy iteration as a diffusion noise regression problem.
- Enable a direct representation of target policies as diffusion models.
- The resulting noise prediction target involves a soft Q-guidance term that adjusts the Q-gradient guidance according to the noise scales.



Score-based generative modeling

- Specifically, starting from any prior distribution $x_0 \sim \pi(x)$,

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \sqrt{2\epsilon}\, \mathbf{z}_i, \quad i = 0, 1, \cdots, K,$$

where $z_i \sim N(0, I)$.

- Compared to standard SGD, stochastic gradient Langevin dynamics injects Gaussian noise into the parameter updates to avoid collapses into local minima.

DAC: Motivation

- It turns out that

$$\nabla_{\mathbf{x}_t} \log p_t^*(\mathbf{x}_t|\mathbf{s}) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{s}) + \frac{1}{\eta}\nabla_{\mathbf{x}_t} Q^{\pi_k}(\mathbf{s}, \mathbf{x}_t),\ \mathbf{x}_t \in \mathbb{R}^d,$$

where $p_t^*(\mathbf{x}_t|\mathbf{s}) = \int q_t(\mathbf{x}_t|\mathbf{a})\pi^*(\mathbf{a}|\mathbf{s})d\mathbf{a}$ and $p_t(\mathbf{x}_t|\mathbf{s}) = \int q_t(\mathbf{x}_t|\mathbf{a})\pi_\beta(\mathbf{a}|\mathbf{s})d\mathbf{a}$ are noise distributions.

Theorem 1. *Let* $\boldsymbol{\epsilon}^*(\mathbf{x}_t, \mathbf{s}, t) := -\sqrt{1-\bar{\alpha}_t}\,\nabla_{\mathbf{x}_t} \log p_t^*(\mathbf{x}_t|\mathbf{s})$. *Then* $\boldsymbol{\epsilon}^*(\mathbf{x}_t, \mathbf{s}, t)$ *is a Gaussian noise predictor which defines a diffusion model for generating* π^*.

- Hence, we transform the original offline RL problem into a noise regression problem:

$$\arg\min_\theta \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}, \mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{a}), t} ||\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{s}, t) - \boldsymbol{\epsilon}^*(\mathbf{x}_t, \mathbf{s}, t)||^2.$$

DAC: Motivation

- Such learning objective has an equivalent form which is easy to optimize.

Theorem 2. *Training parameters θ according to (14) is equivalent to optimize the following objective:*

$$\arg\min_{\theta} \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t} ||\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{s}, t) - \boldsymbol{\epsilon} + \frac{1}{\eta}\sqrt{1-\bar{\alpha}_t}\,\nabla_{\mathbf{x}_t} Q^{\pi_k}(\mathbf{s}, \mathbf{x}_t)||^2, \tag{15}$$

where $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{a} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$.

- A direct representation through diffusion models.

DAC: Several other key designs

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- Policy evaluation via Q-ensemble: An ensemble of H parameterized Q-networks:

$$\phi_k^h \leftarrow \arg\min_{\phi^h} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D},\mathbf{a}'\sim\pi_{\theta_k}} \left[r + \gamma Q_{\bar{\phi}_{k-1}^h}(\mathbf{s}',\mathbf{a}') - Q_{\phi^h}(\mathbf{s},\mathbf{a})\right]^2, h \in \{1,2,...,H\}.$$

  - However, this may lead to overestimation bias and unsuccessful learning.
- Our approach utilizes the lower confidence bound (LCB) of Q-ensembles:

$$\mathcal{L}_C(\phi^h) = \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D},\mathbf{a}'\sim\pi_{\theta_k}} \left[r + \gamma Q_{\mathrm{LCB}}(\mathbf{s}',\mathbf{a}') - Q_{\phi^h}(\mathbf{s},\mathbf{a})\right]^2,$$

$$Q_{\mathrm{LCB}}(\mathbf{s}',\mathbf{a}') = \mathbb{E}_h[Q_{\bar{\phi}_k^h}(\mathbf{s}',\mathbf{a}')] - \rho\sqrt{\mathrm{Var}_h[Q_{\bar{\phi}_k^h}(\mathbf{s}',\mathbf{a}')]},$$

Algorithm

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Algorithm 1 Diffusion Actor-Critic Training

Require: offline dataset $\mathcal{D}$, batch size B, learning rates α_ϕ, α_θ, α_η and α_{ema}, behavior cloning threshold b, pessimism factor ρ

Initialize: diffusion policy $\boldsymbol{\epsilon}_\theta$, target diffusion policy $\boldsymbol{\epsilon}_{\bar{\theta}} = \boldsymbol{\epsilon}_\theta$, Q-networks Q_{ϕ^h}, target Q-networks $Q_{\bar{\phi}^h} = Q_{\phi^h}$ $(h = 1, 2, ..., H)$, Lagrangian multiplier $\eta = \eta_{\text{init}}$

while training not convergent **do**

  Sample a batch of B transitions $\{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')\} \subset \mathcal{D}$

  Sample $\mathbf{a}' = \mathbf{x}_0$ through denoising process using noise predictor $\boldsymbol{\epsilon}_{\bar{\theta}}(\mathbf{x}_t, t, \mathbf{s})$.

  for h in $\{1, 2, ..., H\}$ **do**

    Update $\phi^h \leftarrow \phi^h - \alpha_\phi \nabla_{\phi^h} \mathcal{L}_C(\phi^h)$ (18) ▷ Critic learning

  end for

  Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \text{Unif}(0, T)$ and compute $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{a} + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$

  Estimate Q-gradient $\nabla_{\mathbf{x}_t} Q^{\pi_k}(\mathbf{s}, \mathbf{x}_t)$ using (19)

  $\theta \leftarrow \theta - \alpha_\theta \nabla_\theta \mathcal{L}_A(\theta)$ (16) ▷ Actor learning

  $\eta \leftarrow \eta + \alpha_\eta(||\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{s}, t) - \boldsymbol{\epsilon}||^2 - b)$ ▷ Dual gradient ascent (optional)

  $\bar{\theta} \leftarrow (1 - \alpha_{\text{ema}})\bar{\theta} + \alpha_{\text{ema}}\theta$

  $\bar{\phi}^h \leftarrow (1 - \alpha_{\text{ema}})\bar{\phi}^h + \alpha_{\text{ema}}\phi^h$ ▷ Update target networks using EMA

end while

Experiments

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Dataset	TD3+BC	AWAC	Onestep-RL	CQL	IQL	IVR	EQL	Diffuser	SfBC	DQL	IDQL-A	DAC (ours)
halfcheetah-m	48.3	43.5	48.4	44.0	47.4	48.3	48.3	44.2	45.9	51.1	51.0	**59.1** ± 0.4
hopper-m	59.3	57.0	59.6	58.5	66.3	75.5	74.2	58.5	57.1	90.5	65.4	**101.2** ± 2.0
walker2d-m	83.7	72.4	81.8	72.5	78.3	84.2	84.2	79.7	77.9	87.0	82.5	**96.8** ± 3.6
halfcheetah-m-r	44.6	40.5	38.1	45.5	44.2	44.8	45.2	42.2	37.1	47.8	45.9	**55.0** ± 0.2
hopper-m-r	60.9	37.2	97.5	95.0	94.7	**99.7**	**100.7**	96.8	86.2	**101.3**	92.1	**103.1** ± 0.3
walker2d-m-r	81.8	27.0	49.5	77.2	73.9	81.2	82.2	61.2	65.1	**95.5**	85.1	**96.8** ± 1.0
halfcheetah-m-e	90.7	42.8	93.4	91.6	86.7	94.0	**94.2**	79.8	92.6	**96.8**	**95.9**	**99.1** ± 0.9
hopper-m-e	98.0	55.8	103.3	105.4	91.5	**111.8**	**111.2**	**107.2**	**108.6**	**111.1**	**108.6**	**111.7** ± 1.0
walker2d-m-e	**110.1**	74.5	**113.0**	**108.8**	**109.6**	**110.2**	**112.7**	**108.4**	**109.8**	**110.1**	**112.7**	**113.6** ± 3.5
locomotion-v2 total	677.4	450.7	684.6	698.5	749.7	749.7	752.9	678.0	680.3	791.2	739.2	**836.4**
antmaze-u	78.6	56.7	64.3	74.0	87.5	93.2	93.8	-	92.0	93.4	94.0	**99.5** ± 0.9
antmaze-u-div	71.4	49.3	60.7	**84.0**	62.2	74.0	**82.0**	-	**85.3**	66.2	80.2	**85.0** ± 7.9
antmaze-m-play	10.6	0.0	0.3	61.2	71.2	80.2	76.0	-	81.3	76.6	**84.2**	**85.8** ± 5.5
antmaze-m-div	3.0	0.7	0.0	53.7	70.0	79.1	73.6	-	**82.0**	78.6	**84.8**	**84.0** ± 6.2
antmaze-l-play	0.2	0.0	0.0	15.8	39.6	53.2	46.5	-	59.3	46.4	**63.5**	50.3 ± 8.6
antmaze-l-div	0.0	1.0	0.0	14.9	47.5	52.3	49.0	-	45.5	56.6	**67.9**	55.3 ± 10.3
antmaze-v0 total	168.3	107.7	125.3	303.6	378.0	432.0	420.9	-	445.4	417.8	**474.6**	**459.9**

Conclusion

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

- We leverage the generative models, including flow GAN and diffusion models, to avoid the learning of policies that visit OOD points.
- We validate the effectiveness of our methods both theoretically and experimentally.
- Experimental results on the Gym-MuJoCo and AntMaze tasks demonstrate that our methods surpass state-of-the-art competitors, highlighting their superior performance.

Thank you for your attention!
Any questions?

THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY (GUANGZHOU)