Ensemble-based Offline Reinforcement Learning with Adaptive Behavior Cloning

Danyang Wang[1]; Lingsong Zhang[1]

[1]Department of Statistics, Purdue University

Algorithm 1 Ensemble-based Actor Critic with Adaptive Behavior Cloning (EABC)

Input: offline dataset $\mathcal{D}$, number of Q-ensembles K, confidence level $p \in [0,1]$. Initialize critic network ensemble Q_{θ_i} for $i = 1, \ldots K$, and actor network π_ϕ, with random parameters θ_i's, ϕ.
Initialize target networks $\tilde{\theta}_i \leftarrow \theta_i$; $\tilde{\phi} \leftarrow \phi$.
for $i = 1$ **to** T **do**
  Sample batch of N transitions $\{(s, a, r, s', d)\}$ from $\mathcal{D}$.
  $\tilde{a}' \leftarrow \pi_{\tilde{\phi}}(s') + \epsilon, \quad \epsilon \sim clip(\mathcal{N}(0, \tilde{\sigma}), -c, c)$.
  Compute $\texttt{pess}(Q_{\tilde{\theta}}(s', \tilde{a}'))$.
  $y = r + \gamma(1-d)\texttt{pess}(Q_{\tilde{\theta}}(s', \tilde{a}'))$.
  Update critics: $\theta_i \leftarrow \operatorname{argmin}_{\theta_i} N^{-1} \sum (Q_{\theta_i}(s,a) - y)^2$.
  if t % *policy update frequency* $== 0$ **then**
    $\tilde{a} \leftarrow \pi_\phi(s)$.
    Compute $\texttt{pess}(Q_\theta(s, \tilde{a}))$.
    $\lambda = \frac{\alpha}{N^{-1}\sum |\texttt{pess}(Q_\theta(s,\tilde{a}))|}$.
    Sample $w(s,a) \sim Bernoulli(p)$.
    Update actor: $\phi \leftarrow \operatorname{argmin}_{\phi} N^{-1} \sum \left[-\lambda \texttt{pess}(Q_\theta(s,\tilde{a})) + w(s,a)(\pi_\phi(s) - a)^2 \right]$.
    Update target networks: $\tilde{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\tilde{\theta}_i$; $\tilde{\phi} \leftarrow \tau\phi + (1-\tau)\tilde{\phi}$.
  end if
end for

Introduction

- Offline reinforcement learning (RL) algorithm TD3+BC [1] achieved state-of-the-art performance when it was proposed. However, it performs poorly on offline datasets with inferior behavior policy.
- We propose an offline RL algorithm, *Ensemble-based actor-critic with Adaptive Behavior Cloning* (EABC), built on TD3+BC, aiming to improve performance on datasets collected with inferior behavior policy.
- We use a *pessimistic ensemble of Q-value estimates* to reduce variance, and leverage a *weight function* with user-specified confidence level p to adjust the extent of behavior cloning, accounting for the quality of the underlying offline dataset.
- See Algorithm 1 to the left. The code is at the website: https://github.com/Penguin0007/EABC.

Adjust the extent of behavior cloning based on the quality of behavior policy

Given a user specified confidence level $p \in [0,1]$, we adjust the extent of BC through a Bernoulli random variable.

$$w(s,a) = \begin{cases} 1 & \text{with probability } p \\ 0 & \text{with probability } 1-p. \end{cases}$$

- Essentially a hyperparameter fine-tuning issue, a common challenge in almost all algorithms.
- A set of five numbers $\{0.0, 0.25, 0.5, 0.75, 1.0\}$, covering representative scenarios of p.

Boosting the performance with a pessimistic Q value ensemble

Simple implementation of ensemble: $\{Q_{\theta_i}\}_{i=1}^K$, with default number of ensembles K constrained to 10.

$$\bar{Q}_\theta := \frac{1}{K}\left(\sum_{i=1}^K Q_{\theta_i}\right)$$

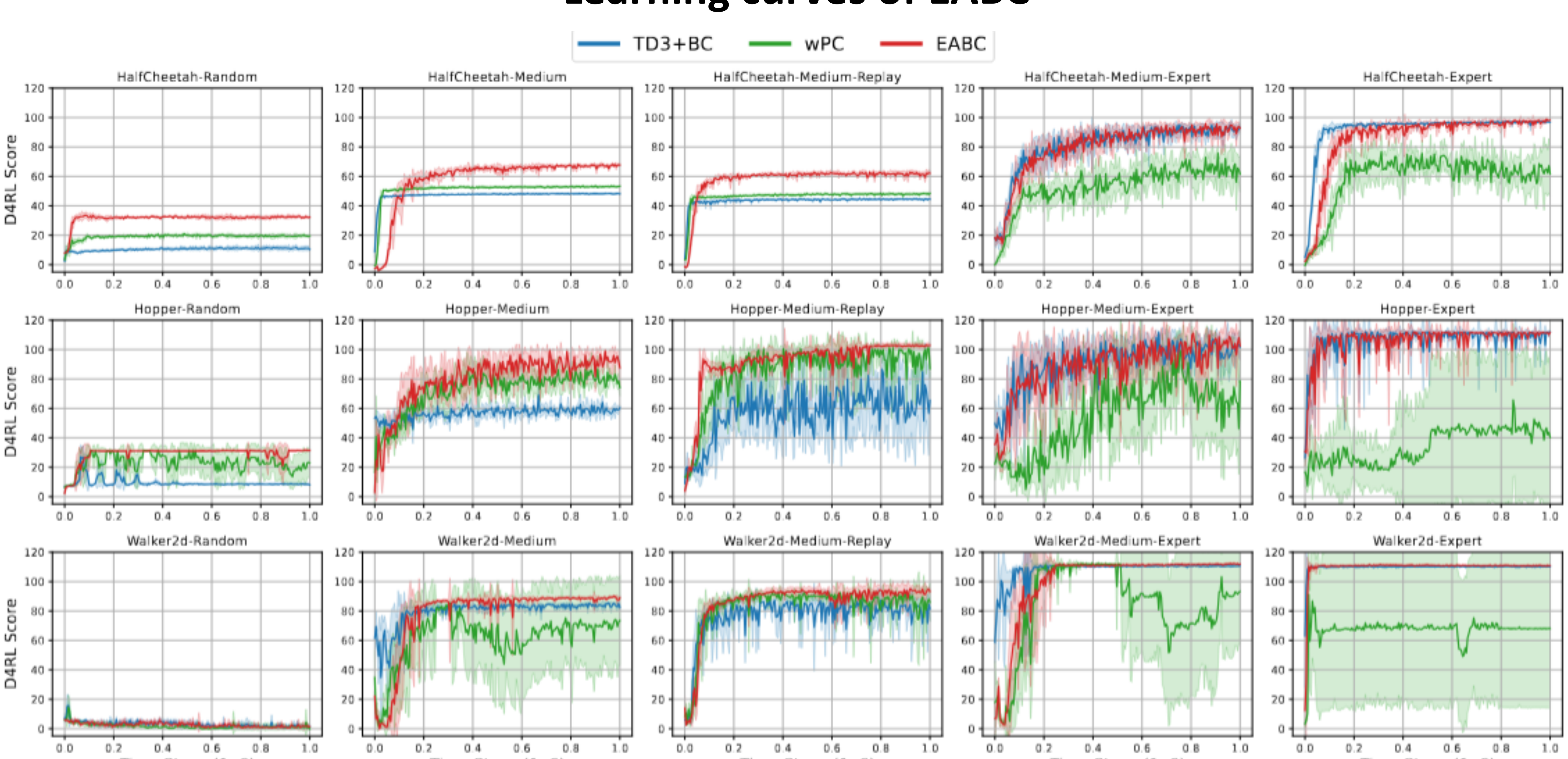
$$\widehat{\sigma} := \sqrt{\frac{1}{K-1}\sum_{i=1}^K (Q_{\theta_i} - \bar{Q}_\theta)^2}$$

$$\texttt{pess}\left(Q_\theta(s,a)\right) = \bar{Q}_\theta(s,a) - \widehat{\sigma}$$

D4RL Benchmark Experiments

Task Name	BC	TD3	TD3+BC	CQL	IQL	wPC	EABC (ours)
halfcheetah-r	2.2±0.0	32.0±2.2	11.0±1.1	17.5±1.5	13.1±1.3	19.7±0.8	**32.4**±0.7
hopper-r	3.7±0.6	26.8±5.1	8.5±0.6	7.9±0.4	7.9±0.2	20.9±9.4	**31.5**±0.4
walker2d-r	1.3±0.1	-0.1±0.2	1.6±1.7	5.1±1.3	**5.4**±1.2	1.3±2.3	1.7±1.7
halfcheetah-m	43.2±0.6	33.8±11.8	48.3±0.3	47.0±0.5	47.4±0.2	53.2±0.3	**67.3**±0.9
hopper-m	54.1±3.8	0.7±0.0	59.3±4.2	53.0±28.5	66.2±5.7	79.4±2.0	**92.4**±3.9
walker2d-m	70.9±11.0	0.6±1.0	83.7±2.1	73.3±17.7	78.3±8.7	71.0±31.6	**89.0**±0.6
halfcheetah-m-r	37.6±2.1	42.3±7.8	44.6±0.5	45.5±0.7	44.2±1.2	48.1±0.4	**61.4**±1.6
hopper-m-r	16.6±4.8	44.4±23.8	60.9±18.8	88.7±12.9	94.7±8.6	94.5±3.8	**102.6**±1.4
walker2d-m-r	20.3±9.8	31.0±14.2	81.8±5.5	81.8±2.7	73.8±7.1	84.0±11.0	**93.2**±2.9
halfcheetah-m-e	44.0±1.6	6.2±7.1	90.7±4.3	75.6±25.7	86.7±5.3	63.7±10.8	**92.9**±1.9
hopper-m-e	53.9±4.7	0.7±0.1	98.0±9.4	**105.6**±12.9	91.5±14.3	64.7±29.1	**104.0**±3.6
walker2d-m-e	90.1±13.2	0.7±1.1	110.1±0.5	107.9±1.6	109.6±1.0	91.4±39.1	**112.0**±0.3
halfcheetah-e	91.8±1.5	-2.7±0.3	96.7±1.1	96.3±1.3	95.0±0.5	64.9±13.0	**97.6**±0.2
hopper-e	107.7±0.7	1.3±0.5	107.8±7	96.5±28.0	109.4±0.5	44.4±49.2	**111.2**±0.3
walker2d-e	106.7±0.2	1.8±0.3	110.2±0.3	108.5±0.5	109.9±1.2	68.1±53.9	**110.8**±0.1
Average	49.6±3.6	14.6±5.0	67.5±3.8	67.3±9.1	68.9±3.8	58.0±17.1	**80.0**±1.4

Learning curves of EABC



Conclusions

- With adjustable behavior cloning, we effectively improved algorithm performance on inferior data.
- Benefiting from the ensemble approach, the performance is stable with low variance.
- EABC has a simple, intuitive structure, and a short runtime, while achieving state-of-the-art performance.
- One potential future extension could involve automating the determination of p.
- Utilization of pre-known expert information can be immensely valuable. By leveraging such supervised information (confidence level p regarding the offline dataset), we can potentially avoid extensive parameter tuning and complex training strategies.

References

[1] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.