Learning Pseudometric-based Action Representations for Offline Reinforcement Learning

Pengjie Gu[1] , Mengchen Zhao[2,*] , Chen Chen[2] , Dong Li[2] , Jianye Hao[3,2] , Bo An[1]

School of Computer Science and Engineering, Nanyang Technological University, Singapore[1]

Noah's Ark Lab, Huawei[2]

College of Intelligence and Computing, Tianjin University[3]

ICML'22

Background

- **Offline RL** is promising for practical applications since it does **not** require **interactions** with **real-world** environments.
- Existing methods focus on environments with **continuous or small discrete action spaces**.
- To address the issue of overestimating the values of out-of -distribution (o.o.d.) actions, they usually **constrain** the learned policy to stay **close** to the data-generating policies

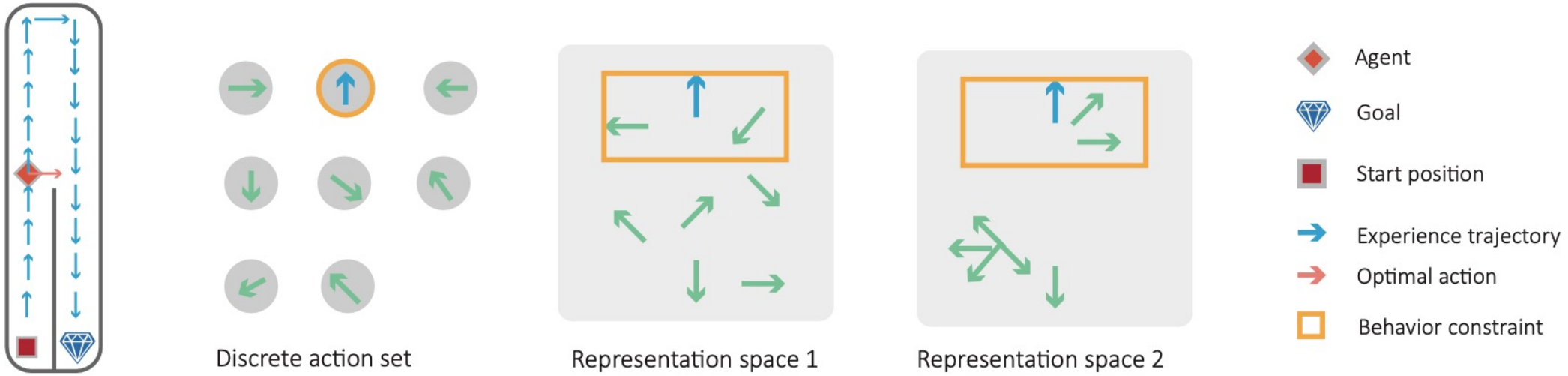
Background

- However, the performance of these algorithms **decreases** drastically **with the size of action space increasing**. Two major reasons:

1. The value function **hardly generalizes** over the entire action space **without proper action representations**.
2. Logged state-action pairs are extremely sparse to the entire state-action space, resulting in **overly restrictive policies**.

Background

- Online RL benefits from using **action representations** to **exploit underlying structures of large action spaces.**
- They fail to learn **reasonable relative distances** between actions, so they cause **Inappropriate behavioral regularizations** of offline RL algorithms.

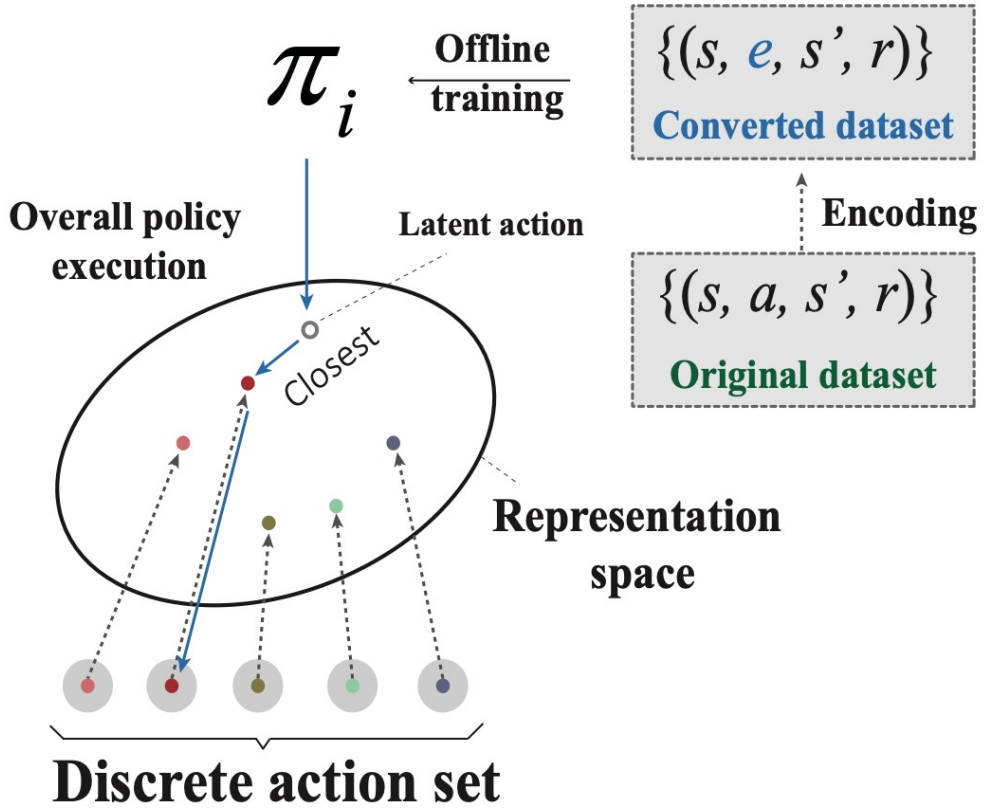


Example

Overview

- A framework for incorporating action represenations into offline RL.
  - A pseudometric function for measuring relations between actions.
  - A relation network architecture for learning action representations.
- Theoretical analysis.

Overall Framework



Incorporating Action Representations into Offline RL

Pseudometric Function for Measuring Relations between Actions

➢We expect that the learned **action representations' relative distances reflect two major relations** between actions:

1. **The behavioral relation** (reflects the difference between the induced transitions and rewards)

$$d(a_i, a_j|s) = |\mathcal{R}_s^{a_i} - \mathcal{R}_s^{a_j}| + \gamma \cdot W_2(\mathcal{P}_s^{a_i}, \mathcal{P}_s^{a_j})$$

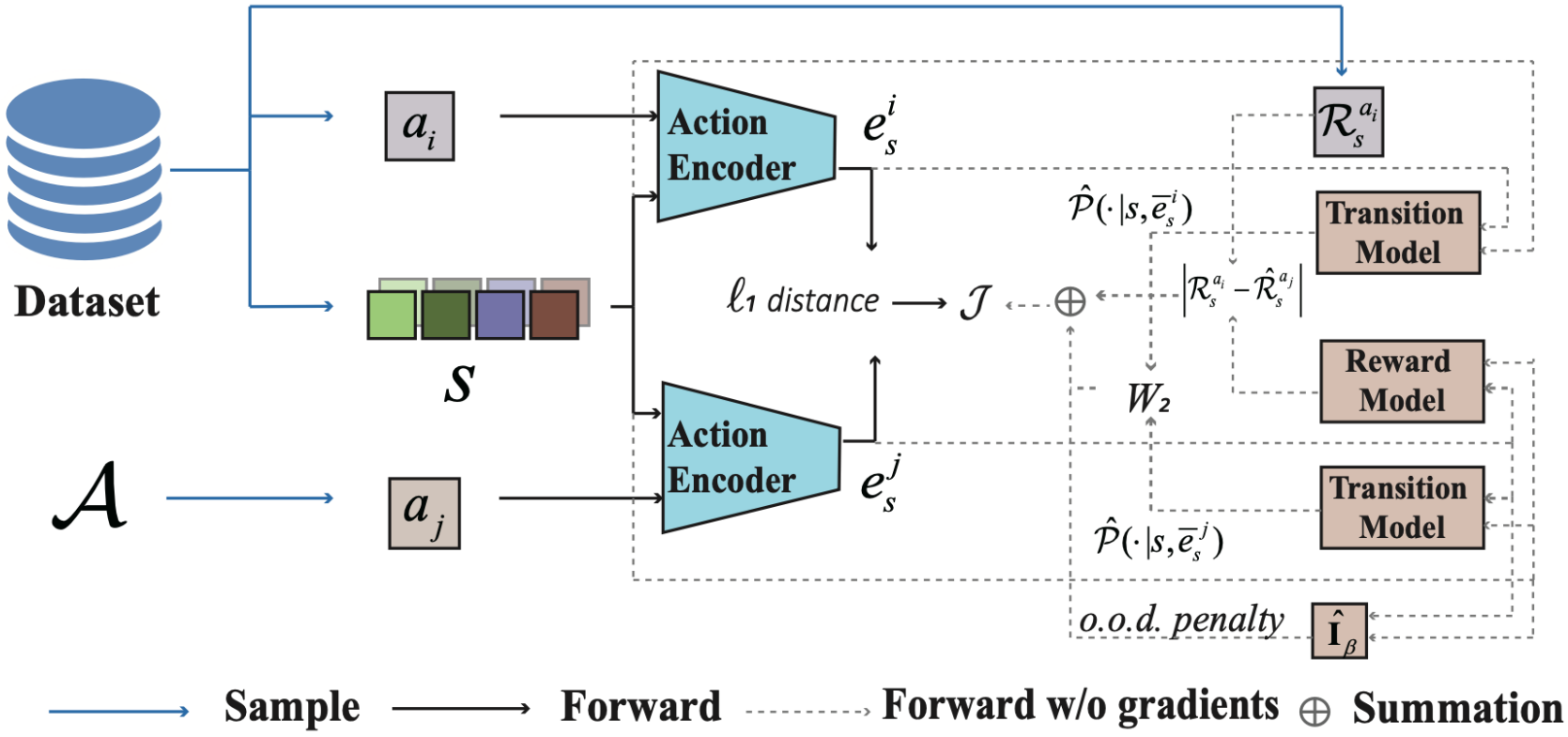
2. **The data-distributional relation** (reflects whether actions are in the same distribution of the experience dataset)

$$d(a_i, a_j|s) = |\mathcal{R}_s^{a_i} - \mathcal{R}_s^{a_j}| + \gamma \cdot W_2(\mathcal{P}_s^{a_i}, \mathcal{P}_s^{a_j}) + p \cdot I_\beta(a_i, a_j|s)$$

Penalty coefficient

equals 1 if two actions are from the same distribution, otherwise, it equals 0.

Learning Pseudometric-based Action Representations



$$J(\phi) = \mathbb{E}_{s,a_i,\mathcal{R}_s^{a_i}\sim\mathcal{D},a_j\sim\mathcal{A}}\left(\|e_s^i - e_s^j\|_1 - \hat{d}(a_i,a_j|s)\right)^2$$

where

$$\begin{aligned}\hat{d}(a_i,a_j|s) =& |\mathcal{R}_s^{a_i} - \hat{\mathcal{R}}(s,\bar{e}_s^j)| \\ &+ \gamma \cdot W_2(\hat{\mathcal{P}}(\cdot|s,\bar{e}_s^i),\hat{\mathcal{P}}(\cdot|s,\bar{e}_s^j)) \\ &+ p \cdot \hat{I}_\beta(a_j|s)\end{aligned}$$

The architecture of pseudometric-based action representation learning

Theoretical Analysis

Theorem 4.3 (Q^π is Lipschiz with respect to d). *Given a policy π, let Q^π be the value function for a given discount factor γ. Q^π is Lipschitz continuous with respect to d with a Lipschitz constant $\frac{1}{1-\gamma}$*

$$|Q^\pi(s, a_i) - Q^\pi(s, a_j)| \leq \frac{1}{1-\gamma} d(a_i, a_j|s) \qquad (8)$$

the value function of the policy would be **Lipschitz continuous** in the action representation space.

1. brings an effective generalization capability
2. reduces the estimation errors of o.o.d. actions

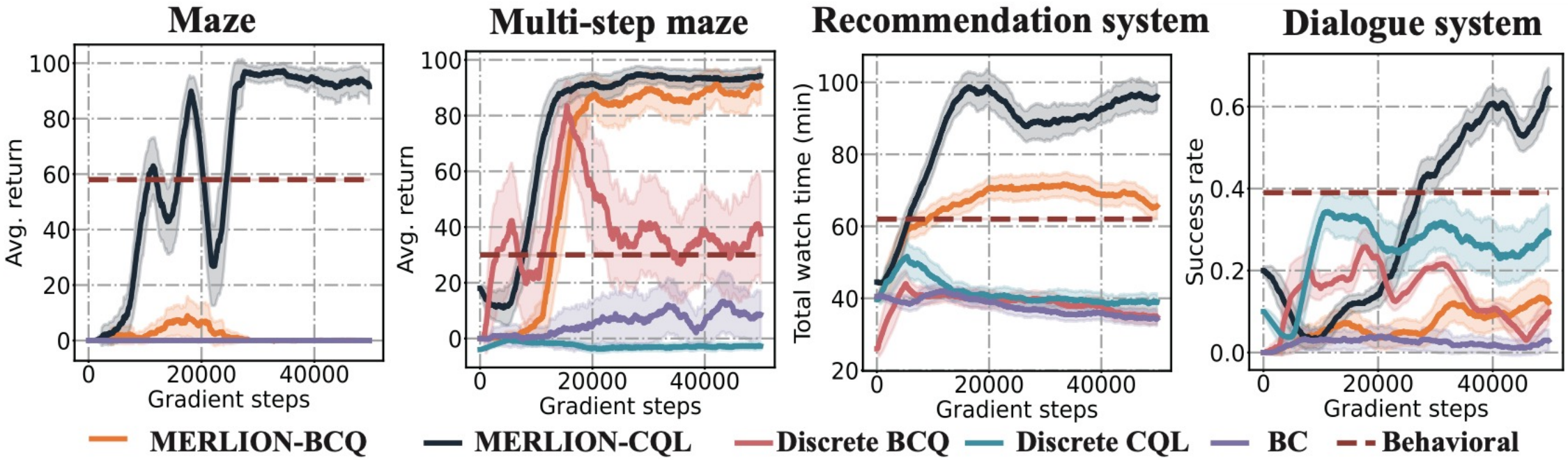
Theorem 4.4 (Performance bound in offline RL). *Let $\pi_i^*(e|s)$ be the policy obtained by CQL performing with MERLION in the constructed MDP $\overline{\mathcal{M}}$ and $\pi_{i,g}^*(a|s)$ refer to the overall policy when $\pi_i^*(e|s)$ is used together with nearest lookup function g. Let $J(\pi, \mathcal{M})$ refer to the expected return of π in $\mathcal{M}$ and $\phi(a; s)$ is the MERLION action encoder, which has a learning error ϵ. Let π_β refer to the behavioral policy generating $\mathcal{D}$ and $\overline{\pi}_\beta(e|s) \equiv e = \phi(a; s), a \sim \pi_\beta(a|s)$.Then, $J(\pi_{i,g}^*, \mathcal{M}) \geq J(\pi_\beta, \mathcal{M}) - k$ where*

$$k = \mathcal{O}\left(\frac{1}{(1-\gamma)^2}\mathbb{E}_{s\sim d_{\hat{\mathcal{M}}}^{\pi_i^*(s)}}\left[\sqrt{|\mathcal{E}|D_{CQL}(\pi_i^*, \overline{\pi}_\beta)(s) + 1}\right]\right) - \frac{\alpha}{1-\gamma}\mathbb{E}_{s\sim d_{\hat{\mathcal{M}}}^{\pi_i^*(s)}}[D_{CQL}(\pi_i^*, \overline{\pi}_\beta)(s)] + \frac{\epsilon + 2\gamma\mathcal{R}_{max}}{1-\gamma} \qquad (9)$$

This bound suggests that **the lower bound over the performance of the learned overall policy** depends on three factors:

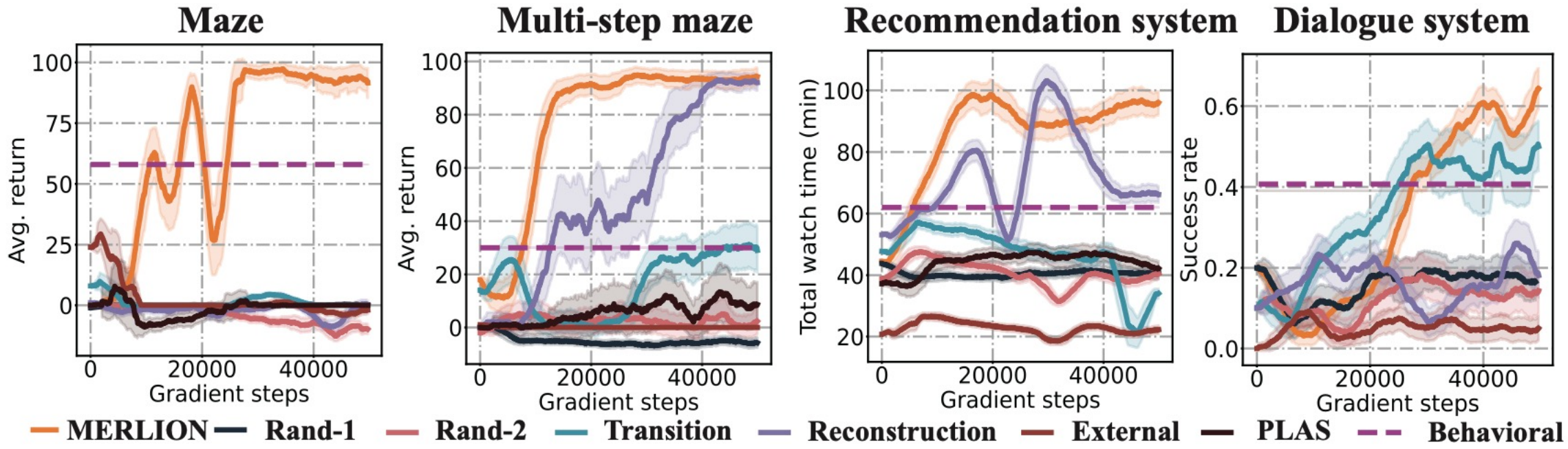
1. The divergence between the learned policy and the behavioral policy $D_{CQL}(\pi_i^*, \overline{\pi}_\beta)(s)$
2. The number of the projected latent actions $|\mathcal{E}|$.
3. The learning error of action encoder ϵ.

Experimental Results



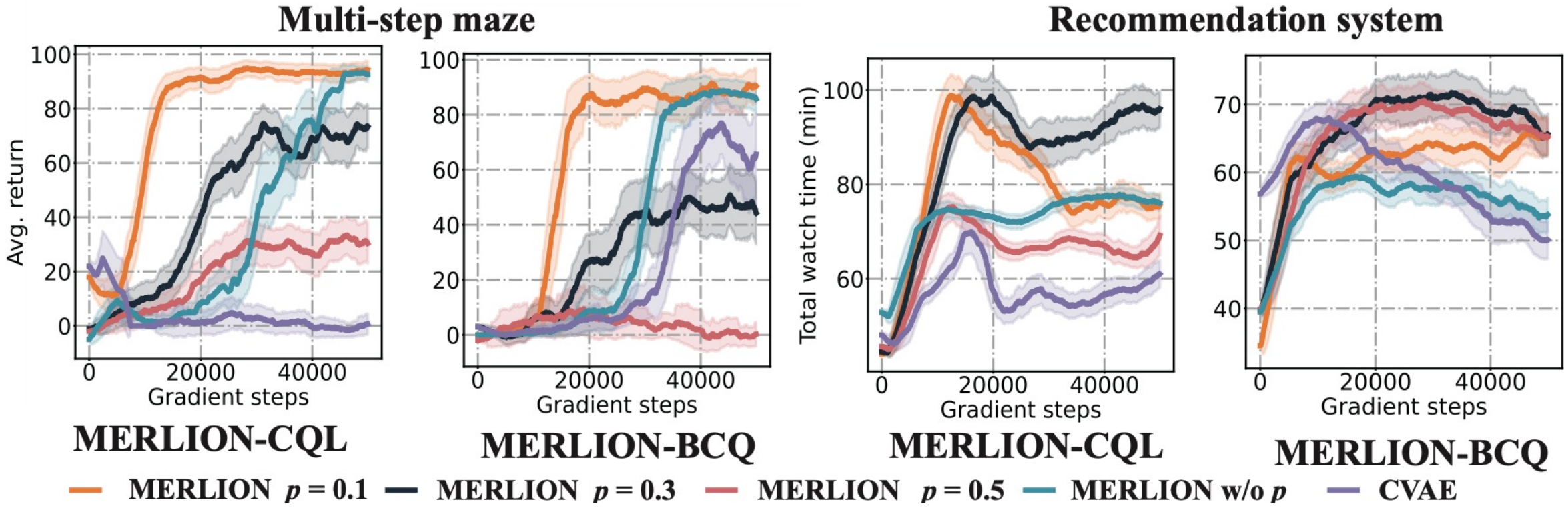
Comparing MERLION equipped with BCQ and CQL against directly training offline RL algorithms (Discrete BCQ, Discrete CQL, and BC) on the original action spaces in 4 environments with large action spaces.

Experimental Results



Comparing the performance of MERLION against other widely used action representations

Ablations



We consider MERLION with **different penalty distances**, removing the penalty distance from the learning objective (**MERLION w/o p**), and removing the distance learning objective from the learning procedure (**CVAE**).

Thanks!