ROBUST OFFLINE REINFORCEMENT LEARNING FROM LOW-QUALITY DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

In practice, due to the need for online interaction with the environment, deploying reinforcement learning (RL) agents in safety-critical scenarios is challenging. Offline RL offers the promise of directly leveraging large, previously collected datasets to acquire effective policies without further interaction. Although a number of offline RL algorithms have been proposed, their performance is generally limited by the quality of dataset. To address this problem, we propose an Adaptive Policy constrainT (AdaPT) method, which allows effective exploration on out-of-distribution actions by imposing an adaptive constraint on the learned policy. We theoretically show that AdaPT produces a tight upper bound on the distributional deviation between the learned policy and the behavior policy, and this upper bound is the minimum requirement to guarantee policy improvement at each iteration. Formally, we present a practical AdaPT augmented Actor-Critic (AdaPT-AC) algorithm, which is able to learn a generalizable policy even from the dataset that contains a large amount of random data and induces a bad behavior policy. The empirical results on a range of continuous control benchmark tasks demonstrate that AdaPT-AC substantially outperforms several popular algorithms in terms of both final performance and robustness on four datasets of different qualities.

1 INTRODUCTION

Reinforcement learning (RL) provides a mathematical formalism for learning-based sequential decision. Existing RL algorithms typically require to adopt an online learning paradigm, which involves online interactions with the environment and learns from its own collected experience. However, this limits their applicability in real-world scenarios (Garcıa & Fernández, 2015; Dulac-Arnold et al., 2019), where online interaction is expensive (e.g., in robotics (Dasari et al., 2019)) or there is a safety consideration with updating the policy online (e.g., in healthcare (Shortreed et al., 2011; Nemati et al., 2016)). As a feasible solution, offline RL (also known as batch RL (Lange et al., 2012)) concerns a more realistic setting in which an agent has access to only a fixed dataset of logged experience, without any further interactions with the environment. This setting is of particular interest for industrial applications such as recommendation system (Bottou et al., 2013) and autonomous driving (Yu et al., 2018). Nevertheless, the inability to interact with the environment directly poses a major challenge to offline RL, *especially in the case of low-quality offline datasets*.

While off-policy algorithms can in principle learn from arbitrary data, the performance is usually poor in the offline setting. These failures can generally be attributed to distributional shift, a fundamental issue of offline RL, which leads to large extrapolation error when the policy learned from one distribution is evaluated on a different distribution (Fujimoto et al., 2019). To overcome this challenge, a commonly used approach is to constrain the learned policy against the behavior policy induced by the dataset (Kumar et al., 2019; Wu et al., 2019). However, some of policy constraint based methods are developed exclusively for offline datasets of specific source and thus are vulnerable to the low-quality dataset, a common case in real scenarios. In addition, the performance of these methods is limited to behaviors within the data manifold. For example, when an offline dataset contains a large amount of random data and induces a bad behavior policy, existing policy constraint based methods struggle to learn a generalizable policy due to the difficulty to trade off error accumulation and suboptimality of the learned policy (Levine et al., 2020). Therefore, how to adaptively adjust the policy constraint imposed on the learned policy regardless of the quality of dataset remains a major issue in the field of offline RL.

In this paper, we seek to develop an Adaptive Policy constrainT (AdaPT) method that enables the agent to learn a generalizable policy even from low-quality datasets. To that end, we make the observation that the optimal distributional deviation between the learned policy and the behavior policy can be seen as an upper bound for AdaPT. Inspired by the idea underlying policy consistency learning (Nachum et al., 2017), we can derive an adaptive form for this upper bound in the behavior regularized offline RL framework. The resulting adaptive upper bound is related to the intensity of regularization and thus reflects the suboptimality of the learned policy. In fact, there are several appealing advantages to apply AdaPT. First, the policy constraint can be adjusted adaptively according to the performance of the learned policy. Second, the feasible policy set induced by the policy constraint is large enough to include the learned policy at the last iteration. While the former makes it possible to adaptively control the constraint imposed on the learned policy, the latter ensures an improved policy can always be obtained theoretically at each iteration.

The primary contribution of this work is a model-free AdaPT augmented Actor-Critic (AdaPT-AC) algorithm, which can perform well even in the case of low-quality offline datasets. Our AdaPT-AC is developed directly by augmenting the vanilla behavior regularized policy iteration with the proposed AdaPT, the standard form of which can be readily derived based on a behavior consistency relation among the learned policy, the learned value functions and the behavior policy. We theoretically show that AdaPT produces a tight upper bound on the distributional deviation between the learned policy and the behavior policy, and this upper bound is the minimum requirement to guarantee policy improvement at each iteration. In particular, our AdaPT-AC enables the agent to learn a policy that can be improved to a great extend and is allowed to deviate far from the behavior policy. We empirically compare the AdaPT-AC to several popular algorithms on very complex and high-dimensional continuous control tasks, such as the Humanoid benchmark in OpenAI Gym (Brockman et al., 2016). For each task, we construct four offline datasets of different qualities. The results demonstrate that AdaPT-AC substantially outperforms the prior methods on these offline datasets, and is the only method that still performs well on low-quality datasets, such as our *very low* dataset where **80 percent** of the data is completely random.

2 Related Work

As a safe way to deploy RL agents in the real world, offline RL shows great potential for application in industrial scenarios. Here we briefly review the existing offline RL approaches, which can be partitioned into two broad categories: *policy constraint based methods* and *explicit uncertainty estimation based methods*.

The basic idea underlying *policy constraint based methods* is to constrain, explicitly or implicitly, the learned policy to stay close to the behavior policy, and ensure the Bellman backup is never executed on out-of-distribution actions. The methods in this family differ in terms of the probability metric used to measure "closeness" (such as KL-divergence (Jaques et al., 2019; Peng et al., 2019; Siegel et al., 2020), Wasserstein distance (Wu et al., 2019), or MMD (Gretton et al., 2007; Kumar et al., 2019)) and how this constraint is introduced and enforced. However, a notorious challenge associated with these methods is that the degree of improvement beyond the behavior policy is restricted by error accumulation (Fujimoto et al., 2019; Simão et al., 2019), and a small divergence from the behavior policy at each step can give rise to a policy that diverges away from the behavior distribution and performs very poorly (Levine et al., 2020). Aside from directly constraining the policy, another conceptually attractive approach to mitigating the effect of out-of-distribution actions is to estimate the epistemic uncertainty of the Q-function. The intuition behind *uncertainty estimation based methods* is that the uncertainty is expected to be substantially large for out-of-distribution actions, and thus can be utilized to produce conservative target values in such cases. In general, such methods learn an uncertainty set or distribution over possible Q-functions by maintaining a lower confidence bound (Auer et al., 2009; Kumar et al., 2020), or directly representing samples from the distribution over Q-functions, for example via bootstrap ensembles (Osband et al., 2016; Anschel et al., 2017; Agarwal et al., 2020) or parameterization (ODonoghue et al., 2018). However, offline RL requires a calibrated uncertainty estimation to directly capture the trustworthiness of the Q-function, which is often very difficult, especially with modern high-capacity function approximators, such as deep neural networks (DNNs).

Besides the above two established approaches, there have also been significant efforts on enabling standard off-policy RL algorithms (Precup et al., 2001; Bellemare et al., 2017; Fujimoto et al., 2018; Dabney et al., 2018) in the offline setting on large-scale datasets (Pietquin et al., 2011; De Bruin

et al., 2015; Gu et al., 2017; Zhang & Sutton, 2017; Kalashnikov et al., 2018; Cabi et al., 2019). These methods do not explicitly correct for the distributional deviation between the learned policy and the behavior policy, and thus require to make unrealistic assumptions on the quality or source of dataset. In contrast, this work aims to adaptively adjust the policy constraint according to the performance of the learned policy, and the proposed method is able to learn a generalizable policy even from low-quality datasets that contain a large amount of random data.

3 PRELIMINARIES

3.1 NOTATION

We consider an infinite-horizon Markov decision process (MDP) setting. An MDP is typically defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, p_0, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, \infty)$ is a conditional probability distribution that describes the dynamics of the system, p_0 defines the initial state distribution $p_0(s)$, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines a reward function, and $\gamma \in (0, 1]$ is a scalar discount factor. The goal of an RL agent is to learn a policy π, which is a mapping from each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$ to the conditional probability $\pi(a|s)$ of taking action a when in state s. Throughout this paper, we will use $\rho_\pi(s_t)$ and $\rho_\pi(s_t, a_t)$ to denote the state and state-action marginal of the trajectory distribution induced by the policy $\pi(a_t|s_t)$.

In this work, we address the offline RL problem for complex continuous domains. Specifically, we aim to learn a policy $\pi(a_t|s_t)$ from a fixed dataset $\mathcal{D}$ consisting of single-step transitions $(s_t, a_t, r(s_t, a_t), s_{t+1})$. Similar to prior works (Wu et al., 2019; Siegel et al., 2020), we define the behavior policy $\pi_b(a_t|s_t)$ as the conditional distribution $p(a_t|s_t)$ observed in $\mathcal{D}$. Under this definition, such a behavior policy $\pi_b(a_t|s_t)$ is always well-defined even if the dataset was collected by multiple, distinct behavior policies, and can be fitted with max-likelihood method over $\mathcal{D}$.

3.2 BEHAVIOR REGULARIZED OFFLINE REINFORCEMENT LEARNING

Standard RL learns a policy by directly maximizing the expected sum of rewards $\sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t)]$. In contrast, the main idea underlying behavior regularization methods for offline RL is to ensure that the learned policy $\pi(a_t|s_t)$ is constrained against the behavior policy $\pi_b(a_t|s_t)$. Therefore, we will consider a behavior regularized objective, which augments the standard objective with the expected sum of policy penalties over $\rho_\pi(s_t)$

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi} \left[r(s_t, a_t) - \alpha D(\pi, \pi_b, s_t) \right], \tag{1}$$

where $D(\pi, \pi_b, s_t)$ is a distributional deviation function between $\pi(\cdot|s_t)$ and $\pi_b(\cdot|s_t)$ over actions (e.g. KL divergence or Wasserstein distance). The temperature parameter $\alpha \geq 0$ determines the relative importance of the policy penalty term against the reward, and thus control the degree to which the learned policy π is allowed to deviate from the behavior policy π_b. In particular, the standard objective can be recovered in the limit as $\alpha \to 0$. It is common in prior works (Wu et al., 2019; Kumar et al., 2019) to use a fixed α or adaptively train α as a Lagrangian multiplier by setting a fixed threshold for the average distributional deviation.

How to determine the degree of behavior regularization is a key challenge in behavior regularized offline RL. Existing methods generally use a *fixed* α or train it to satisfy a policy constraint that has a *specified* upper bound. A major limitation of these approaches is that they prefer to use a conservative upper bound and thus keep the distributional deviation small to control errors due to out-of-distribution actions. However, in practice, the distributions do need to deviate in order for the learned policy to improve over the behavior policy, especially in cases where the dataset $\mathcal{D}$ contains a large amount of random data and induces a bad behavior policy. In this work, we will discuss how we can devise a model-free offline RL algorithm, which imposes an adaptive policy constraint on the learned policy regardless of the quality of dataset.

4 METHODOLOGY

Our offline RL algorithm can be derived starting from a general formulation of *Adaptive Policy constrainT* (AdaPT), which is the key idea of our work. In this section, we first present this derivation to obtain the main results. Then the proposed AdaPT is applied to develop a practical deep RL algorithm for learning from offline data.

4.1 ADAPTIVE POLICY CONSTRAINT

In this work, we aim to achieve robust offline RL even from low-quality datasets, such as the dataset that contains a large amount of random data, which usually induces a bad behavior policy. To learn a policy whose performance is not largely limited by the behavior policy, we propose a novel method, i.e., *Adaptive Policy constrainT* (AdaPT), which allows effective exploration on out-of-distribution actions by imposing an adaptive constraint on the distributional deviation between the learned policy and the behavior policy. Our AdaPT can be derived starting from a behavior regularized variant of softmax temporal consistency (Nachum et al., 2017). This variant arises by optimizing the objective (1) with the distributional deviations $D(\pi, \pi_b, \cdot)$ having a form of KL divergence. In this setting, the state value function (also known as value function) $V^\pi(s)$, describes the expected sum of discounted future rewards and policy penalties when starting in state s and thereafter following policy π. Then we can define state value $V^\pi(s)$ recursively as

$$V^\pi(s) = \mathbb{E}_{a\sim\pi, s'}\big[r(s,a) - \alpha \log \pi(a|s) + \alpha \log \pi_b(a|s) + \gamma V^\pi(s')\big]. \tag{2}$$

Let $\pi^* = \arg\max_\pi \mathbb{E}_s V^\pi(s)$ denote the optimal policy corresponding to the above Bellman equation (2), and let $V^* = V^{\pi^*}$ denote the optimal value function. Then the optimal action value function (also known as Q-function) can be expressed as $Q^*(s,a) = r(s,a) + \gamma \mathbb{E}_{s'}\big[V^*(s')\big]$. It is worth noting that when $\alpha > 0$, the optimal policy π^* is no longer deterministic, since the relative entropy term prefers the use of policies with more uncertainty. By reformulating the right-hand side (RHS) of Equation (2) as a KL divergence between the learned policy π and the rest, we can explicitly solve for the optimal policy π^* as a Boltzmann distribution of the form

$$\pi^*(a|s) = \exp\big\{Q^*(s,a)/\alpha + \log \pi_b(a|s) - V^*(s)/\alpha\big\}. \tag{3}$$

Taking the $\log$ of both sides of Equation (3) reveals an important connection among the optimal values $Q^*(s,a)$, $V^*(s)$, and corresponding action probabilities under the log-policy $\log \pi^*(a|s)$ and $\log \pi_b(a|s)$. Then we can formally characterize the distributional deviation between π^* and π_b in terms of Q and value functions as

$$\log \pi^*(a|s) - \log \pi_b(a|s) = (Q^*(s,a) - V^*(s))/\alpha. \tag{4}$$

We point out that similar notion of behavior consistency established in Equation (4) has been studied in previous works (Nachum et al., 2017; 2018). For example, Nachum et al. (2018) has proposed to restrict the learned policy within a relative entropy trust region around a prior policy for more stable learning. Here we make new observations to achieve robust offline RL even from low-quality datasets. We note that the behavior consistency (4) gives the optimal distributional deviation between the learned policy π and the behavior policy π_b for a specific α, and thus shows the possibility to construct an adaptive policy constraint for offline RL. The formal result is given in Proposition 1.

Proposition 1 (Adaptive Policy constrainT)**.** *Suppose that the policy set* Π *is composed of all the policies* π *satisfying the consistency constraint*

$$\mathbb{E}_{s\sim\mathcal{D}, a\sim\pi}\left[\log \frac{\pi(a|s)}{\pi_b(a|s)}\right] \le \mathbb{E}_{s\sim\mathcal{D}, a\sim\pi^*}\left[\log \frac{\pi^*(a|s)}{\pi_b(a|s)}\right]. \tag{5}$$

Then, when optimizing the objective (1) within the policy set Π*, the following inequality holds*

$$\mathbb{E}_{s\sim\mathcal{D}, a\sim\pi}\left[\log \frac{\pi(a|s)}{\pi_b(a|s)}\right] \le \mathbb{E}_{s\sim\mathcal{D}}\left[\frac{Q^*(s, \arg\max_a Q^*(s,a)) - V^*(s)}{\alpha}\right]. \tag{6}$$

Proof. See Appendix A. □

The proposed AdaPT has several good properties. First, the search space of policy is restricted to a small set Π to avoid unstable global exploration, which may lead to large extrapolation error on out-of-distribution actions. Second, the policy set Π corresponds to the minimal radius that includes the optimal policy π^*, around the behavior policy. Third, the upper bound in Equation (5) can adapt to the behavior policy π_b, which generally reflects the quality of dataset. More concretely, the constraint imposed on the learned policy can be adjusted adaptively according to the quality of dataset, encouraging the agent to explore in space away from the behavior policy.

4.2 ADAPTIVE POLICY CONSTRAINT BASED POLICY ITERATION

We now derive AdaPT based policy iteration, a general algorithm that alternates between policy evaluation and policy improvement in the adaptive policy constraint framework. Our derivation is based on a tabular setting to enable theoretical analysis.

In the policy evaluation step, we wish to compute the action value of a policy π according to the behavior regularized objective (1). For a fixed policy π and temperature parameter α, the action value can be computed iteratively, starting from any Q-function and repeatedly applying a modified Bellman backup operator $\mathcal{T}^{\pi,\alpha}$ given by

$$\mathcal{T}^{\pi,\alpha}Q(s_t,a_t) := r(s_t,a_t) + \gamma \mathbb{E}_{s_{t+1},a_{t+1}\sim\pi}\left[Q(s_{t+1},a_{t+1}) - \alpha \log \frac{\pi(a_{t+1}|s_{t+1})}{\pi_b(a_{t+1}|s_{t+1})}\right]. \quad (7)$$

Then, for an initial Q-function Q^0, we can obtain the optimal Q-function Q^π for any policy π by repeatedly applying $\mathcal{T}^{\pi,\alpha}$ to the old Q-function, namely, $Q^{n+1} = \mathcal{T}^{\pi,\alpha}Q^n$. The convergence is readily proved by defining a relative entropy augmented reward and then applying the standard convergence results for policy evaluation (Sutton & Barto, 2018).

In the policy improvement step, we update the policy by directly applying the new Q-function to (3). Specifically, for each state and action, we update the policy within the policy set Π_k according to

$$\pi_{k+1}(a_t|s_t) = \arg\min_{\pi\in\Pi_k} D_{KL}\big(\pi, \exp\{Q^{\pi_k}(s_t,a_t)/\alpha + \log \pi_b(a_t|s_t) - \log Z^{\pi_k,\alpha}(s_t)\}\big), \quad (8)$$

where $Z^{\pi_k,\alpha}(s_t)$ is an action-independent normalization function. and the policy set Π_k is composed of all the policies π that satisfy

$$\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[\log\frac{\pi(a|s)}{\pi_b(a|s)}\right] \le \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_k}\left[\frac{Q^{\pi_k}(s,a) - V^{\pi_k}(s)}{\alpha}\right] \le \mathbb{E}_{s\sim\mathcal{D}}\left[\frac{Q^{\pi_k}(s,a') - V^{\pi_k}(s)}{\alpha}\right], \quad (9)$$

where $a' = \arg\max_a Q^{\pi_k}(s,a)$. This particular choice of update can be guaranteed to result into an improved policy in terms of its Q-function and avoid optimizing in the whole policy space. We show that the new policy has a higher Q-value than the old policy with respective to the objective in Equation (1). This result is formalized in Theorem 1.

Theorem 1. *Let π_{k+1} is updated using Equation* (8)*. Then the second term of Equation* (9) *is the minimum upper bound to ensure that $Q^{\pi_{k+1}}(s_t,a_t) \ge Q^{\pi_k}(s_t,a_t)$ for all $(s_t,a_t)\in\mathcal{D}$.*

Proof. See Appendix B. □

The convergence of the whole algorithm can be readily proved based on Theorem 1, and we refer the reader to the Theorem 1 of Haarnoja et al. (2018) for a complete analysis. Although this algorithm will provably find the optimal solution, we can perform it in its exact form only in the tabular case. In the next section, we will approximate the algorithm for continuous domains.

4.3 Adaptive Policy Constraint Augmented Actor-Critic

As discussed above, large, continuous domains require us to derive a practical approximation to AdaPT based policy iteration. Here we present a practical model-free offline actor-critic algorithm, which relies on function approximator to represent the Q-value and policy. In particular, we employ an adaptive scheme to update the temperature parameter α based on the proposed AdaPT. For the remainder of this paper, we will consider a parameterized Q-function $Q_\theta(s_t,a_t)$ and a tractable policy $\pi_\phi(a_t|s_t)$. In the following, we will derive update rules for network parameters (θ,ϕ) and temperature parameter α.

The critic network approximates the Q-function. According to Equation (7), the parameters of critic can be optimized directly by minimizing the following Bellman residual

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\mathcal{D}}\big[Q_\theta(s_t,a_t) - r_t - \gamma V(s_{t+1})\big]^2, \quad (10)$$

where $\mathcal{D}$ is a fixed offline dataset, and

$$V(s_{t+1}) = \mathbb{E}_{a_{t+1}\sim\pi_\phi}\big[Q_{\theta'}(s_{t+1},a_{t+1}) - \alpha\log\pi_\phi(a_{t+1}|s_{t+1}) + \alpha\log\pi_b(a_{t+1}|s_{t+1})\big]. \quad (11)$$

Note that θ' is the parameter vector of a target critic network. There is no need to include a separate function approximator for V, since it is related to the Q-function, the learned policy and the behavior policy as shown in Equation (11). Here we evaluate this quantity using actions sampled from π_ϕ. In addition, we also parameterize two Q-functions $Q_{\theta_i}(i=1,2)$ and train them independently to mitigate positive bias in the policy improvement step that is known to degrade the performance of value-based methods (Fujimoto et al., 2018; Haarnoja et al., 2018).

The actor network approximates the policy, the parameters of which can be trained by directly minimizing the KL divergence on the RHS of Equation (8), which we reproduce here in parameterized form for completeness

$$J_\pi(\phi) = D_{KL}\big(\pi_\phi(\cdot|s_t)\big\|\exp\big\{Q_\theta(s_t,\cdot)/\alpha + \log\pi_b(\cdot|s_t) - \log Z_\theta^\alpha(s_t)\big\}\big)\,. \tag{12}$$

There are several options to minimize J_π, depending on the choice of the policy class. A general likelihood ratio gradient estimator that can be used to update the actor network is given as follows

$$\nabla_\phi J_\pi(\phi) = \mathbb{E}_{a_t\sim\pi_\phi}\big[\nabla_\phi \log\pi_\phi(a_t|s_t)\big(\log\pi_\phi(a_t|s_t) - \log\pi_b(a_t|s_t) - Q_\theta(s_t,a_t)/\alpha + 1\big)\big]\,, \tag{13}$$

which can be derived directly by replacing the entropy term of soft policy gradient (Shi et al., 2019) with our relative entropy term. In our implementation, we employ a Gaussian policy with mean and covariance predicted by DNNs. In this special setting, an alternative is to apply the reparameterization trick, which allows us to backpropagate the gradient from the critic network.

The temperature parameter α controls the degree to which the learned policy π_ϕ is allowed to deviate from the behavior policy π_b. However, the original form of AdaPT (9) is intractable, since the action with the highest Q-value is unavailable in continuous cases. To tackle this problem, we instead utilize a *Max-Q* approximation π_q, which samples a few actions from π_ϕ and then returns the one with the highest Q-value Q_θ. While in principle we could choose any update rule, it will turn out to be convenient to use the stochastic gradient descent as follows

$$\nabla_\alpha J_\alpha(\alpha) = \beta\,\mathbb{E}_{a_t\sim\mathcal{N}(\pi_q,\sigma)}\left[\frac{Q_\theta(s_t,a_t) - V(s_t)}{\alpha}\right] - \mathbb{E}_{a_t\sim\pi_\phi}\left[\log\frac{\pi_\phi(a_t|s_t)}{\pi_b(a_t|s_t)}\right], \tag{14}$$

where $V(s_t)$ is evaluated using actions sampled from π_ϕ as in Equation (11), $\mathcal{N}(\pi_q,\sigma)$ is a Gaussian policy with a covariance matrix σ, and β is a positive correction coefficient to mitigate the bias introduced by the *Max-Q* sampling policy. Moreover, to stabilize the training, the first term of Equation (14) is bounded by a maximum target divergence B. We emphasize that the above update rule shows an adaptive scheme to learn α. When the learned policy π_ϕ is too close to the behavior policy π_b, α will be decreased to result in a relaxed policy constraint according to Equation (14).

The complete algorithm, which we call AdaPT augmented Actor-Critic (AdaPT-AC), is summarized in Appendix D. AdaPT-AC alternates between updating the parameters of networks and temperature parameter using the stochastic gradients on batches sampled from a fixed offline dataset.

5 Experiments

The goal of our experimental evaluation is to verify the effectiveness of our method on offline datasets, especially on those datasets that contain a large amount of random data and induce bad behavior policies. To that end, we evaluate our AdaPT-AC on a range of challenging MuJoCo continuous control tasks (Todorov et al., 2012) from the OpenAI Gym benchmark suite (Brockman et al., 2016), including Humanoid-v2, Reacher-v2, Walker2d-v2, Hopper-v2, Ant-v2 and HalfCheetah-v2. For each task, we construct four datasets of different qualities, as described in Section 5.1. We compare our algorithm against several baselines, including behavioral cloning (BC), offline implementation of soft actor-critic (offline SAC) (Haarnoja et al., 2018), batch-constrained deep Q-learning (BCQ) (Fujimoto et al., 2019), bootstrapping error accumulation reduction (BEAR) (Kumar et al., 2019), behavior regularized actor-critic with value penalty (BRAC-v) (Wu et al., 2019) and the $\mathcal{H}$-variant of conservative Q-learning (CQL($\mathcal{H}$)) (Kumar et al., 2020). For more details on the experimental setup and hyperparameters of each method, see Appendix C.

5.1 Offline Dataset

To comprehensively evaluate our method, we follow (Wu et al., 2019) to construct four offline datasets of different qualities (**mixed**, **medium**, **low**, **very low**) for each task. All these datasets have one million (1M) transitions collected with different combinations of multiple policies, as described below. **mixed**: all transitions observed during a training run of an online SAC algorithm until a specified iteration step is reached. **medium**: 0.2M transitions with a random policy π_r, 0.4M transitions with a partially trained policy π_p and 0.4M transitions with a perturbed version of π_p, i.e., $\tilde{\pi}_p$. **low**: 0.6M transitions with π_r, 0.2M transitions with π_p and 0.2M transitions with $\tilde{\pi}_p$. **very low**: 0.8M transitions with π_r, 0.1M transitions with π_p and 0.1M transitions with $\tilde{\pi}_p$. The performance of the partially trained policies (distinct from the behavior policies which have injected noise) and more details about how to generate these datasets are given in Appendix C.

Table 1: Evaluation results comparing BC, offline SAC, BCQ, BEAR, BRAC-v, CQL($\mathcal{H}$) and our AdaPT-AC on four datasets. All results are reported as the average undiscounted return of the learned policy over three independent trials with random seeds. Here we omit BRAC with policy regularization (BRAC-p) (Wu et al., 2019) because BRAC-v generally obtains higher performance than BRAC-p. The results of offline SAC are given in Appendix E due to the limitation of space.

Tasks	Datasets	BC	BCQ	BEAR	BRAC-v	CQL($\mathcal{H}$)	AdaPT-AC (ours)
Humanoid	mixed	4007±437	2347±230	2417±200	2168±250	271±10	**5001**±151
	medium	2631±211	1590±130	624±87	1537±334	136±59	**4282**±523
	low	582±82	966±152	259±15	1200±144	283±216	**4496**±381
	very low	406±100	574±53	219±39	866±150	178±94	**4487**±373
Walker2d	mixed	3711±116	3614±616	2347±257	3926±261	4669±14	**4724**±24
	medium	1683±12	1733±365	2246±131	2956±216	3347±74	**3785**±138
	low	851±426	1076±47	1474±845	2240±309	929±55	**3452**±12
	very low	1000±631	1248±31	1162±236	1550±588	441±18	**3360**±44
Hopper	mixed	3519±12	1753±285	2654±34	3532±76	3605±11	**3614**±16
	medium	1809±186	1641±495	2136±199	2729±559	2686±9	**3461**±22
	low	464±439	1401±509	2658±213	1789±953	2819±467	**3325**±41
	very low	720±398	1104±106	1975±782	747±406	**3045**±76	2740±121
Reacher	mixed	-3.3±0.3	-3.2±0.0	-3.2±0.2	-3.3±0.3	-3.3±0.0	**-2.8**±0.1
	medium	-4.7±0.2	-4.1±0.2	-3.7±0.1	-4.1±0.0	-4.0±0.0	**-3.3**±0.1
	low	-4.9±0.2	-4.4±0.1	-4.0±0.4	-4.3±0.1	-4.5±0.1	**-3.4**±0.2
	very low	-5.3±0.1	-4.2±0.0	-4.8±0.2	-4.3±0.2	-4.3±0.2	**-3.5**±0.3
HalfCheetah	mixed	8417±25	7782±201	8205±33	9201±26	8896±20	**9343**±29
	medium	4651±12	5392±99	4533±15	6268±35	5920±28	**6764**±77
	low	3590±490	5343±13	4805±27	6189±14	5899±12	**6672**±82
	very low	1823±914	5299±10	4753±76	6149±23	5750±66	**6286**±105
Ant	mixed	4922±55	3600±282	4974±102	4041±329	**5026**±135	4982±216
	medium	2276±64	2451±152	2514±40	3198±154	3031±70	**3324**±29
	low	2215±26	2272±87	2712±45	3241±96	2908±40	**3412**±70
	very low	2205±38	2246±88	2710±41	2911±49	2753±45	**3250**±53

It is worth noting that we do not directly use the D4RL (Fu et al., 2020), which is a recently proposed standard benchmark suite for offline RL, due to several reasons. First, the D4RL includes only three MuJoCo continuous tasks (HalfCheetah, Hopper and Walker2d). Second, we want to generate lower-quality datasets for evaluation, such as our **low** and **very low** datasets which contain *a large amount of random data* and induce bad behavior policies. Third, our **mixed** and **medium** datasets correspond to the *mixed* and *random-medium* datasets of D4RL, respectively. Therefore, our dataset suite, which includes two lower-quality datasets (i.e., **low** and **very low**), can be seen as a straightforward and important extension to D4RL. In addition, we do not compare different algorithms on completely random dataset, since almost all algorithms fail to achieve satisfactory performance.

5.2 Comparative Evaluation

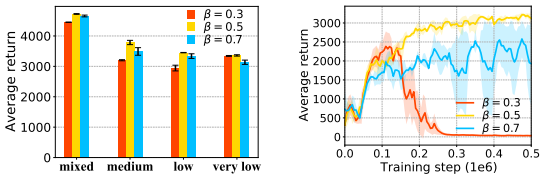
We compare AdaPT-AC against baselines by training on these fixed datasets and evaluating the learned policies on the real environments. Table 1 summarizes the performance of different algorithms across all tasks and datasets. The training curves are given in Appendix E. The empirical results show that AdaPT-AC substantially outperforms all baselines on almost all tasks and datasets, verifying the effectiveness of our method. On the hardest task, Humanoid-v2, AdaPT-AC achieves the best performance on all datasets, whereas the others fail to make great progress. In addition, we can observe that as the quality of dataset becomes worse or as the difficulty of dataset increases (i.e., from **medium** to **very low**), almost all baseline methods have an obvious performance degradation, since their performance is largely limited by that of the behavior policy. In contrast, our AdaPT-AC can still perform well on the **low** and **very low** datasets that contain a large amount of random data and induce bad behavior policies. This conclusion indicates that our AdaPT method is able to adaptively adjust the constraint imposed on the learned policy according to the quality of dataset, as discussed in Section 4.1. Moreover, it can be seen that although the **low** and **very low** datasets

induce bad behavior policies, our AdaPT-AC is able to learn a policy that can be improved to a great extend and is allowed to deviate far from the behavior policy.

5.3 Ablation Studies

To understand how each component in AdaPT-AC affects performance, we further conduct a through ablation study to examine how sensitive AdaPT-AC is to the choice of two main hyperparameters, namely correction coefficient β and maximum target divergence B. In practice, we found β and B to be the only two hyperparameters that require tuning for new tasks.

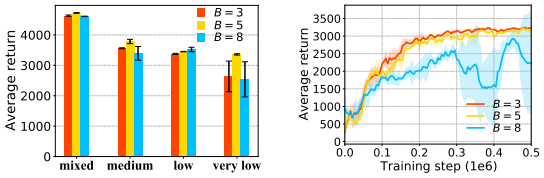
Correction coefficient. Since the induced policy of Q-function in the second term of Equation (9) is intractable when a parameterized policy is used, we instead apply a greater upper bound to update the temperature parameter, as shown in Equation (14). To mitigate the bias introduced by a greater upper bound, the target divergence is multiplied by a correction coefficient $\beta \in (0, 1)$. Figure 1 shows how learning performance and learning curve change when the correction coefficient is varied. The learning curves on other datasets are given in Appendix E. It can be seen that AdaPT-AC is slightly sensitive to the value of correction coefficient β. Specifically, a small β leads to a strict constraint on the learned policy, hence the learning is prone to collapse when bad transitions are sampled. For a large β, the agent is allowed to explore in a larger action space, resulting in unstable learning. With the right correction coefficient, the model balances policy constraint and exploration, leading to better performance and stable learning. In practice, we found that correction coefficient is not sensitive to different tasks, and 0.5 is an appropriate choice for most tasks.



(a) Walker2d (b) Walker2d (**low** dataset)

Figure 1: Learning performance and curves of AdaPT-AC with varying correction coefficient β.

Maximum target divergence. In principle, it is unnecessary to restrict the target divergence with B if the expectation can be calculated exactly in Equation (14). However, in our implementation, we have to approximate it by sampling a few actions for continuous domains. To avoid extreme distributional deviation resulted from out-of-distribution actions and stabilize the learning, the target divergence is bounded by a maximum value B. Figure 2 compares the final performance and learning curves of our AdaPT-AC with three different values of B. The learning curves on other datasets are given in Appendix E. We can see that AdaPT-AC is also slightly sensitive to the choice of maximum target divergence B. A large B allows the agent to explore on out-of-distribution actions that are far from the behavior policy, but at the cost of unstable learning. In practice, we just need to make a coarse grid search to choose an appropriate B.



(a) Walker2d (b) Walker2d (**low** dataset)

Figure 2: Learning performance and curves of AdaPT-AC with varying maximum target divergence B.

6 Conclusion

In this work, we proposed Adaptive Policy constrainT augmented Actor-Critic (AdaPT-AC), a practical model-free offline RL algorithm that can learn a generalizable policy even from the datasets that contain a large amount of random data. AdaPT-AC can adaptively adjust the constraint imposed on the learned policy to allow effective exploration on out-of-distribution actions. We theoretically show that AdaPT produces a tight upper bound on the distributional deviation between the learned policy and the behavior policy, and this upper bound is the minimum requirement to guarantee policy improvement at each iteration. In particular, our AdaPT-AC enables the agent to learn a policy that can be improved to a great extend and is allowed to deviate far from the behavior policy. Empirically, we demonstrate that AdaPT-AC outperforms several prior algorithms on a range of very complex and high-dimensional continuous control tasks, especially in the case of low-quality offline datasets. Our results suggest that AdaPT provides a promising avenue for achieving near-optimal performance with policy constraint based offline RL methods, and further exploring explicit and direct implementations of AdaPT is an exciting direction for future work.

REFERENCES

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

Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 176–185. PMLR, 2017.

Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 89–96, 2009.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 2017.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1):3207–3260, 2013.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, et al. A framework for data-driven robotics. *arXiv preprint arXiv:1909.12200*, 2019.

Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

Tim De Bruin, Jens Kober, Karl Tuyls, and Robert Babuška. The importance of experience replay database composition in deep reinforcement learning. In *Deep Reinforcement Learning Workshop, NeurIPS*, 2015.

Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 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, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. 2018.

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

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alexander J Smola. A kernel approach to comparing distributions. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, pp. 1637, 2007.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation*, pp. 3389–3396. IEEE, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 2018.

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.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673, 2018.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pp. 11784–11794, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 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.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2775–2785, 2017.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Trust-pcl: An off-policy trust region method for continuous control. In *International Conference on Learning Representations*, 2018.

Shamim Nemati, Mohammad M Ghassemi, and Gari D Clifford. Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 2978–2981. IEEE, 2016.

Brendan ODonoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The uncertainty bellman equation and exploration. In *International Conference on Machine Learning*, pp. 3836–3845, 2018.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, pp. 4026–4034, 2016.

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

Olivier Pietquin, Matthieu Geist, Senthilkumar Chandramohan, and Hervé Frezza-Buet. Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transactions on Speech and Language Processing*, 7(3):1–21, 2011.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *International Conference on Machine Learning*, pp. 417–424, 2001.

Wenjie Shi, Shiji Song, and Cheng Wu. Soft policy gradient method for maximum entropy deep reinforcement learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 3425–3431. AAAI Press, 2019.

Susan M Shortreed, Eric Laber, Daniel J Lizotte, T Scott Stroup, Joelle Pineau, and Susan A Murphy. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1-2):109–136, 2011.

Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. In *International Conference on Learning Representations*, 2020.

Thiago D Simão, Romain Laroche, and Rémi Tachet des Combes. Safe policy improvement with an estimated baseline policy. *arXiv preprint arXiv:1909.05236*, 2019.

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

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

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

Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Shangtong Zhang and Richard S Sutton. A deeper look at experience replay. *arXiv preprint arXiv:1712.01275*, 2017.

A PROOF OF PROPOSITION 1

Proof. By substituting Equation (4) into the RHS of Equation (5), we have

$$\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[\log\frac{\pi(a|s)}{\pi_b(a|s)}\right] \leq \mathbb{E}_{s\sim\mathcal{D},a\sim\pi^*}\left[\frac{Q^*(s,a)-V^*(s)}{\alpha}\right] \tag{15}$$

$$\leq \mathbb{E}_{s\sim\mathcal{D}}\left[\frac{Q^*(s,\arg\max_a Q^*(s,a))-V^*(s)}{\alpha}\right]. \tag{16}$$

The final inequality is obtained since the expected Q-value $\mathbb{E}_{a\sim\pi}[Q^*(s,a)]$ under any policy π is always not greater that $\max_a Q^*(s,a)$. □

B PROOF OF THEOREM 1

Proof. We will drop the state and action arguments from the following derivation for improved readability. According to the Bellman equation, the policy π_k at the last iteration step satisfies

$$\mathbb{E}_{\mathcal{D},\pi_k}\left[\log\frac{\pi_k}{\pi_b}\right] = \mathbb{E}_{\mathcal{D},\pi_k}\left[\frac{Q^{\pi_k}-V^{\pi_k}}{\alpha}\right] \leq \mathbb{E}_{\mathcal{D}}\left[\frac{\max_a Q^{\pi_k}-V^{\pi_k}}{\alpha}\right], \tag{17}$$

hence we get $\pi_k \in \Pi_k$. In the policy improvement step, since π_{k+1} is the optimal solution to Equation (8), it must be the case that

$$\begin{aligned} D_{KL}\big(\pi_{k+1}, \exp\{Q^{\pi_k}/\alpha + \log\pi_b - \log Z^{\pi_k,\alpha}\}\big) \leq \\ D_{KL}\big(\pi_k, \exp\{Q^{\pi_k}/\alpha + \log\pi_b - \log Z^{\pi_k,\alpha}\}\big), \end{aligned} \tag{18}$$

By rewriting the KL divergence in its tractable form and further replacing $\mathbb{E}_{\pi_k}[Q^{\pi_k} - \alpha\log\pi_k + \alpha\log\pi_b]$ with V^{π_k}, we obtain

$$V^{\pi_k} \leq \mathbb{E}_{\pi_{k+1}}\left[Q^{\pi_k} - \alpha\log\frac{\pi_{k+1}}{\pi_b}\right]. \tag{19}$$

Next, consider the behavior regularized Bellman equation:

$$Q^{\pi_k} = r + \gamma\mathbb{E}_{s'}[V^{\pi_k}] \tag{20}$$

$$\leq r + \gamma\mathbb{E}_{s',a'\sim\pi_{k+1}}\left[Q^{\pi_k} - \alpha\log\frac{\pi_{k+1}}{\pi_b}\right] \tag{21}$$

$$\leq r + \gamma\mathbb{E}_{s',a'\sim\pi_{k+1}}\left[r' - \alpha\log\frac{\pi_{k+1}}{\pi_b} + \gamma\mathbb{E}_{s''}[V^{\pi_k}]\right] \tag{22}$$

$$\vdots$$

$$\leq Q^{\pi_{k+1}}. \tag{23}$$

We keep expanding Q^{π_k} and then applying Equation (19) on the RHS, which converges to $Q^{\pi_{k+1}}$. It is worth noting that the above inequality only holds when $\pi_k \in \Pi_k$. Therefore, the second term of Equation (9) is the minimum to guarantee policy improvement in terms of its Q-function. □

C EXPERIMENTAL DETAILS

Dataset collection. We consider six challenging MuJoCo continuous control tasks. For each task, we generate four offline datasets of different qualities, i.e., **mixed**, **medium**, **low** and **very low**. Each dataset contains 1 million (1M) transitions, which are collected using a mixture of multiple policies of different qualities. To simulate the real-world scenarios, we first obtain a partially trained policy π_p by training a policy with off-policy SAC and performing online interaction until the policy performance achieves a performance threshold. The performance of the partially trained policies and corresponding performance thresholds are listed in Table 2. Then, we perturb π_p with ϵ-greedy ($\epsilon = 0.1$) noise, resulting in an exploration policy $\tilde{\pi}_p$. That is, at each step, $\tilde{\pi}_p$ has 0.1 probability to take a uniformly random action, otherwise takes the action sampled from π_p. In addition to π_p and $\tilde{\pi}_p$, we also use a uniform-at-random policy π_r to generate some random data, since there generally exists a large amount of random data in many industrial scenarios. The specific dataset collections are shown below:

Table 2: The performance of the partially trained policies and corresponding performance thresholds for each task.

Tasks	performance threshold	performance of the partially trained policy
Humanoid-v2	1500	1821
Walker2d-v2	1000	1092
Hopper-v2	1000	1136
Reacher-v2	-6.0	-5.8
HalfCheetah-v2	2500	2855
Ant-v2	1000	1007

- **mixed**: The dataset consists of all transitions observed during a training run of an online SAC algorithm until the maximum iteration step is reached.
- **medium**: The dataset is a mixture of three parts: 0.2M transitions are collected by the random walk policy π_r, 0.4M transitions are collected by purely executing π_p, and the remaining 0.4M transitions are collected by the perturbed policy $\tilde{\pi}_p$.
- **low**: This dataset is also a mixture of three parts: 0.6M transitions are collected by the random walk policy π_r, 0.2M transitions are collected by purely executing π_p, and the remaining 0.2M transitions are collected by the perturbed policy $\tilde{\pi}_p$.
- **very low**: Similar to the **medium** and **low** datasets, 0.8M transitions are collected by the random walk policy π_r, 0.1M transitions are collected by purely executing π_p, and the remaining 0.1M transitions are collected by the perturbed policy $\tilde{\pi}_p$.

Implementation details. Most of our baselines build on the open source implementations [1] released by the authors of BRAC, hence we refer the readers to the Appendix A of Wu et al. (2019) for a detailed description. Here we only give the implementation details of our AdaPT-AC algorithm. For all function approximators, we use fully-connected neural networks with RELU activations. For policy networks, we use tanh (Gaussian) on outputs following BEAR (Kumar et al., 2019). The sizes of policy network and Q-network are (200, 200) and (300, 300), respectively. As in other deep RL algorithms, we maintain source and target Q-functions with an update rate 0.005 per iteration. The distributional deviation is calculated by sampling 4 actions. As discussed previously, the target divergence or the adaptive upper bound is calculated by sampling 4 actions with a Gaussian policy. While the expectation of this Gaussian policy is obtained by sampling 10 actions from the learned policy π_θ and then take the one with the highest learned Q-value, the variance is directly chosen as the variance of the learned policy. We use Adam for actor, critic and temperature parameter optimizers, the learning rates of which are 3e-4, 1e-3 and 3e-5, respectively. The maximum training step is set to 1×10^6. At test time, we follow (Wu et al., 2019; Kumar et al., 2019) by sampling 10 actions from π_θ at each step and take the one with the highest learned Q-value.

Hyperparameters. For all baselines, we use the hyperparameters reported in Wu et al. (2019), which has performed a grid search over some important hyperparameters. For our AdaPT-AC, we do a grid search over correction coefficient β and maximum target divergence B. Specifically, we search correction coefficient over three values (0.3, 0.5, 0.7) on walker2d-v2 and then use the best one for all tasks. The maximum target divergence is sensitive to different tasks, hence we search over five values (3, 5, 10, 20, 50) for each task. The correction coefficient and maximum target divergence used on each task are listed in Table 3. Other hyperparameters, were kept identical to the BRAC algorithm implementation, inlcuding the twin Q-function trick, soft-target updates, etc.

[1] https://github.com/google-research/google-research/tree/master/behavior_regularized_offline_rl

Table 3: The correction coefficient and maximum target divergence used on each task for our AdaPT-AC algorithm.

Tasks	correction coefficient β	maximum target divergence B			
		mixed	**medium**	**low**	**very low**
Humanoid-v2	0.5	10	10	10	10
Walker2d-v2	0.5	5	3	5	5
Hopper-v2	0.5	3	3	3	3
Reacher-v2	0.5	10	10	10	20
HalfCheetah-v2	0.5	50	50	50	50
Ant-v2	0.5	10	5	10	10

D ALGORITHM

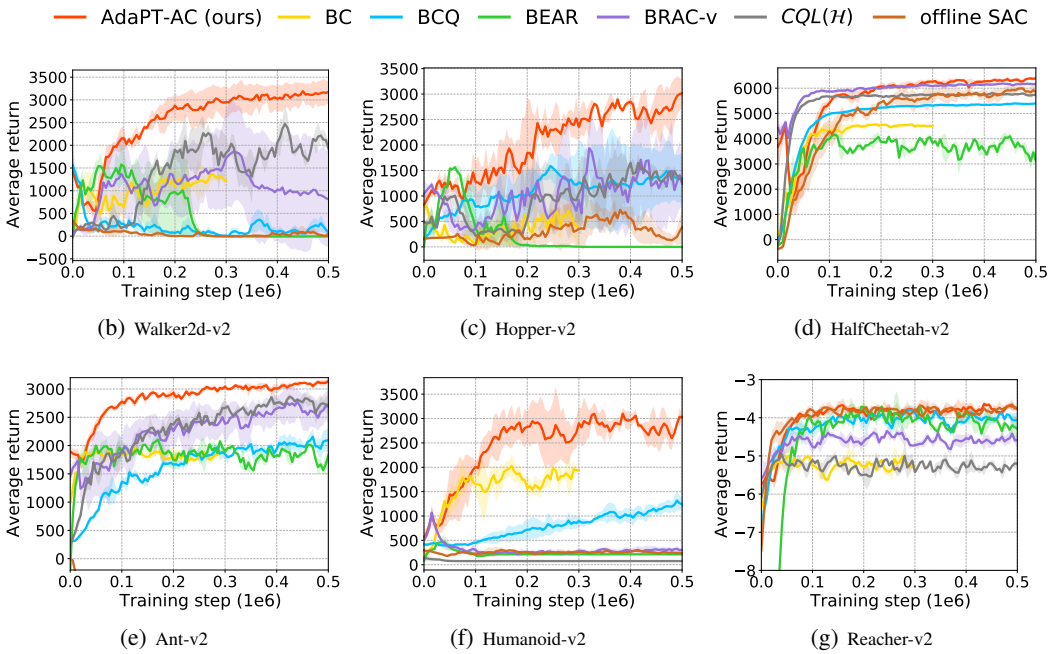
Algorithm 1: AdaPT-AC: Adaptive Policy constrainT augmented Actor-Critic

Randomly initialize critic Q_θ and actor π_ϕ with weights θ and ϕ;
Initialize target network $Q_{\theta'}$ with $\theta' \leftarrow \theta$, and temperature parameter α with $\alpha \leftarrow \alpha_0$;
Set offline dataset $\mathcal{D}$, target update rate τ, minibatch size N, learning rates $\{\lambda_\theta, \lambda_\phi, \lambda_\alpha\}$;
for *each gradient step* **do**
    Sample N transitions (s, a, r, s') from $\mathcal{D}$;
    Update critic network: $\theta \leftarrow \theta - \lambda_\theta \nabla_\theta J_Q(\theta)$
    Update actor network: $\phi \leftarrow \phi - \lambda_\phi \nabla_\phi J_\pi(\phi)$
    Update temperature parameter: $\alpha \leftarrow \alpha - \lambda_\alpha \nabla_\alpha J_\alpha(\alpha)$
    Update target network: $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$
end

E ADDITIONAL RESULTS

Table 4: Evaluation results of offline SAC on four datasets. All results are reported as the average undiscounted return of the learned policy over three independent trials with random seeds.

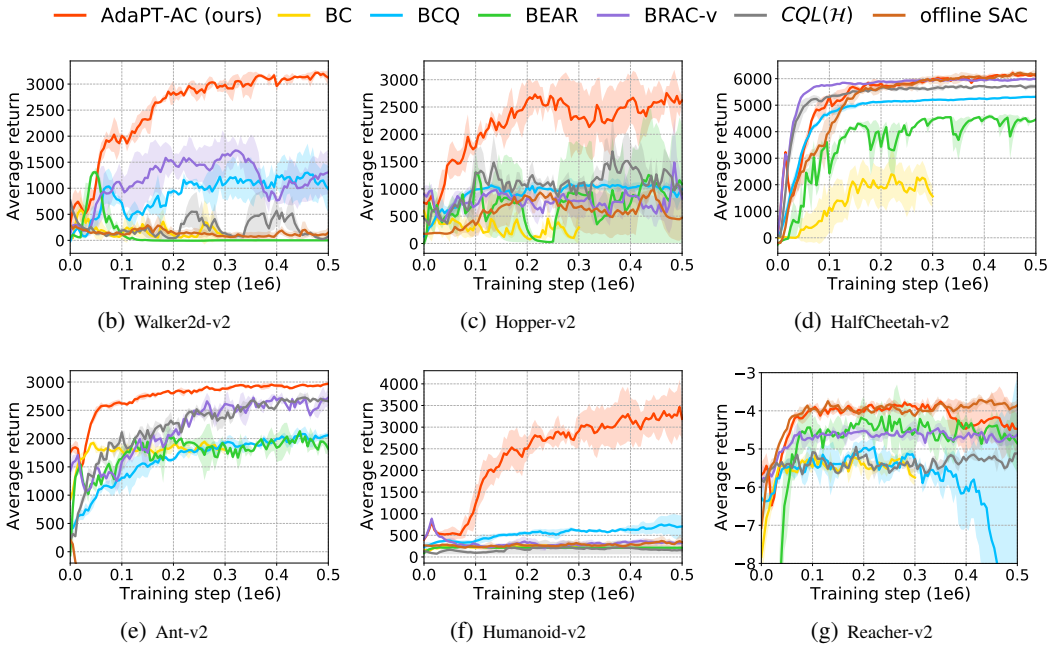
Tasks	Datasets	offline SAC	Tasks	Datasets	offline SAC
Humanoid	mixed	1047±488	Walker2d	mixed	1731±416
	medium	451±23		medium	916±70
	low	629±25		low	892±82
	very low	672±58		very low	827±65
Hopper	mixed	3584±18	Reacher	mixed	-2.8±0.1
	medium	2716±464		medium	-3.2±0.2
	low	1730±830		low	-3.2±0.1
	very low	1618±721		very low	-3.2±0.1
HalfCheetah	mixed	9010±25	Ant	mixed	99±151
	medium	6585±34		medium	229±235
	low	6538±92		low	240±248
	very low	6292±98		very low	240±244



(b) Walker2d-v2 (c) Hopper-v2 (d) HalfCheetah-v2

(e) Ant-v2 (f) Humanoid-v2 (g) Reacher-v2

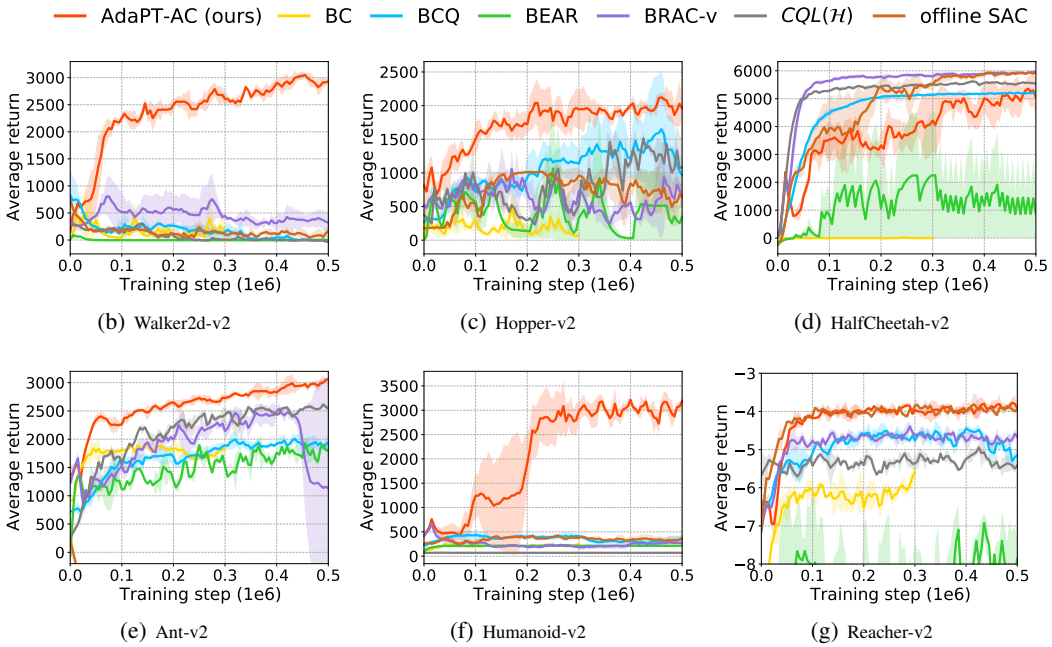
Figure 3: Learning curves of different algorithms on the **medium** datasets.



(b) Walker2d-v2 (c) Hopper-v2 (d) HalfCheetah-v2

(e) Ant-v2 (f) Humanoid-v2 (g) Reacher-v2

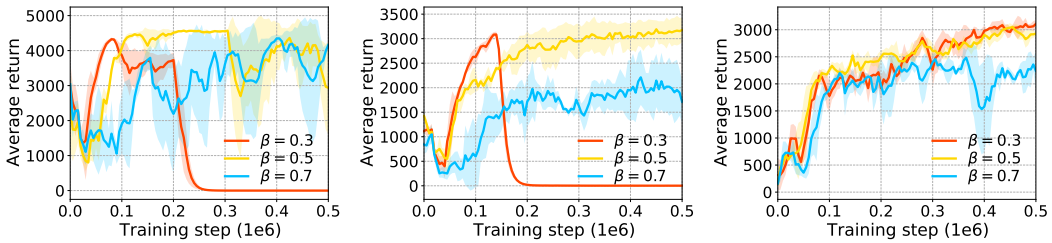
Figure 4: Learning curves of different algorithms on the **low** datasets.



(b) Walker2d-v2 (c) Hopper-v2 (d) HalfCheetah-v2

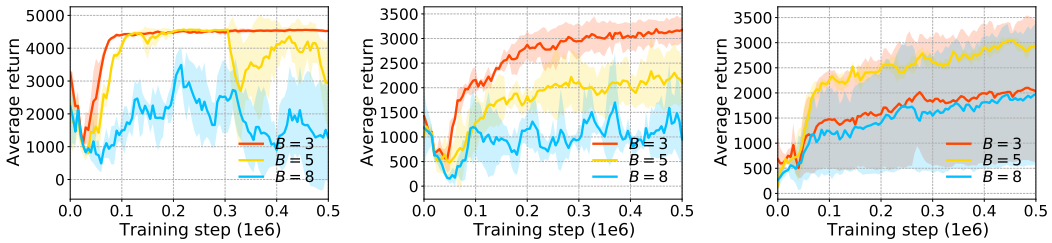
(e) Ant-v2 (f) Humanoid-v2 (g) Reacher-v2

Figure 5: Learning curves of different algorithms on the **very low** datasets.



(a) Walker2d (**mixed** dataset) (b) Walker2d (**medium** dataset) (c) Walker2d (**very low** dataset)

Figure 6: Learning curves of AdaPT-AC with varying correction coefficient β.



(a) Walker2d (**mixed** dataset) (b) Walker2d (**medium** dataset) (c) Walker2d (**very low** dataset)

Figure 7: Learning curves of AdaPT-AC with varying maximum target divergence B.