PESSIMISTIC POLICY ITERATION FOR OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline reinforcement learning suffers from extrapolation error in the Q-value function. In addition, most methods enforce a consistent constraint on the policy during training, regardless of its out-of-distribution level. We propose pessimistic policy iteration, which guarantees that the Q-value error is small under the trained policy's distribution and bounds the sub-optimality gap of the trained policy's value function. At the same time, pessimistic policy iteration's core component is a horizon-flexible uncertainty quantifier, which could set a constraint according to regional uncertainty. The empirical study shows that the proposed method could boost the performance of baseline methods and is robust to the scale of the constraint. Also, a flexible horizon of uncertainty is necessary to identify out-of-distribution regions.

1 INTRODUCTION

Offline reinforcement learning (RL) is promising to be applied to medical treatment (Gottesman et al., 2018; Wang et al., 2018), advertisement systems (Chen et al., 2019), and autonomous driving (Sallab et al., 2017; Kendall et al., 2019), due to its safety and the availability of large datasets.

However, offline RL suffers from out-of-distribution (OOD) issue heavily (Fujimoto et al., 2019). From the viewpoint of restriction, recent attempts to mitigate OOD issue can be divided into two branches: policy-restriction methods (Wu et al., 2019; Kumar et al., 2019) and Q-restriction methods (Kumar et al., 2020; Cheng et al., 2022). The former regularize the trained policy to be close to the behavior policy while the latter penalize OOD actions' Q-values.

Although current methods have obtained desirable performance, several issues have not been tackled yet. The most important challenge is that the in-distribution constraints still cannot eliminate the error in Q-value estimation introduced by iterative off-policy evaluation (Brandfonbrener et al., 2021). The error could be controlled by enforcing a strong constraint, but it might limit the potential to discover the optimal policy hidden in the dataset. IQL (Kostrikov et al., 2022) and One-step RL (Brandfonbrener et al., 2021) could avoid extrapolation error by utilizing in-sample learning without querying the values of unseen actions in the dataset. However, in-sample learning might also limit the policy's generalization ability. We expect that the target action in Bellman equation is still produced by the trained policy, thus enjoying generalization benefit. Under this situation, we hope that the Q-value error is small under the trained policy's distribution.

Another issue is that, most methods enforce a consistent constraint on the policy, regardless of its OOD level. Uncertainty estimation has been introduced to deal with the issue. Among policy-restriction methods, UWAC (Wu et al., 2021) utilizes MC-dropout (Gal & Ghahramani, 2016) to estimate the uncertainty of a given state-action pair. However, the uncertainty estimator only focuses on the local transition's uncertainty while ignoring the future trajectory's uncertainty, dubbed global uncertainty. Taking Fig.1 as an example, the black and blue regions denote OOD and in-distribution regions, respectively. The aim of the agent is to move from left to right. In Fig.1(a), the local uncertainty is identical for the two orange points, but the agent should be more conservative at the left point than at the right one. So a global uncertainty estimator is necessary. However, there is still a lack of study on global uncertainty for policy-restriction methods. For Q-restriction methods, SAC-N (An et al., 2021), MOPO (Yu et al., 2020), and PBRL (Bai et al., 2022) take the local uncertainty as an intrinsic reward explicitly or implicitly and embed it into Q-value. Global uncertainty is learned but it must share the same horizon with the Q-value evaluation, which is close to 1 in general. The

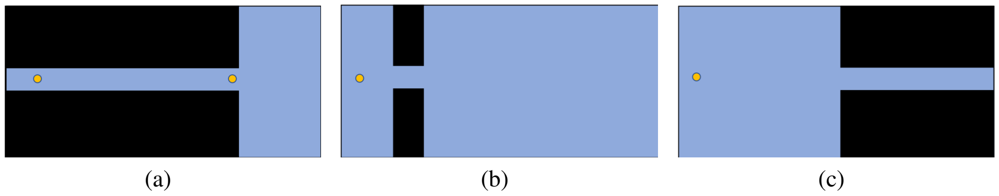


Figure 1: (a) demonstrates of the defect of local uncertainty. (b) and (c) show the defect of using a large horizon for global uncertainty.

large horizon might make the policy think too far ahead in the future, ignoring dangers nearby. The orange points in Fig.1(b) and Fig.1(c) have the same global uncertainty when using Q-value's discount factor. However, the agent at the orange point in Fig.1(b) should be more conservative than in Fig.1(c). Thus, a global uncertainty estimator with a flexible horizon is required.

In this paper, we analyze error propagation for policy iteration in offline RL and propose pessimistic policy iteration based on a global uncertainty quantifier to bound the sub-optimality gap of the trained policy's value function. The global uncertainty is estimated independently in Bellman-style to allow uncertainty propagation. We instantiate pessimistic policy iteration on a policy-restriction baseline to update the policy according to uncertainty. Also, Q-restriction methods such as CQL (Kumar et al., 2020) could be augmented with global uncertainty to enable a malleable penalty. In addition, the global uncertainty estimator can have a different horizon with Q-value evaluation, which could have flexible attention in the time domain to allow regional uncertainty. We show that a smaller horizon will also bring a tighter generalization bound. The experiment on "Go Through the Forest" shows that global uncertainty helps identify OOD regions, and choosing a moderate horizon is important. The experimental results demonstrate that our proposed method outperforms baselines on most D4RL tasks (Fu et al., 2020) and is robust to the scale of the constraint.

2 Related works

Offline RL. Offline RL methods aim to learn a policy when presented with a static dataset (Ernst et al., 2005; Lange et al., 2012; Levine et al., 2020). In this setting, standard off-policy methods (Mnih et al., 2015; Lillicrap et al., 2015) generally exhibit undesirable performance since they have no chance to correct the OOD actions' Q-values (Fujimoto et al., 2019). Recently, various methods have been proposed to tackle the OOD issue (Agarwal et al., 2020; Buckman et al., 2020; Ghasemipour et al., 2021). BEAR (Kumar et al., 2019) allows the trained policy to deviate from the dataset within a tolerance threshold and introduces a support-set constraint. CQL (Kumar et al., 2020) learns a lower bound for the policy's value function by minimizing OOD actions' Q-values. Many works learn the policy according to policy iteration (Fujimoto & Gu, 2021; Wu et al., 2019) but there is still a lack of analysis on the error propagation of policy iteration for offline RL. Our method explores the sub-optimality gap for offline policy iteration and restricts the Q-value error under the trained policy's distribution. On the other hand, many previous methods keep the pessimism level to be consistent during training and ignore the OOD degree of the policy, which might hurt the policy's generalization ability. By contrast, our method adjusts the constraint according to the policy's uncertainty, which is adaptive and could discover well-behaved actions near the dataset.

Uncertainty Estimation. Our proposed method is in line with uncertainty estimation for offline RL. UWAC (Wu et al., 2021) takes advantage of MC-dropout to estimate local uncertainty and reweights the RL objective with uncertainty. But UWAC only focuses on the current transition and ignores the future trajectory's uncertainty. SAC-N (An et al., 2021) learns hundreds of Q-networks and takes the minimum of the ensemble as the Bellman target to penalize the OOD actions. It implicitly calculates global uncertainty and embeds it into Q-value. It requires a large number of Q-value networks, which is computationally inefficient. PBRL (Bai et al., 2022) takes the standard deviation of a Q-value ensemble as a penalty for Q-value estimation. For SAC-N and PBRL, the global uncertainty evaluation is mixed with the Q-value estimation and shares the same horizon with the latter task, which limits its flexibility and cannot be applied to policy-restriction methods. Our proposed method learns independent global uncertainty, which could be introduced to policy-restriction methods and have a different horizon with the Q-value evaluation for flexible attention.

3 Background

RL. We consider the standard reinforcement learning setting, which is always formalized with a Markov decision process M, with the state space S, the action space A, the transition function P, the reward function R, and the discount factor γ (Sutton & Barto, 2018). In reinforcement learning, an agent starts from a state s and executes an action a at each time step. By interacting with the environment, the agent observes a next state s' and receives a reward r. The aim of reinforcement learning is to learn a policy π to receive accumulative rewards as much as possible:$\max_\pi \ \mathbb{E}_\pi[\sum_{t=1}^{\infty}\gamma^{t-1}r_t]$.

When an agent takes an action a at state s and following a policy π, the expectation of the cumulative rewards is defined as Q-value function: $Q^\pi(s,a)=\mathbb{E}_\pi[\sum_{t=1}^{\infty}\gamma^{t-1}r_t|s,a]$. The value function of a given state s is defined as $V^\pi(s)=\mathbb{E}_{a\sim\pi}[Q(s,a)]$. In general, $\hat{Q}$ and $\hat{V}$ are used to denote the estimated Q-value function and value function, respectively. The Q-learning algorithm updates the Q-value function according to Bellman operator $\mathcal{T}$ as: $\mathcal{T}Q(s,a):=r+\gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[\max_{a'}Q(s',a')]$.

Compared with Q-learning methods, actor-critic methods employ a stochastic policy $\pi(a|s)$ or a deterministic policy $\pi(s)$ to sample actions and update via policy iteration. We use $Q(s,\pi)$ to denote $\mathbb{E}_{a\sim\pi(\cdot|s)}Q(s,a)$ when the policy is stochastic.The update rule of Q-value evaluation is $\mathcal{T}^\pi Q = r+\gamma P^\pi Q$, where $P^\pi Q$ is the expectation of Q-value with respect to transition matrix P and policy π. Since it is impossible to enumerate all states and actions, the Q-value function is generally updated using an empirical Bellman operator $\hat{\mathcal{T}}^\pi$. Recently, DDPG (Lillicrap et al., 2016) employs deep neural networks to approximate the true Q-value function and a deterministic policy. Based on the neural network, the Q-value loss function is defined as:

$$\mathbb{E}_{s,a,r,s'\sim\mathcal{D}}[Q(s,a;\theta)-(r+\gamma(Q(s',\pi(s';\zeta);\theta))]^2, \tag{1}$$

where $\mathcal{D}$ is a replay buffer containing the collected samples, θ is the parameter of the Q-value network, and ζ is the parameter of the policy. In this manner, the policy can be updated by:

$$\max_\zeta Q(s,\pi(s;\zeta);\theta). \tag{2}$$

Offline RL. Offline RL aims to learn a policy given a static dataset $\mathcal{D}$ without interaction with the environment. The behavior policy is denoted as π_β. The standard reinforcement learning algorithms might derive a policy that generates OOD actions when applied in this setting. They have no chance to correct the Q-values of these actions, thus leading to failure. CQL (Kumar et al., 2020) proposes to minimize the Q-values of OOD state-action pairs. The Q-network is updated as:

$$\min_Q \quad \alpha\mathbb{E}_{s\sim\mathcal{D}}[Q(s,\pi)]+\mathbb{E}_{s,a,r,s'\sim\mathcal{D}}\left[\left(Q(s,a)-\hat{\mathcal{T}}^\pi Q(s,a)\right)^2\right],$$

where α is a hyperparameter for controlling the degree of conservativeness with respect to the Bellman update. TD3BC (Fujimoto & Gu, 2021) builds on top of TD3 (Fujimoto et al., 2018). It mainly focuses on plugging a behavior cloning regularization term into the policy training in TD3. The objective of the policy is

$$\max_\pi \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[Q(s,\pi(s))-\nu(\pi(s)-a)^2\right],$$

where the behavior cloning scale ν is used to make a balance between reinforcement learning and imitation learning objectives.

4 Pessimistic policy iteration

For offline RL, some works analyze and discuss value iteration, such as BEAR (Kumar et al., 2019) and PEVI (Jin et al., 2021). However, for continuous action space, it is infeasible to apply value iteration since enumerating all actions and selecting the target action which has the highest Q-value are challenging. Generally, a policy is always introduced to this setting, such as TD3 (Fujimoto et al., 2018) and SAC (Haarnoja et al., 2018), which conduct policy iteration. During policy iteration, off-policy evaluation will introduce value error due to distribution shift, and policy improvement might choose overestimated actions. In offline RL, the Q-value error will increase during training unless a strong policy constraint is enforced (Brandfonbrener et al., 2021). But a strong constraint might hurt the policy's generalization ability.

First, we discuss general policy iteration within Π: $\max_{\pi\in\Pi}\mathbb{E}_{a\sim\pi(\cdot|s)}Q(s,a)$, where Π is a set of constrained policies. In offline RL, one can specify Π as the policy set: $\Pi = \{\pi \mid \pi(a|s) = 0 \text{ whenever } \pi_\beta(a|s) < \epsilon\}$. We denote the optimal policy in Π as $\pi^{\#}$. The distribution-constrained Bellman operator (Kumar et al., 2019) is defined as $\mathcal{T}^{\Pi}Q(s,a) \overset{\text{def}}{=} \mathbb{E}\left[R(s,a) + \gamma\mathbb{E}_{P(s'|s,a)}\left[\max_{\pi\in\Pi}\mathbb{E}_{\pi}Q(s',a')\right]\right]$, and the sub-optimality constant is defined as $\alpha(\Pi) = \max_{s,a}\left|\mathcal{T}^{\Pi}Q^*(s,a) - \mathcal{T}Q^*(s,a)\right|$.

Theorem 1. *Execute policy evaluation with $\mathcal{T}^{\pi}$ and policy improvement with $\max_{\pi\in\Pi}\mathbb{E}_{a\sim\pi(\cdot|s)}Q(s,a)$. For any state distribution μ, assume that we have $||V^k - V^{\pi_k}||_{\mu_k} \leq \delta$, where V_k is the estimated value function of the k-th policy iteration and V^{π_k} is the true value function of policy π_k, and $\mu_k = \mu\frac{(1-\gamma)^2}{2}\left(I-\gamma P^{\pi^{\#}}\right)^{-1}\left[P^{\pi_{k+1}}\left(I-\gamma P^{\pi_{k+1}}\right)^{-1}\left(I+\gamma P^{\pi_k}\right) + P^{\pi^{\#}}\right]$. It follows that*

$$\lim_{k\to\infty}||V^k - V^*||_{\mu} \leq \frac{1}{1-\gamma}\alpha(\Pi) + \frac{2\gamma\delta}{(1-\gamma)^2}. \tag{3}$$

The sub-optimality is bounded by $\alpha(\Pi)$ and δ, where $\alpha(\Pi)$ denotes the distance between the optimal policy and the constrained policy set Π and δ is the error brought by policy evaluation. Note that different from BEAR (Kumar et al., 2019), which considers value iteration, Theorem 1 bounds the error between the value function of the trained policy and the optimal policy for policy iteration.

The sub-optimality gap in Eq.3 depends on the bound of $V^k - V^{\pi_k}$ under distribution μ_k. For commonly used methods, the Q-value function is directly adopted to conduct policy evaluation instead of a value function. The objective can be approximately translated into bounding $\mathbb{E}_{s\sim\mu_k,a\sim\pi_k(\cdot|s)}|Q^k(s,a) - Q^{\pi_k}(s,a)|$, which is unknown and might be large.

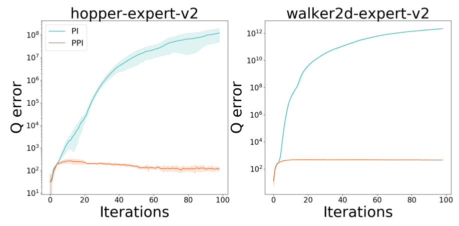


Figure 2: Value error $|Q^k(s,a) - Q^{\pi_k}(s,a)|$ when a weak constraint is applied. PI denotes general policy iteration and PPI denotes pessimistic policy iteration.

We introduce the following definition for policy evaluation which is related to the theoretical study on provable efficiency in offline RL.

Definition 2. *(ξ-Uncertainty Quantifier (Jin et al., 2021)). For a penalty function $\omega(\cdot,\cdot) : S\times A \to \mathbb{R}$, if it holds that $P(|\mathcal{T}^{\pi}\hat{Q}(s,a) - \hat{\mathcal{T}}^{\pi}\hat{Q}(s,a)| \leq \omega(s,a)) \geq 1-\xi$ for all $(s,a)\in S\times A$, where $\hat{Q}$ is an estimated action-value function, $\mathcal{T}^{\pi}$ is the Bellman operator, and $\hat{\mathcal{T}}^{\pi}$ is the empirical Bellman operator, then we say the penalty ω is a ξ-uncertainty quantifier.*

We then define the global ξ-uncertainty quantifier for a given policy π which satisfies $\Omega^{\pi}(s,a) = \omega(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}\mathbb{E}_{a'\sim\pi(\cdot|s')}\Omega(s',a')$. It represents the cumulative error for approximated Q-function introduced by the empirical Bellman update. As an alternative to $\mathbb{E}_{s\sim\mu_k,a\sim\pi_k(\cdot|s)}|Q^k(s,a) - Q^{\pi_k}(s,a)|$, we bound a surrogate objective $\mathbb{E}_{s\sim\mu_k,a\sim\pi'_k(\cdot|s)}|Q^k(s,a) - Q^{\pi_k}(s,a)|$, where $\pi'_k(a|s) = \frac{\pi_k(a|s)}{\Omega^{\pi_k}(s,a)Z(s)}$, and $Z(s)$ is the normalization term: $Z(s) = \int\frac{\pi_k(a|s)}{\Omega^{\pi_k}(s,a)}da$. For the surrogate objective, the following result holds,

Theorem 3. *Assume that there exists a constant Z such that for all s, $Z(s)\geq Z$. For the penalty function which is a 0-uncertainty quantifier, the following result holds,*

$$\mathbb{E}_{s\sim\mu_k,a\sim\pi'_k(\cdot|s)}|Q^k(s,a) - Q^{\pi_k}(s,a)| \leq \frac{1}{Z}. \tag{4}$$

Theorem 3 shows that for the policy distribution weighted by the global uncertainty quantifier, we could bound the gap between the empirical Q-value function and the true expected Q-value function, which approximates δ in Theorem 1.

We now introduce pessimistic policy iteration, which reformulates the policy improvement step in general policy iteration. The objective translates into $\max_{\pi\in\Pi}\mathbb{E}_{a\sim\pi(\cdot|s)}\frac{Q(s,a)}{\Omega^{\pi}(s,a)}$. It expects the policy to have a high expected return and low accumulative Q-value error. Fig.2 shows that the Q-value error for pessimistic policy iteration is far smaller than policy iteration when a weak constraint is applied. It indicates that pessimistic policy iteration relies less on strong constraints and might have better generalization ability. Note that it requires the value of $|\mathcal{T}^\pi \hat{Q}(s,a) - \hat{\mathcal{T}}^\pi \hat{Q}(s,a)|$, which is unavailable. In the following section, we introduce an approximation to $\Omega(s,a)$ and propose a practical implementation of our method.

5 Global uncertainty estimation

In this section, we introduce a global uncertainty estimator corresponding to $\Omega(s,a)$. Then we incorporate it into a typical policy-restriction method to realize pessimistic policy iteration and introduce it to a Q-restriction method in the next section.

5.1 Estimation of global uncertainty quantifier

For a linear MDP whose transition matrix and reward function are linear, it has been shown that $\omega(s,a) = \beta \left[\phi(s,a)^\top \Lambda^{-1} \phi(s,a)\right]^{1/2}$ is a ξ-uncertainty quantifier, where $\Lambda = \sum_{i=1} \phi(s_i,a_i)\phi(s_i,a_i)^\top + \eta \cdot \mathbf{I}$, $\phi(\cdot,\cdot)$ is the feature function for states and actions, and β is a constant (Jin et al., 2021).

Now we estimate $\omega(s,a)$ by maintaining N Q-networks and define the local uncertainty of (s,a) as

$$u(s,a) = \text{std}(Q^{\text{ensemble}}(s,a)) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Q_i(s,a) - \bar{Q}(s,a))^2}, \tag{5}$$

where $\bar{Q}(s,a) = \frac{1}{N}\sum_{i=1}^{N} Q_i(s,a)$. We point out that $\beta u(s,a)$ is a parametric estimation to $\beta \left[\phi(s,a)^\top \Lambda^{-1} \phi(s,a)\right]^{1/2}$ in Linear MDP (Bai et al., 2022). In addition, Q-ensemble produces desirable uncertainty estimation (Ciosek et al., 2019). Regarding practical uncertainty estimation, another choice is to replace the Q-ensemble with a dynamics ensemble (Yu et al., 2020). However, the dynamics ensemble will consider unimportant states, which might be task-irrelevant.

Then we define the global uncertainty of (s,a) as $U(s,a)$, which is updated by the Bellman operator,

$$U(s,a) \leftarrow u(s,a) + \lambda \mathbb{E}_{s' \sim P(\cdot|s,a)} \mathbb{E}_{a' \sim \pi(\cdot|s')} U(s',a'), \tag{6}$$

where λ is the discount factor for uncertainty estimation. We denote the true expected value of uncertainty for policy π as $U^{\pi,\lambda}(s,a)$. When $\lambda = \gamma$, it is actually an estimation of $\Omega(s,a)$.

5.2 View of uncertainty estimation and horizon

Previous works also utilize uncertainty estimation for offline RL (Wu et al., 2021; An et al., 2021). Unlike these methods, our method separates uncertainty learning from Q-value evaluation. In this manner, uncertainty learning can have a different horizon with Q-value evaluation, which is impossible for previous works in offline RL.

We have shown that choosing $\lambda = \gamma$ could bound the sub-optimality gap. However, a smaller λ corresponds to regional uncertainty and could identify the risk of executing OOD actions in recent steps because nearby OOD actions need to be corrected immediately. Otherwise, the policy might diverge from the dataset. Note that $U(s,a)$ differs from Uncertainty Bellman Equation (O'Donoghue et al., 2018) for exploration, which calculates the posterior variance for Q-value function and has a horizon γ^2. Different from online RL, which requires consistent exploration to reach remote regions in the future, offline RL demands the policy to stay close to the dataset. On the other hand, when $\lambda = 0$, it resembles UWAC (Wu et al., 2021), whose uncertainty is calculated based on the Q-value of the current state-action pair. However, the Q-value function updated by the Bellman equation cannot propagate uncertainty (Osband et al., 2018). Therefore, a moderate value for λ is important for offline RL. Note that $\lambda = 0$ or $\lambda = \gamma$ also might be favorable for some settings.

We now show the validity of choosing a smaller λ from the viewpoint of multi-task learning. To simplify analysis, we consider maximizing the expected return while minimizing cumulative uncertainty: $\max_\pi \mathbb{E}_{s\sim\mathcal{D}}[Q(s,\pi) - U(s,\pi)]$, which corresponds to PBRL (Bai et al., 2022). Denote $\bar{u}(s,a) = -(u(s,a) - \max(u(\cdot,\cdot)))$ and the maximum of $\bar{u}(s,a)$ as $U_{\max}$. We rewrite the MDP as $M = (S, A, P, R, \gamma, \bar{u}, \gamma)$, where the first γ is for reward function and the second γ is for uncertainty estimation. Let $\hat{M} = (S, A, \hat{P}, \hat{R}, \gamma, \hat{u}, \lambda)$ be an estimated MDP whose transition matrix, reward function, and uncertainty function are estimated by the offline dataset with the size n. The discount factor of uncertainty estimation for $\hat{M}$ is λ. If we learn an optimal policy in $\hat{M}$, the following result holds.

Proposition 4. *Consider the task* $\max\sum_{t=1}^{\infty}\gamma^{t-1}r_t + \gamma^{t-1}\bar{u}_t$ *under* M *and the task* $\max\sum_{t=1}^{\infty}\gamma^{t-1}\hat{r}_t + \lambda^{t-1}\hat{u}_t$ *under* $\hat{M}$*. The optimal policies for the two tasks are* π^*_M *and* $\pi^*_{\hat{M}}$*, respectively. With the probability of at least* $1-\delta$*, the optimal policy for the latter task has a bound*

$$\|V_M^{\pi^*_M} - V_M^{\pi^*_{\hat{M}}}\| \leq \frac{\gamma-\lambda}{(1-\gamma)(1-\lambda)}U_{\max} + \left[\frac{2R_{\max}}{(1-\gamma)^2} + \frac{2U_{\max}}{(1-\lambda)^2}\right]\sqrt{\frac{1}{2n}\log\frac{4|S||A|\,|\Pi|}{\delta}},$$

where $V_M^{\pi^*_M}$ *is the expected value function of* π^*_M *evaluated under* M*,* $V_M^{\pi^*_{\hat{M}}}$ *is the expected value function of* $\pi^*_{\hat{M}}$ *evaluated under* M*, and* $|\Pi|$ *is the policy class complexity.*

The first term is in negative relation to λ, which encourages λ to be large. But the second term requires a small λ to tighten the bound. Thus, a moderate λ should be chosen to balance the two terms.

6 METHOD

6.1 TD3BC-U

Now we propose an instance of pessimistic policy iteration based on the global uncertainty estimator. As Section 4 shows, the policy improvement objective is $\max_{\pi\in\Pi}\mathbb{E}_{a\sim\pi(\cdot|s)}\frac{Q(s,a)}{\Omega^\pi(s,a)}$. From the practical side, the constrained policy set Π could be realized by adding a regularization term to the policy, such as behavior cloning in TD3BC (Fujimoto & Gu, 2021) and MMD loss in BEAR (Kumar et al., 2019). We select TD3BC for its simplicity and propose an uncertainty-based variant according to pessimistic policy iteration. We introduce the global uncertainty estimator to TD3BC by restricting the actor according to uncertainty: $\max_\pi \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\frac{Q(s,\pi(s))}{U(s,\pi(s))} - \nu(\pi(s)-a)^2\right]$.

When $U(s,\pi(s))$ approaches 0, the scale of the gradient for the first term approaches infinity. To stabilize the training process, we multiply the objective with $U(s,\pi(s))$ and rewrite the objective as:

$$\max_\pi \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[Q(s,\pi(s)) - \nu U(s,\pi(s))(\pi(s)-a)^2\right]. \tag{7}$$

It demands the policy to be more conservative when the global uncertainty is high at state s. Otherwise, it relaxes the constraint so that the policy could update more according to the Q-value. We refer to this algorithm as TD3BC-U.

6.2 CQL-U

Besides policy-restriction methods, the proposed global uncertainty estimator could also boost Q-restriction methods. CQL is a Q-restriction method that penalizes OOD actions' Q-values. We introduce the global uncertainty estimator to CQL by restricting the critic according to uncertainty,

$$\min_Q \mathrm{E}_{(s,a,r,s')\sim\mathcal{D}}\,\alpha U(s,\pi)Q(s,\pi) + \left[\Big(Q(s,a) - \hat{\mathcal{T}}^\pi Q(s,a)\Big)^2\right]. \tag{8}$$

We refer to this algorithm as CQL-U.

6.3 PRACTICAL IMPLEMENTATION

For both algorithms, we maintain 8 Q-networks and a policy network. For the training of the uncertainty network, we maintain a target network to stabilize the training process.

For TD3BC-U, the behavior cloning scale ν is set to 1. Note that the pessimism level ν influences the policy's performance a lot (Fujimoto & Gu, 2021). Considering that the average magnitude of $U(s,a)$ will influence the overall conservativeness, which could be regarded as choosing a different ν, we rescale $U(s,a)$ at each training step to make sure the average uncertainty over a batch is 1. By doing so, the average pessimism magnitude of the training process is still ν. Then we could make a fair comparison between our proposed method and the baseline algorithm. The algorithm is described in Alg.1.

For CQL-U, the hyperparameter α is set to 2. We use a deterministic version of CQL where a deterministic policy is applied. Thus, pessimism is introduced by only penalizing the Q-value of a single action for a state, which reduces the computational complexity of the original CQL. The magnitude of U is also rescaled as the TD3BC-U part. The algorithm is described in Alg.2.

Algorithm 1 TD3BC-U

Input: offline dataset $\mathcal{D}$, update iterations t_{max} for policy and Q-value networks, hyperparameter ν.
Parameter: uncertainty network U, policy network π, Q-networks $Q_1, Q_2, ..., Q_N$ and target networks.
Output: learned policy network π.

```
1:  Initialize the uncertainty network, policy network,
    and Q-networks.
2:  while $t < t_{max}$ do
3:      Sample a mini-batch of samples from $\mathcal{D}$.
4:      Calculate local uncertainty according to Eq.5
5:      Update uncertainty network according to Eq.6
6:      Calculate $U(s, \pi(s))$ for current policy.
7:      Update the Q-networks according to Eq.1.
8:      Update the policy network according to Eq.7.
9:      Update the target networks via soft update.
10: end while
```

Algorithm 2 CQL-U

Input: offline dataset $\mathcal{D}$, update iterations t_{max} for policy and Q-value networks, hyperparameter α.
Parameter: uncertainty network U, policy network π, Q-networks $Q_1, Q_2, ..., Q_N$ and target networks.
Output: learned policy network π.

```
1:  Initialize the uncertainty network, policy network,
    and Q-networks.
2:  while $t < t_{max}$ do
3:      Sample a mini-batch of samples from $\mathcal{D}$.
4:      Calculate local uncertainty according to Eq.5
5:      Update uncertainty network according to Eq.6
6:      Calculate $U(s, \pi)$ for current policy.
7:      Update the Q-networks according to Eq.8.
8:      Update the policy network according to Q-
        network $\max_\pi \mathbb{E}_{a\sim\pi} Q(s,a)$.
9:      Update the target networks via soft update.
10: end while
```

7 Experiment

In this section, we conduct several experiments to show the effectiveness of our proposed method.

7.1 Uncertainty estimation

To illustrate the efficiency of our proposed uncertainty estimator, we first provide an example called "Go Through the Forest", where we show local ($\lambda = 0$), regional ($\lambda = 0.8$), and global ($\lambda = 0.99$) uncertainty in 2D heat maps. We design this "Go Through the Forest" environment in Fig.3, which involves a single agent with 2D observation space and 1D action space. There are three kinds of areas, "Aerobic area"(green), "Toxic area"(red), and "Ordinary area"(white). The goal of the agent is to reach the opposite side ($x = 1$). When getting into the "Aerobic area" for the first time, the agent will receive a reward of 1 and when "Toxic area" that of -1. We consider two environment settings. Setting $\boldsymbol{A}$: Staring from the point $(0, 0)$, at each time step, the agent walks ahead in a selected direction under a fixed step length $L = 0.04$. For simplicity, the agent is restricted to select those directions in which the abscissa x becomes larger. Setting $\boldsymbol{B}$: Staring from the point $(0, y)$ (y is randomly chosen from [-0.5, 0.5]), the y-axis of the agent decays exponentially in an episode. Concretely, $y_t = \beta^t y_0$ (β is set to 0.98). At each time step, the agent walks in the x-axis direction under a selected step length $L \in [0, 0.01]$. For each setting, we generate a random offline

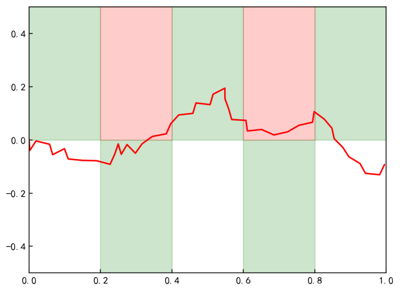
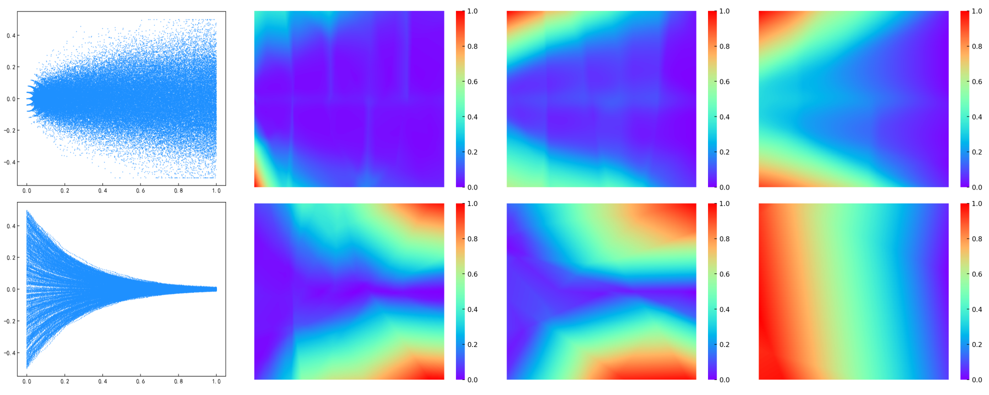


Figure 3: The designed "Go Through the Forest" environment. The red line means one trajectory from the start point to the end in the Setting $\boldsymbol{A}$.



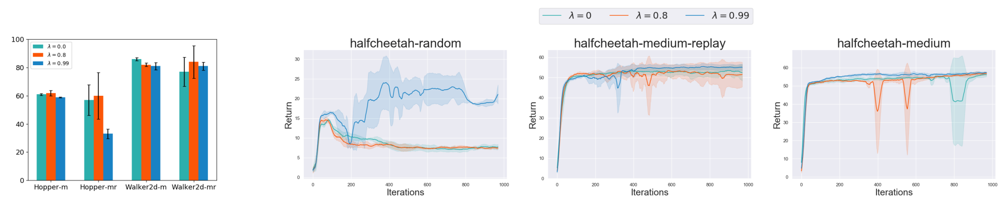
(a) s distribution in D (b) $\lambda = 0$ (c) $\lambda = 0.8$ (d) $\lambda = 0.99$

Figure 4: Visualization of the "Go Through the Forest" example. (a) State distributions in the random offline dataset. (b) $\lambda = 0$ means local uncertainty. (c) $\lambda = 0.8$ means regional uncertainty. (d) $\lambda = 0.99$ means global uncertainty. Note: we visualize the uncertainty in 2D heat maps after training 1e6 steps respectively. The uncertainty is normalized to 0-1 range.

dataset (size of 100,000) of an agent taking random actions. As shown in Fig.4(a), the first and second rows correspond to setting $\boldsymbol{A}$ and setting $\boldsymbol{B}$, respectively.

Due to the heterogeneous state-action distribution in the whole 2D observation space, there exist lots of out-of-distribution states and actions, especially those around the edge of the environment. Under these settings, local uncertainty ($\lambda = 0$), regional uncertainty ($\lambda = 0.8$), and global uncertainty ($\lambda = 0.99$) are visualized in 2D heat maps (Fig. 4(b), 4(c), and 4(d) respectively) after training 1e6 steps. Our method can detect the OOD regions around which the dataset contains much less data than those in distribution.

What's more, we illustrate that the detectability of local uncertainty ($\lambda = 0$) or global uncertainty ($\lambda = 0.99$) can be too extreme to work well. For the setting $\boldsymbol{A}$, using only local uncertainty can not fully detect the OOD state-action pairs. Using global uncertainty has a stronger perception of the environment than local uncertainty. However, setting $\boldsymbol{B}$ demonstrates using global uncertainty may overestimate the uncertainty under some circumstances, especially when the state-action distribution is very heterogeneous, thus hurting OOD detectability. Taking both settings into account, regional uncertainty is milder and more robust than the other two types, which means it can be applied to a broader range of the dataset and act as a more efficient tool to detect the OOD regions.



(a) TD3BC-U (b) CQL-U

Figure 5: Varying the scale of λ. Mean and standard deviation are plotted over six seeds.

7.2 D4RL RESULTS

To evaluate the validity of our proposed methods in high-dimensional settings, we perform experiments using the MuJoCo control suits in D4RL benchmarks (Fu et al., 2020). We compare our proposed methods to TD3BC and CQL algorithms. Both TD3BC and TD3BC-U use the same number of Q-networks and the same hyperparameter ν. Also, both CQL and CQL-U use the same number of Q-networks and the same hyperparameter α. The results are shown in Table.1. TD3BC-U performs better than TD3BC on most tasks and CQL-U outperforms CQL on most tasks. It can be shown that pessimistic policy iteration, which introduces global uncertainty estimation, boosts the

Table 1: Results on offline MuJoCo tasks. The results are averaged over six seeds.

Task Name	TD3BC	TD3BC-U	CQL	CQL-U
HalfCheetah-random	10.9 ± 0.6	$\mathbf{17.5 \pm 0.8}$	17.9 ± 0.7	$\mathbf{21.7 \pm 3.9}$
HalfCheetah-medium	48.6 ± 0.1	$\mathbf{49.9 \pm 0.6}$	56.7 ± 0.6	$\mathbf{58.8 \pm 0.5}$
HalfCheetah-medium-replay	44.1 ± 0.4	$\mathbf{45.2 \pm 0.2}$	51.9 ± 0.2	$\mathbf{54.6 \pm 2.3}$
HalfCheetah-medium-expert	87.3 ± 1.8	72.7 ± 6.3	37.3 ± 6.5	$\mathbf{38.4 \pm 10.3}$
Hopper-random	29.0 ± 1.4	$\mathbf{31.4 \pm 1.2}$	9.3 ± 2.2	$\mathbf{31.1 \pm 0.3}$
Hopper-medium	57.1 ± 1.6	$\mathbf{62.3 \pm 1.9}$	62.8 ± 18.7	$\mathbf{75.4 \pm 8.3}$
Hopper-medium-replay	47.2 ± 5.2	$\mathbf{60.0 \pm 16.5}$	100.7 ± 0.9	$\mathbf{101.2} \pm 0.5$
Hopper-medium-expert	104.8 ± 1.3	82.6 ± 9.1	38.7 ± 10.1	22.5 ± 9.6
Walker2d-random	0.1 ± 0.1	0.1 ± 0.0	0.0 ± 0.1	0.0 ± 0.3
Walker2d-medium	79.3 ± 1.6	$\mathbf{82.4 \pm 1.1}$	85.0 ± 7.5	$\mathbf{92.8 \pm 3.4}$
Walker2d-medium-replay	61.0 ± 11.0	$\mathbf{84.0} \pm 11.4$	96.4 ± 0.3	$\mathbf{99.7 \pm 0.9}$
Walker2d-medium-expert	110.5 ± 0.1	$\mathbf{113.1 \pm 0.4}$	108.9 ± 3.0	$\mathbf{111.0 \pm 5.7}$

performance of the baseline algorithms. Note that in some tasks, the improvement is not large. The reason is that the pessimism scale is the most important factor which affects the performance. We use the same pessimism scale to make a fair comparison.

Discount factor for uncertainty estimation. To show what impact the uncertainty discount factor λ has on our method, we test $\lambda \in \{0, 0.8, 0.99\}$ for TD3BC-U and CQL-U. $\lambda = 0$ denotes only considering the local uncertainty and $\lambda = 0.99$ denotes the case global uncertainty shares the same horizon with Q-value estimation. The results are shown in Fig.5. For TD3BC-U, choosing $\lambda = 0.8$ is better than $\lambda = 0.99$. Choosing $\lambda = 0.99$ is more appropriate for CQL-U. The results indicate that a moderate horizon for global uncertainty estimation is important for performance.

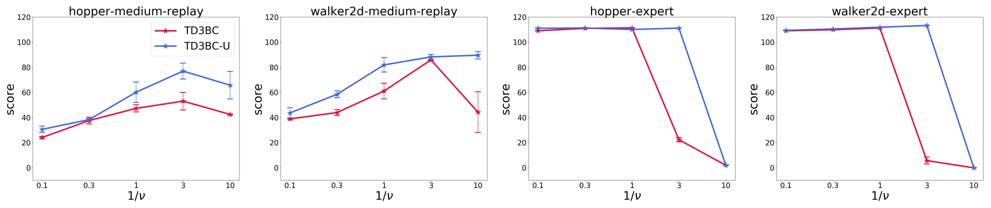


Figure 6: TD3BC-U is more robust to the scale of $1/\nu$ than TD3BC.

Robustness. It is known that the behavior cloning scale ν impacts the trained policy a lot (Fujimoto & Gu, 2021). A small ν might lead to inadequate pessimism and failure, while a large ν may restrict the policy too much and hurt the policy's performance. We show that the pessimistic policy iteration is robust to the scale of ν. Fig.6 shows the performances of TD3BC-U and TD3BC when varying $1/\nu$. As $1/\nu$ increases, the constraint is weaker, and TD3BC suffers a performance drop. When $1/\nu = 3$, TD3BC fails on hopper-expert and walker2d-expert tasks, which have narrow distributions. TD3BC-U is more robust than TD3BC in this setting. Note that when $1/\nu = 10$, the performance TD3BC-U also drops because the constraint is too weak to restrict the policy to be near the behavior policy.

8 Conclusion

In this paper, we propose pessimistic policy iteration to restrict the Q-evaluation error and to bound the sub-optimality gap of the trained policy's value function. We realize pessimistic policy iteration by learning a global uncertainty and introducing it to a policy-restriction method. We also build a Q-restriction variant based on global uncertainty to allow flexible attention. The proposed method boosts the baselines' performance and it is robust to the coefficient of the constraint. We find that it is important to find a moderate horizon for the uncertainty estimation. For future work, we expect to find a better local uncertainty estimator than the Q-ensemble, such as random network distillation (Burda et al., 2019).

REFERENCES

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

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34, 2021.

Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhi-Hong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Y4cs1Z3HnqL`.

David Brandfonbrener, William F Whitney, Rajesh Ranganath, and Joan Bruna. Offline RL without off-policy evaluation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=LU687itn08w`.

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

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=H1lJJnR5Ym`.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 456–464, 2019.

Ching-An Cheng, Tengyang Xie, Nan Jiang, and Alekh Agarwal. Adversarially trained actor critic for offline reinforcement learning. In *ICML*, 2022.

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. *Advances in Neural Information Processing Systems*, 32, 2019.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

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

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 20132–20145, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/a8166da05c5a094f7dc03724b41886e5-Abstract.html`.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

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

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. 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.

Omer Gottesman, Fredrik Johansson, Joshua Meier, Jack Dent, Donghun Lee, Srivatsan Srinivasan, Linying Zhang, Yi Ding, David Wihl, Xuefeng Peng, et al. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint arXiv:1805.12298*, 2018.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Nan Jiang, Alex Kulesza, Satinder Singh, and Richard Lewis. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1181–1189. Citeseer, 2015.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5084–5096. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/jin21e.html.

Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8248–8254. IEEE, 2019.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pp. 45–73. Springer, 2012.

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

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1509.02971.

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. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Rémi Munos. Error bounds for approximate policy iteration. In *ICML*, volume 3, pp. 560–567, 2003.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Vlad Mnih. The uncertainty Bellman equation and exploration. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3839–3848. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/odonoghue18a.html.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Lu Wang, Wei Zhang, Xiaofeng He, and Hongyuan Zha. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2447–2456, 2018.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. *arXiv preprint arXiv:2105.08140*, 2021.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

9 Proofs

9.1 Error bound for approximate policy iteration

Lemma 5. *For any state distribution μ, run approximate policy iteration. Assuming that $\mu_k = \mu\frac{(1-\gamma)^2}{2}\left(I-\gamma P^{\pi^*}\right)^{-1}\left[P^{\pi_{k+1}}\left(I-\gamma P^{\pi_{k+1}}\right)^{-1}\left(I+\gamma P^{\pi_k}\right)+P^{\pi^*}\right]$, the error bound for approximate policy iteration without policy improvement error follows:*

$$\lim_{k\to\infty}\sup_k ||V^k - V^*||_\mu \le \lim_{k\to\infty}\sup_k \frac{2\gamma||V^k - V^{\pi_k}||_{\mu_k}}{(1-\gamma)^2}$$

Proof. Seen in Munos (2003). □

9.2 Proof of Theorem 1

Proof. First we define the fixed point of $\mathcal{T}^\Pi$ as V^Π. We have

$$\begin{aligned}||V^k - V^*||_\mu &= ||V^k - V^\Pi + V^\Pi - V^*||_\mu \\ &\le \underbrace{||V^k - V^\Pi||_\mu}_{L_1} + \underbrace{||V^\Pi - V^*||_\mu}_{L_2}\end{aligned}$$

For L_2, the following inequality holds:

$$\begin{aligned}\left\|V^\Pi - V^*\right\|_\mu &\le \left\|V^\Pi - V^*\right\|_\infty \\ &= \left\|\mathcal{T}^\Pi V^\Pi - \mathcal{T}V^*\right\|_\infty \\ &\le \left\|\mathcal{T}^\Pi V^\Pi - \mathcal{T}^\Pi V^*\right\|_\infty + \left\|\mathcal{T}^\Pi V^* - \mathcal{T}V^*\right\|_\infty \\ &\le \gamma\left\|V^\Pi - V^*\right\|_\infty + \alpha(\Pi)\end{aligned}$$

Thus, we have $L_2 \le \frac{\alpha(\Pi)}{1-\gamma}$.

For L_1, it is straightforward to apply Lemma 5 by replacing V^* with V^Π. Thus, we have

$$\begin{aligned}L_1 &\le \lim_{k\to\infty}\sup_k \frac{2\gamma||V^k - V^{\pi_k}||_{\mu_k}}{(1-\gamma)^2}. \\ &\le \frac{2\gamma\delta}{(1-\gamma)^2}\end{aligned}$$

□

9.3 Proof of Theorem 3

Proof. Considering that $\pi'_k(a|s) = \frac{\pi_k(a|s)}{\Omega^{\pi_k}(s,a)Z(s)}$, we have

$$\mathbb{E}_{s\sim\mu_k, a\sim\pi'_k(\cdot|s)}|Q^k(s,a) - Q^{\pi_k}(s,a)| = \mathbb{E}_{s\sim\mu_k, a\sim\pi_k(\cdot|s)}\frac{|Q^k(s,a) - Q^{\pi_k}(s,a)|}{\Omega^{\pi_k}(s,a)Z(s)}. \tag{9}$$

Since Q^k is the result of applying $\hat{T}^{\pi_k}$ repeatedly, it holds that Q^k is the fixed point of $\hat{T}^{\pi_k}$: $\hat{T}^{\pi_k}Q^k = Q^k$, then we have

$$|T^{\pi_k}Q^k(s,a) - Q^k(s,a)| = |T^{\pi_k}Q^k(s,a) - \hat{T}^{\pi_k}Q^k(s,a)|).$$

According to the definition of ξ-uncertainty quantifier, it follows that

$$|T^{\pi_k}Q^k(s,a) - Q^k(s,a)| = |T^{\pi_k}Q^k(s,a) - \hat{T}^{\pi_k}Q^k(s,a)|) \le \omega(s,a).$$

For $|Q^k(s,a) - Q^{\pi_k}(s,a)$ which is the numerator in Eq.9, it follows that

$$\begin{aligned}
|Q^k(s,a) - Q^{\pi_k}(s,a)| &= |Q^k(s,a) - T^{\pi_k}Q^k(s,a) + T^{\pi_k}Q^k(s,a) - \{T^{\pi_k}\}^2 Q^k(s,a) + ... \\
&+ \{T^{\pi_k}\}^{\infty} Q^k(s,a) - Q^{\pi_k}(s,a)| \\
&\leq |Q^k(s,a) - T^{\pi_k}Q^k(s,a| + |T^{\pi_k}Q^k(s,a) - \{T^{\pi_k}\}^2 Q^k(s,a)| + ... \\
&+ |\{T^{\pi_k}\}^{\infty} Q^k(s,a) - Q^{\pi_k}(s,a)| \\
&\leq \omega(s,a) + \gamma \mathbb{E}_{s',a' \sim P^{\pi_k}(\cdot,\cdot|s,a)}[\omega(s',a') + \gamma \mathbb{E}_{s'',a'' \sim P^{\pi_k}(\cdot,\cdot|s',a')} \omega(s'',a'') + ...] \\
&= \mathbb{E}_{\pi_k}[\sum_{t=1}^{\infty} \gamma^{t-1} \omega(s_t,a_t) | (s_1,a_1) = (s,a)] \\
&= \Omega^{\pi_k}(s,a)
\end{aligned}$$

According to the assumption that for all $s, Z(s) \geq Z$, thus we have

$$\mathbb{E}_{s \sim \mu_k, a \sim \pi_k(\cdot|s)} \frac{|Q^k(s,a) - Q^{\pi_k}(s,a)|}{\Omega^{\pi_k}(s,a) Z(s)} \leq \frac{1}{Z}. \tag{10}$$

□

9.4 Lemmas for Proposition 4

Lemma 6. *For MDP $M_1 = (S, A, P, R, \gamma, \bar{u}, \lambda)$, any policy π and $\lambda \leq \gamma$, it holds that*

$$V^{\pi}_{M_1} \leq V^{\pi}_{M} \leq V^{\pi}_{M_1} + \frac{\gamma - \lambda}{(1-\gamma)(1-\lambda)} U_{\max}$$

The proof is similar to Jiang et al. (2015).

Proof.

$$\begin{aligned}
\|V^{\pi}_{M_1} - V^{\pi}_{M}\| &= \|\sum_{t=1}^{\infty} (\lambda^{t-1} - \gamma^{t-1}) \{T^{\pi}\}^{t-1} \bar{u}(s_t,a_t) + \sum_{t=1}^{\infty} (\gamma^{t-1} - \gamma^{t-1}) \{T^{\pi}\}^{t-1} r(s_t,a_t)\| \\
&\leq \sum_{t=1}^{\infty} (\lambda^{t-1} - \gamma^{t-1}) U_{\max} \\
&= \frac{\gamma - \lambda}{(1-\gamma)(1-\lambda)} U_{\max}
\end{aligned}$$

□

Lemma 7. *For MDP $\hat{M}_1 = \hat{M} = (S, A, \hat{P}, \hat{R}, \gamma, \hat{u}, \lambda)$, it holds that with probability at least $1-\delta$,*

$$\|V^{\pi^*_{M_1}}_{M_1} - V^{\pi^*_{\hat{M}_1}}_{M_1}\| \leq \frac{2R_{\max}}{(1-\gamma)^2} \sqrt{\frac{1}{2n} \log \frac{4|S||A|\,|\Pi|}{\delta}} + \frac{2U_{\max}}{(1-\lambda)^2} \sqrt{\frac{1}{2n} \log \frac{4|S||A|\,|\Pi|}{\delta}},$$

where $|\Pi|$ is the policy class complexity.

Proof. It follows that

$$\|V^{\pi^*_{M_1}}_{M_1} - V^{\pi^*_{\hat{M}_1}}_{M_1}\| \leq 2 \max_{\pi} \|V^{\pi}_{M_1} - V^{\pi}_{\hat{M}_1}\| \tag{11}$$

Let Q_1 be the expected action-value function for reward r and Q_2 be the expected action-value function for reward $\bar{u}$.

$$\|Q^{\pi}_{1,M_1} - Q^{\pi}_{1,\hat{M}_1}\| \leq \frac{1}{1-\gamma} \max_{s,a} |\hat{r}(s,a) + \gamma < \hat{P}, V^{\pi}_{1,M_1} > - Q^{\pi}_{1,M_1}(s,a)| \tag{12}$$

$$\|Q^{\pi}_{2,M_1} - Q^{\pi}_{2,\hat{M}_1}\| \leq \frac{1}{1-\lambda} \max_{s,a} |\hat{u}(s,a) + \lambda < \hat{P}, V^{\pi}_{2,M_1} > - Q^{\pi}_{2,M_1}(s,a)| \tag{13}$$

Eq.11, Eq.12, Eq.13 hold according to Lemma 3 and Lemma 4 in Jiang et al. (2015). Note that $\hat{r}(s,a) + \gamma < \hat{P}, V^{\pi}_{1,M_1} >$ is sampled from the distribution which has the mean $Q^{\pi}_{1,M_1}(s,a)$ and the bound $[0, \frac{R_{\max}}{1-\gamma}]$. Also, $\hat{u}(s,a) + \lambda < \hat{P}, V^{\pi}_{2,M_1} >$ is sampled from the distribution which has the mean $Q^{\pi}_{2,M_1}(s,a)$ and the bound $[0, \frac{U_{\max}}{1-\lambda}]$.

$$\begin{aligned}
\|V_{M_1}^{\pi^*_{M_1}} - V_{M_1}^{\pi^*_{\hat{M}_1}}\| &\leq 2\max_{\pi} \|V^{\pi}_{M_1} - V^{\pi}_{\hat{M}_1}\| \\
&\leq 2\max_{\pi} \|Q^{\pi}_{M_1} - Q^{\pi}_{\hat{M}_1}\| \\
&= 2\max_{\pi} \|Q^{\pi}_{1,M_1} + Q^{\pi}_{2,M_1} - Q^{\pi}_{1,\hat{M}_1} - Q^{\pi}_{2,\hat{M}_1}\| \\
&\leq 2\max_{\pi} \|Q^{\pi}_{1,M_1} - Q^{\pi}_{1,\hat{M}_1}\| + 2\max_{\pi} \|Q^{\pi}_{2,M_1} - Q^{\pi}_{2,\hat{M}_1}\| \\
&\leq 2\max_{\pi,s,a} |Q^{\pi}_{1,M_1}(s,a) - Q^{\pi}_{1,\hat{M}_1}(s,a)| + 2\max_{\pi,s,a} |Q^{\pi}_{2,M_1}(s,a) - Q^{\pi}_{2,\hat{M}_1}(s,a)|
\end{aligned}$$

According to Hoeffding inequality and union bound, we have with probability $1-\delta$,

$$\|V_{M_1}^{\pi^*_{M_1}} - V_{M_1}^{\pi^*_{\hat{M}_1}}\| \leq \frac{2R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{4|S||A|\,|\Pi|}{\delta}} + \frac{2U_{\max}}{(1-\lambda)^2}\sqrt{\frac{1}{2n}\log\frac{4|S||A|\,|\Pi|}{\delta}}.$$

□

9.5 Proof for Proposition 4

Proof.

$$V_M^{\pi^*_M}(s) - V_M^{\pi^*_{\hat{M}}}(s) = V_M^{\pi^*_M}(s) - V_{M_1}^{\pi^*_M}(s) + V_{M_1}^{\pi^*_M}(s) - V_M^{\pi^*_{\hat{M}}}(s)$$

According to Lemma 6, it follows that for the first term,

$$V_M^{\pi^*_M}(s) - V_{M_1}^{\pi^*_M}(s) \leq \frac{\gamma - \lambda}{(1-\gamma)(1-\lambda)} U_{\max}$$

For the second term,

$$\begin{aligned}
V_{M_1}^{\pi^*_M}(s) - V_M^{\pi^*_{\hat{M}}}(s) &= V_{M_1}^{\pi^*_M}(s) - V_M^{\pi^*_{\hat{M}_1}}(s) && (14) \\
&\leq V_{M_1}^{\pi^*_M}(s) - V_{M_1}^{\pi^*_{\hat{M}_1}}(s) && (15) \\
&\leq V_{M_1}^{\pi^*_{M_1}}(s) - V_{M_1}^{\pi^*_{\hat{M}_1}}(s) && (16) \\
&\leq \frac{2R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{4|S||A|\,|\Pi|}{\delta}} + \frac{2U_{\max}}{(1-\lambda)^2}\sqrt{\frac{1}{2n}\log\frac{4|S||A|\,|\Pi|}{\delta}} && (17)
\end{aligned}$$

Eq.14 holds since $\hat{M} = \hat{M}_1$. Eq.15 holds because of Lemma 5. Eq.16 holds since $\pi^*_{M_1}$ is the optimal policy under M_1. The last inequality is derived from Lemma 7.

□

10 Experiment Details

It is known that taking the minimum of a large ensemble could reduce the overestimation error (An et al., 2021). To not make the ensemble influence the Q-value update a lot, we use 8 Q-networks and spit the ensemble into 4 groups. Each group is constituted of 2 networks. By doing so, the target Q-value is the minimum of two networks, which is similar to TD3BC and CQL. We update the uncertainty network and the Q-network with the same frequency. Note that for TD3BC and TD3BC-U, we also normalize the Q-value to make the Q-value term for policy loss in an appropriate range. The hyperparameters for TD3BC-U are shown in Table 2.

For CQL-U, since we use a deterministic of policy, there is no entropy term for the policy loss and Q-value loss. The hyperparameters for CQL-U are shown in Table 3.

Table 2: TD3BC-U hyperparameters.

	Hyperparameter	Value
TD3BC-U hyperparameters	Optimizer	Adam
	Critic learning rate	3e-4
	Actor learning rate	3e-4
	Mini-batch size	256
	Training step	1e6
	Soft update rate	0.005
	Weight of Q-value term for policy loss	2.5
	Policy update frequency	1
	Discount factor γ	0.99
	ν	1
	λ for Table 1	0.8
Architecture	Critic hidden dim	256
	Critic layer dim	256
	Critic activation function	ReLU
	Actor hidden dim	256
	Actor layer dim	256
	Actor activation function	ReLU
	Number of critics	8

Table 3: CQL-U hyperparameters.

	Hyperparameter	Value
CQL-U hyperparameters	Optimizer	Adam
	Critic learning rate	3e-4
	Actor learning rate	3e-4
	Mini-batch size	256
	Training step	1e6
	Soft update rate	0.005
	Weight of Q-value term for policy loss	2.5
	Policy update frequency	2
	Discount factor γ	0.99
	α	2
	λ for Table 1	{0 for walker2d-medium and walker2d-medium-expert, 0.99 for others}
Architecture	Critic hidden dim	256
	Critic layer dim	256
	Critic activation function	ReLU
	Actor hidden dim	256
	Actor layer dim	256
	Actor activation function	ReLU
	Number of critics	8

11 TRAINING CURVES

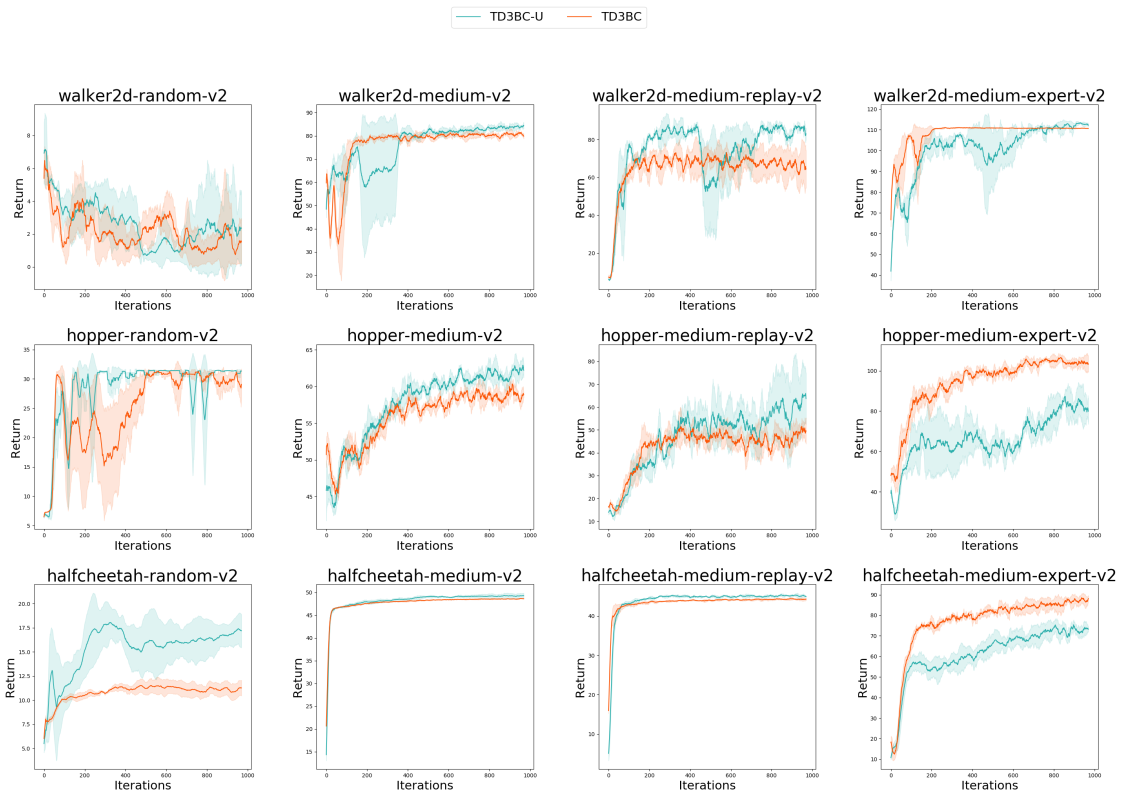


Figure 7: Training curve of TD3BC-U and TD3BC. Mean and variance are calculated over six seeds.