Boosting Offline Reinforcement Learning with Action Preference Query

Qisen Yang [* 1]   Shenzhi Wang [* 1]   Matthieu Gaetan Lin [2]   Shiji Song [1]   Gao Huang [1]

Abstract

Training practical agents usually involve offline and online reinforcement learning (RL) to balance the policy's performance and interaction costs. In particular, online fine-tuning has become a commonly used method to correct the erroneous estimates of out-of-distribution data learned in the offline training phase. However, even limited online interactions can be inaccessible or catastrophic for high-stake scenarios like healthcare and autonomous driving. In this work, we introduce an interaction-free training scheme dubbed Offline-with-Action-Preferences (OAP). The main insight is that, compared to online fine-tuning, querying the preferences between pre-collected and learned actions can be equally or even more helpful to the erroneous estimate problem. By adaptively encouraging or suppressing policy constraint according to action preferences, OAP could distinguish overestimation from beneficial policy improvement and thus attains a more accurate evaluation of unseen data. Theoretically, we prove a lower bound of the behavior policy's performance improvement brought by OAP. Moreover, comprehensive experiments on the D4RL benchmark and state-of-the-art algorithms demonstrate that OAP yields higher (29% on average) scores, especially on challenging AntMaze tasks (98% higher).

1. Introduction

Traditionally, reinforcement learning (RL) algorithms iteratively collect experience by interacting with the environment (Sutton & Barto, 1998). However, such interactions are impractical in many applications, either because online data collection is expensive or dangerous (Yue et al., 2023), *e.g.* in robotics(Singh et al., 2022), educa-

tional agents (Singla et al., 2021), healthcare (Liu et al., 2020), and self-driving (Kiran et al., 2021). Therefore, offline RL, where agents only learn from a pre-collected dataset without any additional online interaction, emerges and thrives (Levine et al., 2020).
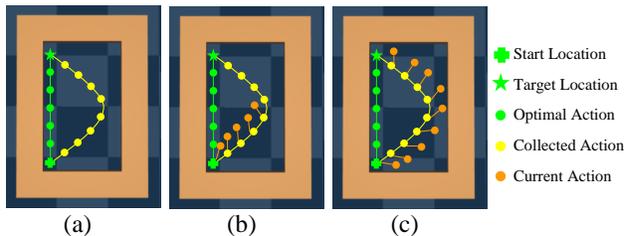
Figure 1. An intuitive example of action preferences. (a) The task is to find the shortest path. (b) If the current policy acts better than the pre-collected action, it shows that the current learning direction is right and should be supported. (c) Otherwise, the OOD data are overestimated, and the distributional shift should be suppressed.

Moreover, learning from a static dataset also makes offline RL suffer from the *distributional shift*, *i.e.*, the gap between state-action distributions of the training data and the test environment. It hinders the agent's performance due to off-policy bootstrapping error accumulation caused by out-of-distribution (OOD) data, *i.e.*, data that is out of the distribution of the offline dataset. This problem is inevitable because there exists a counterfactual inference problem: given data that resulted from a set of decisions, infer the consequence of a different set of decisions (Levine et al., 2020). Hence, offline RL agents are usually fine-tuned by further online training (Nair et al., 2020; Kostrikov et al., 2022), where erroneous estimates of OOD data could be corrected through accurate rewards and real-time transitions. Notably, unlike regular online algorithms, the online fine-tuning phase usually has a limited interaction budget.

Nevertheless, for high-risk scenarios like healthcare and autonomous driving, even a few online interactions with the environment can cause catastrophic losses. Additionally, designing an appropriate reward function takes significant effort for many real-world environments. In this case, developing a safer approach to boosting offline RL without any online interactions is valuable. Supposing that there is a measure where the optimality of given actions can be queried, it may be equally or even more helpful to the erro-

---
*Equal contribution [1]Department of Automation, BNRist, Tsinghua University, Beijing, China [2]Department of Computer Science, BNRist, Tsinghua University, Beijing, China. Correspondence to: Gao Huang <gaohuang@tsinghua.edu.cn>.
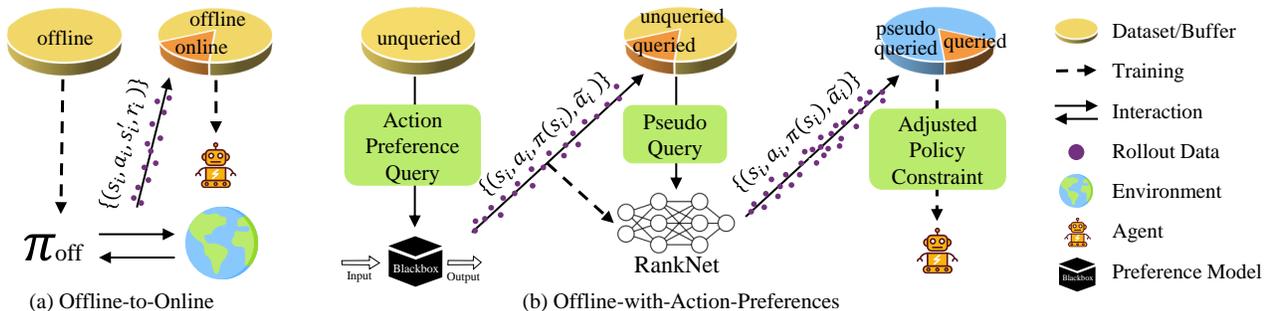
Figure 2. Schematic illustrations of the commonly-used Offline-to-Online and the proposed OAP paradigms. **(a)** The agent learns a policy π_{off} from the pre-collected offline dataset and collects new experiences by interacting with the environment to further fine-tune the policy. **(b)** Some samples in the offline dataset are selected and annotated with action preferences by the proprietary preference model in a blackbox way. Then a RankNet learns from the queried samples and conducts pseudo queries on the rest of the data. All the annotated data are used to train the offline agent with the adjusted updating objective. The entire process does not involve any online interactions.

neous estimate problem if we adaptively support or suppress the agent to learn unseen data in the training process, compared to the online fine-tuning approach. As in Figure 1, the task is to find the optimal path to the target location from the start point. There are pre-collected actions in the static offline dataset. The agent may learn OOD actions because of the policy improvement objective. Intuitively, if OOD actions are better than the collected ones, the current learning direction is encouraged. Conversely, if they are worse than the collected ones, we suppress the divergence and urge the agent to imitate the offline dataset.

This paper proposes a novel query-based offline training scheme, dubbed Offline-with-Action-Preferences (OAP), to achieve the adaptive constraint. Unlike the commonly used interaction-based method, we periodically query preferences between pre-collected and learned actions during offline training and adjust optimization directions according to these action preferences. Instead of high-performing demonstrations, acquiring action preferences are viable in real-world deployment because available expert models are usually proprietary and can only be accessed in an interaction-free and blackbox way (Yu et al., 2020; Chi et al., 2021). Specifically, OAP involves three steps: (1) optionally selecting samples in the offline dataset and querying for preferences, (2) learning the preference pattern using a neural network and pseudo-annotating the rest of the data, (3) training the agent with the adjusted optimization objective. Theoretically, we prove that OAP brings a stable performance improvement of the behavior policy, and even inaccurate preference annotations could help. Empirically, we instantiate OAP with state-of-the-art offline RL algorithms and perform proof-of-concept investigations on the D4RL benchmark (Fu et al., 2020). Surprisingly, compared to online fine-tuning, OAP is safer because online interactions are not required, leading to significantly higher scores, especially on challenging tasks (up to 115% higher).

2. Preliminaries

RL formulation. RL tasks are usually modeled as a Markov decision process (MDP) which can be denoted as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, \rho^0, R, \gamma)$. $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $T(s_{t+1}|s_t, a_t)$ defines the transition function of the environment E, $\rho^0(s_0)$ is the initial state distribution, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the reward function, and $\gamma \in (0, 1]$ is a scalar discount factor. The objective of RL is to learn a policy $\pi(a_t|s_t)$ that maximizes the accumulated rewards.

Offline RL. Offline RL can be seen as a data-driven formulation of the reinforcement learning problem. The agent cannot interact with the environment and only learns from a previously collected dataset $\mathcal{D} = \{(s_i, a_i, s_i', r_i) \mid i = 1, 2, \cdots, N\}$. The distribution over states and actions in $\mathcal{D}$ is denoted as the behavior policy π_β. Finding a balance between increased generalization and avoiding unwanted behaviors outside of distribution is one of the core problems of offline RL (Prudencio et al., 2022). Generally speaking, the optimization objectives of popular offline RL algorithms (Kumar et al., 2020b; Nair et al., 2020; Fujimoto & Gu, 2021; Kostrikov et al., 2022) make trade-offs between policy improvement and policy constraint, either explicitly or implicitly. It can be formulated as follows:

$$\pi^* = \operatorname{argmax}_\pi \mathbb{E}_{(s,a)\sim\mathcal{D}} \, \mathrm{F} \Big(\underbrace{L_{pi}(Q, \pi, s, a)}_{\text{policy improvement term}},$$
$$\underbrace{L_{pc}(Q, \pi, s, a)}_{\text{policy constraint term}}, \underbrace{d_c}_{\text{constraint degree}} \Big), \tag{1}$$

where π is a policy, π^* is the optimal policy, $Q(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a state-action value function estimating the expected sum of discounted rewards after taking action a at state s. (Sutton & Barto, 1998; Levine et al., 2020)

3. Method

The commonly-used Offline-to-Online scheme is based on the pretrain-finetune paradigm, as shown in Figure 2(a). A policy π_{off} is first learned from the offline dataset that stores pre-collected experiences. This process does not involve online interactions and is thus friendly to high-stake applications. Considering the erroneous estimate problem caused by learning from a static dataset, π_{off} is further fine-tuned by a few online interactions with the environment. The second online process can achieve significant performance improvement for the agent but also brings high risk and cost. We aim to propose a new training scheme that exempts the agent from online interactions and meanwhile improves the offline policy π_{off}. As shown in Figure 2(b) and Algorithm 1, our OAP scheme first queries a few samples for action preferences (Section 3.1). Secondly, the rest of the unqueried data are pseudo-queried by a learned RankNet (Section 3.2). Thirdly, these queried and pseudo-queried data are used to train the agent with the adjusted policy constraint (Section 3.3). The three main elements and further theoretical analyses of OAP are detailedly described below.

3.1. Action Preference Query

Offline RL aims to optimize the policy π by an offline dataset $\mathcal{D} = \{(s_i, a_i, s'_i, r_i) \mid i = 1, 2, \cdots, N\}$. As in preference-based RL, an action preference compares two actions for the same state (Wirth et al., 2017). Given a state-action pair $(s_i, a_i) \in \mathcal{D}$ and a preference function G, the action preference query can be formulated as $\tilde{a}_i = G(s_i, a_i, \pi(s_i))$, where π is the current policy and $\tilde{a}_i$ is the preferred action. In real-world applications, the available preference model for queries is usually proprietary, and the preferred action can be annotated in a blackbox way. In this paper, we train an expert policy beforehand and utilize its state-action value function $Q^*(s, a)$ end-to-end to serve as the proprietary model. The preference function is:

$$G(s_i, a_i, \pi(s_i)) = \underset{a \in \{a_i, \pi(s_i)\}}{\arg\max} \ Q^*(s_i, a). \quad (2)$$

For fair comparisons, the number of queries is limited to 100k as the interaction steps in the Offline-to-Online scheme, which is usually accessible and economical in real-world scenarios. Therefore, only the most divergent actions, which may suffer more from the distributional shift problem, are considered worthy of being queried. In other words, millions of samples in the dataset are ranked according to the divergence criterion, and the top ones are selected for action preferences. We simply adopt the Euclidean norm as the *ranking criterion*: $l_i = (\pi(s_i) - a_i)^2$.

3.2. Pseudo Query with RankNet

To take full advantage of query information, all queried samples are collected as a query dataset $\mathcal{D}_q =$

$\{(s_k, a_k, \pi^k(s_k), \tilde{a}_k) \mid k = 1, 2, \cdots, M\}$. Meanwhile, the action preference problem can be viewed as ranking two options under the same state. Considering that practical recommendation systems can learn a ranking function based on a few query pairs, we attempt to obtain pseudo query results by learning from the query dataset $\mathcal{D}_q$. One of the typical Learning-to-Rank approaches, RankNet (Burges et al., 2005), is adopted in our method. It models the underlying ranking function f_r by a neural network with 3 MLP layers. Denote the modeled posterior $P(f_r(s_k, a_k) > f_r(s_k, \pi^k(s_k)))$ by $P_k, k = 1, 2, \cdots, M$, and let $\bar{P}_k$ be the logged target values for those posteriors. Define $o_k = f_r(s_k, a_k) - f_r(s_k, \pi^k(s_k)))$. The pairwise cost function of RankNet is formulated as follows:

$$C_k = C(o_k) = -\bar{P}_k \log P_k - (1 - \bar{P}_k)\log(1 - P_k), \quad (3)$$

where the map from outputs to probabilities is modeled using a logistic function $P_k = e^{o_k}/(1 + e^{o_k})$.

After querying the selected samples and training RankNet by Equation (3), the rest of the samples in the offline dataset are pseudo-queried by RankNet. Then the preference function in this process is changed to:

$$G(s_i, a_i, \pi(s_i)) = \underset{a \in \{a_i, \pi(s_i)\}}{\arg\max} \ f_r(s_i, a). \quad (4)$$

3.3. Adjusted Policy Constraint

Offline RL requires reconciling two conflicting aims (Kostrikov et al., 2022): policy improvement and policy constraint. Generally, a tight constraint would hinder the policy from improving over the behavior policy that collected the offline dataset. A loose constraint may result in a policy that suffers from distributional shift (Levine et al., 2020) and fails on OOD states. Therefore, we aim to achieve a better policy constraint by adaptively deviating from the fixed dataset based on query results. Specifically, when the current policy acts better than the pre-collected action, it is unnecessary to exert a strong constraint. On the contrary, if the action conducted by the current policy is worse than that in the dataset, the policy would remain constrained near the behavior policy. Taking TD3+BC (Fujimoto & Gu, 2021) as an example, the original training objective is:

$$\pi = \underset{\pi}{\arg\max} \, \mathbb{E}_{(s,a)} \Big[\underbrace{\lambda Q(s, \pi(s))}_{\text{policy improvement}} - \underbrace{(\pi(s) - a)^2}_{\text{policy constraint}} \Big], \quad (5)$$

where λ is a hyperparameter and Q is the state-value function. After the action preference is acquired, the policy constraint term is adjusted, and the objective becomes:

$$\pi = \underset{\pi}{\arg\max} \, \mathbb{E}_{(s,a)} \left[\lambda Q(s, \pi(s)) - (\pi(s) - \tilde{a})^2 \right], \quad (6)$$

where $\tilde{a}$ refers to the preferred action in Equation (2) and Equation (4). Consequently, if the updating direction is correct, then the constraint is loosened near the current policy; but if the direction induces worse actions, the constraint reduces to the behavior policy. Such adaptive policy constraints would encourage more aggressive policy improvement and meanwhile filter out wrong moves.

3.4. Theoretical Analysis

In this section, we theoretically validate the superiority of OAP. Considering the blackbox policy that provides action preferences is optimal, Proposition 3.1 suggests that the trained policy is constrained to a better behavior policy. Meanwhile, suppose the preferences are faulty because of the proprietary preference model or the learned RankNet. In that case, Proposition 3.2 demonstrates that OAP can still bring performance improvement.

For any deterministic policy π, its performance (return) can be formulated as $\eta(\pi) = \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{+\infty} \gamma^t r(s_t, a_t)\right]$. Denote the behavior policy of the pre-collected offline dataset $\mathcal{D}$ as π_β, and the behavior policy revised with action preferences as $\tilde{\pi}_\beta$. For any policy π, ρ_π is the (unnormalized) discounted visitation frequency, defined as $\rho_\pi(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s)$, where $s_0 \sim \rho^0(s_0)$ and the trajectory $(s_0, s_1, \ldots)$ is sampled by the policy π. By the definition, $\rho_\pi(s) \in [0, \frac{1}{1-\gamma}]$.

Proposition 3.1 (Perfect preference case). *Consider the case with perfect preferences, i.e., $\forall(s, a)$, the state-action value function $Q^*(s, a)$ used for the action preference query is accurate. Then π_β and $\tilde{\pi}_\beta$ satisfy:*

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \\ \approx \mathbb{E}_{s \sim \mathcal{D}}\left[Q^*(s, \tilde{\pi}_\beta(s)) - Q^*(s, \pi_\beta(s))\right] \geq 0. \quad (7)$$

Proof. The proof is deferred to Appendix A.1. □

For value functions Q_1, Q_2, we define the total variation distance $D_{\mathrm{TV}}^\pi(Q_1, Q_2) = \max_s |Q_1(s, \pi(s)) - Q_2(s, \pi(s))|$.

Proposition 3.2 (Imperfect preference case). *Consider the case where preferences probably have errors. Denote the accurate state-action value function as $Q^*(s, a)$ and the faulty function as $\hat{Q}^*(s, a)$. Then $\forall \hat{Q}^*$ satisfying $D_{\mathrm{TV}}^{\tilde{\pi}_\beta}(\hat{Q}^*, Q^*) \leq \tilde{\alpha}, D_{\mathrm{TV}}^{\pi_\beta}(\hat{Q}^*, Q^*) \leq \alpha$, it holds that*

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \gtrsim \mathbb{E}_{s \sim \mathcal{D}}\Big[\hat{Q}^*(s, \tilde{\pi}_\beta(s)) \\ - \hat{Q}^*(s, \pi_\beta(s))\Big] - 2(\tilde{\alpha} + \alpha)\overline{\rho}_{\pi_\beta}, \quad (8)$$

where $\overline{\rho}_{\pi_\beta} = \sup\{\rho_{\pi_\beta}(s), s \in \mathcal{S}\} \in \left[\frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)}, \frac{1}{1-\gamma}\right]$ ($|\mathcal{S}_\mathcal{D}|$ denotes the number of different states in $\mathcal{D}$).

Proof. Please refer to Appendix A.2. □

The first term in the RHS of Equation (8) is non-negative because $\forall s \in \mathcal{S}, \hat{Q}^*(s, \tilde{\pi}_\beta(s)) - \hat{Q}^*(s, \pi_\beta(s)) \geq 0$ according to Equation (2). The second term $-2(\tilde{\alpha} + \alpha)\overline{\rho}_{\pi_\beta}$ relates to the quality of the blackbox policy. A more accurate blackbox policy would lead to smaller error bounds $(\alpha, \overline{\alpha})$ and thus a larger performance lower bound of $\tilde{\pi}_\beta$. Furthermore, in offline RL, the offline dataset usually contains a large number of samples with various states. Therefore, $\tilde{\rho}_{\pi_\beta}$ would be small and close to its lower bound $\frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)} \ll 1$. It indicates that OAP has a tolerance for faulty annotations, and coarse preference results can improve performance when a variety of offline samples are available.

Algorithm 1 Offline-with-Action-Preferences

Require: Offline dataset $\mathcal{D}$, query dataset $\mathcal{D}_q$, training steps N_{train}, query intervals M_{inter}, query limit K_{total}.
Ensure: Policy π after optimization.
  Initialize policy π and RankNet f_r.
  Let the preferred actions $\tilde{a}_i = a_i, a_i \in \mathcal{D}$.
  for $t = 1 \to N_{\mathrm{train}}$ **do**
    Update the policy π by Equation (6).
    if $t \mod M_{\mathrm{inter}} = 0$ **then**
      Select $\frac{K_{\mathrm{total}} M_{\mathrm{inter}}}{N_{\mathrm{train}}}$ samples from the offline dataset $\mathcal{D}$ according to the ranking criterion.
      Conduct action preference query by Equation (2).
      Add queried samples into the query dataset $\mathcal{D}_q$.
      for epoch $= 0, 1, \cdots$, until convergence **do**
        Train the RankNet f_r with the query dataset $\mathcal{D}_q$ by Equation (3).
      end for
      Conduct pseudo queries on the rest of the samples in the offline dataset $\mathcal{D}$ by Equation (4).
    end if
  end for

---

4. Experiments

We investigate different training schemes on various domains to find a better way of utilizing real-world resources. Firstly, a range of dataset compositions and training schemes are introduced in Section 4.1. Then, we instantiate these schemes on a popular offline RL algorithm and compare their performances in Section 4.2. Finally, the high-performing schemes are instantiated on more state-of-the-art algorithms for generalized conclusions in Section 4.3.

4.1. Setup

Datasets. We consider three different domains of tasks in D4RL (Fu et al., 2020) benchmark: Gym, AntMaze, and Adroit. The Gym-Mujoco tasks include datasets in various environments (*e.g.*, halfcheetah, hopper, and walker) with different qualities (*e.g.*, random, medium, medium-replay, and medium-expert). AntMaze and Adroit tasks are more

Table 1. Comparisons of five training schemes.

Requirement	Offline	Online	Online-Mix	Offline-to-Online	OAP
Pre-collected Offline Data	✓		✓	✓	✓
Training on Offline Data	✓			✓	
Available State Transition		✓	✓	✓	
Predefined Reward Function		✓	✓	✓	
Action Preference Query					✓

Table 2. Average normalized D4RL score (Fu et al., 2020) over the final 10 evaluations and 5 random seeds. Different training schemes are instantiated on the commonly-used TD3+BC (Fujimoto & Gu, 2021) algorithm. OAP is safer than the popular Offline-to-Online scheme but performs significantly better on a variety of tasks. The standard error of AntMaze is usually large since the return is binomial.

Dataset	Offline	Online	Online-Mix	Offline-to-Online	OAP
halfcheetah-random-v2	11.1 ± 1.3	19.2 ± 4.1	28.3 ± 2.1	32.3 ± 1.9	24.0 ± 1.6
hopper-random-v2	8.7 ± 1.6	14.5 ± 12.5	10.1 ± 0.4	8.6 ± 1.2	8.8 ± 1.8
walker2d-random-v2	1.8 ± 1.5	1.0 ± 2.5	2.2 ± 1.5	7.7 ± 8.0	5.1 ± 5.1
halfcheetah-medium-v2	48.1 ± 0.2	19.2 ± 4.1	48.1 ± 0.2	49.2 ± 0.2	56.4 ± 4.3
hopper-medium-v2	55.8 ± 1.2	14.5 ± 12.5	58.4 ± 1.7	57.8 ± 2.0	82.0 ± 6.6
walker2d-medium-v2	83.2 ± 0.5	1.0 ± 2.5	79.2 ± 9.9	85.1 ± 0.9	85.6 ± 1.2
halfcheetah-medium-replay-v2	44.9 ± 0.1	19.2 ± 4.1	46.0 ± 0.4	48.3 ± 0.3	53.4 ± 1.9
hopper-medium-replay-v2	57.2 ± 10.2	14.5 ± 12.5	48.4 ± 3.4	76.3 ± 13.3	98.5 ± 2.5
walker2d-medium-replay-v2	81.1 ± 2.8	1.0 ± 2.5	76.7 ± 14.1	85.6 ± 1.7	84.3 ± 2.7
halfcheetah-medium-expert-v2	85.4 ± 3.3	19.2 ± 4.1	82.2 ± 5.2	94.5 ± 0.6	83.4 ± 5.3
hopper-medium-expert-v2	88.7 ± 2.9	14.5 ± 12.5	97.0 ± 7.6	102.5 ± 4.5	85.9 ± 6.6
walker2d-medium-expert-v2	110.9 ± 0.6	1.0 ± 2.5	110.1 ± 0.8	110.8 ± 0.4	111.1 ± 0.6
Gym Average	**56.4 ± 2.2**	**11.6 ± 6.4**	**57.2 ± 3.9**	**63.2 ± 2.9**	**64.9 ± 3.3**
antmaze-umaze-v0	94.4 ± 2.7	0	0	72.8 ± 36.8	90.4 ± 5.2
antmaze-umaze-diverse-v0	51.0 ± 16.8	0	0	62.5 ± 31.2	75.0 ± 19.0
antmaze-medium-play-v0	1.4 ± 0.8	0	0	0	62.0 ± 10
antmaze-medium-diverse-v0	1.0 ± 1.9	0	0	0.3 ± 0.4	54.5 ± 23.3
antmaze-large-play-v0	0	0	0	0	0
antmaze-large-diverse-v0	0	0	0	0	9.4 ± 8.4
AntMaze Average	**24.6 ± 3.7**	**0**	**0**	**22.6 ± 11.4**	**48.6 ± 11.0**
pen-human-v1	84.8 ± 11.2	4.3 ± 7.4	8.5 ± 10.1	79.1 ± 14.5	101.2 ± 11.5
pen-cloned-v1	56.2 ± 16.3	4.3 ± 7.4	57.2 ± 29.5	66.8 ± 11.1	73.5 ± 13.0
Adroit Average	**70.5 ± 13.7**	**4.3 ± 7.4**	**32.8 ± 19.8**	**72.9 ± 12.8**	**87.4 ± 12.2**
Average	**48.3 ± 3.8**	**7.4 ± 4.6**	**37.6 ± 4.3**	**52.0 ± 6.4**	**62.2 ± 6.5**

challenging, and even online RL algorithms struggle to complete them. The AntMaze domain involves navigation tasks that require an 8-DoF "Ant" quadruped robot to reach a goal location. There are three maze layouts (umaze, medium, large) with different location types (play and diverse). The Adroit datasets are mostly collected by human behavior and aim at controlling a 24-DoF robotic hand.

Schemes. Given that an offline dataset and limited interactions/queries are available, there are five schemes of training an agent, as shown in Table 1. The *Offline* scheme requires pre-collected offline data, and the agent only learns from that fixed dataset. The *Online* scheme trains the agent through real-time interactions with the environment, where the state transition function and reward function are necessary. The *Online-Mix* scheme is similar to the *Online*

one except for adding the offline data into the online replay buffer. The *Offline-to-Online (O2O)* scheme pre-trains the agent in an *Offline* way and fine-tunes it in an *Online-Mix* way. The *Offline-with-Action-Preferences (OAP)* scheme learns a policy from the offline dataset and periodically queries a blackbox model for action preferences. For all the schemes above, we consider online interaction steps or action preference queries limited to 100k, which is usually accessible and acceptable in real-world applications. More implementation details are in Appendix B.

4.2. Comparisons among Schemes

We investigate the different schemes presented above using the popular offline RL algorithm, TD3+BC (Fujimoto & Gu, 2021). Comprehensive experiments are conducted
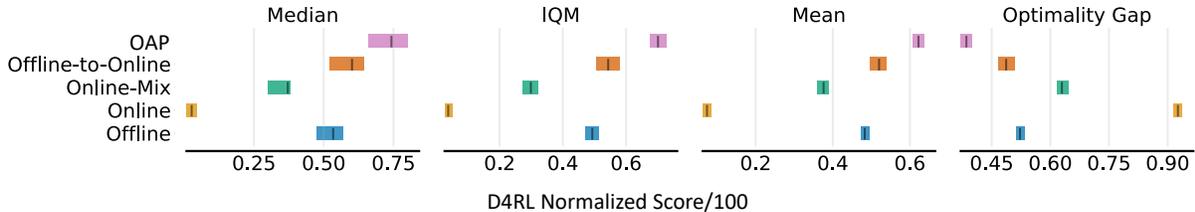
Figure 3. Statistical results of different training schemes instantiated on TD3+BC by rliable (Agarwal et al., 2021) over 5 random seeds.
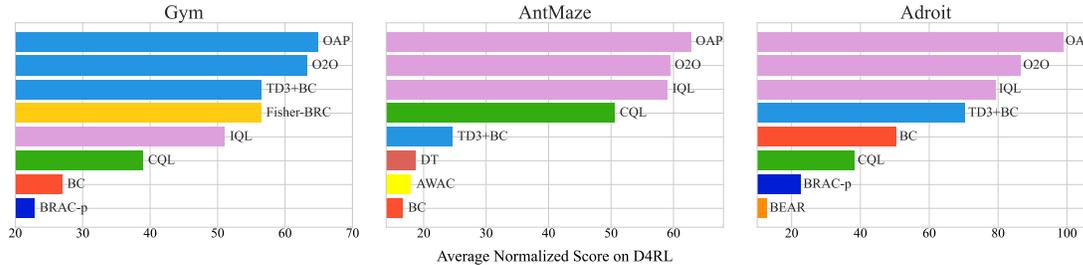
Figure 4. Results of the Offline-to-Online (O2O) and Offline-with-Action-Preferences (OAP) schemes instantiated on SOTA offline RL algorithms. OAP further improves the best-performing algorithms in all three domains. We reproduce IQL and TD3+BC following author-provided implementations, and other offline results are from (Fujimoto & Gu, 2021; Kostrikov et al., 2022).

to evaluate each scheme on various tasks, as shown in Table 2. Firstly, we observe that the Online scheme fails on all tasks because of insufficient data (100k). Secondly, the performances of the Online-Mix scheme are comparable to that of the Offline scheme on Gym tasks but show a tremendous drop in more challenging domains (AntMaze and Adroit). As shown in previous work (Fu et al., 2019; Kumar et al., 2020a; Yue et al., 2022; Anonymous, 2023), the distributional gap between offline data and newly-added online data may harm the training. Thirdly, the Offline-to-Online scheme improves the pre-trained policy on some tasks but still suffers from the distributional gap on some challenging tasks (*e.g.* antmaze-umaze and pen-human). Finally, compared to other schemes, our method OAP dramatically improves upon the Offline baseline on all three domains. In addition, compared to previous work based on the O2O scheme, our method has the following advantages: (1) it does not require real-world interactions nor (2) a well-designed reward function.

Statistical validation. In addition to the point estimates of aggregate performance in Table 2, we present a more rigorous evaluation in Figure 3. These metrics from rliable (Agarwal et al., 2021) increase the results' confidence by accounting for the statistical uncertainty in a handful of runs. Four metrics are considered: median, interquartile mean (IQM), mean, and optimality gap. IQM (also called 25% trimmed mean) and optimality gap are robust alternatives to median and mean respectively. Higher mean, median and IQM scores and lower optimality gap are better. Results in Figure 3 statistically support the conclusions in Table 2 and validate the superiority of our OAP method.

4.3. Results on SOTA Baselines

To validate the generalization of our method, we report results on various D4RL domains(Fu et al., 2020) using state-of-the-art algorithms with Offline-to-Online and OAP schemes. For policy regularization-based methods, we consider BC, BEAR(Kumar et al., 2019), BRAC(Wu et al., 2019), AWAC(Nair et al., 2020), Fisher-BRC(Kostrikov et al., 2021), TD3+BC(Fujimoto & Gu, 2021), and IQL(Kostrikov et al., 2022). For Q-value constraint and sequence modeling methods, we include CQL (Kumar et al., 2020b) and Decision Transformer (DT) (Chen et al., 2021).

We first compare different offline RL algorithms on the Offline scheme and select the one with the highest score. Then, the most powerful algorithm in this domain is equipped with the Offline-to-Online (O2O) or Offline-with-Action-Preferences (OAP) scheme. As shown in Figure 4, the SOTA algorithm on Gym is TD3+BC, and IQL is the best for AntMaze and Adroit. We can observe that the two schemes both improve the performance of the SOTA algorithm. Meanwhile, our OAP scheme brings stable performance gain compared to the O2O scheme on all three domains, especially on the harder AntMaze and Adroit tasks.

5. Discussion

This section presents in-depth investigations of our proposed method. Experiments are based on TD3+BC (Fujimoto & Gu, 2021) and the benchmark is D4RL (Fu et al., 2020). In particular, we investigate OAP's four main components: the blackbox policy, the adjusted policy constraint, the periodi-
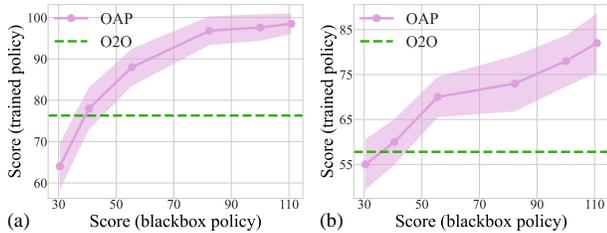
Figure 5. OAP queries a blackbox policy (π^*) for action preferences. Blackbox policies of various quality are compared in this figure (a. hopper-medium-replay, b. hopper-medium). Results validate that even sub-optimal π^* can lead to significant performance improvement, which makes OAP more practical in the real world.

cal queries, and the RankNet. Hence, analytical experiments are designed around the following four questions.

5.1. Can OAP work with faulty annotations?

We denote the policy that provides action preferences as the *blackbox policy* (π^*) and the one that is trained with OAP as the *trained policy* (π). The policy that achieves a full normalized D4RL score (≥ 100) can be considered expert quality. In real-world applications, the proprietary preference model usually performs at the expert quality, and it's feasible to adopt it as π^*. Meanwhile, considering faulty annotations and how OAP works correspondingly are equally valuable since security and stability are always highly prioritized in practice. As shown in Figure 5, policies of different quality (*i.e.*, high or low D4RL score) are adopted as π^* in, OAP and the performance of trained policies are compared. Taking the gym tasks where OAP works most effectively as examples, we can observe that OAP brings more considerable performance improvement to the trained policy than O2O even with poor-performing balckbox policy (score $<$ 50). These results suggest that OAP has a high tolerance for faulty annotations and works well with sub-optimal π^*, which coincides with our theoretical analyses in Section 3.4.

5.2. Do action preferences help policy constraint?

Section 3.3 shows that OAP can adaptively constrain the policy on either actions given by the dataset or by the learned policy. In particular, given a sample (s_t, a_t) from the offline dataset, the preference model selects between a_t and the learned action $\pi(s_t)$. If a_t is preferred, we constrain the policy on the offline dataset. Conversely, if $\pi(s_t)$ is preferred, the constraint is loosened to the trained policy's vicinity.

We adopt *action divergence* and *value gain* to measure the quality of policy constraint. For a well-trained policy π and a pair (s_t, a_t) in the offline dataset, the action divergence calculates the distance of $\pi(s_t)$ and a_t for the whole dataset. Given an accurate value function $Q^*(s, a)$, the value gain is $Q^*(s_t, \pi(s_t)) - Q^*(s_t, a_t)$, representing how many values

the policy gains by conducting $\pi(s_t)$ instead of a_t. A large action divergence shows that the policy takes an aggressive move, and a large value gain means this move is worthy.

In Figure 6, the Offline, O2O, and OAP schemes are compared, and there are three observations. Firstly, the policy trained from OAP has the largest action divergence, and that from the Offline scheme has the smallest, which means OAP learns more OOD actions. Secondly, compared to the Offline scheme, O2O learns a more aggressive policy (*i.e.*, more divergent actions) but meanwhile suffers from the overestimation problem (*i.e.*, more negative value gains). Thirdly, the most divergent actions in OAP usually have large positive value gains, which means the policy is beneficially aggressive. Hence, it is validated that OAP facilitates more adaptive policy constraints by encouraging high-value divergences and restraining harmful ones.

5.3. What if queries were focused instead of spaced?

O2O allows interactions with the environment after training on the offline dataset, which is a typical procedure of pretrain-finetune. Contrastively, OAP spaces the queries throughout the training process because we assume that timely corrections of policy constraints are better than changing a convergent policy. Hence, we investigate whether spacing the queries matters or not and whether O2O also benefits from spaced interactions or not.

We denote the pretrain-finetune procedure as *FT* and the periodical interactions/queries as *Interval*. In Figure 7, *O2O* means interacting with the environment, and *OAP* means querying for action preferences. O2O-FT and OAP-Interval correspond to the original O2O and OAP. It is noted that all four methods in Figure 7 conform to the 100k limitations of interactions/queries. Results show that O2O-FT and O2O-Interval generally have comparable performances in three domains. OAP-Interval performs similarly to OAP-FT on Gym but is much better on harder AntMaze and Adroit. Therefore, we suggest that spaced queries are necessary for OAP, but periodical interactions cannot benefit O2O.

5.4. Is there a necessity for RankNet?

During training, since the 100k queries only cover a small set of samples, it is natural to assume that its effect on the policy constraint is limited. As described in Section 3.2, RankNet extends the number of queries by pseudo-annotating the unqueried samples. Ideally, we would like RankNet to perfectly learn to rank two actions and provide expert preferences. Specifically, we compare two variants of

---

[3]HC = Halfcheetah, Hop = Hopper, W = Walker, r = random, m = medium, mr = medium-replay, me = medium-expert; AM = antmaze, u = umaze, m = medium, l = large, d = diverse, p = play, h = human, c = cloned.
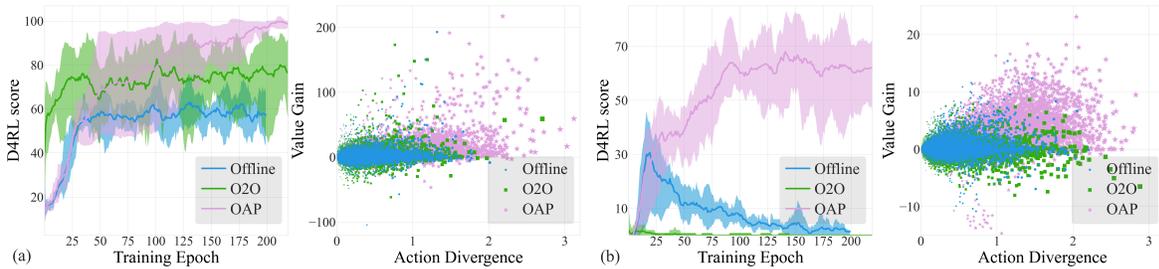
Figure 6. The training curve (Training Epoch *vs.* D4RL Score) and policy constraint quality (Action Divergence *vs.* Value Gain) of two typical tasks in Gym (a. hopper-medium-replay) and AntMaze (b. antmaze-medium-play) domains. OAP performs better than O2O because its out-of-distribution data (*i.e.,* data with large action divergences) usually have positive and high value gains.
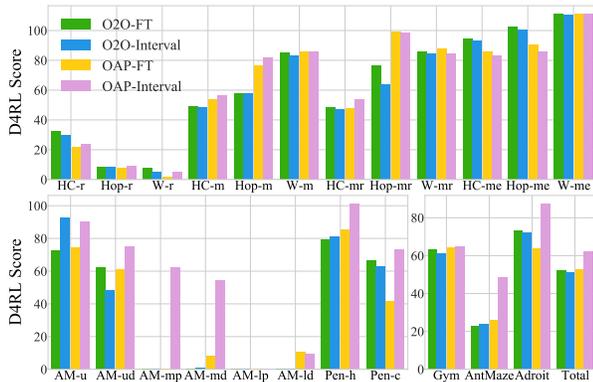
Figure 7. Normalized D4RL score of O2O, OAP, and variants on Gym, AntMaze, and Adroit tasks. [3] *FT* means pre-training on offline data and fine-tuning with interactions/queries. *Interval* means spreading interactions/queries throughout the training process.

Table 3. Average normalized D4RL score (Fu et al., 2020) over the final 10 evaluations and 5 random seeds. OAP(inf) means OAP with unlimited times of queries. OAP(w/RN) means OAP without pseudo queries from RankNet but with limited querying times.

Dataset	OAP	OAP (inf)	OAP (w/ RN)
HC-r	24.0 ± 1.6	22.0 ± 1.4	11.0 ± 1.1
Hop-r	8.8 ± 1.8	13.2 ± 7.4	8.1 ± 0.8
W-r	5.1 ± 5.1	2.3 ± 2.0	1.9 ± 0.9
HC-m	56.4 ± 4.3	59.2 ± 1.5	48.2 ± 0.3
Hop-m	82.0 ± 6.6	92.8 ± 4.1	45.9 ± 1.5
W-m	85.6 ± 1.2	86.6 ± 0.3	84.8 ± 2.4
HC-mr	53.4 ± 1.9	51.3 ± 0.6	28.4 ± 3.1
Hop-mr	98.5 ± 2.5	101.9 ± 2.0	31.6 ± 3.5
W-mr	84.3 ± 2.7	84.9 ± 9.9	74.8 ± 5.5
HC-me	83.4 ± 5.3	84.1 ± 3.2	84.3 ± 5.3
Hop-me	85.9 ± 6.6	92.2 ± 8.2	80.6 ± 2.6
W-me	111.1 ± 0.6	109.4 ± 1.5	111.0 ± 0.4
Avg.	$\mathbf{64.9} \pm 3.3$	$\mathbf{66.7} \pm 3.5$	$\mathbf{50.9} \pm 2.3$

OAP to validate the effectiveness of RankNet. OAP(inf) allows action preference queries for each sample in the offline dataset instead of the 100k limitation in the normal OAP. OAP(w/RN) maintains the 100k limitation of queries, and the unqueried samples cannot have pseudo annotations from RankNet. As shown in Table 3, OAP(inf) performs slightly better than OAP, while the performance of OAP(w/RN) has a significant drop. The former result suggests that RankNet successfully learns the pattern of an expert's action preferences in most cases. The latter result shows that the 100k queries are not enough, and RankNet is necessary for OAP.

6. Related Work

Offline RL. As aforementioned, offline RL suffers from *distributional shift* caused by the gap between pre-training and OOD data. Existing methods generally constrain or regularize the learned policy to limit the deviation from the behavior policy. This may be implemented by an explicit density model (Wu et al., 2019; Fujimoto et al., 2019; Kumar et al., 2019; Ghasemipour et al., 2021), implicit divergence constraints (Peters & Schaal, 2007; Peng et al., 2019; Nair et al., 2020; Wang et al., 2020; Kostrikov et al.,

2022), pessimistic estimations of state-action values (Kumar et al., 2020b; Kostrikov et al., 2021), or directly adding a behavior cloning term to the policy improvement loss (Nair et al., 2018; Booher, 2019; Fujimoto & Gu, 2021). In the same way that powerful computer vision and NLP models are often pre-trained on large, general-purpose datasets and then fine-tuned on task-specific data, practical instantiations of RL for real-world applications usually involve pre-training on the offline dataset and then fine-tuning with online interactions (Nair et al., 2020; Kostrikov et al., 2022).

Preference Learning. Roughly speaking, preference learning involves inducing predictive preference models from empirical data (Wirth et al., 2017). A preference learning task consists of some set of items for which preferences are known, and the task is to learn a function that predicts preferences for a new set of items, or the same set of items in a different context (Fürnkranz & Hüllermeier, 2010). One of the most extensively studied tasks in preference learning is learning to rank (LTR). The commonly used LTR algorithms are mainly pointwise (Li et al., 2007), pairwise (Burges et al., 2005), and listwise (Burges et al., 2006; Burges, 2010). The pointwise and pairwise approaches treat relevance degree as real values or categories, while the pairwise methods reduce ranking to classifying the order between each pair.

In preference-based RL (PbRL), the main problem is learning a policy using preferences between states, actions, or trajectories without any numeric reward signal. It replaces reward values in online RL by preferences to better elicit human opinions on the target objective, especially when numerical reward values are hard to design or interpret. The process usually involves two actors: an agent that acts according to a given policy and an expert evaluating its behavior (Wirth et al., 2017) , which is similar to OAP. On the other hand, unlike PbRL algorithms, our method only uses a small amount of preference information and learns from a static offline dataset. Preferences are used to help agents distinguish good or bad policy improvement rather than directly taken as training feedback or reformulated as reward functions. As far as we know, OAP takes the first step to utilize preference learning for boosting offline RL and achieving more extensive success than online fine-tuning methods.

7. Conclusion

We present Offline-with-Action-Preferences (OAP), a general training scheme for practical offline RL that brings higher returns than online fine-tuning but dispenses with the risk of real-world interactions. To our knowledge, OAP is the first method that avoids the challenges of online fine-tuning and meanwhile achieves better performance than previous methods that leverage online fine-tuning. This has a number of significant benefits since online fine-tuning is usually essential for offline agents in real-world applications. Firstly, our method is more practical for high-stake scenarios like healthcare and self-driving because interactions with the environment are not required. Secondly, we don't require well-defined reward signals as online feedback, which can be hard to design in many tasks. Thirdly, OAP exploits limited queries efficiently to ensure it is economical and accessible. Finally, besides the safety and efficiency of this method, we show that it attains superior performance across D4RL tasks, either compared to other high-risk training schemes or instantiated on more state-of-the-art algorithms. This work is a step towards our larger vision of more practical RL, where the key is to address the uncertainty and risk of online interactions. We hope it can inspire future research, such as offline RL with weak action preferences, intricate policy constraints, state/trajectory preferences, revised Q-learning, and adaptive uncertainty estimation.

Acknowledgments

References

Agarap, A. F. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 34:29304–29320, 2021.

Anonymous. Actor-critic alignment for offline-to-online reinforcement learning. In *Submitted to International Conference on Learning Representations*, 2023.

Booher, J. Bc + rl : Imitation learning from non-optimal demonstrations. 2019.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. Learning to rank using gradient descent. In *International Conference on Machine Learning*, pp. 89–96, 2005.

Burges, C., Ragno, R., and Le, Q. Learning to rank with non-smooth cost functions. *Advances in Neural Information Processing Systems*, 19, 2006.

Burges, C. J. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81, 2010.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021.

Chi, H., Liu, F., Yang, W., Lan, L., Liu, T., Han, B., Cheung, W., and Kwok, J. Tohan: A one-step approach towards few-shot hypothesis adaptation. *Advances in Neural Information Processing Systems*, 34:20970–20982, 2021.

Fu, J., Kumar, A., Soh, M., and Levine, S. Diagnosing bottlenecks in deep q-learning algorithms. In *International Conference on Machine Learning*, pp. 2021–2030. PMLR, 2019.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4RL: datasets for deep data-driven reinforcement learning. *CoRR*, abs/2004.07219, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Fürnkranz, J. and Hüllermeier, E. (eds.). *Preference Learning*. Springer, 2010.

Ghasemipour, S. K. S., Schuurmans, D., and Gu, S. S. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pp. 3682–3691. PMLR, 2021.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018.

Kakade, S. and Langford, J. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*. Citeseer, 2002.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., and Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23 (6):4909–4926, 2021.

Kostrikov, I., Fergus, R., Tompson, J., and Nachum, O. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Kumar, A., Gupta, A., and Levine, S. Discor: Corrective feedback in reinforcement learning via distribution correction. *Advances in Neural Information Processing Systems*, 33:18560–18572, 2020a.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020b.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, abs/2005.01643, 2020.

Li, P., Wu, Q., and Burges, C. Mcrank: Learning to rank using multiple classification and gradient boosting. *Advances in Neural Information Processing Systems*, 20, 2007.

Liu, S., See, K. C., Ngiam, K. Y., Celi, L. A., Sun, X., Feng, M., et al. Reinforcement learning for clinical decision support in critical care: comprehensive review. *Journal of Medical Internet Research*, 22(7):e18477, 2020.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Overcoming exploration in reinforcement learning with demonstrations. In *International Conference on Robotics and Automation*, 2018.

Nair, A., Dalal, M., Gupta, A., and Levine, S. Accelerating online reinforcement learning with offline datasets. *CoRR*, abs/2006.09359, 2020.

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *CoRR*, abs/1910.00177, 2019.

Peters, J. and Schaal, S. Reinforcement learning by reward-weighted regression for operational space control. In *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pp. 745–750. ACM, 2007.

Prudencio, R. F., Máximo, M. R. O. A., and Colombini, E. L. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *CoRR*, abs/2203.01387, 2022.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897. PMLR, 2015.

Singh, B., Kumar, R., and Singh, V. P. Reinforcement learning in robotic applications: a comprehensive survey. *Artif. Intell. Rev.*, 55(2):945–990, 2022.

Singla, A., Rafferty, A. N., Radanovic, G., and Heffernan, N. T. Reinforcement learning for education: Opportunities and challenges. *CoRR*, abs/2107.08828, 2021.

Sutton, R. S. and Barto, A. G. Introduction to reinforcement learning, 1998.

Wang, Z., Novikov, A., Zolna, K., Merel, J. S., Springenberg, J. T., Reed, S. E., Shahriari, B., Siegel, N., Gulcehre, C., Heess, N., et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33: 7768–7778, 2020.

Wilmer, E., Levin, D. A., and Peres, Y. Markov chains and mixing times. *American Mathematical Soc., Providence*, 2009.

Wirth, C., Akrour, R., Neumann, G., Fürnkranz, J., et al. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136): 1–46, 2017.

Wu, Y., Tucker, G., and Nachum, O. Behavior regularized offline reinforcement learning. *CoRR*, abs/1911.11361, 2019.

Yu, Y., Min, X., Zhao, S., Mei, J., Wang, F., Li, D., Ng, K., and Li, S. Dynamic knowledge distillation for black-box hypothesis transfer learning. *CoRR*, abs/2007.12355, 2020.

Yue, Y., Kang, B., Ma, X., Xu, Z., Huang, G., and YAN, S. Boosting offline reinforcement learning via data rebalancing. *Advances in Neural Information Processing Systems, Workshop*, 2022.

Yue, Y., Kang, B., Xu, Z., Huang, G., and Yan, S. Value-consistent representation learning for data-efficient reinforcement learning. *Association for the Advancement of Artificial Intelligence*, 2023.

A. Theoretical Proofs

A.1. Proof of Proposition 3.1

We first start with a lemma considering the policy improvement as follows:

Lemma A.1. *Given any two policy π_1 and π_2,*

$$\eta(\pi_1) - \eta(\pi_2) = \int_{s \in \mathcal{S}} \rho_{\pi_1}(s)(Q_{\pi_2}(s, \pi_1(s)) - V_{\pi_2}(s)) \, \mathrm{d}s \tag{9}$$

Proof. The derivation of this lemma is related to Schulman et al. (2015) and Kakade & Langford (2002).

According to Kakade & Langford (2002),

$$\eta(\pi_1) = \eta(\pi_2) + \mathbb{E}_{\tau \sim \pi_1} \left[\sum_{t=0}^{\infty} \gamma^t Q_{\pi_2}(s_t, a_t) - V_{\pi_2}(s_t) \right]. \tag{10}$$

Here, $\mathbb{E}_{\tau \sim \pi_1}$ denotes sampling trajectories with the policy π_1.

It follows that

$$\eta(\pi_1) = \eta(\pi_2) + \sum_{t=0}^{\infty} \int_{s \in \mathcal{S}} P(s_t = s | \pi_1) \gamma^t (Q_{\pi_2}(s, \pi_1(s)) - V_{\pi_2}(s)) \, \mathrm{d}s \tag{11}$$

$$= \eta(\pi_2) + \int_{s \in \mathcal{S}} \sum_{t=0}^{\infty} \gamma^t P(s_t = s | \pi_1) (Q_{\pi_2}(s, \pi_1(s)) - V_{\pi_2}(s)) \, \mathrm{d}s \tag{12}$$

$$= \eta(\pi_2) + \int_{s \in \mathcal{S}} \rho_{\pi_1}(s)(Q_{\pi_2}(s, \pi_1(s)) - V_{\pi_2}(s)) \, \mathrm{d}s. \tag{13}$$

Therefore Lemma A.1 is proven. $\qquad \square$

According to Lemma A.1, it follows that

$$\eta(\tilde{\pi}_\beta) - \eta(\pi^*) = \int_{s \in \mathcal{S}} \rho_{\tilde{\pi}_\beta}(s)(Q^*(s, \tilde{\pi}_\beta(s)) - V^*(s)) \, \mathrm{d}s \tag{14}$$

$$\eta(\pi_\beta) - \eta(\pi^*) = \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)(Q^*(s, \pi_\beta(s)) - V^*(s)) \, \mathrm{d}s \tag{15}$$

Combining Equation (14) and Equation (15), we can infer that $\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta)$ satisfies:

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) = (\eta(\tilde{\pi}_\beta) - \eta(\pi^*)) - (\eta(\pi_\beta) - \eta(\pi^*)) \tag{16}$$

$$= \int_{s \in \mathcal{S}} \rho_{\tilde{\pi}_\beta}(s)(Q^*(s, \tilde{\pi}_\beta(s)) - V^*(s)) \, \mathrm{d}s - \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)(Q^*(s, \pi_\beta(s)) - V^*(s)) \, \mathrm{d}s \tag{17}$$

$$\approx \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)(Q^*(s, \tilde{\pi}_\beta(s)) - Q^*(s, \pi_\beta(s))) \, \mathrm{d}s \tag{18}$$

The derivation from Equation (17) to Equation (18) is because there are only a few queries compared with the huge number of offline data, it follows $\rho_{\tilde{\pi}_\beta}(s) \approx \rho_{\pi_\beta}(s)$.

Based on Equation (2), it holds that $\forall s \in \mathcal{S}, Q^*(s, \tilde{\pi}_\beta(s)) \geq Q^*(s, \pi_\beta(s))$. Considering $\forall s \in \mathcal{S}, \rho_{\pi_\beta}(s) \geq 0$, it follows that

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \approx \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)(Q^*(s, \tilde{\pi}_\beta(s)) - Q^*(s, \pi_\beta(s))) \, \mathrm{d}s \geq 0. \tag{19}$$

Noting that $\int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s) \cdot \mathrm{d}s$ is equivalent to $\mathbb{E}_{s \sim \mathcal{D}}[\cdot]$, Proposition 3.1 follows. **Q.E.D.**

12

A.2. Proof of Proposition 3.2

For the imperfect query case, we denote that

$$Q^*(s,a) = \hat{Q}^*(s,a) + \delta(s,a), \tag{20}$$

where $\delta(s,a)$ is the approximation error of the estimated state-value function $\hat{Q}^*(s,a)$.

Combining Equation (18) and Equation (20), it follows that

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \approx \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)(Q^*(s,\tilde{\pi}_\beta(s)) - Q^*(s,\pi_\beta(s)))\,\mathrm{d}s \tag{21}$$

$$= \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)\left[(\hat{Q}^*(s,\tilde{\pi}_\beta(s)) + \delta(s,\tilde{\pi}_\beta(s))) - (\hat{Q}^*(s,\pi_\beta(s)) + \delta(s,\pi_\beta(s)))\right]\mathrm{d}s \tag{22}$$

$$= \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right]\mathrm{d}s - \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)[\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))]\,\mathrm{d}s. \tag{23}$$

Considering the condition in Proposition 3.2 that $D_{\mathrm{TV}}^{\tilde{\pi}_\beta}(\hat{Q}^*, Q^*) \leq \tilde{\alpha}, D_{\mathrm{TV}}^{\pi_\beta}(\hat{Q}^*, Q^*) \leq \alpha$, we have

$$\forall s \in \mathcal{S}, \quad |\delta(s,\tilde{\pi}_\beta(s))| = \left|Q^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\tilde{\pi}_\beta(s))\right| \leq D_{\mathrm{TV}}^{\tilde{\pi}_\beta}(\hat{Q}^*, Q^*) \leq \tilde{\alpha}, \tag{24}$$

$$\forall s \in \mathcal{S}, \quad |\delta(s,\pi_\beta(s))| = \left|Q^*(s,\pi_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right| \leq D_{\mathrm{TV}}^{\pi_\beta}(\hat{Q}^*, Q^*) \leq \alpha. \tag{25}$$

Therefore,

$$\forall s \in \mathcal{S}, \quad |\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))| \leq |\delta(s,\tilde{\pi}_\beta(s))| + |\delta(s,\pi_\beta(s))| = \tilde{\alpha} + \alpha. \tag{26}$$

To prepare for further derivation, We need to introduce Lemma A.2.

Lemma A.2 (Wilmer et al. (2009)). *Suppose that μ and ν are two probability distributions on $\mathcal{X}$, then*

$$\max_{x \in \mathcal{X}} |\mu(x) - \nu(x)| = \frac{1}{2}\sum_{x \in \mathcal{X}} |\mu(x) - \nu(x)|. \tag{27}$$

Proof. Please refer to Proposition 4.2 in Wilmer et al. (2009). $\square$

Denote $\overline{\rho}_{\pi_\beta} = \sup\{\rho_{\pi_\beta}(s), s \in \mathcal{S}\}$. Based on Equation (23), it follows

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \approx \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right]\mathrm{d}s - \int_{s \in \mathcal{S}} \rho_{\pi_\beta}(s)[\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))]\,\mathrm{d}s \tag{28}$$

$$= \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - \overline{\rho}_{\pi_\beta} \int_{s \in \mathcal{S}} \frac{\rho_{\pi_\beta}(s)}{\overline{\rho}_{\pi_\beta}}[\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))]\,\mathrm{d}s \tag{29}$$

$$\geq \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - \overline{\rho}_{\pi_\beta} \int_{s \in \mathcal{S}} \left|\frac{\rho_{\pi_\beta}(s)}{\overline{\rho}_{\pi_\beta}}[\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))]\right|\mathrm{d}s. \tag{30}$$

Due to $\forall s \in \mathcal{S}, \frac{\rho_{\pi_\beta}(s)}{\overline{\rho}_{\pi_\beta}} \in [0,1]$,

$$\eta(\tilde{\pi}_\beta) - \eta(\pi_\beta) \gtrsim \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - \overline{\rho}_{\pi_\beta} \int_{s \in \mathcal{S}} \left|\frac{\rho_{\pi_\beta}(s)}{\overline{\rho}_{\pi_\beta}}[\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))]\right|\mathrm{d}s \tag{31}$$

$$\geq \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - \overline{\rho}_{\pi_\beta} \int_{s \in \mathcal{S}} |\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))|\,\mathrm{d}s \tag{32}$$

$$= \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - 2\overline{\rho}_{\pi_\beta} \max_{s \in \mathcal{S}} |\delta(s,\tilde{\pi}_\beta(s)) - \delta(s,\pi_\beta(s))| \quad \text{(Lemma A.2)} \tag{33}$$

$$\geq \mathbb{E}_{s \sim \mathcal{D}}\left[\hat{Q}^*(s,\tilde{\pi}_\beta(s)) - \hat{Q}^*(s,\pi_\beta(s))\right] - 2(\tilde{\alpha} + \alpha)\overline{\rho}_{\pi_\beta}. \quad \text{(Equation (26))} \tag{34}$$

Last, we consider the value range of $\overline{\rho}_{\pi_\beta}$. Considering an extreme case that π_β gets stuck in some state s', *i.e.* $\forall t, P(s_t = s') = 1$, then $\overline{\rho}_{\pi_\beta} = \sum_{t=0}^{\infty} \gamma^t = \frac{1}{1-\gamma}$. This is obviously the upper bound of $\overline{\rho}_{\pi_\beta}$. On the other hand, considering another extreme case that π_β visits all the states in the offline dataset $\mathcal{D}$ equiprobably, *i.e.* $\forall t, \forall s' \in \mathcal{D}, P(s_t = s') = \frac{1}{|\mathcal{S}_\mathcal{D}|}$, then $\forall s \in \mathcal{S}_\mathcal{D}, \rho_{\pi_\beta}(s) = \sum_{t=0}^{\infty} \gamma^t \frac{1}{|\mathcal{S}_\mathcal{D}|} = \frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)}$. It is impossible for any $\overline{\rho}_{\pi_\beta}$ to be less than $\frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)}$ (otherwise $\int_{s \in \mathcal{S}_\mathcal{D}} \overline{\rho}_{\pi_\beta}\, ds < \int_{s \in \mathcal{S}_\mathcal{D}} \frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)}\, ds = \frac{1}{1-\gamma} = \int_{s \in \mathcal{S}_\mathcal{D}} \rho_{\pi_\beta}(s)\, ds$, which contradicts $\forall s, \overline{\rho}_{\pi_\beta} \geq \rho_{\pi_\beta}(s)$). Therefore, $\overline{\rho}_{\pi_\beta} \in \left[\frac{1}{|\mathcal{S}_\mathcal{D}|(1-\gamma)}, \frac{1}{1-\gamma} \right]$.

Thus, Proposition 3.2 is proven. **Q.E.D.**

B. Implementation Details

Blackbox Policy. To play the role of proprietary preference models in real-world deployment, blackbox policies that provide action preferences are pre-trained using the SOTA algorithms and unlimited training resources in this paper. For Gym and Adroit tasks, we adopt the online training scheme and train the SAC (Haarnoja et al., 2018) algorithm for 3Mil steps, following D4RL (Fu et al., 2020) and the rlkit[4] repository. For AntMaze tasks, online algorithms fail, and we adopt the Offline-to-Online training scheme with the IQL (Kostrikov et al., 2022) method, following its author-provided implementations (*i.e.*, 1Mil steps for offline pre-training and 1Mil steps for online fine-tuning). Table 4 shows the performance of well-trained blackbox policies over 100 random rollouts.

Table 4. The training schemes and normalized D4RL scores of adopted blackbox policies.

Dataset	HC	Hop	W	Pen	AM-u	AM-ud	AM-ml	AM-md	AM-lp	AM-ld
Training Scheme		Online					Offline-to-Online			
Normalized D4RL Score	118	110	105	125	91	78	85	83	47	34

Trained Policy. We train the policy for 1Mil steps in the Offline, Online, and Online-Mix schemes, and additional 100k steps in the Offline-to-Online and OAP schemes. Other technical details of the Offline-to-Online scheme follow the AWAC and IQL (Nair et al., 2020; Kostrikov et al., 2022) methods. Based on the three parts in Section 3, the pseudo-code describing the entire training process of OAP is presented in Algorithm 1. The hyperparameters of OAP instantiated on TD3+BC (Fujimoto & Gu, 2021) and IQL (Kostrikov et al., 2022) are presented in Table 5.

C. Potential Negative Societal Impact

RL agents may take suboptimal or even unreasonable actions during the trial-and-error training process. Meanwhile, online interactions with the environment can be high-risk and high-cost in real-world applications, such as autonomous driving and medical treatment. Hence, offline RL provides a more feasible solution than online RL by leveraging the offline logged data to dispense with online interactions during the training phase. However, a limitation of offline RL is that agents' performances are primarily affected by the quality and quantity of previously collected data. Moreover, this may include potentially damaging applications such as biased datasets and biased agents. For the proposed Offline-with-Action-Preferences (OAP) method, preference learning is involved in offline reinforcement learning. We foresee the impact of our work is probably to help explore user-adaptive RL agents. However, this characteristic may facilitate harmful applications like biased agents at the same time. Therefore, we advocate that RL-based robotics systems, game AI and other applications should follow fair and safe principles.

---

[4] https://github.com/rail-berkeley/rlkit, commit ID c81509d982b4d52a6239e7bfe7d2540e3d3cd986.

Table 5. Hyperparameters of OAP instantiated on the TD3+BC (Fujimoto & Gu, 2021) and IQL (Kostrikov et al., 2022) algorithms on Gym/ AntMaze/ Adroit domains. "Unqueried First" means selecting unqueried samples preferentially.

	Hyperparameter	Value
TD3 Hyperparameters	Optimizer	Adam (Kingma & Ba, 2015)
	Critic learning rate	3e-4
	Actor learning rate	3e-4
	Mini-batch size	256
	Discount factor	0.99
	Target update rate	5e-3
	Policy noise	0.2
	Policy noise clipping	(-0.5, 0.5)
	Policy update frequency	2
TD3 Architecture	Critic hidden dim	256
	Critic hidden layers	2
	Critic activation function	ReLU (Agarap, 2018)
	Actor hidden dim	256
	Actor hidden layers	2
	Actor activation function	ReLU (Agarap, 2018)
TD3+BC Hyperparameters	λ	2.5 / 2.5 / 0.1
	State normalization	True
IQL Hyperparameters	Optimizer	Adam (Kingma & Ba, 2015)
	Policy learning rate	3e-4
	Mini-batch size	256
	Dropout rate	0.0 / 0.0 / 0.1
	Beta	3 / 10 / 0.5
	Quantile	0.7 / 0.9 / 0.7
OAP Hyperparameters	Training steps	1e6
	Query limit	1e5
	Periodical steps	1e5
	Unqueried First	True
	L2R training epochs	100
RankNet Architecture	Hidden dim	512, 256
	Hidden layers	2
	Dropout Rate	0.5
	Activation function	ReLU (Agarap, 2018)