TAMING POLICY CONSTRAINED OFFLINE REINFORCEMENT LEARNING FOR NON-EXPERT DEMONSTRATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We are concerned with offline reinforcement learning (RL) for which, a promising paradigm is to constrain the learned policy to stay close to the dataset behaviors, known as policy constrained offline RL. However, existing works rely heavily on the purity of the data, exhibiting performance degradation or even catastrophic failure when learning from *contaminated datasets* containing trajectories of diverse levels, e.g., expert level, medium level, etc., while offline contaminated data logs exist commonly in the real world. To mitigate this, we first introduce *gradient penalty* over the learned value function to tackle the exploding Q-function gradients induced by the failed closeness constraint on non-expert states. We then relax the harmful closeness constraints towards non-expert dataset actions with *critic weighted constraint relaxation*. Experimental results show that the proposed techniques effectively tame the policy constrained offline RL for non-expert trajectories, evaluated on a set of contaminated D4RL Mujoco and Adroit datasets.

1 INTRODUCTION

Effective offline reinforcement learning (RL) should be able to extract policies with the maximum possible utility out of the static demonstrations without interacting with the environment (Lange et al., 2012; Fujimoto et al., 2019; Levine et al., 2020). One typical way of offline RL is to use policy constraint, enforcing the learned policy to stay close to the behavior policy that generated the dataset, involving various closeness metrics (Fujimoto et al., 2019; Aviral et al., 2019; Wu et al., 2019a; Kostrikov et al., 2021).

However, we find many policy constrained offline RL methods suffer *performance degradation* and even *catastrophic failure* (please see Figure 3) when trained on datasets containing different levels of policy trajectories. For example, methods in Figure 1 show better performance on the expert dataset while achieving lower scores on the medium-expert dataset. This is undesired as the medium-expert datasets contain more dynamics, i.e., both expert and medium-level data (Fu et al., 2020).

Many real-world applications demand robust offline RL algorithms, such as robotic controlling tasks with datasets for multiple tasks or incomplete demonstrations (Sun & Ma, 2019; Fu et al., 2020), recommendation tasks with datasets containing non-user logs (Gunes et al., 2014; Huang et al., 2021), and autonomous driving tasks with trajectories with various levels. In these cases, the dataset contains both expert demonstrations and trajectories from non-experts who have not mastered the task. Filtering out non-expert trajectories with human effort is either expensive or impossible, necessitating robust offline RL algorithms that are resistant to the effects of non-expert trajectories.

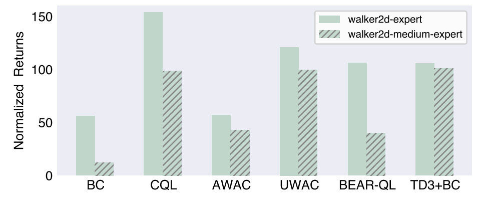

Figure 1: Performance degradation. We report the paper results of CQL (Kumar et al., 2020), AWAC (Nair et al., 2020), UWAC (Wu et al., 2021b), TD3+BC (Fujimoto & Gu, 2021). For BEAR-QL (Aviral et al., 2019), we report the result from D4RL (Fu et al., 2020).

Why do the observed performance degradation and catastrophic failure occur? To answer this question, we first introduce *contaminated datasets*, which contain trajectories from both expert and

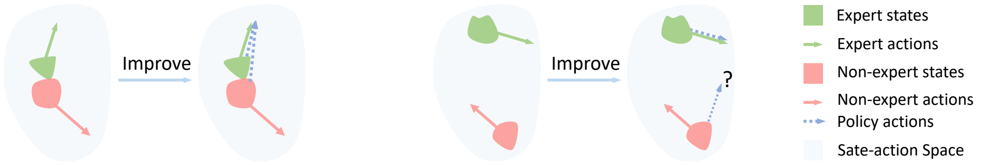

Figure 2: Non-expert data inhibit in two different ways. For non-expert states overlap with expert dataset states (left), they influence the policy improvement in a supervised fashion. For non-expert states far away from expert dataset states (right), policy improvement may lead to OOD actions.

non-expert behavior policies, including *medium*, *cloned*, and *random* levels (Fu et al., 2020). By analyzing the learning behaviors on such datasets, we identify two key paths by which non-expert trajectories inhibit policy constrained offline RL.

First, non-expert data can inhibit policy constrained offline RL in a supervised manner (see Figure 2, left). The closeness constraint explicitly regresses the policy to both expert and non-expert actions, leading to a compromised policy when the states visited by expert overlap that of non-experts (Wu et al., 2019b). To tackle this issue, we propose critic weighted constraint relaxation (**+ CR**), which leverages a polished Q-function to relax the harmful closeness constraint towards non-expert actions.

A more important finding of this work is non-expert trajectories can destroy the learned Q-function via out-of-distribution (OOD) actions (Figure 2, right). Policy improvements on the contaminated dataset make the learned policy closer to dataset expert actions while moving it away from non-expert decisions. This implicitly leads to failed closeness constraint on non-expert state-action pairs when expert and non-expert states follow different distributions, resulting in OOD actions and in turn give rise to unstable Q-values (see Theorem 3.1), sharp Q-function gradients, and finally catastrophic failures (as observed in Figure 3). We introduce the gradient penalty technique (**+ GP**) to suppress the observed exploding Q-function gradients induced from the failed closeness constraint (OOD actions). To justify the proposed GP technique, we theoretically show that there exists an upper bound for the norm of (optimal) Q-function gradients (see Theorem 4.1).

We integrate the proposed two techniques on the top of BEAR-QL (Aviral et al., 2019) and TD3+BC (Fujimoto & Gu, 2021), attaining BEAR++ and TD3BC++. Evaluations on the contaminated datasets for D4RL mujoco and adroit tasks demonstrate that the proposed techniques together could serve as a general plugin to tame the policy constrained offline RL algorithms.

2 Preliminaries

RL A Markov decision process (MDP) can be represented by $M = \langle \mathcal{S}, \mathcal{A}, T, d_0, r, \gamma \rangle$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probability $T(s_{t+1}|s_t, a_t)$, initial state distribution d_0, reward function $r(s_t, a_t)$, and discount factor γ. RL methods aim to find a policy $\pi(a_t|s_t)$, to maximize the expected (discounted) cumulative reward $\mathbb{E}_{\tau \sim p_\pi(\tau)}\left[\sum_{t=0}^{|\tau|} \gamma^t r(s_t, a_t)\right]$, with the trajectory distribution $p_\pi(\tau) = d_0(s_0) \prod_{t=0}^{|\tau|} \pi(a_t|s_t) T(s_{t+1}|s_t, a_t)$.

Offline RL Offline RL algorithms aim to learn policies from a static set of interactions, $D = \{(s_t, a_t, s_{t+1}, r(s_t, a_t))\}_{t=0}^{N}$. One main challenge in offline RL lies in the distribution shift issue (Fu et al., 2019) or the extrapolation error (Fujimoto et al., 2019) during training. For example, a Q-function is trained on *dataset actions* $\mu(a_t|s_t)$ but evaluated on *policy actions* $\pi(a_{t+1}|s_{t+1})$:

$$Q^{k+1}(s_t, a_t) = \mathop{\mathbb{E}}_{s_t, a_t, r, s_{t+1} \sim \mathcal{D}} [r + \gamma Q^k(s_{t+1}, \pi(a|s_{t+1}))] \tag{1}$$

The learned policy may generate *out-of-distribution* (OOD) actions that differ from dataset actions since its optimization objective makes no other guarantees except for generating high-value actions:

$$\pi^{k+1} = \arg\max_{\pi} Q(s_t, \pi(s_t)) \tag{2}$$

Policy improvements implicitly drive the policy to explore OOD actions (Hu et al., 2021), and policy evaluations exploit these OOD actions and in turn affect policy improvements.

Policy constrained offline RL One main approach to offline RL is to enforce the learned policy to stay close to the behavior policy that generated the dataset (Levine et al., 2020):

$$\pi^{k+1} = \arg\max_{\pi} Q(s_t, \pi(s_t)), \qquad s.t. \text{ closeness constraint} \tag{3}$$

We try to address the performance degradation and catastrophic failure issues in two policy constrained offline RL algorithms, TD3+BC (Fujimoto & Gu, 2021) and BEAR-QL (Aviral et al., 2019), without loss of generality.

TD3+BC adds a behavior cloning term on the top of TD3 (Fujimoto et al., 2018), resulting in:

$$\pi = \arg\max_{\pi} \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\Big[\frac{\alpha}{\mathbb{E}[|Q(s_t,a_t)|]} Q(s_t,\pi(s_t)) - (\pi(s_t)-a_t)^2\Big], \tag{4}$$

where α is a hyperparameter controlling the strength of the regularizer.

BEAR-QL constrains the learned policy to have non-negligible support under the data distribution:

$$\pi = \arg\max_{\pi} \mathbb{E}_{s_t\sim\mathcal{D}}\Big[Q(s_t,\pi(s_t))\Big] \text{ s.t. } \mathbb{E}_{s_t\sim\mathcal{D}}[\mathrm{MMD}(\beta(s_t),\pi(s_t))] \leq \epsilon, \tag{5}$$

with β approximating the behavior policy and ϵ being a threshold parameter set to 0.05.

3 Catastrophic failure happens with exploding Q-gradients

Policy constrained offline RL methods fail to learn meaningful policies on contaminated datasets that contain significantly multi-modal state distributions, e.g., expert-cloned, expert-random. We call this catastrophic failure as it happens with exploding Q-function gradients (see Figure 3).

3.1 Catastrophic failure on contaminated datasets

In order to mimic real-world logs that contain multi-level trajectories, we introduce contaminated datasets, which can be generated by contaminating an expert dataset with non-expert demonstrations. For instance, ER3, short for expert-random-30, refers to a dataset in which 70% are expert trajectories and 30% are from random behavior policies. Please refer to Appendix A for the detailed statistics and discussion about the contaminated datasets.

We run TD3+BC and BEAR-QL on contaminated datasets, walker2d-expert-random-v0, as depicted in Figure 3. The catastrophic failure occurs after the percentage of random data exceeds 30%.

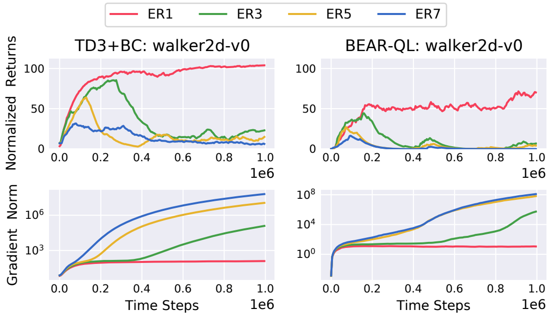

Figure 3: Catastrophic failure. **ER1** is short for expert-random-10, which denotes a contaminated dataset with 10% random trajectories and 90% expert demonstrations.

3.2 Analysis with distribution-constrained Q-iteration

Why does catastrophic failure occur, and why is it so always after the learned policy's performance has improved? In order to give some insights, we use the analysis tool from Aviral et al. (2019), which involves a constrained Bellman backup operator, defined as:

$$\mathcal{T}^{\Pi}Q(s_t,a_t) := \mathbb{E}\big[r + \gamma \max_{\pi\in\Pi} \mathbb{E}_{T(s_{t+1}|s_t,a_t)}[V_\pi(s_{t+1})]\big], \tag{6}$$

with state value function $V_\pi(s_t) := \mathbb{E}_\pi[Q(s_t,\pi(a_t|s_t))]$ and restricted set of policies Π.

Theorem 3.1. *The performance of distribution-constrained Q-iteration can be bounded as:*

$$\lim_{k\to\infty} \mathbb{E}_{d_0}\Big[\Big|V^{\pi^k}(s_t) - V^{\Pi}(s_t)\Big|\Big] \leq \frac{2\gamma}{(1-\gamma)^2} C_{\Pi,\mu}\mathbb{E}_\mu\Big[\max_{\pi\in\Pi}\mathbb{E}_\pi[\delta(s_t,a_t)]\Big] \tag{7}$$

with the concentrability coefficient $C_{\Pi,\mu}$ for quantifying how far the conditional distribution of the policy action $\pi(a_t|s_t) \sim \Pi$ is from the corresponding dataset action $\mu(a_t|s_t)$, and V^{Π} denotes the fixed point of $\mathcal{T}^{\Pi}$, d_0 denotes the initial state distribution.

To understand why the catastrophic failure happens, we simplify the concentrability coefficient $C_{\Pi,\mu}$ (Munos & Szepesvári, 2008) as the distance between the decisions from π and μ.

Before the discussion, we first consider learning from a pure dataset generated by policies with similar decision-making capabilities, e.g., D4RL expert datasets. The policy improvement implicitly drives the learned policy out of the dataset distribution (Hu et al., 2021), resulting in a large concentrability coefficient (Aviral et al., 2019) for all dataset states. Policy constrained offline RL algorithms force the learned policy $\pi \sim \Pi$ to be close to the behavior policy μ, yielding a low concentrability coefficient and thus making it possible to learn RL policy from static datasets.

For the contaminated dataset having two (or more) behavior policies with significantly different decision-making capacities, policy constrained offline RL faces a dilemma:

Adhering to the non-expert dataset decisions (low $C_{\Pi,\mu}$) leads to bad policies, but driving out of the non-expert trajectories (large $C_{\Pi,\mu}$) give rise to OOD actions.

In this case, behavior policies show different state visits, as depicted by Figure 2 and detailed in Figure 9. Policy improvements implicitly drive the learned policy to be different from the non-expert actions and thus leads to the failed closeness constraint on non-expert states, resulting in OOD actions and a large $C_{\Pi,\mu}$. We visualize this in Figure 4. The catastrophic failure (bottom) correlates with the failed constraint towards non-expert decisions (top), i.e., the increasing divergence between the policy and non-expert dataset actions. Please note that the learned policy stays close to dataset expert actions throughout.

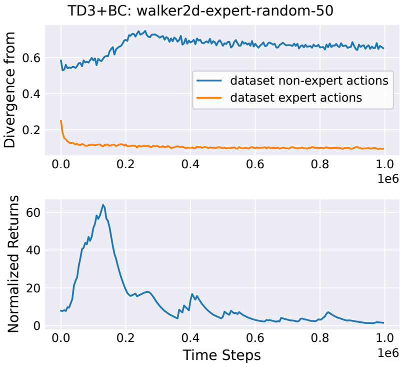

Figure 4: Failed closeness constraint on non-expert state-action pairs correlate with catastrophic failures. **Divergence:** the 75th percentile of the squared error between the policy decisions and the corresponding dataset actions.

The closeness constraint on non-expert state-action pairs is destroyed by the policy improvement, which is why it always occurs after achieving good performance. Such failed closeness constraints lead to OOD actions, which consitute the main challenge for offline RL that induces erroneous Q-values, overestimation problems, and bad policies. However, due to the dilemma above, it is hard to find a proper closeness metric for contaminated datasets.

How to save the policy when OOD actions are inevitable? To the best of our knowledge, this work is the first to observe that OOD actions correlate with extreme sharpness of the Q-values (with respect to actions) and not just overly large values. This inspires us to attenuate the effect of OOD actions from the perspective of gradient regularization.

4 Recovering from catastrophic failure via gradient penalty

We now introduce the gradient penalty technique to reduce the impact of OOD actions induced by the failed closeness constraint on non-expert state-action pairs, and then give theoretical insights on the proposed gradient penalty, followed by a discussion on the difference between our method and a previous work.

4.1 Penalizing the unstable gradients

Recall that the policy improvement step with neural network approximation is $\pi_\theta^{k+1} = \arg\max_\pi Q(s_t, \pi(s_t))$. In practice, this can be achieved by gradient ascent in the parameter space: $\theta = \theta + \alpha \cdot \nabla_{a_t} Q(s_t, a_t)\Big|_{a_t=\pi_\theta(s_t)} \cdot \nabla_\theta \pi_\theta(s_t)$.

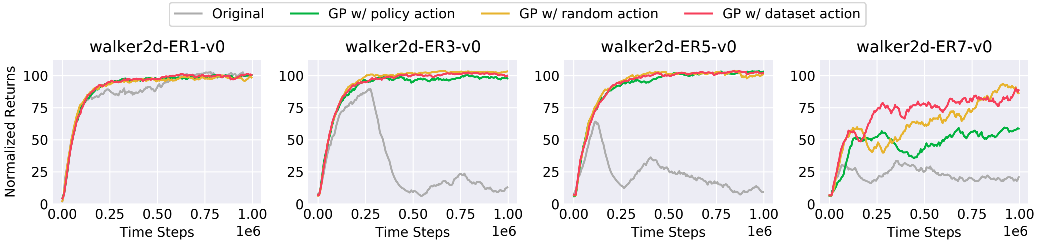

Figure 5: Performance of original TD3+BC algorithm (**Original**), and TD3+BC with gradient penalty w.r.t. actions from different sampling strategies (**Dataset action** for $a \sim \mu$, **Policy action** for $a \sim \pi$, and **Random action** for $a \sim \mathcal{A}$) on the contaminated D4RL datasets. **ER1** is short for expert-random-10. Best viewed in color.

Recall that the improved policy generates OOD actions on non-expert states (with large $C_{\Pi,\mu}$), which cause a loose performance bound 7 and unstable Q-values. The policy resulted from the abnormal Q gradients in turn produces OOD actions. To break the pathological loop, we propose our first modification for policy constrained offline RL methods, i.e., gradient penalty term in the critic loss:

$$\mathcal{L}_{GP} = \lambda_{GP} \mathop{\mathbb{E}}_{s_t \sim \mathcal{D},\ a} \left[\mathrm{ReLU} \left(\left\| \nabla_a Q(s_t, a) \right\|_F - 1 \right) \right]^2 \tag{8}$$

We introduce a one-sided penalty to encourage the norm of the Q-function gradient w.r.t. non-expert actions stays below 1 while avoiding over-punishment for expert alike actions. λ_{GP} controls the contribution of the gradient penalty term. In order to improve computational efficiency, we execute the gradient penalty in every N training steps. We empirically set N to 5 in our experiments.

Note that we do not specify the sampling distribution for action a, as we find there is no significant performance difference between the following three sampling strategies: 1) the current policy action $a \sim \pi$, 2) the dataset action distribution $a \sim \mu$, and 3) random sampling over the action space $a \sim \mathcal{A}$. We will discuss the different motivations behind these choices later.

4.2 Lipschitz property of the learned Q-function

To motivate the proposed gradient penalty technique, we show that the Frobenius norm of the learned Q-function gradient w.r.t. input actions is bounded.

Theorem 4.1. *Suppose a policy $\pi(a_t|s_t)$ on an MDP $M = \langle \mathcal{S}, \mathcal{A}, r, \gamma, T \rangle$ with dynamics T satisfies the inequality $\left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} \right\|_F \leq L_{\pi,T} < 1$ and the reward function $r(s_t, a_t)$ satisfies $\left\| \frac{\partial r(s_t,a_t)}{\partial a_t} \right\| \leq L_r$. If we denote the dimension of the action space as N, then the magnitude of the gradient of the (optimal) Q-function w.r.t. action can be upper bounded as:*

$$\left\| \nabla_{a_t} Q^{\pi}(s_t, a_t) \right\|_F \leq \frac{\sqrt{N} L_r}{1 - \gamma L_{\pi,T}} \tag{9}$$

Proof. See Appendix B. □

Remark 4.2. Theorem 4.1 holds for offline RL setting as the offline MDP is equal to an modified online MDP with a constrained Bellman backup operator (Aviral et al., 2019). It indicates that the Q-prediction should not vary much for perturbations in the action space, implying that the observed exploding Q-function gradient is unreasonable and thus motivating our gradient penalty technique.

4.3 Difference from Fisher-BRC

One may note that the proposed gradient penalty resembles the Fisher divergence term in Fisher-BRC (Kostrikov et al., 2021):

$$\mathop{\mathbb{E}}_{s_t \sim \mathcal{D}} \left[\mathrm{Fisher} \left(\frac{\exp Q(s_t, \cdot)}{\sum_a \exp Q(s_t, a)}, \mu(\cdot|s_t) \right) \right] = \mathop{\mathbb{E}}_{s_t \sim \mathcal{D}, a \sim \pi_{emb}(\cdot|s_t)} \left[\| \nabla_a Q(s_t, a) - \nabla_a \log \mu(a|s_t) \|_F^2 \right].$$

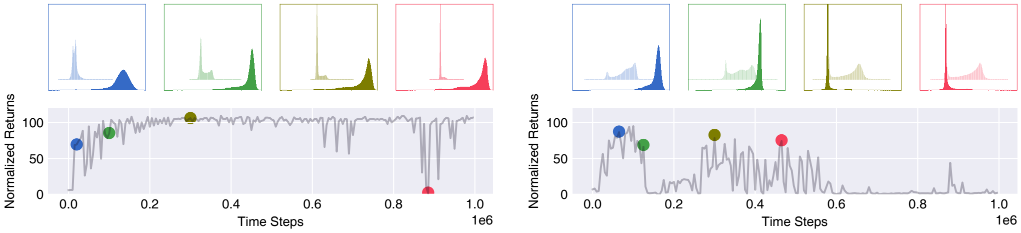

Figure 6: Q-function with gradient penalty can distinguish expert and random actions. We plot the Q-value distributions of dataset actions (top) in the training process (bottom) of a TD3+BC agent **with (left)** and **without (right)** gradient penalty. Diluted and raised histogram for random actions, heavy color for expert actions. Task name is walker2d-expert-random-50-v0. Best viewed in color.

In fact, they are different because Fisher-BRC utilizes gradients to measure the Fisher information distance between the learned policy and the behavior policy μ, while our method serves to reduce the negative impact of OOD actions. With different motivations, our method 1) does not require an entropy regularizer for recovering the Boltzmann policy π_{emb}, and 2) should be insensitive to the action sampling distribution in contrast to Fisher-BRC which needs $a \sim \pi_{emb}$. Figure 5 demonstrates the insensitivity property of our method, where three types of sampling strategies show no performance difference for expert-random-10 (ER1), ER3, and ER5 settings. We perform gradient penalty w.r.t random actions in the section of experiments.

5 Constraint relaxation with polished Q-function

The harmful closeness constraints toward non-expert dataset actions make the learned policy deviate from optima, see also in Wu et al. (2019b); Sasaki & Yamashina (2020). We further relax it by critic weighted constraint relaxation (+ CR) in this section.

The key challenge to relaxing the harmful constraints is to identify the optimality of the dataset actions, for which in offline RL we may make it by the learned Q-function. As depicted in Figure 6, the polished Q-function could successfully discriminate expert decisions (dark colors) and random actions (light colors), even when the policy performs not so well (left). On the other hand, without the gradient penalty, the Q-function is not accurate even if the performance is good (right).

We use the Q-value to indicate the optimality, with a min-max normalization over a mini-batch:

$$W(s_t, a_t) = \frac{Q(s_t, a_t) - Q_{min}}{Q_{max} - Q_{min}} \tag{10}$$

We then could rewrite the regularizer term in BEAR-QL as:

$$\pi = \arg\max_{\pi} \mathbb{E}_{s_t \sim \mathcal{D}} \Big[Q(s_t, \pi(s_t)) \Big] \;\; \mathrm{s.\,t.}\, \mathbb{E}_{s_t \sim \mathcal{D}} [\mathrm{MMD}\left(\beta(s_t), \pi(\cdot|s_t)\right) \cdot W\big(s_t, \beta(s_t)\big)] \leq \epsilon, \tag{11}$$

and for TD3+BC we have:

$$\pi = \arg\max_{\pi} \mathop{\mathbb{E}}_{(s_t, a_t) \sim \mathcal{D}} \left[\frac{\alpha}{\mathbb{E}[|Q(s_t, a_t)|]} Q(s_t, \pi(s_t)) - (\pi(s_t) - a_t)^2 \cdot W(s_t, a_t) \right] \tag{12}$$

Note that we do not propagate gradient through the relaxation weight, $W(s_t, a_t)$.

6 Experiments

We proposed two modifications for policy constrained offline RL: 1) gradient penalty (+ GP) to alleviate the negative impacts of OOD actions induced from the failed closeness constraint and 2) critic weighted constraint relaxation (+ CR) for the harmful closeness constraint. We integrated them in TD3+BC and BEAR-QL, attaining TD3BC++ and BEAR++.

Table 1: Evaluation on the D4RL Mujoco Gym tasks. ER1 is short for Expert-random-10. We rerun all algorithms. With the proposed two techniques, BEAR++ and TD3BC++ could address the performance degradation and catastrophic failures issues. The highest performing scores are bolded.

Task	Setting	BC	10%BC	CQL	BEAR-QL	TD3+BC	10%TD3+BC	Fisher-BRC	UWAC	IQL	BEAR ++	TD3BC ++
Walker2d	Expert	66.1 ± 22.7	76.0 ± 17.5	104.0 ± 6.7	75.1 ± 15.7	**104.5** ± 5.0	72.5 ± 19.6	75.6 ± 42.0	64.3 ± 23.9	**105.6** ± 3.7	97.2 ± 8.3	102.9 ± 4.3
	Expert-medium	11.3 ± 8.0	77.5 ± 19.3	102.4 ± 13.0	56.1 ± 11.7	101.6 ± 10.4	98.2 ± 14.8	103.3 ± 5.3	14.8 ± 9.5	**105.1** ± 4.7	74.1 ± 9.0	**104.3** ± 6.7
	ER1	7.1 ± 15.5	74.8 ± 19.7	100.8 ± 10.8	68.9 ± 13.5	98.8 ± 20.3	100.9 ± 8.9	100.0 ± 16.0	6.6 ± 14.4	**105.2** ± 3.6	94.5 ± 9.5	**104.1** ± 5.2
	ER3	0.8 ± 0.1	60.4 ± 24.8	97.8 ± 13.4	2.2 ± 5.0	14.2 ± 21.7	100.5 ± 10.5	95.0 ± 25.8	9.9 ± 19.9	**102.9** ± 9.0	95.1 ± 8.1	**104.3** ± 3.1
	ER5	1.0 ± 0.3	30.5 ± 22.5	**93.2** ± 21.9	5.2 ± 5.5	8.6 ± 16.3	72.2 ± 48.2	82.5 ± 26.0	4.0 ± 10.4	92.2 ± 12.6	87.0 ± 11.4	**104.4** ± 5.2
	ER7	3.4 ± 8.0	16.3 ± 21.0	77.0 ± 28.3	-0.2 ± 0.7	19.6 ± 23.0	47.7 ± 44.4	69.3 ± 33.7	2.2 ± 4.0	67.6 ± 29.5	**73.1** ± 12.4	**100.2** ± 9.0
Hopper	Expert	111.7 ± 1.7	109.6 ± 6.5	111.7 ± 2.3	61.5 ± 54.3	112.2 ± 0.2	69.3 ± 41.0	112.2 ± 0.7	106.8 ± 10.8	**112.5** ± 0.2	111.4 ± 2.7	**112.3** ± 0.2
	Expert-medium	77.0 ± 38.6	111.0 ± 3.1	**112.1** ± 0.3	85.1 ± 20.9	112.0 ± 0.4	66.0 ± 31.7	112.3 ± 0.3	70.8 ± 33.3	**112.5** ± 0.4	110.3 ± 3.8	112.1 ± 0.3
	ER1	106.6 ± 17.0	112.1 ± 2.3	112.1 ± 0.4	104.4 ± 12.8	112.2 ± 0.2	85.5 ± 24.6	112.3 ± 0.2	91.5 ± 23.6	**112.6** ± 0.1	111.6 ± 3.6	**112.3** ± 0.3
	ER3	25.8 ± 25.9	112.1 ± 1.2	111.2 ± 2.8	82.0 ± 13.8	112.1 ± 0.2	80.4 ± 30.2	112.1 ± 0.7	9.9 ± 0.3	**112.4** ± 0.2	104.8 ± 11.1	**112.2** ± 0.2
	ER5	15.8 ± 20.8	111.7 ± 1.2	112.0 ± 1.8	27.1 ± 11.1	**112.2** ± 0.2	**112.3** ± 0.19	**112.2** ± 0.2	9.9 ± 0.2	111.6 ± 2.4	92.0 ± 12.0	**112.2** ± 0.3
	ER7	9.6 ± 0.2	105.9 ± 9.9	17.9 ± 21.0	10.0 ± 0.1	112.0 ± 0.7	**112.1** ± 0.3	**112.1** ± 0.8	9.7 ± 0.2	**112.5** ± 0.1	45.8 ± 40.4	**112.1** ± 0.2
Halfcheetah	Expert	105.8 ± 2.4	68.6 ± 17.8	94.7 ± 7.3	103.8 ± 6.0	105.3 ± 4.3	67.5 ± 16.6	**106.5** ± 3.5	95.1 ± 10.2	102.4 ± 3.8	104.5 ± 3.4	**105.9** ± 3.4
	Expert-medium	65.9 ± 19.0	95.3 ± 8.6	33.3 ± 10.9	49.3 ± 9.5	94.9 ± 6.3	**96.4** ± 10.3	95.3 ± 9.9	38.0 ± 4.4	81.9 ± 7.3	91.0 ± 9.3	**105.3** ± 2.3
	ER1	89.6 ± 11.1	68.6 ± 13.4	83.0 ± 11.4	93.9 ± 17.5	**101.5** ± 5.2	64.9 ± 16.4	93.3 ± 11.3	63.5 ± 19.3	76.1 ± 9.1	100.4 ± 7.6	**105.1** ± 3.9
	ER3	66.1 ± 17.6	63.0 ± 16.7	62.0 ± 14.4	82.2 ± 19.4	98.4 ± 7.2	77.3 ± 12.3	67.8 ± 21.0	22.6 ± 17.4	64.2 ± 12.6	**103.0** ± 5.3	**103.8** ± 4.2
	ER5	30.1 ± 15.5	66.2 ± 16.0	55.8 ± 11.5	43.4 ± 20.8	90.1 ± 9.7	70.3 ± 12.3	46.9 ± 17.7	2.3 ± 0.1	53.0 ± 10.1	**100.3** ± 8.3	**105.2** ± 2.2
	ER7	2.5 ± 1.5	61.0 ± 16.9	40.2 ± 13.0	2.3 ± 0.0	67.6 ± 9.6	75.0 ± 15.2	29.0 ± 12.5	2.3 ± 0.0	31.0 ± 10.9	**101.8** ± 4.6	**99.8** ± 4.7
Total		796.2 ± 225.9	1420.6 ± 238.4	1521.2 ± 191.3	952.1 ± 238.4	1577.8 ± 140.9	1469.1 ± 356.7	1637.9 ± 227.3	624.1 ± 201.7	1663.3 ± 124.1	1697.9 (+78.3%)	1918.5 (+21.6%)

6.1 Setup

Datasets We consider three types of contaminated datasets, expert-medium, expert-cloned, and expert-random (see Appendix A). For performance on original tasks, please refer to Appendix C.3.

Evaluation We train each algorithm for 1 million time steps, evaluate them every 5000 time steps, and finally report the mean and standard deviation of the normalized scores (Fu et al., 2020) over the final 500 episodes (10 trajectories, 10 evaluations, and 5 seeds). Please note that 5-seed evaluation is a common setting for offline RL literature (Aviral et al., 2019; Wu et al., 2019a; Fujimoto & Gu, 2021; Ma et al., 2021; Wu et al., 2021b; Sinha et al., 2022).

Baselines We compare TD3BC++ and BEAR++ with BC (Pomerleau, 1991), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2022), UWAC (Wu et al., 2021b), Fihser-BRC(Kostrikov et al., 2021) and original BEAR-QL (Aviral et al., 2019), TD3+BC (Fujimoto & Gu, 2021). We also examine percentile BC and percentile TD3+BC, i.e., run BC and TD3+BC on the top 10% of trajectories with higher rewards. Our BC performs similarly to that in Fujimoto & Gu (2021).

6.2 Results and discussion

Performance degradation (Table 1) For the expert-medium setting, in which states visited by expert and medium-level policies show great overlaps (visualized in Figure 9), baseline algorithms suffer performance degradation. In contrast, the proposed TD3BC++ and BEAR++ show resistance to performance degradation, i.e., agents trained on expert-medium datasets perform as experts for all 3 mujoco gym tasks.

Catastrophic failure (Table 1 and 2) BEAR-QL and TD3+BC suffer catastrophic failure when learning on datasets containing trajectories from distinct behavior policies, e.g., expert-random and expert-cloned datasets. Fortunately, the proposed methods alleviate the catastrophic failure issues for all 7 tasks, and they could even help TD3+BC perform as well on ER7 as on the expert datasets, in 6 out of 7 tasks.

Further penalization on OOD actions (Figure 7) Recall that the failed closeness constraint on non-expert decisions produces OOD policy actions, and our proposed gradient penalty successfully recovers the Q-function by penalizing the unstable sharp Q gradients. Based on the result, we conjecture that the proposed gradient penalty could also contribute to reducing the strength of the required closeness constraint for policy constrained offline RL.

To investigate this, we run TD3+BC plus gradient penalty, and change the hyperparameter α to control the strength of BC term (Equation 4). Note that the agent with $\alpha = 1$ prefers imitation while with $\alpha = 4$ for RL. Figure 7 demonstrates that GP reduces the dependence on policy constraints for TD3+BC, and thus may stop from degrading to behavioral cloning. This is not surprising, because the failed closeness constraint can be caused not only by the policy improvement on non-expert demonstrations but also by poor closeness metrics.

Ablation study (Figure 8) We ablate the effects of the two proposed techniques when applied individually. For datasets contain many low-level demonstrations (ER3 and ER5 settings), the gra-

Table 2: Evaluation on the D4RL Adroit domain, involves controlling a 24-DoF robotic hand to perform different tasks. EC1 is short for Expert-cloned-10, with cloned trajectories for non-expert behaviors. The highest performing scores are bolded.

Task	Setting	BC	%BC	CQL	BEAR-QL	TD3+BC	%TD3+BC	Fisher-BRC	UWAC	IQL	BEAR ++	TD3BC ++
Door	Expert	104.6 ± 1.1	104.8 ± 2.3	102.9 ± 5.0	104.8 ± 0.5	103.7 ± 3.5	**105.7** ± 3.12	49.4 ± 23.5	104.5 ± 1.2	**105.6** ± 1.4	104.8 ± 0.7	105.1 ± 0.3
	EC3	102.3 ± 14.7	**105.3** ± 2.0	101.9 ± 3.1	104.4 ± 1.0	0.0 ± 0.0	103.9 ± 4.49	-0.0 ± 0.1	104.0 ± 1.4	**104.3** ± 2.5	104.5 ± 0.8	**105.2** ± 0.6
	EC5	103.5 ± 1.9	**105.3** ± 0.8	-0.2 ± 0.0	82.4 ± 20.9	-0.1 ± 0.0	102.5 ± 5.9	-0.0 ± 0.1	101.9 ± 3.0	104.2 ± 2.8	**104.6** ± 0.8	104.4 ± 1.8
	EC7	52.2 ± 39.2	**104.7** ± 1.5	-0.2 ± 0.1	-0.2 ± 0.1	0.0 ± 0.0	102.5 ± 4.4	-0.0 ± 0.1	92.3 ± 9.9	104.3 ± 2.5	103.0 ± 1.4	**104.5** ± 1.4
Hammer	Expert	126.6 ± 0.5	123.8 ± 7.9	-	126.9 ± 0.3	**127.8** ± 0.6	**129.9** ± 0.3	35.9 ± 33.9	126.2 ± 0.6	119.7 ± 12.5	126.9 ± 0.5	126.8 ± 0.5
	EC3	126.9 ± 0.7	**128.6** ± 0.6	-	84.7 ± 59.7	128.0 ± 0.4	**129.5** ± 2.4	0.2 ± 0.1	126.6 ± 0.6	124.9 ± 5.8	126.7 ± 0.6	126.9 ± 0.5
	EC5	120.4 ± 18.2	127.9 ± 2.0	-	90 ± 42.0	**128.4** ± 0.7	**129.5** ± 0.5	0.2 ± 0.0	125.4 ± 4.1	126.8 ± 2.4	127.0 ± 0.4	127.1 ± 0.5
	EC7	73.7 ± 28.0	127.2 ± 1.8	-	21.0 ± 46.5	0.8 ± 0.6	**128.6** ± 0.6	0.3 ± 0.2	107.9 ± 19.5	127.6 ± 0.6	127.0 ± 0.7	**127.9** ± 1.9
Pen	Expert	**157.5** ± 5.4	78.5 ± 30.6	94.7 ± 25.8	155.5 ± 2.0	132.5 ± 26.3	85.1 ± 24.4	-	155.1 ± 2.6	**155.8** ± 5.4	155.0 ± 2.3	150.3 ± 9.1
	EC3	145.8 ± 24.4	12.9 ± 17.6	66.1 ± 50.0	-3.7 ± 0.4	100.4 ± 10.3	85.5 ± 28.9	-	154.5 ± 2.4	**156.1** ± 5.1	**154.3** ± 1.8	128.9 ± 42.3
	EC5	67.9 ± 38.1	-1.8 ± 2.4	-1.6 ± 2.1	-2.6 ± 0.2	67.1 ± 37.1	78.9 ± 25.0	-	152.6 ± 2.2	**154.3** ± 6.1	**153.9** ± 2.8	141.6 ± 17.8
	EC7	61.8 ± 33.7	-0.8 ± 3.3	-1.6 ± 2.4	-2.4 ± 0.1	65.5 ± 25.5	38.0 ± 23.5	-	59.1 ± 15.6	**154.6** ± 6.3	63.8 ± 15.5	**101.5** ± 21.2
Relocate	Expert	102.3 ± 3.6	52.9 ± 15.0	-	**105.2** ± 1.5	**105.2** ± 2.3	50.8 ± 13.9	3.9 ± 6.1	105.1 ± 2.8	104.9 ± 4.4	**105.2** ± 2.5	103.5 ± 4.1
	EC3	103.2 ± 3.8	65.4 ± 9.5	-	-0.3 ± 0.0	103.9 ± 3.3	48.1 ± 13.2	-0.0 ± 0.1	104.1 ± 3.7	**107.1** ± 2.7	**105.9** ± 1.4	104.4 ± 2.5
	EC5	82.1 ± 23.5	66.8 ± 14.7	-	-0.3 ± 0.0	97.5 ± 9.0	64.8 ± 11.8	-0.0 ± 0.2	103.2 ± 3.5	**106.2** ± 3.7	**105.4** ± 1.7	103.4 ± 2.4
	EC7	40.1 ± 27.4	75.9 ± 10.5	-	-0.3 ± 0.0	27.4 ± 34.0	76.2 ± 15.8	-0.0 ± 0.1	74.9 ± 9.7	**107.0** ± 3.1	**102.4** ± 3.0	99.2 ± 6.7
Total		1571.1 ± 264.2	1270.1 ± 123.0	-	865.0 ± 175.2	1188.1 ± 153.7	1459.4 ± 178.6	-	1797.5 ± 82.9	1963.1 ±68.7	1870.3 (+116.2%)	1860.8 (+56.6%)

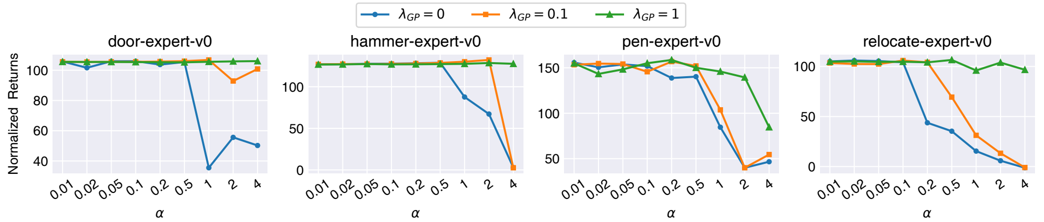

Figure 7: Gradient penalty alleviates the dependence on policy constraints. We run TD3+BC plus different strengths of gradient penalty ($\lambda_{GP} = 0, 0.1, 1$) and different strengths of BC term (X-axis, $\alpha = 0$ for entire BC and $\alpha = 4$ for RL) on Adroit tasks.

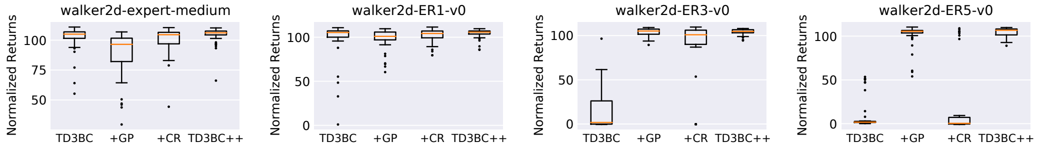

Figure 8: Ablation stduy. Box plot. We run original TD3+BC, TD3+BC with gradient penalty (+GP), TD3+BC with critic weighted constraint relaxation (+CR), and TD3BC++ on walker2d tasks.

dient penalty stabilizes Q-values and prevents from catastrophic failure, and the constraint relaxation with polished Q-weights brings performance back up to the expert level. For datasets that contain a medium level or a low proportion of random demonstrations (EM and ER1 settings), catastrophic failures do not occur. In this case, constraint relaxtion alone is effective, and it performs better in conjunction with a polished Q-function.

Comparison with the naïve solution (Table 1 and 2) %BC and %TD3+BC show slight resistance to the two issues. We also set X to $X \pm 10$ and find it performs worse. This tells that simply discarding non-expert demonstrations may devastate expert trajectories.

Gradient penalty w.r.t. input states We investigate the effect of gradient penalty w.r.t. states. However, we find experimentally that it performs much worse. This finding may prevent other approaches to modify the Q-function gradients, e.g., spectral normalization (Gogianu et al., 2021).

Computational cost comparison We train TD3+BC and TD3BC++ agents for 1 million time steps. The wall clock time of TD3+BC is 160m, and 173m for TD3BC++, indicating that the two techniques proposed in this paper are light and efficient plugins for policy constrained offline RL.

7 Related work

Policy constrained offline RL One main approach for offline RL is to enforce the learned policy stay close to the behavior policy, involved with various closeness measurements such as KL-divergence (Jaques et al., 2019), maximum mean discrepancy (MMD) (Aviral et al., 2019), Wasserstein distance (Wu et al., 2019a), Fisher divergence (Kostrikov et al., 2021) and even Euclidean distance (Fujimoto & Gu, 2021). Closeness constraints could help avoid OOD actions. However, when training on contaminated datasets with non-expert demonstrations, a common setting in real-world applications, these methods show performance degradation and even catastrophic failure in our observation. The proposed two techniques serve to mitigate such issues.

Value regularizated offline RL Another offline RL approach is modifying the Q-values to prevent overestimation on OOD actions. This can be achieved by directly penalizing the Q-values of OOD actions in the regression target, e.g., CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2022), SAC-N (An et al., 2021) or discounting them with uncertainty measurements e.g., UWAC (Wu et al., 2021b), EDAC (An et al., 2021), PBRL (Bai et al., 2022), RORL (Yang et al., 2022). The proposed gradient penalty aims not to prevent the OOD actions but rather to minimize their negative impact, i.e., unstable Q-function gradients. In this work, this is caused by the failed closeness constraint on non-expert states.

Lipschitzness in reinforcement learning Our method penalizes the sharp gradient derived from the critic, which is similar to enforcing the learned Q-function to be locally Lipschitz-continuous. Lipschitzness is often used for stabilizing generative adversarial network (GAN) training. It can be achieved by gradient penalty (Thanh-Tung et al., 2019), spectral normalization (Miyato et al., 2018), gradient normalization (Wu et al., 2021a), etc. In online RL, Gogianu et al. (2021) use spectral normalization to better the optimization dynamics of the Bellman backups. Lecarpentier et al. (2020) utilize Lipschitz continuity between MDPs to transfer knowledge for lifelong RL tasks. Memarian et al. (2021) promotes a local Lipschitz discriminator for robust generative adversarial imitation learning (GAIL) algorithms. Our method aims to minimize the effects of the OOD policy actions non-expert trajectories, which carries a different motivation.

Learning from non-expert trajectories This work focus on the influence of non-expert trajectories in the offline RL setting. Similarly, Zhang et al. (2021a) proposes an algorithm to address this issue, assuming clustering methods can recognize transitions from different behavior policies. Besides, Zhang et al. (2021c) consider the task of training policy from datasets with adversarial corruptions. Our method does not rely on such assumptions. In addition, Nair et al. (2020) proposes an offline RL algorithm with advantage-based critic weight, which is theoretically superior to our CR technique. We leave this for further work.

Learning from non-expert data is also a key challenge in imitation learning. Methods in this topic can be mainly divided into two types. Ranking-based methods learn a policy from demonstrations annotated with rankings (Akrour et al., 2011; Brown et al., 2019; 2020; Chen et al., 2020). Confidence-based methods construct or learn a confidence value function describing the quality of demonstrations and then reweight training samples for imitation (Wang et al., 2018; Wu et al., 2019b; Zhu et al., 2020; Tangkaratt et al., 2020; Sasaki & Yamashina, 2020; Cao & Sadigh, 2021; Zhang et al., 2021b; Wang et al., 2021). Our method utilizes the learned Q-function to indicate the optimality of the transition, which is close to the confidence-based methods. l

8 CONCLUSION

By analyzing the learning behavoirs on datasets generated by multiple distinct behavior policies, we identify two senarios where non-expert trajectories inhibit policy constrained offline RL: 1) the harmful closeness constraint towards non-expert actions on overlaping states, and 2) the failed closeness constraint on non-expert states that causes OOD actions. The proposed CR and GP techniques are tailored to handle the two issues, respectively, and their effectiveness is empirically evaluated in expert-medium, expert-cloned, and expert-random settings.

The combination of the proposed two techniques extends the applicability of the policy constrained offline RL to contaminated datasets. Particularly, the proposed gradient penalty can help mitigate the negative impacts of OOD actions when the policy constraint fails (on contaminated datasets) or when the constraint needs be weakened (in order to outperform the behavior policies). Hopefully, our work would attract more attention to offline reinforcement learning in a way different from the Q-value regularization or policy constraint.

REFERENCES

Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 12–27. Springer, 2011.

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*, 34:7436–7447, 2021.

Kumar Aviral, Fu Justin, Soh Matthew, Tucker George, and Levine Sergey. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pp. 11761–11771, 2019.

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.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International Conference on Machine Learning*, pp. 783–792. PMLR, 2019.

Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pp. 330–359. PMLR, 2020.

Zhangjie Cao and Dorsa Sadigh. Learning from imperfect demonstrations from agents with varying dynamics. *IEEE Robotics and Automation Letters*, 6(3):5231–5238, 2021.

Letian Chen, Rohan Paleja, and Matthew Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. In *Conference on Robot Learning*, pp. 1262–1277. PMLR, 2020.

Justin Fu, Aviral Kumar, Matthew Soh, and Sergey Levine. Diagnosing bottlenecks in deep Q-learning algorithms. In *International Conference on Machine Learning*, pp. 2021–2030. PMLR, 2019.

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. *Advances in Neural Information Processing Systems*, 34, 2021.

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.

Florin Gogianu, Tudor Berariu, Mihaela Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: an optimisation perspective. In *International Conference on Machine Learning*, pp. 3734–3744. PMLR, 2021.

Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review*, 42(4):767–799, 2014.

Yuzheng Hu, Ziwei Ji, and Matus Telgarsky. Actor-critic is implicitly biased towards high entropy optimal policies. *arXiv preprint arXiv:2110.11280*, 2021.

Hai Huang, Jiaming Mu, Neil Zhenqiang Gong, Qi Li, Bin Liu, and Mingwei Xu. Data poisoning attacks to deep learning based recommender systems. *arXiv preprint arXiv:2101.02644*, 2021.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

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

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *The Tenth International Conference on Learning Representations*, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

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

Erwan Lecarpentier, David Abel, Kavosh Asadi, Yuu Jinnai, Emmanuel Rachelson, and Michael L Littman. Lipschitz lifelong reinforcement learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pp. 8270–8278. AAAI Press, 2020.

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.

Yecheng Ma, Dinesh Jayaraman, and Osbert Bastani. Conservative offline distributional reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Farzan Memarian, Abolfazl Hashemi, Scott Niekum, and Ufuk Topcu. Robust generative adversarial imitation learning via local Lipschitzness. *arXiv preprint arXiv:2107.00116*, 2021.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

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

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Emmanuel Rachelson and Michail G Lagoudakis. On the locality of action domination in sequential decision making. In *International Symposium on Artificial Intelligence and Mathematics*, 2010.

Fumihiro Sasaki and Ryota Yamashina. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*, 2020.

Samarth Sinha, Ajay Mandlekar, and Animesh Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pp. 907–917. PMLR, 2022.

Mingfei Sun and Xiaojuan Ma. Adversarial imitation learning from incomplete demonstrations. *arXiv preprint arXiv:1905.12310*, 2019.

Voot Tangkaratt, Bo Han, Mohammad Emtiyaz Khan, and Masashi Sugiyama. Variational imitation learning with diverse-quality demonstrations. In *International Conference on Machine Learning*, pp. 9407–9417. PMLR, 2020.

Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. In *International Conference on Learning Representations*, 2019.

Qing Wang, Jiechao Xiong, Lei Han, Peng Sun, Han Liu, and Tong Zhang. Exponentially weighted imitation learning for batched historical data. In *Advances in Neural Information Processing Systems*, pp. 6291–6300, 2018.

Yunke Wang, Chang Xu, Bo Du, and Honglak Lee. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, pp. 10961–10970. PMLR, 2021.

Yi-Lun Wu, Hong-Han Shuai, Zhi-Rui Tam, and Hong-Yu Chiu. Gradient normalization for generative adversarial networks. In *International Conference on Computer Vision*, pp. 6373–6382, 2021a.

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

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. In *International Conference on Machine Learning*, pp. 11319–11328. PMLR, 2021b.

Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pp. 6818–6827. PMLR, 2019b.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *arXiv preprint arXiv:2206.02829*, 2022.

Hongchang Zhang, Jianzhun Shao, Yuhang Jiang, Shuncheng He, and Xiangyang Ji. Reducing conservativeness oriented offline reinforcement learning. *arXiv preprint arXiv:2103.00098*, 2021a.

Songyuan Zhang, Zhangjie Cao, Dorsa Sadigh, and Yanan Sui. Confidence-aware imitation learning from demonstrations with varying optimality. *Advances in Neural Information Processing Systems*, 34:12340–12350, 2021b.

Xuezhou Zhang, Yiding Chen, Jerry Zhu, and Wen Sun. Corruption-robust offline reinforcement learning. *arXiv preprint arXiv:2106.06630*, 2021c.

Zhuangdi Zhu, Kaixiang Lin, Bo Dai, and Jiayu Zhou. Learning sparse rewarded tasks from sub-optimal demonstrations. *arXiv preprint arXiv:2004.00530*, 2020.

Appendices

A THE CONTAMINATED D4RL DATASETS

We first provide details about the contaminated D4RL datasets to accommodate reproducibility. Then we give evidence to support the description of the different state overlaps in Figure 2. And finally, we provide some perspectives on the proposed contaminated D4RL datasets.

A.1 DATASET STATISTICS

The contaminated D4RL mujoco gym datasets Each dataset contains trajectories from two different levels of policies. We use the D4RL medium-expert datasets for expert-medium settings, which are combinations of expert and medium-level trajectories and are about twice the size of the corresponding expert or medium datasets.

We also contaminate the expert demonstrations with random-levels. For example, ER-1, short for Expert-random-10, represents a dataset constructed by first loading an expert dataset and then replacing the final 10 percent transitions with tuples from random trajectories, i.e., the first 10 percent in the corresponding random dataset. We provide statistics:

Table 3: Statistics of the contaminated D4RL mujoco gym datasets (expert-random).

Task	Setting	Total transition	Expert transition	Random transition	Averaged reward
Hopper	Expert-random-10	999,034	899,131	99,903	3.53
	Expert-random-30	999,034	699,324	299,710	3.33
	Expert-random-50	999,034	499,517	499,517	3.13
	Expert-random-70	999,034	299,711	699,323	2.93
Walker	Expert-random-10	999,304	899,374	99,930	4.25
	Expert-random-30	999,304	699,513	299,791	3.33
	Expert-random-50	999,304	499,652	499,652	2.39
	Expert-random-70	999,304	299,792	699,512	1.45
Halfcheetah	Expert-random-10	998,999	899,100	99,899	10.94
	Expert-random-30	998,999	699,300	299,699	8.44
	Expert-random-50	998,999	499,500	499,499	5.95
	Expert-random-70	998,999	299,700	699,299	3.46

The contaminated D4RL adroit datasets The contaminated D4RL Adroit datasets can be constructed in a similar way, except that the non-expert trajectories are from cloned agents, i.e., imitation policies trained from the human-level demonstrations. Statistics of the contaminated D4RL adroit datasets used in our evaluations are:

Table 4: Statistics of the contaminated D4RL Adroit Datasets (expert-cloned).

Task	Setting	Total transition	Expert transition	Cloned transition	Averaged reward
Door	Expert-cloned-10	995,000	895,500	99,500	13.08
	Expert-cloned-30	995,000	696,500	298,500	10.13
	Expert-cloned-50	995,000	497,500	497,500	7.16
	Expert-cloned-70	995,000	298,500	696,500	4.83
Hammer	Expert-cloned-10	995,000	895,500	99,500	55.31
	Expert-cloned-30	995,000	696,500	298,500	42.79
	Expert-cloned-50	995,000	497,500	497,500	30.06
	Expert-cloned-70	995,000	298,500	696,500	19.32
Pen	Expert-cloned-10	495,000	445,500	49,500	30.73
	Expert-cloned-30	495,000	346,500	148,500	25.96
	Expert-cloned-50	495,000	247,500	247,500	21.05
	Expert-cloned-70	495,000	148,500	346,500	20.61
Relocate	Expert-cloned-10	995,000	895,500	99,500	19.44
	Expert-cloned-30	995,000	696,500	298,500	15.10
	Expert-cloned-50	995,000	497,500	497,500	10.80
	Expert-cloned-70	995,000	298,500	696,500	8.27

A.2 Different state overlaps

In Figure 2, We highlight two distinct situations involving different expert and non-expert state overlaps. When states visited by experts show great overlaps with non-expert states, the harmful closeness constraint toward non-expert decisions inhibits. For situation that expert states and non-expert states are well-distinguished, the failed closeness constraint happens as the learned policy is improved, showing different policy actions for dataset non-expert states.

We here provided some visualizations of the distribution of expert and non-expert states in the expert-medium, expert-random, and expert-cloned settings.

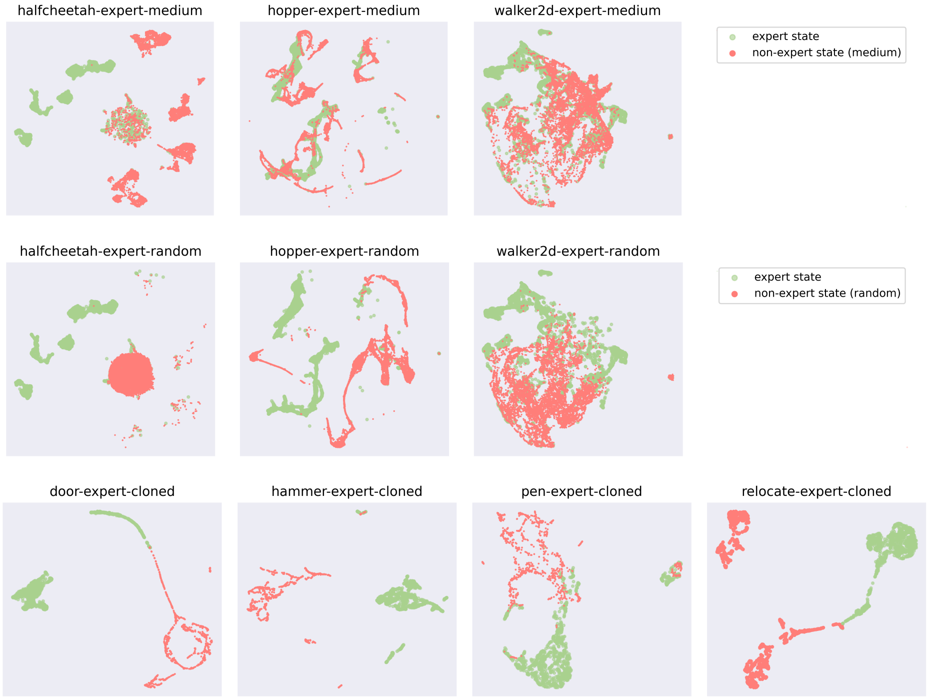

Figure 9: We use UMAP to reduce the dimensionality of states in different D4RL tasks. Expert states are visited by expert behavior policies, and non-expert states are from the medium, random or the cloned policies. We enlarge the dot size of expert states for clarity.

Great state overlaps In this situation, e.g., expert-medium datasets, states visited by expert-level behavior policies show great overlap with that of medium agents. Therefore, the closeness constraint towards non-expert actions may prevent the learned policy from moving closer to the expert decisions. Although offline RL with support-based policy constraints, e.g., BEAR, holds the promise to solve such issues, their exquisite metrics are often difficult to achieve. We alleviate the observed performance degradation by introducing a Q-weight for policy constraint methods (+CR).

Less state overlaps For datasets contaminated by low-level demonstrations, e.g., random and cloned level data, the expert and non-expert states show greatly different distributions. In this case, policy improvement inevitably changes the policy actions on non-expert states, increasing the probability of generating OOD decisions. This can be dangerous as OOD actions have been widely recognized as the source of exploding value function and the failed learning process. We suppress the OOD actions with the proposed GP technique.

The success of BC on adriot tasks For the simple mujoco tasks (controlling 3 or 6-DoF robotics), states visited by expert policies show great overlap with those visited by non-expert policies. With overlapped states, constraints toward non-expert actions affect the decision quality on expert states. In contrast, such impacts are eliminated with fewer overlaps under the complex adroit tasks (24-DoF robotics). The records of non-expert state-action pairs less influence the decisions for expert states, thus leading to the success of BC agents on complex adroit tasks.

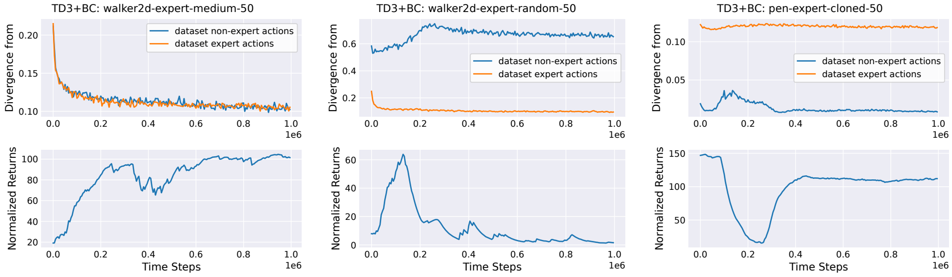

Figure 10: Visulation of the harmful closeness constraint (left) and the faild closeness constraint (middle and right). **Divergence:** the 75th percentile of the squared error between the decisions from the learned policy and the corresponding dataset actions.

A.3 The harmful and the failed closeness constraint.

The harmful closeness constraint When expert and non-expert behavior policies share great state overlaps, two similar dataset states may correspond to two (or more) completely different actions. The closeness constraint towards non-expert one would inhibit the policy improvement in a supervised fashion.

We visualize it in Figure 10, left. The divergence between decisions from the learned policy and the expert behaviors becomes smaller as the policy improves. However, the distance to the non-expert dataset actions also becomes smaller. This contributes to the observed performance degradation. Although ideal support-based policy constraint methods hold the promise to handle this situation, empirically, their performance deteriorates.

The failed closeness constraint The main contribution of this paper is the finding that the policy improvement induces the failed closeness constraint on non-expert dataset states. That is, the policy improvement implicitly drives the learned policy to be different from the decisions recorded for non-expert states, inducing dangerous OOD actions.

In the middle and the right-hand side of Figure 10, we visualize the failed closeness constraint on non-expert dataset states, which happens after the policy achieves a good performance.

A.4 Remarks

Are expert-random datasets too extreme for mimicking real-life scenarios? The proposed contaminated dataset can be used to simulate the training behavior on a dataset containing two distinct behavior policies. In this context, what matters is not the non-expert behavior policies' quality but the states' overlap between the experts and non-experts.

Such datasets do not necessarily have to be constructed by expert and random policies. For example, the catastrophic failures on expert-cloned datasets, see Table 2, indicate that the learned Q-functions are destroyed by the sharp Q-function gradients, though the cloned behavior policies are far away from randoms.

Difference with the D4RL replay datasets This work focuses on training from contaminated datasets, including three types, expert-medium, expert-cloned, and expert-random. Another similar setting is the medium-replay or the full-replay dataset, which records all the interactions during the training. However, we think there is a significant difference between the two settings.

Firstly, the contaminated dataset better fits offline reinforcement learning scenarios. Recall that the primary motivation of offline RL is to avoid the risky interactions for training policy from random initializations. Thus, it is unfeasible to collect logs like medium-replay or full-replay datasets in most situations. On the other hand, the contaminated dataset is used to simulate the training behavior on a dataset where two different behavioral policies exist. We believe a dataset with multiple behavior policies is a really common setting for real-life applications.

Another difference is that the medium-replay and full-replay dataset have a wider distribution of state-action pairs and thus a lower probability of inducing OOD actions than the contaminated dataset considered in this paper. The proportions of expert (medium) trajectories in the replay datasets may also be smaller.

B Lipschitz property of the learned Q-function over action domain

In this section, we provide proof for Theorem 4.1. In order to prove the desired Lipschitz continuity property of the learned Q-function, we need to give an upper bound of the magnitude of the Q-function gradient with respect to the input action, i.e., $\|\frac{\partial Q(s_t,a_t)}{\partial a_t}\|_F$ is bounded.

For notational clarity, we use $\mathbb{E}_{s_{t+k}|s_t}[\cdot]$ to denote the expectation of the argument with respect to the conditional distribution of future state s_{t+k} given that the agent starts from current s_t and follows the policy $\pi(a_t|s_t)$, i.e., $\mathbb{E}_{s_t}\Big[\big(\prod_{j=0}^{k}\pi(a_{t+j}|s_{t+j})T(s_{t+1+j}|s_{t+j},a_{t+j})\big)[\cdot]\Big]$. Then we can rewrite the learned Q-function $Q^{\pi}(s_t,a_t)$ as $\sum_{k=0}^{\infty}\gamma^k\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]$. Our proof starts from the upper bound of the Jacobian of the Q-function w.r.t. one dimension of the action space. In this case, we denote the $i-th$ dimension of the action space as a_t^i. We then drive to the case of multi-dimensional action space and complete our proof.

Proposition B.1. *Suppose a policy π on an MDP $M=\langle\mathcal{S},\mathcal{A},T,d_0,r,\gamma\rangle$ with dynamics T satisfies the following inequality for any given non-negative integer t:*

$$\Big\|\frac{\partial\pi(a_{t+1}|s_{t+1})}{\partial a_t}\Big\|_F \le L_{\pi,T}, \tag{13}$$

then it holds for any given non-negative integer k, and t:

$$\Big|\nabla_{a_t^i}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big| \le L_{\pi,T}\cdot\mathbb{E}_{s_{t+1}|s_t}\Big|\nabla_{a_{t+1}^i}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\Big|. \tag{14}$$

Proof.

$$\begin{aligned}\Big|\nabla_{a_t^i}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big| &= \Big|\nabla_{a_{t+1}^i}\mathbb{E}_{s_{t+1}|s_t}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\cdot\frac{\partial a_{t+1}^i}{\partial a_t^i}\Big|\\ &\le \Big|\nabla_{a_{t+1}^i}\mathbb{E}_{s_{t+1}|s_t}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\Big|\cdot\Big|\frac{\partial a_{t+1}^i}{\partial a_t^i}\Big|\\ &= \Big|\frac{\partial a_{t+1}^i}{\partial a_t^i}\Big|\cdot\mathbb{E}_{s_{t+1}|s_t}\Big|\nabla_{a_{t+1}^i}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\Big|\\ &\le L_{\pi,T}\cdot\mathbb{E}_{s_{t+1}|s_t}\Big|\nabla_{a_{t+1}^i}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\Big|\end{aligned}$$

□

The above proposition gives a derivation from a mild assumption, which is helpful for our next step proof.

Proposition B.2. *Suppose a policy π on an MDP $M=\langle\mathcal{S},\mathcal{A},T,d_0,r,\gamma\rangle$ with dynamics T satisfies the following inequality for any given non-negative integer t:*

$$\Big\|\frac{\partial\pi(a_{t+1}|s_{t+1})}{\partial a_t}\Big\|_F \le L_{\pi,T} \tag{15}$$

$$\Big\|\frac{\partial r(s_t,a_t)}{\partial a_t}\Big\|_F \le L_r, \tag{16}$$

then it holds for any given non-negative integer t:

$$\Big|\nabla_{a_t^i}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big| \le L_{\pi,T}^k\cdot L_r. \tag{17}$$

Proof.

$$\begin{aligned}
\Big|\nabla_{a^i_t}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big| &\le L_{\pi,T}\cdot\mathbb{E}_{s_{t+1}|s_t}\Big|\nabla_{a^i_{t+1}}\mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k},a_{t+k})]\Big| \\
&\le L_{\pi,T}\cdot\mathbb{E}_{s_{t+1}|s_t}\cdots L_{\pi,T}\cdot\mathbb{E}_{s_{t+k}|s_{t+k-1}}\Big|\nabla_{a^i_{t+k}}\mathbb{E}_{s_{t+k}|s_{t+k}}[r(s_{t+k},a_{t+k})]\Big| \\
&= L^k_{\pi,T}\cdot\mathbb{E}_{s_{t+k}|s_t}\Big|\nabla_{a^i_{t+k}}\mathbb{E}_{s_{t+k}|s_{t+k}}[r(s_{t+k},a_{t+k})]\Big| \\
&= L^k_{\pi,T}\cdot\mathbb{E}_{s_{t+k}|s_t}\Big|\nabla_{a^i_{t+k}}r(s_{t+k},a_{t+k})\Big| \\
&\le L^k_{\pi,T}\cdot\mathbb{E}_{s_{t+k}|s_t}\cdot L_r \\
&= L^k_{\pi,T}\cdot L_r
\end{aligned}$$

□

Then we consider the case of multi-dimensional action space. An upper bound formulation of the learned Q-function gradient w.r.t. action can be derived by using Proposition B.1 and Proposition B.2.

Theorem 4.1. *Suppose a policy $\pi(a_t|s_t)$ on an MDP $M = \langle\mathcal{S},\mathcal{A},T,d_0,r,\gamma\rangle$ with dynamics T satisfies the inequality $\left\|\frac{\partial\pi(a_{t+1}|s_{t+1})}{\partial a_t}\right\|_F \le L_{\pi,T} < 1$ and the reward function $r(s_t,a_t)$ satisfies $\left\|\frac{\partial r(s_t,a_t)}{\partial a_t}\right\| \le L_r$. If we denote the dimension of the action space as N, then the magnitude of the gradient of the learned Q-function w.r.t. action can be upperbounded as:*

$$\Big\|\nabla_{a_t}Q^\pi(s_t,a_t)\Big\|_F \le \frac{\sqrt{N}L_r}{1-\gamma L_{\pi,T}}. \tag{18}$$

Proof.

$$\begin{aligned}
\Big\|\nabla_{a_t}Q^\pi(s_t,a_t)\Big\|_F^2 &= \sum_{i=0}^{N}\Big(\nabla_{a^i_t}Q^\pi(s_t,a_t)\Big)^2 \\
&= \sum_{i=0}^{N}\Big(\sum_{k=0}^{\infty}\gamma^k\nabla_{a^i_t}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big)^2 \\
&\le \sum_{i=0}^{N}\Big(\sum_{k=0}^{\infty}\gamma^k\Big|\nabla_{a^i_t}\mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k},a_{t+k})]\Big|\Big)^2 \\
&= \sum_{i=0}^{N}\Big(\sum_{k=0}^{\infty}\gamma^k\cdot L^k_{\pi,T}\cdot L_r\Big)^2 \\
&= N\big(L_r\sum_{k=0}^{\infty}(\gamma L_{\pi,T})^k\big)^2,
\end{aligned}$$

finally, we have:

$$\begin{aligned}
\Big\|\nabla_{a_t}Q^\pi(s_t,a_t)\Big\|_F &\le \sqrt{N}L_r\sum_{k=0}^{\infty}(\gamma L_{\pi,T})^k \\
&= \frac{\sqrt{N}L_r}{1-\gamma L_{\pi,T}}
\end{aligned}$$

□

To better understand the proposed bound (18), we give some perspective on the constants in this formulation. Clearly, L_r is the Lipschitz constant of the reward function w.r.t. the input action. Then we consider the meaning of $L_{\pi,T}$. $\left\|\frac{\partial\pi(a_{t+1}|s_{t+1})}{\partial a_t}\right\|_F$ measures the change in the policy action

a_{t+1} at next state s_{t+1} if we give an infinitesimal perturbation in the current policy action a_t. We denote its upper bound as $L_{\pi,T}$ as the Jacobian is related with the policy π and the environment dynamics T:

$$\begin{aligned}\frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} &= \frac{\partial}{\partial a_t}\pi\Big(a_{t+1}|T(s_{t+1}|s_t,a_t)\Big)\\ &= \frac{\partial T(s_{t+1}|s_t,a_t)}{\partial a_t}\cdot\frac{\partial \pi(a_{t+1}|s')}{\partial s'}\Big|_{s'=T(s_{t+1}|s_t,a_t)}\end{aligned}$$

Then we can derive the upper bound of the Jacobian as:

$$\begin{aligned}\Big\|\frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t}\Big\|_F &= \Big\|\frac{\partial T(s_{t+1}|s_t,a_t)}{\partial a_t}\cdot\frac{\partial \pi(a_{t+1}|s')}{\partial s'}\Big|_{s'=T(s_{t+1}|s_t,a_t)}\Big\|_F\\ &\leq \Big\|\frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial s_{t+1}}\Big\|_F\cdot\Big\|\frac{\partial T(s_{t+1}|s_t,a_t)}{\partial a_t}\Big\|_F\end{aligned}$$

The proposed constant $L_{\pi,T}$ is related with two Lipschitz constants, the first one for the policy π w.r.t. the input states and another one for the environment dynamics T w.r.t. the inpuit action.

We refer the interested readers to Memarian et al. (2021) for the proof of the upper bound for optimal Q-function gradients w.r.t. input states. For the Lipschitz continuity of the value function, see Rachelson & Lagoudakis (2010).

C EXPERIMENT DETAILS

We run our experiments on a single machine with RTX3090 GPUs. All D4RL datasets use the v0 version.

C.1 BASELINES

	Walker2d	Hopper	Halfcheetah	Door	Hammer	Pen	Relocate
min_q_weight	10	20	20	20	-	50	-

Table 5: Hyperparameter for CQL. We sweep it within the range of {5, 10, 20, 50, 100}.

	Walker2d	Hopper	Halfcheetah	Door	Hammer	Pen	Relocate
f_reg	1	1	1	5	10	0.01	0.1

Table 6: Hyperparameter for Fisher-BRC.

- CQL. We use a modular PyTorch implementation of CQL[1]. We are very sorry that we cannot reproduce it on adroit hammer and relocate tasks. To be more specific, for these omitted, the final D4RL normalized scores we got, acoss all swept paremeters, are about zero (random). We thus have to omit these irrational scores to prevent distress or offense to other readers and authors. Table 5 shows the hyperparameters used in our experiments.
- BEAR-QL. We use the recommended Github implementation [2]. We follow the recommended settings for mujoco tasks, and for four adroit tasks, we use the Gaussian kernel.
- UWAC. We use the official implementation [3], with default hyperparameters.
- IQL. We use the authors' implementaion in JAX [4], which is really really fast.

[1] Code and license: https://github.com/young-geng/cql

[2] Code and license: https://github.com/rail-berkeley/d4rl_evaluations

[3] Code and license: https://github.com/apple/ml-uwac

[4] Code and license: https://github.com/ikostrikov/implicit_q_learning

- Fisher-BRC. We use the author's implementation [5]. We sweep the best hyperparameters for D4RL adroit expert tasks and follow the suggested settings for D4RL mujoco tasks.

C.2 The proposed method

Implementation We recommend interested readers to reproduce results of TD3BC++ on the top of TD3+BC [6], which is really a minimalist approach to offline RL. The proposed plugin involves two algorithmic modifications:

```
    # Compute critic loss
    critic_loss = F.mse_loss(current_Q1, target_Q) + F.mse_loss(current_Q2, target_Q)
+   if self.total_it % N == 0: # We empirically set N to 5.
+       _state_rep = state.clone().detach().repeat(16, 1).requires_grad_(True)
+       _random_action = torch.rand(
+           size=self.actor(_state_rep).size(),
+           requires_grad=True
+           ) * 2 - 1.0
+       _random_action= _random_action.to(device)
+       _current_Q1, _current_Q2 = self.critic(_state_rep, _random_action)
+       grad_q1_wrt_random_action = torch.autograd.grad(
+           outputs=_current_Q1.sum(),
+           inputs =_random_action,
+           create_graph=True
+           )[0].norm(p=2, dim=-1)
+       grad_q2_wrt_random_action = torch.autograd.grad(
+           outputs=_current_Q2.sum(),
+           inputs =_random_action,
+           create_graph=True
+           )[0].norm(p=2, dim=-1)
+       grad_q_wrt_random_action = F.relu(grad_q1_wrt_random_action - self.k) **2 +\
+           F.relu(grad_q2_wrt_random_action - self.k) **2
+       critic_loss = critic_loss +  grad_q_wrt_random_action.mean() * self.lambda_GP
...
    # Compute actor loss
-   # actor_loss = -lmbda * Q. mean() + F. mse_loss(pi, action)
+   current_Q = ((current_Q1 + current_Q2) * 0.5).squeeze().detach()
+   actor_loss = -lmbda * Q.mean() + \
+       (F.mse_loss(pi, action, reduction=’none’).mean(axis=-1) * current_Q).mean()
```

Listing 1: The proposed two small changes on the top of TD3+BC.

Hyperparameters used for experiments Our modification involves a weight factor λ_{GP} for gradient penalty loss $\mathcal{L}_{GP}$. As for the backbone algorithm, TD3+BC, we find α, a factor to control the strength of BC term in Equation 4, affects performance the most. Fujimoto & Gu (2021) use $\alpha = 2.5$ for their experiments on D4RL mujoco gym tasks. However, we find it does not work for adroit tasks. We sweep it within the range of $\{0.05, 0.1, 0.2, 0.5, 1, 2, 2.5, 3, 4\}$ and select the maximum possible value that works. Note that, TD3+BC with a low value of α may degenerate to imitation (BC term will dominate the learning) rather than RL. We report the settings used for our experiments:

		Walker2d	Hopper	Halfcheetah	Door	Hammer	Pen	Relocate
TD3+BC	α	2.5	2.5	2.5	0.5	0.2	0.5	0.02
TD3BC++	α	2.5	2.5	2.5	0.5	0.2	0.5	0.02
	λ_{GP}	1	1	1	1	1	1	0.1
BEAR++	λ_{GP}	1	1	1	1	1	1	0.1

Table 7: Hyperparameters for TD3+BC, TD3BC++, and BEAR++.

[5] Code and license: https://github.com/google-research/google-research/tree/master/fisher_brc

[6] Code and license: https://github.com/sfujim/TD3_BC

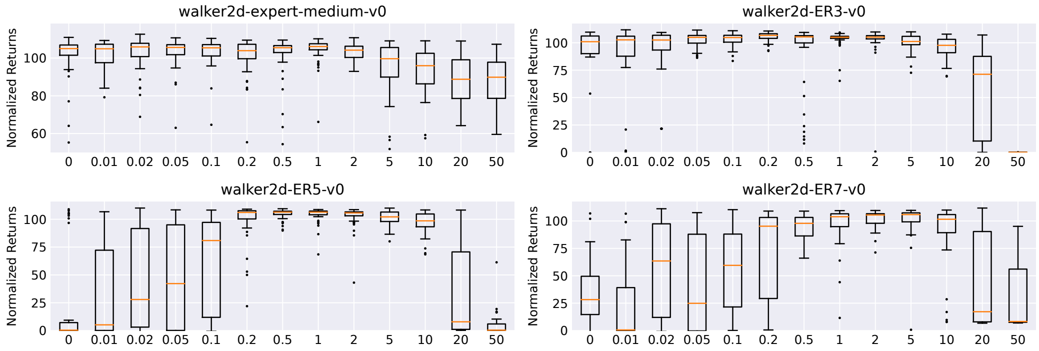

Figure 11: Hyperparameter study. Box plot. We run TD3BC++ with different λ_{GP}.

Hyperparameter study We fix the BC term $\alpha = 2.5$ and vary the gradient penalty term λ_{GP} in TD3BC++, sweeping on four different settings. Results are shown in figure 11. For the EM setting, a small GP term (0.01, 0.02, or 0.05) can have a stabilizing effect on training while an overlarge one would inhibit the learned Q-function. As for the difficult ER3, ER5, and ER7 settings, we recommend practitioners choose a medium value (1, 2, 5) to stabilize learning while avoiding making the Q-function too flat.

C.3 Evaluation on D4RL mujoco gym tasks.

This work focuses on addressing the performance degradation and the catastrophic failure issues for policy constraint offline RL algorithms. Therefore, we are more concerned with the performance on contaminated datasets with non-expert trajectories. In order to show the potential influence of the proposed methods, we report the results of BEAR++ and TD3BC++ on classic D4RL mujoco gym tasks.

Table 8: Evaluation on the original D4RL mujoco gym tasks.

Task	Setting	BC	CQL	Fisher-BRC	AWAC	BEAR	TD3+BC	BEAR ++	TD3BC ++
Halfcheetah	Expert	105.20	82.40	108.40	78.50	103.77	105.70	104.53	105.87
	Medium-expert	67.60	27.10	93.30	36.80	49.25	97.90	91.01	105.26
	Medium	36.60	37.20	41.30	37.40	37.09	42.80	36.85	40.78
	Medium-replay	34.7	41.9	43.3	−.−	37.7	43.3	38.4	43.6
	Random	2.00	21.70	33.30	2.20	2.26	10.20	2.25	6.98
Hopper	Expert	111.50	111.20	112.30	85.20	61.50	112.20	111.36	112.23
	Medium-expert	89.60	111.40	112.40	80.90	85.12	112.20	110.28	111.57
	Medium	30.00	44.20	99.40	72.00	37.89	99.50	39.84	100.13
	Medium-replay	19.7	28.6	35.6	−.−	3.6	31.4	37.8	32.4
	Random	9.50	10.70	11.30	9.60	10.22	11.00	10.04	10.58
Walker2d	Expert	56.00	103.80	103.00	57.00	75.13	105.70	97.20	104.68
	Medium-expert	12.00	68.10	105.20	42.70	56.08	101.10	74.13	104.46
	Medium	11.40	57.50	78.80	30.10	57.87	79.70	62.46	75.79
	Medium-replay	8.3	15.8	42.6	−.−	11.6	25.2	28.9	27.6
	Random	1.20	2.70	1.50	5.10	3.27	1.40	19.90	5.26

We note that for the dataset generated by the single-level behavioral policy (medium, expert, and random), the proposed plug-in has no significant gain, while it does not impair it. For datasets with narrow distributions that generated by several behavior policies, the proposed method has significant gains, e.g., the medium-expert datasets.

Finally, the performance on medium-replay datasets reflects the limitation of this work that the proposed GP plug-in is designed to stabilize the training process by suppressing the unstable sharp Q-functions induced by OOD actions that generated from the policy improvement step on dataset non-expert states. The replay datasets record decision behaviors from random to near-expert policies for each dataset state. Thus the probability of generating OOD actions during the policy improvement steps is much smaller for the replay dataset.

D BROADER IMPACT

Policy constrained offline RL is a crucial approach to data-driven decision-making machines. As it enjoys many advantages, such as easy implementation, small training costs, and no need for extensive domain knowledge, one can apply it to various scenarios. Therefore, we believe that this work will inevitably inherit the social impact of application contexts.

The proposed plugins alleviate the observed performance degradation and catastrophic failure issues for policy constrained offline RL. With them, one can make greater use of static demonstrations to obtain stronger agents. To this degree, we believe our social impact lies in expanding the applicability of policy constrained offline reinforcement learning methods.