Curriculum Offline Reinforcement Learning

Yuanying Cai
Tsinghua University
Beijing, China
cai-yy16@mails.tsinghua.edu.cn

Chuheng Zhang*
Microsoft Research Asia
Beijing, China
zhangchuheng123@live.com

Hanye Zhao
Shanghai Jiao Tong University
Shanghai, China
fineartz@sjtu.edu.cn

Li Zhao
Microsoft Research Asia
Beijing, China
lizo@microsoft.com

Jiang Bian
Microsoft Research Asia
Beijing, China
jiang.bian@microsoft.com

ABSTRACT

Offline reinforcement learning holds the promise of obtaining powerful agents from large datasets. To achieve this, a good algorithm should always benefit from (or at least does not degenerate by) adding more samples, even if the samples are not collected by expert policies. However, we observe that many popular offline RL algorithms do not possess such a property and sometimes suffers from adding heterogeneous or poor samples to the dataset. Empirically we show that, given a stage in the learning process, not all samples are useful for these algorithms. Specifically, the agent can learn more efficiently with only the samples collected by a policy similar to the current policy. This indicates that different samples may contribute to different stages of the training process, and therefore we propose Curriculum Offline Reinforcement Learning (CUORL) to equip the previous methods with the such a favorable property. In CUORL, we select the samples that are likely to be generated by the current policy to train the agent. Empirically, we show that CUORL can prevent the negative impact of adding the samples from poor policies and always improves the performance with more samples (even from random policies). Moreover, CUORL also achieves state-of-the-art performance on standard D4RL datasets, which indicates the potential of curriculum learning for offline RL.

KEYWORDS

Offline Reinforcement Learning; Mixed Dataset; Curriculum Learning

ACM Reference Format:
Yuanying Cai, Chuheng Zhang, Hanye Zhao, Li Zhao, and Jiang Bian. 2023. Curriculum Offline Reinforcement Learning. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

Despite the great success and popularity gained by reinforcement learning (RL) [49], there exists a wide range of real-world applications that can hardly be solved using RL due to the high cost or the risk of online interactions, such as robotics [20], autonomous driving [18], and quantitative trading [12]. To address this problem, researchers study the offline RL paradigm that can learn good policies from datasets consisting of previously collected transitions. This paradigm has achieved many successes in both algorithmic designs [9, 10, 23, 24] and real applications [18, 26, 47] and holds the promise for making it possible to turn large datasets into powerful decision making agents [25].

Intuitively, adding more samples to the dataset should improve the performance of the learned policy. However, we observe that adding more samples does not always lead to performance improvement and sometimes even degenerates the performance in many popular offline RL algorithms, especially when the new samples are collected from a different or poor policy. This phenomenon can be concluded by comparing the performance of popular algorithms on two standard benchmark datasets from D4RL [7]: the expert dataset (with 1 million samples) and the medium-expert dataset (with 1 million samples the expert dataset and 1 million samples from the medium dataset). For a wide range of offline RL algorithms (especially policy-constraint-based algorithms), policies trained on the medium-expert dataset under-perform those trained on the expert dataset (see the experiment results in [9]). To show this, we take two representative offline RL algorithms, CQL and TD3+BC, as the example and show how the performance of the learned policy changes when the dataset is augmented with more samples in Figure 1. We observe a clear trend that, with more random/medium samples, the performance of the resultant policies degenerate monotonically.

In principle, more data reveals more information about the dynamics of the environment and should improve the performance. We argue that the gap between the experiment results (i.e., adding more data degenerates the performance) and this intuition is due to the fact that the samples do not contribute equally in different stages during the training. Specifically, we design experiments to show that the agent can learn more efficiently using samples that are collected by policies similar to the current target policy. Accordingly, a natural idea is to train the target policy using different data in different stages of the learning phase, which follows the curriculum learning paradigm and results in a novel algorithm called CUrriculum Offline Reinforcement Learning (CUORL). In this way, we can utilize a wider variety of samples as well as prevent the negative impact of poor samples. Specifically, in each iteration, we select the samples that are more likely to be collected by the current policy and train the policy based on the selected samples. In this way, new samples form poor behavior policies may be selected and

*Corresponding Author.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

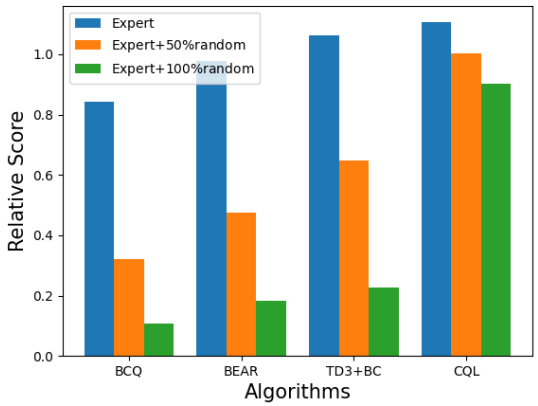

Figure 1: The performance of the policies trained by popular offline RL algorithms when the expert dataset is augmented with new samples collected by random policies. The expert and random samples come from the `walker2d-expert/random-v2` datasets from D4RL [7].

utilized in some early stages of the training process and therefore contribute to the performance and efficiency improvement. Moreover, when the target policy is well-trained and the samples from poor behavior policies cannot provide more useful information, these samples are not likely to be selected which thus avoids the negative impact of the of these poor data.

In our experiments, we show that CUORL outperforms the previous offline RL algorithms on not only the tasks with mixed datasets but also the tasks with samples from a single policy. Moreover, the performance of CUORL is non-decreasing when new samples are added to the dataset regardless of the quality of the behavior policies from which the samples are collected. We also provide a detailed analysis to show which samples are selected during different stages of training in CUORL. The analysis indicates that CUORL essentially follows curriculum learning by first learning from poor samples and then expert samples.

Our contributions are summarized as follows:

- We focus on an understudied but important problem towards RL from large datasets: The performance of existing offline RL algorithms does not improve and sometimes degenerates when heterogeneous or poor samples are added to the dataset.
- Based on the observation that different samples can contribute the different stages of policy learning, we propose a novel algorithm CUORL based on curriculum learning.
- Empirically, we find that CUORL can leverage samples from different behavior policies effectively. Moreover, although CUORL is designed for mixed datasets, it outperforms state-of-the-art offline RL algorithm on not only mixed datasets but also standard datasets from D4RL.

2 RELATED WORK

In this section, we provide a brief survey on previous offline RL methods and curriculum learning methods.

Offline RL. Many previous offline RL methods focus on the problem induced by distribution shift (i.e., the state distribution induced by the learned policy is different from that of the dataset) [see e.g., 24]. Previous solutions can be broadly divided into three categories: 1) the methods that regularize or constrain the learned policy to stay close to the behavior policy either directly [10, 21, 52] or implicitly [3, 9, 22, 23, 33, 38, 51], 2) the methods that regularize the learned value function by penalize the values on out-of-distribution state-action pairs according to different criteria such as the out-of-distribution detection metric [16, 28], the uncertainty quantification metric [4, 15, 53], and other metrics [2, 24, 55], and 3) the methods that estimate the policy gradient of the target policy based on off-policy samples using different importance sampling (IS) techniques such as weighted IS [40], the doubly-robust estimator [14], and marginalized IS [31, 32, 56].

While the first category (e.g., BCQ [10], BEAR [23], ABM [45], CQL [24], and TD3+BC [9]) achieves good performance and gains its popularity in the offline RL community, we find that its performance sometimes degenerates when more data is added. We consider this as a different problem from distribution shift due to the following two reasons: 1) Existing methods with the aim to address the distribution shift problem still suffer from this empirically. 2) Augmenting the dataset can enlarge (or at least does not decrease) the coverage of its state distribution and should alleviate the distribution shift problem, but algorithms can perform worse when adding poor samples.

Curriculum Learning in RL. Curriculum learning (CL) helps RL algorithms to optimize the order of the sub-tasks or the samples learned by the agent with the aim to improve the performance or the training speed on hard tasks [34]. Previously, CL is successful in helping the agent to deal with complex tasks for real-world applications [5, 54] or transfer knowledge between tasks [19, 37]. One category of the work focus on how to generate curriculum sub-tasks to benefit training process [35, 46] or how to optimize the order of the provided sub-task sequences [36, 50]. Among the topics studied in curriculum learning, the most relevant to ours is the category called *sample sequencing* that orders the samples from the replay buffer and results in an implicit curriculum. For example, PER [43] improves the uniform sampling method in DQN [30] by increasing the weights of the samples with large TD errors. DCRL [42] adaptively selects transitions with appropriate difficulty and penalizes frequently replayed samples. Kim and Choi [17] propose the ScreenerNet to learn sample weights jointly with the main task which saves memory compared with PER. However, most of these works focus on the online setting. Although there is a trend to extending CL to offline RL recently, these methods either directly apply the sample sequencing methods from online RL [8], or rely on strong assumptions of the offline dataset [27]. In this paper, we leverage CL to order the offline samples for the learning process to help the agent to maintain or improve the performance when more samples from poor behavior policies are added.

Compared with training the policy directly for the target task, a properly designed subtask sequence can lead to better performance

and higher efficiency [34]. For the online RL setting, the subtasks can be designed as learning to fit different teacher networks [29, 39], self-play with different opponents [48], achieving different goals [6, 41], or learning different skills [13]. For the offline RL setting, we find that a natural way to design the subtasks is to select samples based on which the policy is trained. We leverage this method to solve the problem that the performance degenerates when new samples from poor behavior policies are added. Compared with training the agent with the full dataset or naively filtering out poor samples, our method achieves better performance since we make full use of all the samples, e.g., poor samples that are collected by random policies are utilized at the beginning of the training to improve the learning efficiency and result in a better checkpoint for the later phases.

3 PRELIMINARY

Offline reinforcement learning. We consider a discounted infinite-horizon Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and action space respectively, $P : \mathcal{S} \times \mathcal{A} \to \Delta^{\mathcal{S}}$ is the state transition distribution, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\rho \in \Delta^{\mathcal{S}}$ is the initial state distribution, and $\gamma \in [0, 1]$ is the discounted factor [49]. The objective of online RL is to learn a policy π that maximizes the expected return $J(\pi) = \mathbb{E}_{s_0 \sim \rho, a_0 \sim \pi(\cdot|s_0)}[R_0^\pi]$ where the cumulative discounted reward is defined as $R_t^\pi = \sum_{i=t}^{\infty} \gamma^{i-t}(s_t, a_t)$ with $s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, \cdots$ collected by rolling out the policy π starting from (s_t, a_t). The Q function is defined as $Q^\pi(s, a) = \mathbb{E}[R_0^\pi | s_0 = s, a_0 = a]$ and the expectation is taken over all possible trajectories starting from (s, a) and following the policy π afterwards. The Q function is the fixed point of the Bellman policy operator

$$\mathcal{T}^\pi Q(s, a) := r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')}[Q(s', a')]. \quad (1)$$

The Q function of the optimal policy π^* is represented as $Q^* := Q^{\pi^*}$ which is the fixed point of the Bellman optimality operator

$$\mathcal{T}^* Q(s, a) := r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a'} Q(s', a') \right]. \quad (2)$$

For offline RL, the goal is to learn a good policy based on an offline dataset $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{t=1}^N$ collected by some behavior policies without interaction with the environment.

Curriculum learning. In curriculum learning for reinforcement learning, subtasks $\{T_1, T_2, \cdots, T_N\}$ are designed to train the policy starting from the initial policy π_0 with each of the tasks sequentially. The objective is to obtain a policy that can achieve good performance on the target task T_{target}. We can denote the learned policy under the subtask T_k as π_k.

4 OBSERVATIONS AND ANALYSIS

In this paper, we focus on the problem that adding new samples from poor behavior policies can harm the performance of the learned policy in offline RL. We take TD3+BC [9] and CQL [24] as two representatives of policy-constraint-based and value-constraint-based offline RL algorithms respectively. As we observed in Figure 1, both methods suffer from the problem. In this section, we first conduct experiments to discover how different samples affects the learning process in offline RL in these two algorithms. The observation motivates us to adopt curriculum learning in offline RL. To further justify the use of curriculum learning, we conduct experiments to illustrate how curriculum learning can be used to address the problem and analyze the criteria of the subtask design theoretically.

4.1 Empirical Observations

In this part, we conduct experiments to show how different samples can impact the learning process. The results indicate that offline RL agents learn more efficiently with the samples collected by neighboring policies.

To study how different training samples can affect the learning process in different stages of the offline RL training, we log the policies (as well as other networks) from different checkpoints during the training of an offline RL algorithm and start offline RL training with different datasets with these policies serving as the initialization. Specifically, we log three policies from different stages of an offline RL algorithm which are denoted as $\pi_{1/3}$, $\pi_{1/2}$ and $\pi_{2/3}$ respectively. We collect four datasets, $D_{1/3}$, $D_{1/2}$, $D_{2/3}$, and $D'_{1/2}$, which are obtained by rolling out logged policies. The first three are collected by $\pi_{1/3}$, $\pi_{1/2}$ and $\pi_{2/3}$ respectively, and the last one is collected by another policy that achieves the same performance but different from $\pi_{1/2}$, which has the largest distances between other three behavior policies to exclude the influence of the performance of the behavior policies.

We train the agent using offline RL algorithms (e.g., TD3+BC and CQL) based on these different offline datasets starting from the initial policy $\pi_{1/2}$. We present the results of TD3+BC in Figure 2 and we actually find similar trend of CQL in our experiments. Not surprisingly, the final performance of the TD3+BC agent depends on the quality of the offline dataset. However, we observe that the policy improves more rapidly when using the samples collected by similar or neighboring policies. For example, since the agent is trained starting from $\pi_{1/2}$, learning with $D_{1/2}$ is more efficient than learning with $D_{2/3}$ that contains transitions that can achieve higher scores. This indicates that learning from high-quality datasets is not always the best choice. Still, learning from $D'_{1/2}$ is much slower than learning from $D_{1/3}$ since $D_{1/3}$ is closer to the target policy $\pi_{1/2}$. Moreover, we can also compare the training of $\pi_{1/2}$ with $D_{1/2}$ and $D'_{1/2}$. Although the behavior polices behind these two datasets share similar performance, the agent learns faster on $D_{1/2}$ since the behavior policy behind $D'_{1/2}$ is more distant to $\pi_{1/2}$ than that of $D_{1/2}$ These observations indicate that learning from the samples collected by similar or neighboring policies can make the learning process more efficient.

4.2 How Can Curriculum Learning Help?

Previous observations naturally motivate us to select samples in different training stages such that the agent can learn most efficiently. In this way, we can join the most efficient training processes in different stages. This may be helpful especially when the offline RL agent is trained with the samples collected from a mixture of behavior policies. Accordingly, we design a curriculum-learning-based algorithm (which will be introduced in Section 5) that selects the samples that are likely to be collected by the current policy. Here, we show what samples are selected by this algorithm in different stages in advance to illustrate why it is helpful to address the

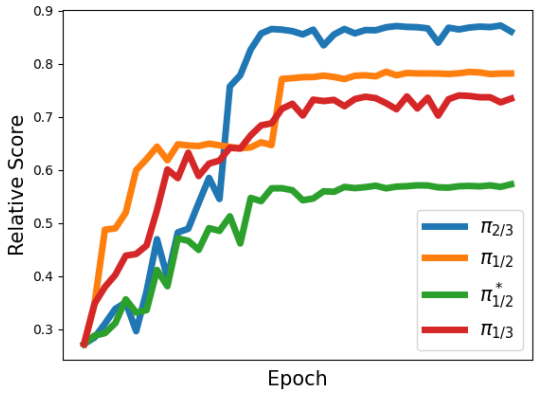

Figure 2: Empirical observations. We choose 3 checkpoints at different stages during the training process of the online SAC agent and collect datasets using these policies. We train the offline RL agent using the policy $\pi_{1/2}$ as the initialization with these datasets and plot the learning curves during training. We also use the dataset collected by another policy $\pi^*_{1/2}$ that has a similar performance as $\pi_{1/2}$ and has the largest distances between other policies to exclude the interference of the the performance. We observe that the agent learns more sample-efficiently at earlier training steps (see the orange curve) with samples from neighboring policies.

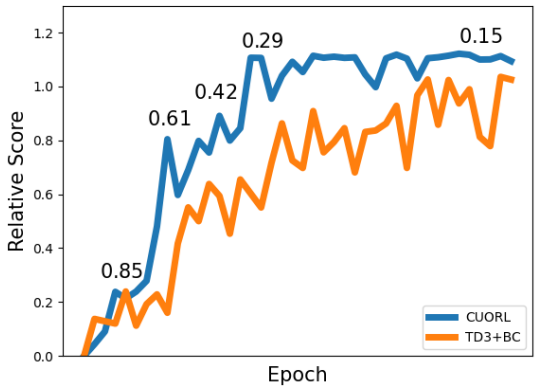

Figure 3: Illustration on how curriculum can help offline RL. We show the performance curve during the training on the medium-expert dataset of HalfCheetah with the numbers indicating the ratio of the samples selected from the medium dataset during curriculum learning. The ratio decreases as the learned policy achieves a better performance.

problem. The experiment is conducted on the halfcheetah-medium-expert-v2 dataset that contains samples from medium and expert behavior policies. We show the performance and the proportion of selected medium samples during the training in Figure 3. We first observe that using curriculum learning to select samples can help improve the performance at convergence as expected compared with sampling from the dataset uniformly as in standard offline RL methods. Moreover, we find that the proportion of selected medium samples decreases during the training process, which indicates the reason why our method works: The samples from poor behavior policies are utilized to boost the learning in early stages which leads to a good starting point of the later stages. In late stages, these samples are filtered out for the agent to focus more on how to match the performance of expert demonstrators.

4.3 Theoretical Analysis

In addition to the previous empirical observations that motivate us to select the samples from neighboring behavior policies for offline RL, we provide theoretical justification for this in this subsection. Specifically, we analyze how the similarity between the behavior policy and the current policy affects the learning efficiency during the offline RL learning process.

Consider updating the current policy π to a new policy $\tilde{\pi}$ based on the samples collected by the behavior policy π_b. We can derive the following performance difference lemma for the offline RL setting:

Lemma 1. *For any policies π, $\tilde{\pi}$ and π_b, we have*

$$\eta(\tilde{\pi}) \geq \eta(\pi) + \mathbb{E}_{s\sim d_{\pi_b}, a\sim\tilde{\pi}(\cdot|s)}[A_\pi(s,a)] - \frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \tilde{\pi}) \tag{3}$$

Note that Lemma 1 extends the standard performance difference lemma in [1] by introducing a behavior policy π_b which leads to an additional term $\frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \tilde{\pi})$ for offline RL setting.

We define $M_i^{\pi_b}(\pi)$ for any policy π as in Eq.(4).

$$M_i^{\pi_b}(\pi) = \eta(\pi_i) + \mathbb{E}_{s\sim d_{\pi_b}, a\sim\pi(\cdot|s)}[A_{\pi_i}(s,a)] - \frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \pi) \tag{4}$$

By taking π and $\tilde{\pi}$ as π_i and π_{i+1} respectively in the inequality (3) and taking π as π_{i+1} in Eq.(4), we observe $M_i(\pi_{i+1})$ is equal to the right hand side of inequality (3). Therefore, according to Lemma 1, we have:

$$\eta(\pi_{i+1}) \geq M_i^{\pi_b}(\pi_{i+1}). \tag{5}$$

On the other hand, by taking π as π_i in Eq.(4), we obtain

$$\begin{aligned} M_i^{\pi_b}(\pi_i) &= \eta(\pi_i) + \mathbb{E}_{s\sim d_{\pi_b}, a\sim\pi_i(\cdot|s)}[A_{\pi_i}(s,a)] - \frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \pi_i) \\ &= \eta(\pi_i) - \frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \pi_i) \end{aligned} \tag{6}$$

by noticing that

$$\mathbb{E}_{s\sim d_{\pi_b}, a\sim\pi_i(\cdot|s)}[A_{\pi_i}(s,a)] = 0.$$

We then reorganize Eq.(6) as

$$\eta(\pi_i) = M_i^{\pi_b}(\pi_i) + \frac{2\gamma}{1-\gamma}\epsilon_\pi D_{TV}^{\pi_b}(\pi_b, \pi_i). \tag{7}$$

Algorithm 1 CUORLv1: Curriculum Offline RL

1: **Inputs:** One update of the actor and critic networks in the base offline RL algorithm $\mathcal{A}$; offline dataset $\mathcal{D}$
2: **Parameters:** β, ζ, h, M, T.
3: Initialize the critic network Q_θ and the actor network π_ϕ
4: Initialize the target networks $Q_{\theta'}$ and $\pi_{\phi'}$
5: Repeat the trajectories in $\mathcal{D}$ for ζ times
6: Initialize buffer $\mathcal{B} = \emptyset$
7: **while** $\mathcal{D} \neq \emptyset$ **do**
8: ▷ *Compute the score of each trajectory in* $\mathcal{D}$
9: **for** each trajectory $\tau_i = \{(s^i_j, a^i_j)\}^h_{j=1} \in B$ **do**
10: $s_{ij} := \|\pi_{\phi'}(s^i_j) - a^i_j\|$ for all $j \in [h]$
11: The score of $s(\tau_i)$ is the β-quantile of $\{s_{ij}\}^h_{j=1}$
12: ▷ *Select trajectories to train the current policy*
13: $\mathcal{B} \leftarrow \min(M, |\mathcal{D}|)$ trajectories with smallest scores in $\mathcal{D}$
14: Remove the selected trajectories from $\mathcal{D}$
15: ▷ *Offline RL training*
16: **for** $t = 1$ to T **do**
17: $\theta, \theta', \phi, \phi' \leftarrow \mathcal{A}(\mathcal{B}; \theta, \theta', \phi, \phi')$

Finally, by combining Eq.(5) and Eq.(7), we obtain

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq M^{\pi_b}_i(\pi_{i+1}) - M^{\pi_b}_i(\pi_i) - \frac{2\gamma}{1-\gamma}\epsilon_\pi D^{\pi_b}_{TV}(\pi_b, \pi_i) \quad (8)$$

Since offline RL methods learn from finite samples collected by π_b, we provide a finite-sample version of Eq.(8).

Theorem 1. *Let π_i be the initialization policy at stage i during curriculum learning, there exists constants V and C, with high probability $\geq 1-\delta$, the following inequality holds:*

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq \hat{M}^{\pi_b}_i(\pi_{i+1}) - \hat{M}^{\pi_b}_i(\pi_i) - C \cdot \sqrt{D^{\pi_b}_{KL}(\pi_i, \pi_b)} - \sqrt{(V - \log\delta)/|D_{\pi_b}|} \quad (9)$$

where $\hat{M}^{\pi_b}_i(\pi) = \eta(\pi) + \mathbb{E}_{s\sim D_{\pi_b}, a\sim\pi}[A_{\pi_i}(s,a)] - C\sqrt{D^{\pi_b}_{KL}(pi_b, \pi)}$, D_{π_b} are the finite state samples following the stationary state distribution induced by π_b, and $|D_{\pi_b}|$ are the number of samples in D_{π_b}.

In order to obtain monotonic performance improvement from the initialization policy π_i, i.e., $\eta(\pi_{i+1}) - \eta(\pi_i) \geq 0$, we should maximize the RHS of (9). Note that given fixed dataset, offline RL methods could optimize $\hat{M}^{\pi_b}_i$ with additional constraints to remain the learned policy stay closed to the behavior policy to mitigate overestimation. Therefore, before using offline RL methods to maximize $\hat{M}^{\pi_b}_i$, we should also choose samples from neighboring policies of π_i to obtain the monotonic performance improvement.

5 METHOD

Based on the empirical observation and theoretical justification for using CL in offline RL, we propose the algorithm, CUrriculum Offline Reinforcement Learning (CUORL), in this section. CUORL can be easily plugged into a wide range of model-free offline RL algorithms that iteratively update the actor and the critic networks. In our later experiments, we combine CUORL with two popular offline RL algorithms TD3+BC [9] and CQL [24].

Algorithm 2 CUORLv2: Curriculum Offline RL

1: **Inputs:** One update of the actor and critic networks in the base offline RL algorithm $\mathcal{A}$ that receives weights for the samples; offline dataset $\mathcal{D}$
2: **Parameters:** h, M.
3: Initialize the critic network Q_θ and the actor network π_ϕ
4: Initialize the target networks $Q_{\theta'}$ and $\pi_{\phi'}$
5: **while** not converged **do**
6: $\mathcal{B} \leftarrow$ a mini-batch of hM transitions in $\mathcal{D}$
7: Compute weights $w_i = f(\|\pi_{\phi'}(s_i) - a\|), \forall i \in [hM]$
8: ▷ *Offline RL training*
9: $\theta, \theta', \phi, \phi' \leftarrow \mathcal{A}(\mathcal{B}, \mathbf{w}; \theta, \theta', \phi, \phi')$

In this section, we will introduce two versions of CUORL. The first version (presented in Algorithm 1) adopts a sample selection procedure that is guaranteed to select the samples collected by neighboring policies. The second version (presented in Algorithm 2) calculates the weights for each sample instead of selecting them which results in a much smaller computational overhead but achieves similar performance.

5.1 CUORL with Sample Selection

We present the procedure in Algorithm 1. In each iteration, we update the networks with the M trajectories selected from $\mathcal{D}$ that are most likely to be collected from neighboring policies of the current policy. To avoid overfitting to part of the trajectories, each trajectory can only be selected for ζ times. This is achieved by the procedure shown in Line 5 and Line 13-14, which can be implemented in a memory-efficient way in practice.

The core of CUORL is the sample selection procedure that calculates the score for each trajectory (cf. Line 8-11) and then selects the trajectories with the smallest scores. The score for each trajectory is calculated based on the probability that the action is selected by the current policy $\pi_{\phi'}(a^i_j|s^i_j)$ for each transition j in the i-th trajectory. Since we consider the continuous control problem in this paper, we use the ℓ_2 distance between the action generated by the target policy and the action in the dataset to replace this (cf. Line 10).

This sample selection procedure has a close connection to the objective derived from our theoretical analysis. Guided by the previous theoretical analysis, we need to select the samples according to the following criteria to ensure the one-step policy improvement:

$$D^{\pi_b}_{\mathrm{TV}}(\pi_b, \pi_k) := \mathbb{E}_{s\sim d_{\pi_b}}[D_{KL}(\pi_b(\cdot|s)||\pi_k(\cdot|s))] \leq \epsilon. \quad (10)$$

Following similar analysis to [27], under the assumption that each trajectories are collected by a deterministic behavior policy π_b with an exploration ratio β, the sample selection procedure can be proved to satisfy the constraint defined in (10).

5.2 CUORL with Weighted Update

Although the previous version has a close connection to the sample selection criteria defined in (10), it suffers from a large computational cost due to iterating over all the samples in the dataset to calculate the similarity score w.r.t. the current policy. To reduce the computational and memory costs of Algorithm 1, we propose an

	2nd-1/2	**Full**	**Full+Filter**	**CUORL**
HalfCheetah: TD3+BC	71.2±5.8	54.1±2.7	65.5±2.3	**101.2**±3.1
HalfCheetah: CQL	88.1±4.7	69.1±6.0	79.7±3.6	**92.6**±5.4
Hopper: TD3+BC	59.4±1.2	23.8±7.5	44.2±2.7	**98.5**±0.7
Hopper: CQL	83.0±13.4	67.7±11.6	76.8±9.4	**95.7**±8.4
Walker2d: TD3+BC	67.4±4.5	19.1±2.7	56.1±1.4	**96.2**±2.9
Walker2d: CQL	75.8±1.9	37.2±1.3	73.9±2.8	**94.5**±2.2
Total	444.9	271.0	396.2	**587.7**
Relative		-39.1%	-12.3%	+32.1%

Table 1: Comparison of different offline RL algorithms on highly mixed datasets collected by saved policies during the learning process of an online SAC agent. We present the mean and the standard deviation over 5 seeds. We highlight the largest scores. 2nd-1/2 and Full represent the performance of TD3+BC and CQL using the datasets collected by the second half of policies and all the policies respectively. We observe the performance degeneration from 2nd-1/2 to Full while CUORL (ours) achieves better performance than 2nd-1/2 using the full datasets. Full+Filter represents a naive method to filter trajectories with lower returns from the full datasets.

	CQL	**CL-CQL**	**TD3+BC**	**CL-TD3+BC**
HalfCheetah-rand	11.2±3.1	**19.8**±1.4	9.1±1.2	**18.7**±1.3
Hopper-rand	8.7±0.7	**12.0**±0.2	7.5±1.5	**14.1**±0.7
Walker2d-rand	2.5±1.1	4.3±1.2	1.4±1.0	5.6±0.8
HalfCheetah-med	44.6±1.8	**50.5**±1.2	41.3±2.1	**49.2**±2.1
Hopper-med	62.9±9.0	**72.4**±6.2	55.0±6.2	**70.8**±3.6
Walker2d-med	81.5±9.8	**94.6**±5.0	84.2±3.1	**91.5**±8.5
HalfCheetah-med-rep	24.7±1.7	**49.3**±1.6	43.6±1.8	**48.6**±1.1
Hopper-med-rep	88.1±1.1	**98.3**±2.1	61.1±1.7	**91.5**±1.8
Walker2d-med-rep	72.1±9.2	**82.5**±3.8	79.8±2.6	88.5±1.8
HalfCheetah-med-exp	72.4±5.1	**112.4**±10.3	90.1±11.3	**108.9**±6.1
Hopper-med-exp	102.0±4.2	**110.5**±2.6	97.4±7.1	**117.9**±4.9
Walker2d-med-exp	90.6±16.8	**109.6**±8.2	96.7±8.4	**116.5**±5.7
HalfCheetah-expert	86.5±13.6	**106.8**±4.5	104.1±6.1	**108.2**±4.4
Hopper-expert	101.4±13.5	**106.4**±1.3	102.4±0.5	**109.1**±1.0
Walker2d-expert	99.1±7.8	**103.7**±5.3	103.7±4.3	**112.8**±4.3
Total	948.3	**1133.1**	977.4	**1151.9**
Relative		+19.5%		+17.9%

Table 2: Comparison of the baseline algorithms and the CL-enhanced algorithms on standar D4RL datasets. We present the mean and the standard deviation over 5 seeds. We highlight the scores where CL enhanced algorithms outperforms the base offline RL algorithms.

efficient version with weighted updates and present it in Algorithm 2. In this version, we compute the distances between the actions generated by the target policy and the actions from the dataset and assign weights for the training samples in the mini-batch according to these distances. We assign higher weights for samples that are more likely collected by a neighboring policy of the learned policy. During the training process of the standard offline RL algorithm, the samples with higher weights have larger impact to the network update.

6 EXPERIMENTS

In the experiment, we evaluate the proposed algorithm CUORL from the following perspectives:

- We evaluate CUORL and baseline algorithms on mixed datasets that contain samples collected by a set of diverse policies of different performance level.
- We evaluate CUORL and baseline algorithms on the standard D4RL datasets, some of which are collected by single behavior policies.

	TD3+BC w/o rand	TD3+BC w/ rand	CUORL w/ rand	CUORL w/o rand
HalfCheetah-med-rep	43.6±1.8	37.8±2.8	53.5±2.4	48.6±1.1
HalfCheetah-med-exp	90.1±11.3	60.6±5.1	108.7±3.0	108.9±6.1
HalfCheetah-exp	104.1±6.1	44.8±4.7	113.5±2.5	108.2±4.4
Hopper-med-rep	61.1±1.7	54.4±2.6	96.1±0.9	91.5±1.8
Hopper-med-exp	97.4±7.1	44.8±2.6	119.3±1.6	117.9±4.9
Hopper-exp	102.4±0.5	66.3±2.9	113.7±0.8	109.1±1.0
Walker-med-rep	79.8±2.6	57.4±3.8	94.3±2.2	88.5±1.8
Walker-med-exp	96.7±8.4	37.7±1.4	119.1±1.9	116.5±5.7
Walker-exp	103.7±4.3	24.9±5.9	116.4±3.6	112.8±4.3
Total	778.9	**428.7**	934.6	**902.0**
Relative		-45.0%		

Table 3: Comparison of TD3+BC and CL-enhanced TD3+BC using standard D4RL datasets. With mixed random dataset, the performance of TD3+BC degeraded by 45% while the CL-enhanced performs better.

6.1 Learning from Mixed Datasets

To evaluate the proposed algorithm on highly mixed dataset that contains samples from behavior policies with diverse performance levels, we first train online agents using SAC [11] on three Mujoco tasks (Halfcheetah, Hopper and Walker) to obtain such datasets. Specifically, we set 50 checkpoints uniformly distributed during the training process of each tasks and save the learned policy at each checkpoint. Then, we roll out each of these policies for 10 trajectories to collect the highly mixed dataset.

We evaluate two base RL algorithms (TD3+BC and CQL) and corresponding CUORL-enhanced versions on the following three datasets: 2nd-1/2, Full, and Full+Filter. The "2nd-1/2" dataset contains the samples collected by the policies logged in the last 25 checkpoints out of the 50 checkpoints, whose data volume is only the half of that of the "Full" dataset. The "Full" dataset is a mixture of the samples collected by all the 50 policies, which has more samples than the "2nd-1/2" dataset but a lower average performance of the behavior policies. The "Full+Filter" dataset is a subset of the "Full" dataset by filtering out the trajectories with low returns and has a equal data volume to the "2nd-1/2" dataset.

We present present the performance of different algorithms on these datasets in Table 1. We find that although the "Full" dataset contains more data than "2nd-1/2", the performance of the learned policy degenerates by 39.1%. This indicates that adding samples from poor behavior policies can harm the performance in CQL and TD3+BC. Filtering out the trajectories with low returns is a naive method to address this problem. However, we find that, although this method improves the performance compared with learning on the whole "Full" dataset, the performance of the learned policies still degenerates by 12.3% compared with learning based on the "2nd-1/2" dataset. This demonstrates that this naive method cannot perfectly address the problem. In contrast, we observe that CUORL achieves a 32.1% performance improvement compared with the learning using the 2nd 1/2 dataset and achieves the best performance due to the fact that CUORL can make full use of the whole dataset.

6.2 Learning from D4RL datasets

We also compare CUORL with the baseline offline RL algorithms on standard D4RL datasets and present the results in Table 2. First, we look at the results of the mixed datasets in the D4RL benchmark such as the medium-expert datasets. We first observe that on the medium-expert datasets, the performance of both TD3+BC and CQL degenerates compared with that on the expert-dataset. In contrast, our algorithm achieves better performance on such mixed datasets compared with the expert datasets due to the use of curriculum learning that can take advantage of the additional medium data. Then, we focus on the performance on the datasets collected by single behavior policies such and the random/expert datasets. We observe that, even on the datasets collected by single policies, CUORL still outperforms the baseline offline RL methods and achieve a performance improvement of 19.5% and 17.9% compared with the baseline of CQL and TD3+BC respectively. This indicates that selecting data during the offline RL training can improve the performance even for datasets collected by single policies, which may benefit from the way training the neural network with easy to hard samples.

To further investigate how the algorithms perform on mixed datasets in the D4RL domain, we additionally mix the random dataset to the others and show the performance on these new mixed datasets in Table 3. This corresponds to many practical scenarios where large-scale datasets can be provided but most samples are collected by poorly-performed policies [44]. We observe that the performance of base offline RL methods degenerates when the samples from random policies are mixed to the dataset, whereas CUORL achieves better performance with these samples.

7 CONCLUSION

In this paper, we study a problem that widely exists in many policy-constraint-based offline RL algorithms: The performance of the agent degenerates when new samples collected by poor policies are added to the dataset. This breaks the promise of offline RL that can obtain effective policies from large-scale datasets with diverse

samples. Motivated by the observation that not all samples are useful given a stage of the offline learning and those from neighboring policies can help the agent learn most efficiently, we propose CUrriculum Offline RL (CUORL) that leverage curriculum learning to select samples that are likely to be collected by the current policy for the training of offline RL agents. CUORL is justified theoretically and can be easily plugged into existing offline RL algorithms. Empirically, CUORL not only be able to benefit from adding new samples (even the samples collected by random policies) to the dataset but also outperforms the baseline offline RL algorithm even on datasets collected by single policies. We also show that CUORL naturally results in a curriculum where poor samples are utilized first for the agent to learn efficiently around the low-reward regions, and expert samples are leveraged at last for the agent to match the expert. The effectiveness of CUORL indicates the potential of such a curriculum for offline RL.

REFERENCES

[1] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. 2019. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep* (2019), 10–4.

[2] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. 2020. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*. PMLR, 104–114.

[3] David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. 2021. Offline rl without off-policy evaluation. *Advances in Neural Information Processing Systems* 34 (2021), 4933–4946.

[4] Jacob Buckman, Carles Gelada, and Marc G Bellemare. 2020. The importance of pessimism in fixed-dataset policy optimization. *arXiv preprint arXiv:2009.06799* (2020).

[5] Rasheed El-Bouri, David Eyre, Peter Watkinson, Tingting Zhu, and David Clifton. 2020. Student-teacher curriculum learning via reinforcement learning: predicting hospital inpatient admission location. In *International Conference on Machine Learning*. PMLR, 2848–2857.

[6] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. 2018. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*. PMLR, 1515–1528.

[7] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219* (2020).

[8] Yuwei Fu, Di Wu, and Benoit Boulet. 2021. Benchmarking Sample Selection Strategies for Batch Reinforcement Learning. (2021).

[9] Scott Fujimoto and Shixiang Shane Gu. 2021. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems* 34 (2021), 20132–20145.

[10] Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*. PMLR, 2052–2062.

[11] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. PMLR, 1861–1870.

[12] Ben Hambly, Renyuan Xu, and Huining Yang. 2021. Recent advances in reinforcement learning in finance. *arXiv preprint arXiv:2112.04553* (2021).

[13] Allan Jabri, Kyle Hsu, Abhishek Gupta, Ben Eysenbach, Sergey Levine, and Chelsea Finn. 2019. Unsupervised curricula for visual meta-reinforcement learning. *Advances in Neural Information Processing Systems* 32 (2019).

[14] Nan Jiang and Lihong Li. 2016. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*. PMLR, 652–661.

[15] Ying Jin, Zhuoran Yang, and Zhaoran Wang. 2021. Is Pessimism Provably Efficient for Offline RL?. In *International Conference on Machine Learning*. PMLR, 5084–5096.

[16] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. 2020. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951* (2020).

[17] Tae-Hoon Kim and Jonghyun Choi. 2018. Screenernet: Learning self-paced curriculum for deep neural networks. *arXiv preprint arXiv:1801.00904* (2018).

[18] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. 2021. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* (2021).

[19] Pascal Klink, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. 2021. Boosted Curriculum Reinforcement Learning. In *International Conference on Learning Representations*.

[20] Jens Kober, J Andrew Bagnell, and Jan Peters. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32, 11 (2013), 1238–1274.

[21] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. 2021. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*. PMLR, 5774–5783.

[22] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169* (2021).

[23] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949* (2019).

[24] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779* (2020).

[25] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* (2020).

[26] Guogang Liao, Ze Wang, Xiaoxu Wu, Xiaowen Shi, Chuheng Zhang, Yongkang Wang, Xingxing Wang, and Dong Wang. 2022. Cross dqn: Cross deep q network for ads allocation in feed. In *Proceedings of the ACM Web Conference 2022*. 401–409.

[27] Minghuan Liu, Hanye Zhao, Zhengyu Yang, Jian Shen, Weinan Zhang, Li Zhao, and Tie-Yan Liu. 2021. Curriculum offline imitating learning. *Advances in Neural Information Processing Systems* 34 (2021), 6266–6277.

[28] Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. 2020. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202* (2020).

[29] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. 2019. Teacher–student curriculum learning. *IEEE transactions on neural networks and learning systems* 31, 9 (2019), 3732–3740.

[30] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[31] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. 2019. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in Neural Information Processing Systems* 32 (2019).

[32] Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. 2019. AlgaeDICE: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074* (2019).

[33] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. 2020. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359* (2020).

[34] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. 2020. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. *Journal of Machine Learning Research* 21, 181 (2020), 1–50.

[35] Sanmit Narvekar, Jivko Sinapov, Matteo Leonetti, and Peter Stone. 2016. Source task creation for curriculum learning. In *Proceedings of the 2016 international conference on autonomous agents & multiagent systems*. 566–574.

[36] Sanmit Narvekar, Jivko Sinapov, and Peter Stone. 2017. Autonomous Task Sequencing for Customized Curriculum Design in Reinforcement Learning.. In *IJCAI*. 2536–2542.

[37] Sanmit Narvekar and Peter Stone. 2019. Learning Curriculum Policies for Reinforcement Learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 25–33.

[38] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. 2019. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177* (2019).

[39] Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. 2020. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*. PMLR, 835–853.

[40] Doina Precup. 2000. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series* (2000), 80.

[41] Sebastien Racaniere, Andrew K Lampinen, Adam Santoro, David P Reichert, Vlad Firoiu, and Timothy P Lillicrap. 2019. Automated curricula through setter-solver interactions. *arXiv preprint arXiv:1909.12892* (2019).

[42] Zhipeng Ren, Daoyi Dong, Huaxiong Li, and Chunlin Chen. 2018. Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning. *IEEE transactions on neural networks and learning systems* 29, 6 (2018), 2216–2226.

[43] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2016. Prioritized Experience Replay. In *ICLR (Poster)*.

[44] Wenjie Shi, Tianchi Cai, Shiji Song, Lihong Gu, Jinjie Gu, and Gao Huang. 2020. Robust Offline Reinforcement Learning from Low-Quality Data. (2020).
[45] Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. 2020. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396* (2020).
[46] Felipe Leno Da Silva and Anna Helena Reali Costa. 2018. Object-oriented curriculum generation for reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 1026–1034.
[47] Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. 2021. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review* (2021), 1–46.
[48] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. 2017. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407* (2017).
[49] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[50] Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. 2017. Automatic curriculum graph generation for reinforcement learning agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.
[51] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. 2020. Critic regularized regression. *Advances in Neural Information Processing Systems* 33 (2020), 7768–7778.
[52] Yifan Wu, George Tucker, and Ofir Nachum. 2019. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361* (2019).
[53] Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. 2021. Uncertainty Weighted Actor-Critic for Offline Reinforcement Learning. *arXiv preprint arXiv:2105.08140* (2021).
[54] Sijia Xu, Hongyu Kuang, Zhuang Zhi, Renjie Hu, Yang Liu, and Huyang Sun. 2019. Macro action selection with deep reinforcement learning in starcraft. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 15. 94–99.
[55] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239* (2020).
[56] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. 2020. GenDICE: Generalized offline estimation of stationary values. *arXiv preprint arXiv:2002.09072* (2020).