A Perspective of Q-value Estimation on Offline-to-Online Reinforcement Learning

Yinmin Zhang1, 2*, Jie Liu1, 3*, Chuming Li1, 2*, Yazhe Niu2, Yaodong Yang4†, Yu Liu2†, Wanli Ouyang2
1 The University of Sydney, SenseTime Computer Vision Group, Australia
2 Shanghai Artificial Intelligence Laboratory
3 Multimedia Laboratory, The Chinese University of Hong Kong
4 Institute for AI, Peking University
yinmin.zhang@sydney.edu.au, jieliu@ie.cuhk.edu.hk, chli3951@uni.sydney.edu.au,
yaodong.yang@pku.edu.cn, liuyuisanai@gmail.com, ouyangwanli@pjlab.org.cn

Abstract
Offline-to-online Reinforcement Learning (O2O RL) aims to improve the performance of offline pretrained policy using only a few online samples. Built on offline RL algorithms, most O2O methods focus on the balance between RL objective and pessimism, or the utilization of offline and online samples. In this paper, from a novel perspective, we systematically study the challenges that remain in O2O RL and identify that the reason behind the slow improvement of the performance and the instability of online finetuning lies in the inaccurate Q-value estimation inherited from offline pretraining. Specifically, we demonstrate that the estimation bias and the inaccurate rank of Q-value cause a misleading signal for the policy update, making the standard offline RL algorithms, such as CQL and TD3-BC, ineffective in the online finetuning. Based on this observation, we address the problem of Q-value estimation by two techniques: (1) perturbed value update and (2) increased frequency of Q-value updates. The first technique smooths out biased Q-value estimation with sharp peaks, preventing early-stage policy exploitation of sub-optimal actions. The second one alleviates the estimation bias inherited from offline pretraining by accelerating learning. Extensive experiments on the MuJoco and Adroit environments demonstrate that the proposed method, named SO2, significantly alleviates Q-value estimation issues, and consistently improves the performance against the state-of-the-art methods by up to 83.1%.

Introduction
In recent years, deep offline Reinforcement Learning (RL) (Yu et al. 2020; Agarwal, Schuurmans, and Norouzi 2020; Bai et al. 2021) has received increasing attention due to its potential for leveraging abundant logged data or expert knowledge (e.g., human demonstrations) to learn high-quality policies. Analogous to trends in computer vision (He et al. 2021; Radford et al. 2021; Bommasani et al. 2021) and natural language processing (Devlin et al. 2019; Brown et al. 2020), where powerful models pretrained on large, diverse datasets generalize to task-specific data through finetuning, offline-to-online reinforcement learning (O2O RL) aims to enhance the performance of a pretrained offline RL policy via finetuning on only a few online collected samples. In O2O RL, the central challenge is to maximize the efficient utilization of additional online samples for further performance improvement.

We find that directly finetuning offline RL policies with online interactions remains inefficient in many cases, as shown in Figure 2a. Most existing works (Zhao et al. 2022; Lee et al. 2022) assume that the above issue results from the state-action distribution shift between the offline and online samples. To handle this distribution shift, they propose different methods to progressively transfer from offline to online finetuning settings, including the balance between RL objective and pessimism (Zhao et al. 2022) and between the usage of offline and online samples (Lee et al. 2022).

The main contribution of this paper is to offer a comprehensive understanding of the problem in O2O RL, focusing on Q-value estimation, a perspective that has been under-explored. While prior research acknowledges this issue, our work delves deeper into it, unveiling two key challenges: (1) biased Q-value estimation and (2) inaccurate rank of Q-value estimation, which represents the distinguishability in the quality of different state-action pairs. Despite most offline RL policies trained in pessimistic manners such as conservative Q-learning (Kumar et al. 2020), action constraints (Fujimoto and Gu 2021), and Q-ensemble (An et al. 2021), we still observe severe overestimation (CQL and TD3-BC) or underestimation (EDAC) compared to an online RL policy trained from scratch with similar performance. These disparities result in a severe bootstrap error, as shown in Figure 2b. Moreover, it is well-known that the inaccurate rank of Q-value estimation, such as inaccurate advantage, results in misleading signals to policy updates. To quantitatively assess the accuracy of Q-value estimation rank, we compute Kendall’s τ coefficient between the estimated and true Q-values, separately for offline RL and online RL policies. Figure 2c shows that offline RL methods (CQL, TD3-BC, and EDAC) have lower Kendall’s τ coefficient compared to the online RL, SAC. This indicates that the offline RL is less accurate in ranking estimated Q-values than the online RL. Consequently, the signal from the inaccurate Q-value estimation used in pol-
ic updates is misleading, which causes instability and slow improvement of performance during online finetuning.

To mitigate the Q-value estimation problems stated above, we identify two widely used techniques: (1) perturbed value update and (2) increased frequency of Q-value updates. Specifically, we perturb the target action with an extra noise to smooth out the biased Q-value estimation with sharp peaks. This prevents the agent from overfitting to a specific action. Consequently, these methods focus on the challenge of significant distribution gaps between offline datasets and online samples during O2O finetuning. Particularly, AWAC (Nair et al. 2020) enforces an implicit constraint on policy updates to align the learned policy with policies in both offline and online samples. (Lee et al. 2022) employs Q-function ensemble and balanced replay, encouraging the use of near-policy samples from offline demonstrations using neural network predictions. AdaptiveBC (Zhao et al. 2022) proposes a randomized Q-function ensemble, dynamically balancing the RL objective with behavior cloning based on behavior performance and training stability. IQL (Kostrikov, Nair, and Levine 2021) approximates each state’s upper bound of Q-values and extracts the policy from the Q-value estimation. ODT (Zheng, Zhang, and Grover 2022) proposes an O2O RL algorithm using sequence modeling and entropy regularizers to unify pertaining and finetuning. PEX (Zhang et al. 2022) introduces a policy expansion approach by incorporating offline policies for additional learning. QDagger (Agarwal et al. 2022) focuses on optimizing knowledge transfer efficiency within the reinharding RL paradigm. While some prior works do acknowledge inaccurate Q-value estimation, they may not have explicitly dissected the various types of inaccuracies and their specific effects on policy updates and online finetuning. Our research extends the existing literature by providing a more comprehensive understanding of these issues within O2O RL. Specifically, we focus on uncovering the challenges related to biased Q-value estimation and inaccurate Q-value ranking inherited from offline RL. These issues result in unreliable signals that exacerbate bootstrap errors during finetuning, leading to instability and suboptimal policy updates. Our investigation and proposed solutions address problems not explored in-depth in previous research.

Related Work

Offline RL Offline RL employs static datasets to train agents but faces Q-value overestimation challenges, especially with unseen state-action pairs (Fujimoto, Meger, and Precup 2019). Solutions can be categorized as follows: (1) Policy constraint methods (Nair et al. 2020; Fujimoto and Gu 2021; Wu, Tucker, and Nachum 2019) enforce policy alignment with the behavior policy in logged datasets, minimizing policy distribution shift through behavior cloning and KL-divergence; (2) Pessimistic value methods (Kumar et al. 2020) regularize the Q-value to mitigate the overestimation by adding a conservative penalty to the RL object; (3) Uncertainty-based methods (An et al. 2021; Kumar et al. 2019; Bai et al. 2021) measure the distribution shift using Q-value uncertainty to guarantee a robust policy update.

Online RL with offline datasets. Several works (Ijspeert, Nakanishi, and Schaal 2002; Kim et al. 2013; Zhu et al. 2019; Nair et al. 2017; Theodorou, Buchli, and Schaal 2010; Gupta et al. 2020) improve online RL by utilizing offline datasets, assuming these datasets contain optimal demonstrations specific to the current environment. However, real-world offline datasets, sourced from various sources, contain numerous sub-optimal demonstrations. Consequently, these methods may not perform well in such scenarios due to estimation bias from sub-optimal datasets. An alternative approach (Nair et al. 2020; Lee et al. 2022) takes a pessimistic view, avoiding the assumption of optimality. These methods focus on the objective of online RL is the expected cumulative Q-value.

Background

RL seeks to learn a policy maximizing the cumulative discounted reward in the Markov Decision Process (MDP) defined by \((S, \mathcal{A}, \mathcal{R}, p, \gamma)\), with the state space \(S\), the action space \(\mathcal{A}\), reward function \(\mathcal{R}\), transition dynamics \(p\), and discount factor \(\gamma \in (0, 1)\). Actor-critic algorithms maintain the Q-value function \(Q(s, a)\) and the learned policy \(\pi(a | s)\).

Offline RL. Offline RL assumes the agent leverages the logged datasets without any interaction with the environment. The agent is trained in a logged dataset \(D\) which is sampled from the environment by a behavior policy \(\pi_{\beta}(a | s)\). For the actor-critic algorithm given dataset \(D = \{(s, a, r, s')\}\), we can represent value iteration and policy improvement by:

\[
Q^{k+1} \leftarrow \arg \min_Q E_{s,a \sim D} \left[ (B^\pi Q^k(s, a) - Q(s, a))^2 \right],
\]

\[
\pi^{k+1} \leftarrow \arg \max_{\pi} E_{s \sim \mathcal{D}, a \sim \pi(a | s)} [Q^{k+1}(s, a)],
\]

where the \(B^\pi\) is the bellman operator on the fixed dataset following the learned policy \(\pi(a | s)\), \(\mathcal{D}\) is the dataset and \(\pi(a' | s)\) is the target policy.

Online RL. Following a given policy \(\pi(a | s)\), the objective of online RL is the expected cumula-
Q-value Estimation Issues in O2O RL

In this section, we systematically study the issues in O2O RL by focusing on Q-value estimation. We begin by assessing the performance of offline RL methods in O2O scenarios and subsequently pinpoint the root cause as Q-value estimation.

Performance. We evaluate the performance of both standard online RL methods and offline RL methods with online finetuning. The online RL methods include SAC and the online variant of EDAC, which are trained without any online dataset. Their curves are labeled with star marks in Figure 2a. The offline RL methods include policy constraints (TD3-BC), conservative Q-value learning (CQL), and Q-ensemble (EDAC) in the O2O RL setting. Additionally, we evaluate the loose variant of those methods by removing the pessimistic constraints during online finetuning after offline pretraining. The performance of standard online RL methods and offline RL methods is averaged on three environments (HalfCheetah, Hopper, and Walker2D). Particularly, for offline RL, each environment has four datasets with different qualities in D4RL benchmark and generates four pretrained policies, resulting in total 12 policies to average on. It is observed in Figure 2a that the improvement of CQL, EDAC, TD3-BC and the loose variant of CQL and EDAC is slow in online finetuning; and the asymptotic performance of the loose variant of TD3-BC is significantly worse than online RL and its initial performance. In summary, while offline algorithms (CQL, TD3-BC and EDAC) achieve excellent performance in offline settings, all of them achieve slow performance improvement or even performance degradation during online finetuning.

Normalized difference of the Q-value. We use normalized difference, also called percent difference in (Fujimoto and Gu 2021), to measure the difference between the estimated Q-value and the true Q-value, formalized by \( \frac{Q_{\text{estimated}} - Q_{\text{true}}}{Q_{\text{true}}} \). \( Q_{\text{true}} \) is computed based on actual returns obtained along an adequately extended trajectory collected by the current policy, providing an accurate reflection of the true Q-value. To highlight the difference between online RL and offline RL algorithms, the normalized differences of offline RL algorithms are subtracted by the normalized difference of the online RL baseline, SAC (Haarnoja et al. 2018). Therefore, a positive normalized difference means that the Q-value estimation of the offline RL algorithm tends to overestimate, compared to the online baseline, and vice versa. Figure 2b demonstrates the subtracted normalized difference of CQL, TD3-BC and EDAC. While offline algorithms deal with the out-of-distribution problem via a pessimistic perspective, TD3-BC and CQL usually overestimate the Q-value for the unseen state-action pairs collected online. Conversely, EDAC faces the issue of Q-value estimation bias, notably leading to significant Q-value underestimation when compared to online reinforcement learning algorithms. This Q-value estimation bias propagates through the Bellman Equation, resulting in suboptimal policy updates or, in some cases, divergence. Specifically, excessively high Q-value estimates can lead to the collapse of Q-value, while overly low Q-value estimates can impede progress, causing slow improvement.

Kendall’s \( \tau \) coefficient over Q-value. Kendall’s \( \tau \) coefficient measures the rank correlation between two sets of variables. To assess the precision of the rank-ordering of estimated Q-values, we employ Kendall’s \( \tau \) coefficient to compare them with true Q-values. We evaluate this metric on policies pretrained by CQL, TD3-BC, and EDAC, as well as online SAC, all of which exhibit comparable performance. This evaluation follows a structured procedure: (1) Roll out with each pretrained policy to collect multiple episodes of state-action pairs; (2) Select collections of pairs within each episode using a sliding window approach, denoting each collection as \( P_i \), where \( i \in [1, ..., M] \) and \( M \) represents the window number (10, as used in reported results); (3) Compute both the estimated Q-value and true Q-value of all state-action pairs within each \( P_i \); (4) Compute Kendall’s ratio \( K_i \) for each collection, and derive the final metric \( K \) as the average, given by \( K = \frac{1}{M} \sum K_i \). Figure 2c exhibits the \( K \) for each RL algorithm. The results reveal that offline RL
algorithms, CQL, TD3-BC, and EDAC, exhibit significantly lower \( K \) values compared to the online RL algorithm SAC. This suggests that the estimated Q-values are not accurate in assessing the relative qualities of different state-action pairs.

**Summary.** While these offline algorithms (CQL, TD3-BC, and EDAC) achieve outstanding performance after pretraining and provide great initial actors for online finetuning, all of them improve slowly with high variances in different environments with different settings, due to the biased estimation and inaccurate rank of Q-value. Thus, how to obtain an accurate Q-value estimation is the bottleneck of O2O RL.

### A Simple but Effective Approach to O2O RL

**Algorithmic Details**

To provide proper guidance for online policy updates, we present an O2O RL approach that improves the accuracy of Q-value estimation. Our algorithm builds on top of the SAC with Q-ensemble, with two modifications, Perturbed Value Update, and Increased Frequency of Q-value Update.

**Perturbed Value Update (PVU).** We modify the update step of the ensemble Q-value, as follows:

\[
\mathcal{T} Q_{\phi_i}(s, a) \leftarrow r + \gamma \left( \hat{Q}_{\phi_i}(s', a' + \epsilon) - \beta \log \pi(a' | s') \right),
\]

\[
\epsilon \sim \text{clip}(N(0, \sigma), -c, c), \quad i = 1, \ldots, N_{\text{ensemble}},
\]

where \( \epsilon \) represents action noise with variance \( \sigma \) and bounded by \( c \). \( \phi_i \) is the parameter of the \( i \)-th Q network in the ensemble method, with a total of \( N_{\text{ensemble}} \) networks. This perturbation enlarges the action distribution used for estimating the target Q-value, resulting in a smoother Q-value estimate. Such smooth estimation prevents extremely exploiting the state on which the action has biased Q-value estimation, often seen as sharp peaks. Hence, this perturbation encourages policy exploration of state-action pairs, even when they have low-estimated Q-values inherited from the offline RL pretraining, and further updates the Q-value estimation of these pairs to find potential strategies with high returns.

**Increased Frequency of Q-value Update.** To alleviate the inaccurate Q-value estimation in offline RL, as discussed in Sec., we increase the update frequency of the Q-value function after each online collection. Specifically, we denote the update frequency after each online sample collection as \( N_{\text{upc}} \), where \( \text{upc} \) means update per collection.

**Implementation Details**

We describe the implementation of SO2 based on SAC with Q-ensemble in DI-engine, a DRL framework utilized by various methods (Li et al. 2023b,a; Liu et al. 2023; Wang et al. 2023). The pseudocode is shown in Algorithm 1, with differences from Q-ensemble SAC in Lines 5-8. After each online collect iteration, we sample a mini-batch \( B \) from the replay buffer. This buffer contains both online and extensive offline dataset samples, enhancing training stability. Subsequently, we perform \( N_{\text{upc}} \) updates to the Q-value network.

**Experiments**

Our experiments aim to investigate the following concerns:

- **Performance:** Can our method improve the performance compared to existing O2O RL approaches and online RL approaches trained from scratch (see Figure 3)?
- **N_{upc}**: Does increasing update frequency per collection effectively enhance performance (see Table 3)?
- **PVU:** Does the proposed Perturbed Value Update stabilize the O2O training (see Table 2)?
- **Q-value estimation:** Whether the proposed method can effectively address Q-value estimation issues including estimation bias and inaccurate rank (see Figure 3b)?
- **Extension:** Does our method generalize to more challenging robotic manipulation tasks (see Table 4)?
We train policies for 100K environment steps and evaluate every 1K step. Our method uses perturbation noise with $\sigma = 0.3$, $c = 0.6$, and $N_{upc} = 10$ as the default setup.

Figure 3 shows the differences in starting performance across different methods arise from each O2O algorithm utilizing distinct baselines. Moreover, the hyperparameters employed in O2O algorithms are meticulously tailored to their respective baselines, rendering them inherently sensitive. Ensuring consistent starting performance across all algorithms is crucial.
Table 1: Normalized average returns on D4RL Gym tasks, averaged over 4 random seeds. SO2 significantly improves the performance. The high variance reveals that even if the learned performance on the Walker2d.

Table 2: SO2 Performance across various $\sigma$ on the Walker2d. W=Walker2d, R=Random, M=Medium, E=Expert.

Table 3: Performance of SO2 over various $N_{\text{upc}}$ pretrained on Walker2d. $N_{\text{upc}}$ denotes the update frequency per collection. W=Walker2d, R=Random, M=Medium, E=Expert.

Analysis on Perturbed Value Update. In this paper, we argue that in the O2O task, adding appropriate noise to the target action can make training more stable and result in better policy performance. To verify the effectiveness of target noise, we conduct an ablation over $\sigma$. As shown in Table 2, the baseline without the target noise ($\sigma = 0$) has significant variances in all offline datasets, up to 1/3 of the average performance. The high variance reveals that even if the learned policies have the same initial performance, some of them may still perform poorly on some episodes, and the learned policy may fluctuate dramatically among each evaluation. On the contrary, the volatile policy, once equipped with the Perturbed Value Updates, achieves consistently impressive performance with low variance across various PVUs.

Analysis on $N_{\text{upc}}$. To verify the effectiveness of $N_{\text{upc}}$, we report the performance of SO2 on Walker2D with different $N_{\text{upc}}$. Table 3 shows that the performance consistently improves with the increase of $N_{\text{upc}}$ in all environments on Walker2D. It is worth noting that increasing $N_{\text{upc}}$ on random data has the most obvious effect. Intuitively, the $N_{\text{upc}}$ term induces the value network to favor frequent updates after each interaction with environments. Consequently, height-
Algorithm 1: SO2

1: Initialized policy parameters $\theta$, Q-function parameters $\{\phi_j\}_{j=1}^N$, target Q-function parameters $\{\phi_j'\}_{j=1}^N$, offline replay buffer $D_{\text{off}}$, online replay buffer $D_{\text{on}}$, epoch number $E$ and frequency of Q-value updates $N_{\text{upc}}$, and batch size $B$.
2: for epoch $e$ in 1, 2, …, $E$ do
3:   Collect a transition $\tau = (s, a, r, s')$ via environment interaction with $\pi_\theta$.
4:   Update online replay buffer $D_{\text{on}} \leftarrow D_{\text{on}} \cup \tau$.
5:   for $N_{\text{upc}}$ iterations do
6:     Sample a mini-batch $b$ of transition $(s, a, r, s')$ from $D_{\text{off}} \cup D_{\text{on}}$.
7:     Compute target Q-values $TQ_{\phi_{j'}}(s, a)$ with the objective in Equation 3.
8:     Update each Q-function $Q_{\phi_j}$ by gradient descent with:
9:         Value Phase
10:        $\nabla_{\phi_j} \frac{1}{B} \sum_{(s, a, r, s') \in b} \left( Q_{\phi_j}(s, a) - TQ_{\phi_{j'}}(s, a) \right)^2$.
11:         Update policy by gradient ascent with: $\nabla_{\theta} \frac{1}{B} \sum_{s \in b} \left( \min_{a} Q_{\phi_j}(s, a_{\tilde{\theta}}(s)) - \beta \log \pi_\theta(a_{\tilde{\theta}}(s) \mid s) \right)$,
12:   end for
13: end for

Table 4: Normalized average scores on the Adroit. We report the finetuning results of baselines with 1M environment steps, while the online finetuning results of ours with 400k steps.

Table 5: Compatibility of SO2. SO2 Integration with OFF2ON and PEX, Reported with 100K Online Samples.

Figure 5: (a): Performance comparing our proposed method against online SAC in HalfCheetah. Although the baseline is initialized from the HalfCheetah-Random dataset with limited performance, our proposed method boosts the baseline algorithm effectively and significantly improves the sample efficiency by up to 17 compared to SAC. (b): Kendall’s $\tau$ coefficient comparing ablation over Perturbed Value Update (PVU), and update per collection ($N_{\text{upc}} = 10$) in the MuJoco. This measures the rank correlation between true Q-value and estimated Q-value for similar state-action pairs.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>AWAC</th>
<th>IQL</th>
<th>Ours (400k)</th>
<th>Ours(400k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pen-human</td>
<td>44.6→70.3</td>
<td>37.4→60.7</td>
<td>51.4→70.6</td>
<td>1M</td>
</tr>
<tr>
<td>door-human</td>
<td>1.3→30.1</td>
<td>0.7→32.3</td>
<td>16.5→66.2</td>
<td>400k</td>
</tr>
<tr>
<td>Total</td>
<td>46.7→103.1</td>
<td>38.1→124.0</td>
<td>67.9→136.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Compatibility of SO2. SO2 Integration with OFF2ON and PEX, Reported with 100K Online Samples.

OFF2ON (Lee et al. 2022), PEX (Zhang, Xu, and Yu 2022), underscores the effectiveness of our approach.

Evaluation on Adroit Tasks

Setup and Results. We also conduct experiments on Adroit. Specifically, the offline policy is trained with limited human demonstrations, and the online finetuning policy is trained with 1M environment steps for baseline, and with only 400k environment steps for ours. Results for AWAC and IQL are quoted from IQL paper. Table 4 shows that our method also outperforms all baselines by a large margin.

Conclusion

In this paper, we have delved into O2O reinforcement learning and systematically studied why this setting is challenging. Different from most existing works, we in-depth analyze the Q-value estimation issues in offline-to-online including the biased estimation and inaccurate rank of the Q-value, besides the bootstrap error resulting from state-action distribution shift. Based on this argument, we propose smoothed offline-to-online (SO2). It effectively and efficiently improves the Q-value estimation by perturbing the target action and improving the frequency of Q-value updates. The proposed method, without any explicit estimation of the state-action distribution shift and complex components to balance offline and online replay buffers, remarkably improves the performance of the state-of-the-art methods by up to 83.1% on the MuJoco and Adroit environments.
Acknowledgments

This research is funded by Shanghai AI Laboratory. This work is partially supported by the National Key R&D Program of China (NO.2022ZD0160100), (NO.2022ZD0160101), and National Natural Science Foundation of China (62376013). This work was done during Yinmin’s internship at Shanghai Artificial Intelligence Laboratory. We would like to thank many colleagues for useful discussions, suggestions, feedback, and advice, including: Zilin Wang and Ruoyu Gao.

References


