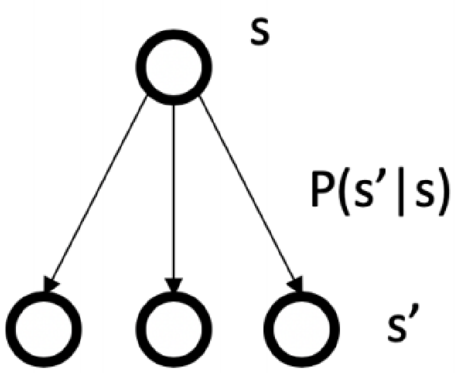
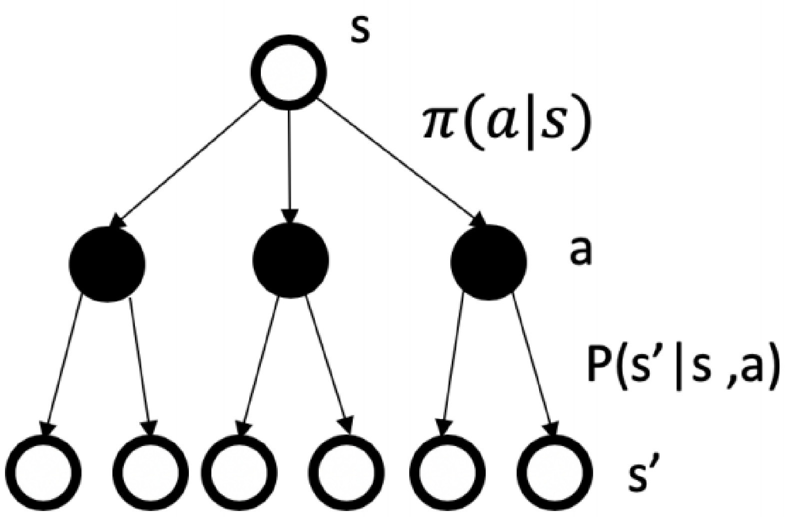
南京航空航天大學
Nanjing University of Aeronautics and Astronautics

Off-Policy Deep Reinforcement Learning without Exploration

Scott Fujimoto [1 2] David Meger [1 2] Doina Precup [1 2]

Background-MDP





$$v^{\pi}(s) = \sum_{a \in A} \pi(a|s)\left(R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) v^{\pi}(s')\right)$$

$$q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \sum_{a' \in A} \pi(a'|s') q^{\pi}(s',a')$$

Background-BatchRL

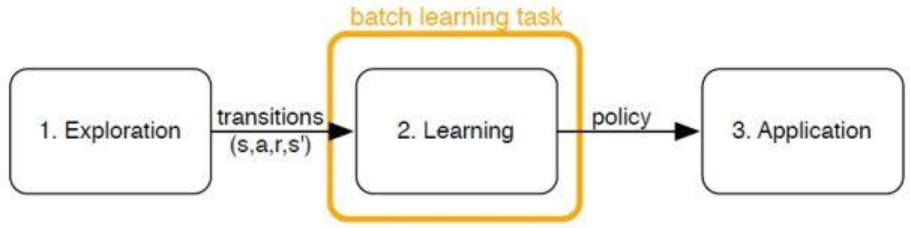


Fig. 1 The three distinct phases of the batch reinforcement learning process: 1: Collecting transitions with an arbitrary sampling strategy. 2: Application of (batch) reinforcement learning algorithms in order to learn the best possible policy from the set of transitions. 3: Application of the learned policy. Exploration is not part of the batch learning task. During the application phase, that isn't part of the learning task either, policies stay fixed and are not improved further.

Differences between RL and BatchRL

· Traditional RL update its Q or V by the experiences collected recently,but Batch RL's experiences come from different policy.Because Traditional RL's experiences are usually collected by the behavior agent which has been iterated several epochs ago,sharing the similar (s,a) distribution with the current behavior agent.But the Batch RL collects its experiences by expert behavior or even some trash agents,so it's hard for Batch RL to learn correctly from these out-of-distribution experiences.

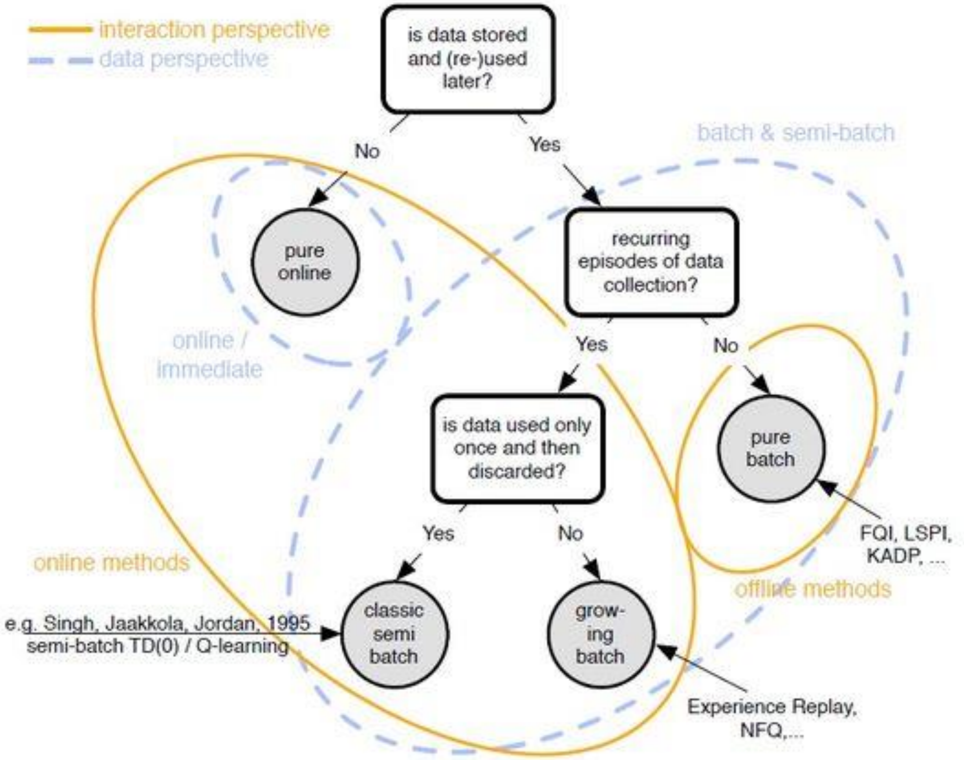


Fig. 4 Classification of batch vs. non-batch algorithms. With the interaction perspective and the data-usage perspective there are at least two different perspectives with which to define the category borders.

Background-BatchRL

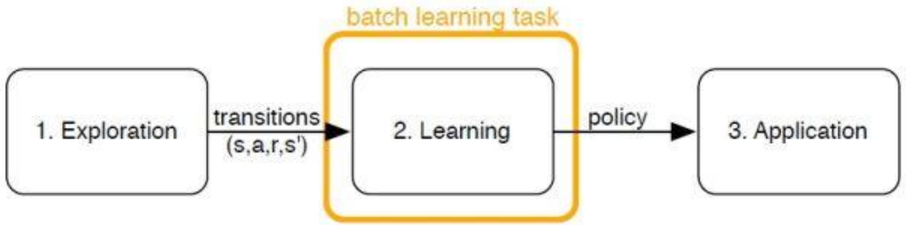


Fig. 1 The three distinct phases of the batch reinforcement learning process: 1: Collecting transitions with an arbitrary sampling strategy. 2: Application of (batch) reinforcement learning algorithms in order to learn the best possible policy from the set of transitions. 3: Application of the learned policy. Exploration is not part of the batch learning task. During the application phase, that isn't part of the learning task either, policies stay fixed and are not improved further.

Differences between IL and BatchRL

· IL usually needs the experiences collected by the expert, because IL extrapolate the actions only by the states.But batch RL can use any experiences(with high coverage of (s,a)),because batch RL can extrapolate the Q or V from the now state,and then choose actions according it.

· In conclusion,IL aims at making connections between states and actions,but Batch RL making connections between states and the estimated Q value of (s,a) or V value of (s).

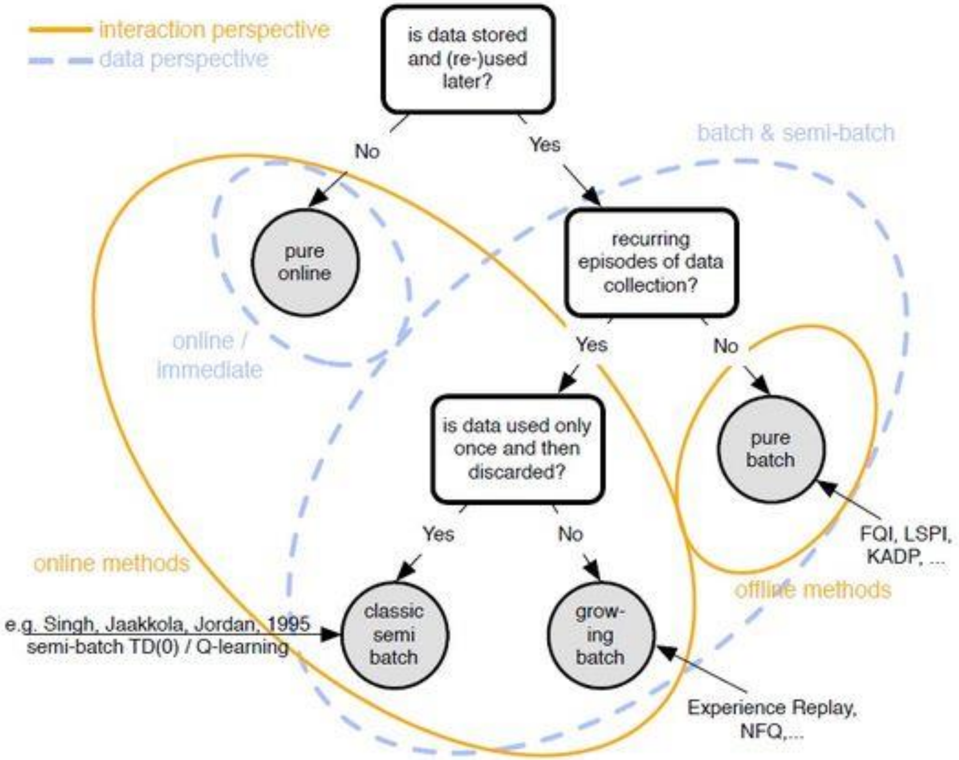


Fig. 4 Classification of batch vs. non-batch algorithms. With the interaction perspective and the data-usage perspective there are at least two different perspectives with which to define the category borders.

Analysis of Extrapolation Error

· **Absent Data**

When (s,a,r,s’) is chosen to update the Q value,we need the optimal action of s’ called a’,then we use Q(s’,a’) and r to update Q(s,a).However,if we didn’t have (s’,a’) in the batch,the Q(s’,a’) will be wrongly estimated which can causes Q(s,a) wrong too.

· **Model bias**

When facing stachostic MDP, choosing a in state s may lead to the state s'_1 and s'_2 with different probabilities. But if the batch only have (s,a,r, s'_1) in,it may cause the equation below being estimated wrongly.

$$\mathcal{T}^{\pi} Q(s,a) \approx \mathbb{E}_{s' \sim \mathcal{B}}[r + \gamma Q(s', \pi(s'))],$$

· **Training mismatch**

The same with other two problems.

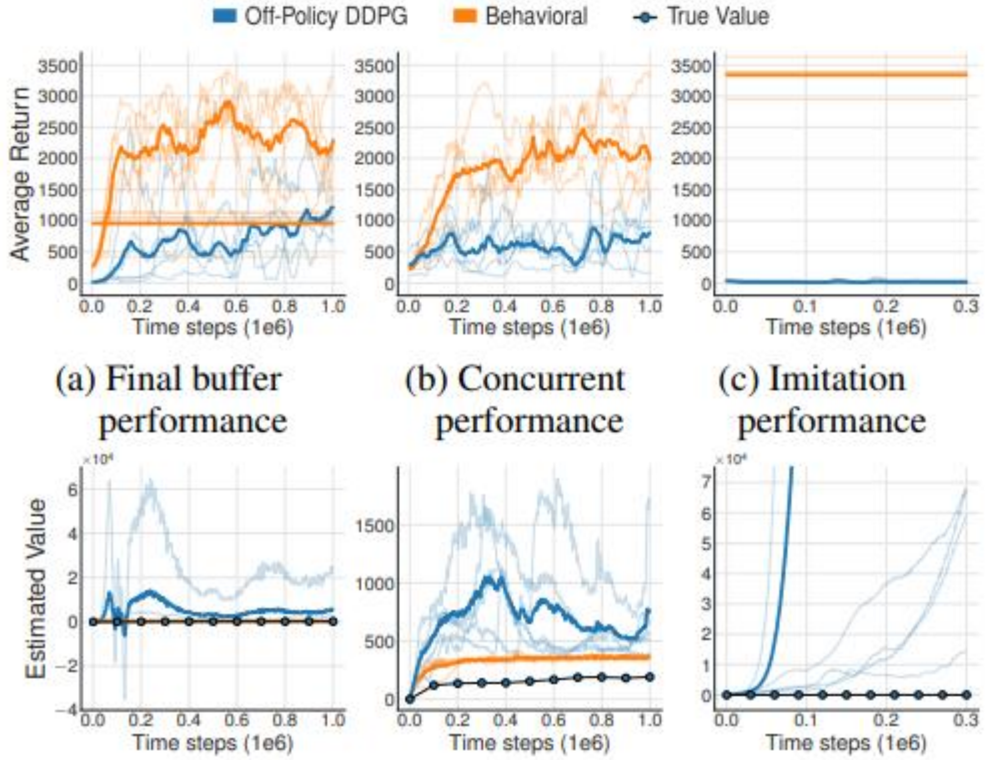
Like PPO and TRPO,off-plicy methods which bound the agent update in a limitation which prevents from out-of-distribution.

Analysis of Extrapolation Error

Batch 1 (Final buffer). We train a DDPG agent for 1 million time steps, adding $\mathcal{N}(0, 0.5)$ Gaussian noise to actions for high exploration, and store all experienced transitions. This collection procedure creates a dataset with a diverse set of states and actions, with the aim of sufficient coverage.

Batch 2 (Concurrent). We concurrently train the off-policy and behavioral DDPG agents, for 1 million time steps. To ensure sufficient exploration, a standard $\mathcal{N}(0, 0.1)$ Gaussian noise is added to actions taken by the behavioral policy. Each transition experienced by the behavioral policy is stored in a buffer replay, which both agents learn from. As a result, both agents are trained with the identical dataset.

Batch 3 (Imitation). A trained DDPG agent acts as an expert, and is used to collect a dataset of 1 million transitions.



(a) Final buffer performance
(b) Concurrent performance
(c) Imitation performance
(d) Final buffer value estimate
(e) Concurrent value estimate
(f) Imitation value estimate

Analysis of Extrapolation Error

Theorem 1. *Performing Q-learning by sampling from a batch $\mathcal{B}$ converges to the optimal value function under the MDP $M_{\mathcal{B}}$.*

Remark 1. *For any policy π and state-action pair (s, a), the error term $\epsilon_{\text{MDP}}(s, a)$ satisfies the following Bellman-like equation:*

$$\begin{aligned}\epsilon_{\text{MDP}}(s,a) = &\sum_{s'} \left(p_M(s'|s,a) - p_{\mathcal{B}}(s'|s,a)\right)\left(r(s,a,s') + \gamma \sum_{a'} \pi(a'|s')\left(Q^{\pi}_{\mathcal{B}}(s',a')\right)\right) \\ &+ p_M(s'|s,a)\gamma \sum_{a'} \pi(a'|s')\epsilon_{\text{MDP}}(s',a').\end{aligned} \tag{16}$$

Lemma 1. *For all reward functions, $\epsilon^{\pi}_{\text{MDP}} = 0$ if and only if $p_{\mathcal{B}}(s'|s,a) = p_M(s'|s,a)$ for all $s' \in \mathcal{S}$ and (s, a) such that $\mu_\pi(s) > 0$ and $\pi(a|s) > 0$.*

Proof. From Remark 1, we note that the form of $\epsilon_{\text{MDP}}(s,a)$, since no assumptions can be made on the reward function and therefore the expression $r(s,a,s') + \gamma \sum_{a'} \pi(a'|s')Q^{\pi}_{\mathcal{B}}(s',a')$, we have that $\epsilon_{\text{MDP}}(s,a) = 0$ if and only if $p_{\mathcal{B}}(s'|s,a) = p_M(s'|s,a)$ for all $s' \in \mathcal{S}$ and $p_M(s'|s,a)\gamma \sum_{a'} \pi(a'|s')\epsilon_{\text{MDP}}(s',a') = 0$.

($\Rightarrow$) Now we note that if $\epsilon_{\text{MDP}}(s,a) = 0$ then $p_M(s'|s,a)\gamma \sum_{a'} \pi(a'|s')\epsilon_{\text{MDP}}(s',a') = 0$ by the relationship defined by Remark 1 and the condition on the reward function. It follows that we must have $p_{\mathcal{B}}(s'|s,a) = p_M(s'|s,a)$ for all $s' \in \mathcal{S}$.

($\Leftarrow$) If we have $\sum_{s'} |p_M(s'|s,a) - p_{\mathcal{B}}(s'|s,a)| = 0$ for all (s,a) such that $\mu_\pi(s) > 0$ and $\pi(a|s) > 0$, then for any (s,a) under the given conditions, we have $\epsilon(s,a) = \sum_{s'} p_M(s'|s,a)\gamma \sum_{a'} \pi(a'|s')\epsilon(s',a')$. Recursively expanding the ϵ term, we arrive at $\epsilon(s,a) = 0 + \gamma 0 + \gamma^2 0 + \ldots = 0$.

Analysis of Extrapolation Error

Theorem 2. *For a deterministic MDP and all reward functions, $\epsilon^{\pi}_{MDP} = 0$ if and only if the policy π is batch-constrained. Furthermore, if $\mathcal{B}$ is coherent, then such a policy must exist if the start state $s_0 \in \mathcal{B}$.*

Theorem 3. *Given the Robbins-Monro stochastic convergence conditions on the learning rate α, and standard sampling requirements from the environment, BCQL converges to the optimal value function Q^*.*

Theorem 4. *Given a deterministic MDP and coherent batch $\mathcal{B}$, along with the Robbins-Monro stochastic convergence conditions on the learning rate α and standard sampling requirements on the batch $\mathcal{B}$, BCQL converges to $Q^{\pi}_{\mathcal{B}}(s, a)$ where $\pi^*(s) = \operatorname{argmax}_{a \text{ s.t.} (s,a)\in\mathcal{B}} Q^{\pi}_{\mathcal{B}}(s, a)$ is the optimal batch-constrained policy.*

Algotirhm

Algorithm 1 BCQ

Input: Batch $\mathcal{B}$, horizon T, target network update rate τ, mini-batch size N, max perturbation Φ, number of sampled actions n, minimum weighting λ.

Initialize Q-networks $Q_{\theta_1}, Q_{\theta_2}$, perturbation network ξ_ϕ, and VAE $G_\omega = \{E_{\omega_1}, D_{\omega_2}\}$, with random parameters θ_1, θ_2, ϕ, ω, and target networks $Q_{\theta'_1}, Q_{\theta'_2}$, $\xi_{\phi'}$ with $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2, \phi' \leftarrow \phi$.

for $t = 1$ **to** T **do**

- Sample mini-batch of N transitions (s, a, r, s') from $\mathcal{B}$
- $\mu, \sigma = E_{\omega_1}(s, a), \quad \tilde{a} = D_{\omega_2}(s, z), \quad z \sim \mathcal{N}(\mu, \sigma)$
- $\omega \leftarrow \mathrm{argmin}_\omega \sum (a - \tilde{a})^2 + D_{\mathrm{KL}}(\mathcal{N}(\mu, \sigma) || \mathcal{N}(0, 1))$
- Sample n actions: $\{a_i \sim G_\omega(s')\}_{i=1}^n$
- Perturb each action: $\{a_i = a_i + \xi_\phi(s', a_i, \Phi)\}_{i=1}^n$
- Set value target y (Eqn. 13)
- $\theta \leftarrow \mathrm{argmin}_\theta \sum (y - Q_\theta(s, a))^2$
- $\phi \leftarrow \mathrm{argmax}_\phi \sum Q_{\theta_1}(s, a + \xi_\phi(s, a, \Phi)), a \sim G_\omega(s)$
- Update target networks: $\theta'_i \leftarrow \tau\theta + (1 - \tau)\theta'_i$
- $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$

end for

$$r + \gamma \max_{a_i} \left[\lambda \min_{j=1,2} Q_{\theta'_j}(s', a_i) + (1 - \lambda) \max_{j=1,2} Q_{\theta'_j}(s', a_i) \right] \tag{13}$$

The equation of Q_target

$$\pi(s) = \operatorname*{argmax}_{a_i + \xi_\phi(s, a_i, \Phi)} Q_\theta(s, a_i + \xi_\phi(s, a_i, \Phi)), \quad \{a_i \sim G_\omega(s)\}_{i=1}^n. \tag{11}$$

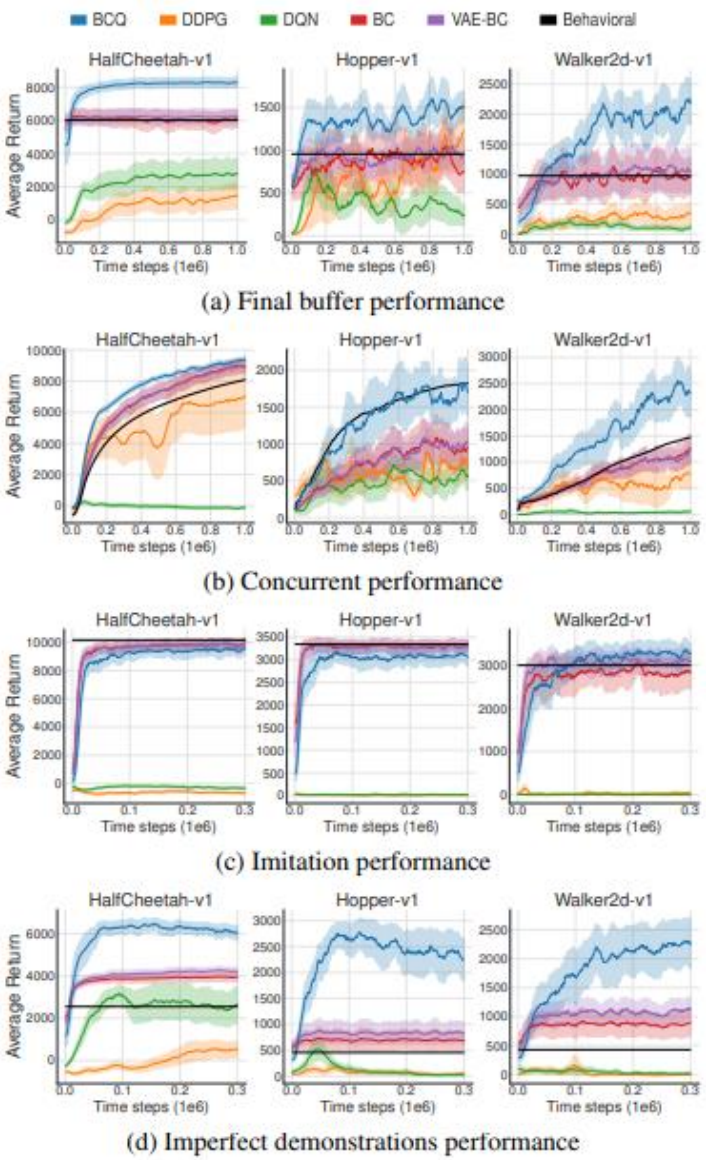
VAE introduce action and action distribution like PPO

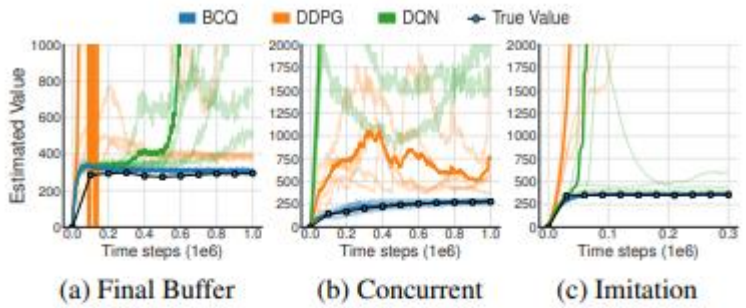
$$\phi \leftarrow \operatorname*{argmax}_\phi \sum_{(s,a) \in \mathcal{B}} Q_\theta(s, a + \xi_\phi(s, a, \Phi)). \tag{12}$$

Update pertube net to find the maximum Q Value

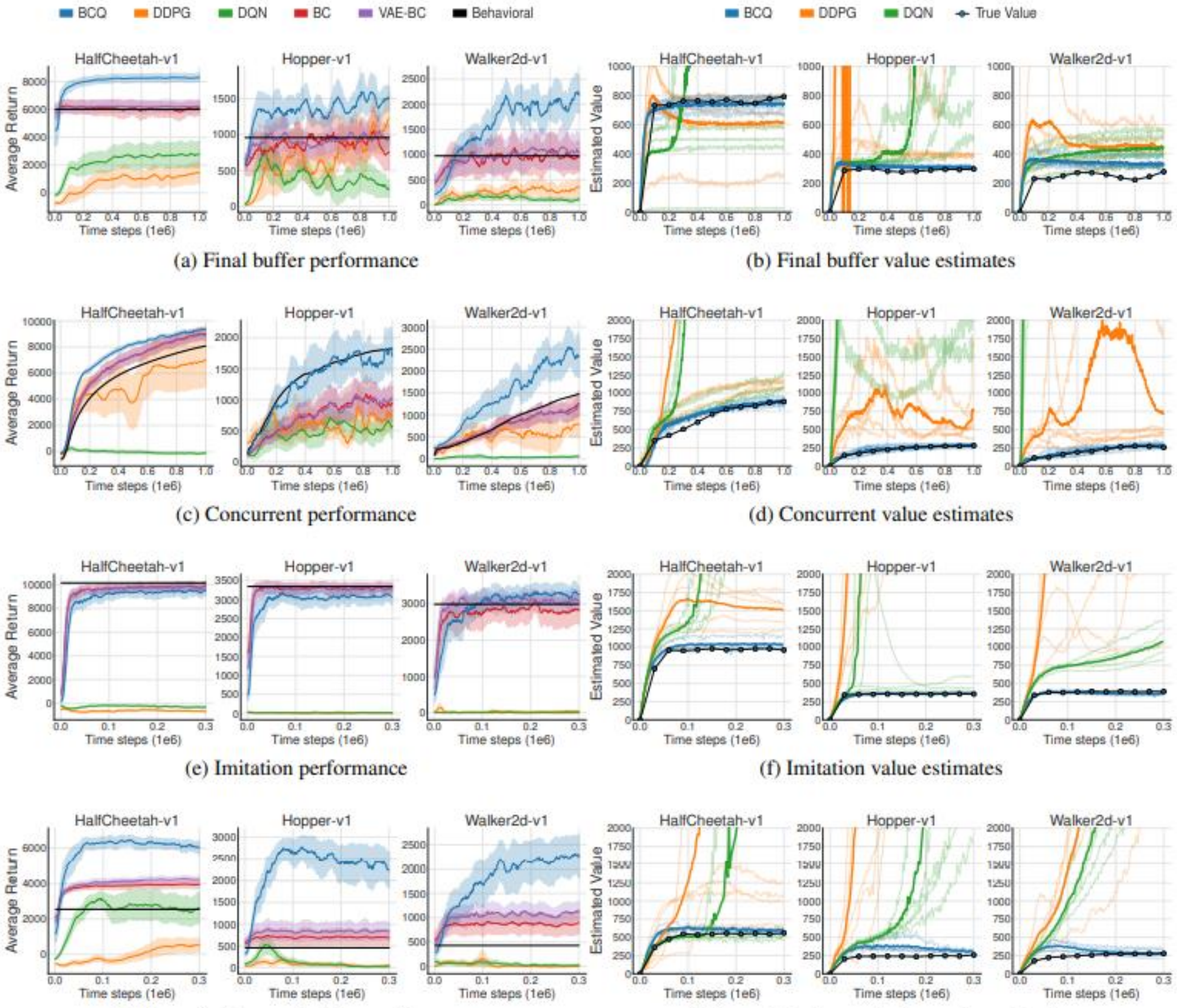
VAE用于bound action的范围，扰动网络用于寻找出最优的动作（diversity）

Experiment


BCQ
DDPG
DQN
BC
VAE-BC
Behavioral
HalfCheetah-v1
Hopper-v1
Walker2d-v1
Average Return
Time steps (1e6)
(a) Final buffer performance
(b) Concurrent performance
(c) Imitation performance
(d) Imperfect demonstrations performance


BCQ
DDPG
DQN
True Value
Estimated Value
Time steps (1e6)
(a) Final Buffer
(b) Concurrent
(c) Imitation

Experiment



(a) Final buffer performance

(b) Final buffer value estimates

(c) Concurrent performance

(d) Concurrent value estimates

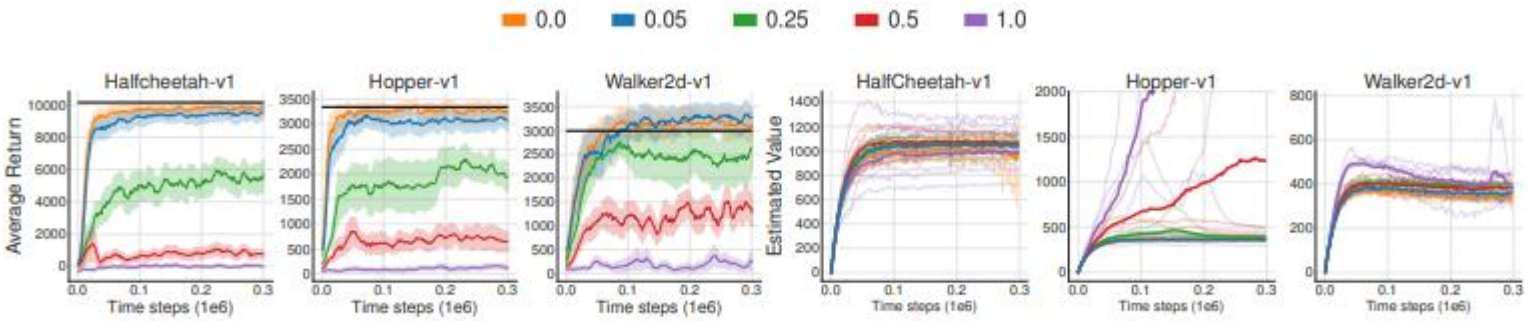
(e) Imitation performance

(f) Imitation value estimates

(g) Imperfect demonstrations performance

(h) Imperfect demonstrations value estimates

Experiment



(a) Imitation performance (b) Imitation value estimates

THANKS