南京航空航天大學

Nanjing University of Aeronautics and Astronautics

ParNeC 模式识别与神经计算研究组

PAttern Recognition and NEural Computing

An Introduction of Batch Reinforcement Learning

2021.11.08

background

Problems

- sample efficiency
- difficult to sample
- safety-critical systems

Algorithm
- Online RL
- Batch RL
- Growing Batch RL

batch RL

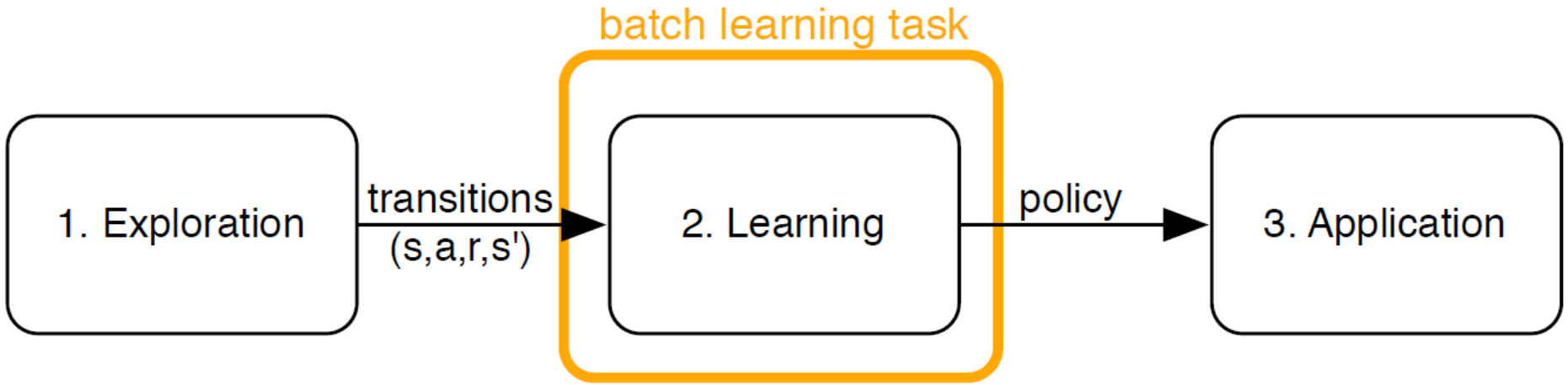


fig1: batch RL设定图

growing batch RL

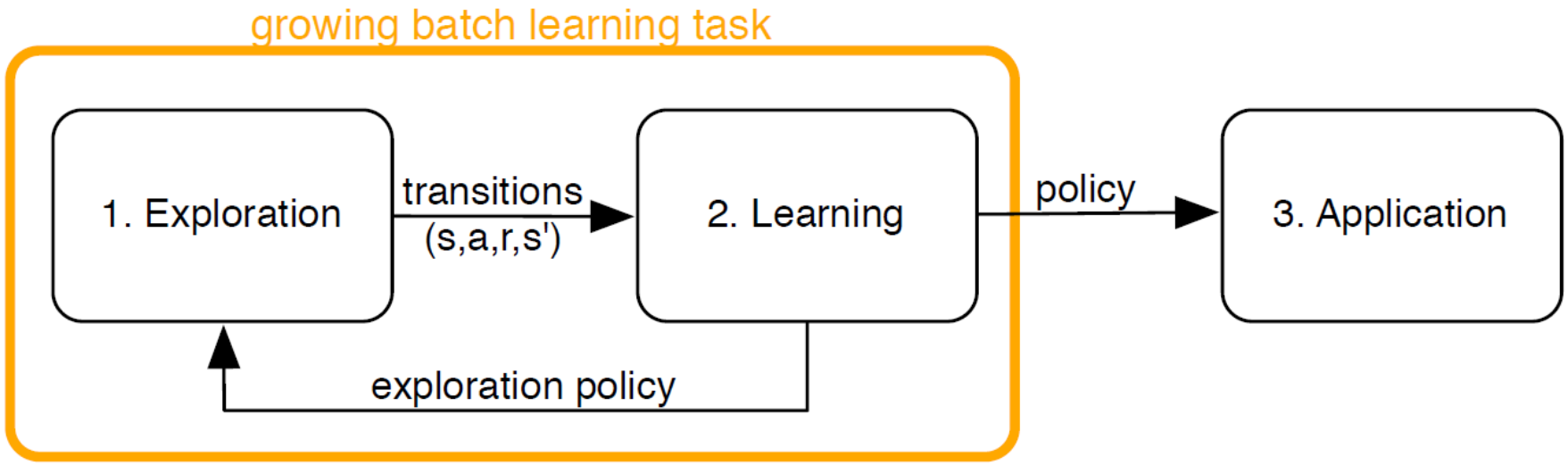


Fig2: growing batch RL设定图

Difference between batch rl and imitation learning

- Batch RL based on the standard off-policy algorithm, usually needs to be optimized through bellman equation or TD; and imitation learning is not required.
- Requirements for the data set: very high-quality data sets or sub-optimal;
- Reward: consider rewards or no reward;
- Imitation learning usually need to label the dataset according to the labels of experts and non-experts; this problem does not need to be considered in Batch RL.

BAIL: Best-Action Imitation Learning for Batch Deep Reinforcement Learning

Xinyue Chen[1] Zijian Zhou[1] Zheng Wang[1]
Che Wang[1,2] Yanqiu Wu[1,2] Keith Ross[1,2*]

[1] New York University Shanghai
[2]New York University

Nips 2020

Main Contributions

- Propose a notion named upper envelope of the data, try to make the best possible estimate of V^* using only the limited information in the batch dataset.
- Propose a method of selecting the best state-action pairs, then train a policy with IL using the selected state-action pairs. Combines both V-learning and IL.
- Conduct experiment on MuJoCo benchmark to demonstrate the superiority.

Best-Action Imitation Learning

BAIL algorithm provide state-of-the-art performance on simulated robotic locomotion tasks, also fast and algorithmically simple.

- First, try to make the best possible estimate of V(s) using only the limited information in the batch dataset.
- Then select state-action pairs from the dataset, whose associated returns G(s, a) are close to V(s).
- Finally, train a policy with Imitation Learning using the selected state-action pairs.

Thus, BAIL combines both V-learning and IL.

Upper envelope of the data

a batch of data $\mathcal{B} = \{(s_i, a_i, r_i, s_i'),\ i = 1, ..., m\}$.

Assumption:

- data in the batch was generated in an episodic fashion.
- data in the batch is ordered accordingly.

For each data point $i \in \{1, ..., m\}$, we calculate the Monte Carlo return G_i as the sum of the discounted rewards from state s_i to the end of the episode as $G_i = \sum_{t=i}^{T} \gamma^{t-i} r_t$

Upper envelope of the data

For a fixed $\lambda \geq 0$, we say that $V_{\phi^\lambda}(s)$ is a λ-regularized upper envelope for $\mathcal{G}$ if ϕ^λ is an optimal solution for the following constrained optimization problem:

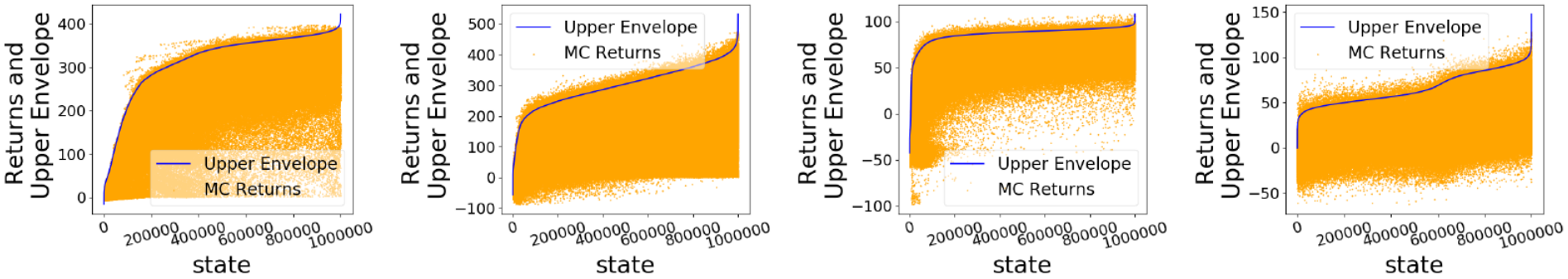
$$\min_{\phi} \sum_{i=1}^{m} [V_\phi(s_i) - G_i]^2 + \lambda \|w\|^2 \qquad s.t. \qquad V_\phi(s_i) \geq G_i, \qquad i = 1, 2, \ldots, m \qquad (1)$$

- When λ is very small, the upper envelope aims to interpolate the data.
- When λ is large, the upper envelope approaches a constant going through the data point with the highest return.

Upper envelope of the data

We solve the constrained optimization problem (1) with a penalty-function approach. Specifically, to obtain an approximate upper envelope of the data $\mathcal{G}$, we solve an unconstrained optimization problem with a penalty loss function (with λ fixed):

$$L^K(\phi) = \sum_{i=1}^{m} (V_\phi(s_i) - G_i)^2 \{\mathbb{1}_{(V_\phi(s_i) \geq G_i)} + K \cdot \mathbb{1}_{(V_\phi(s_i) < G_i)}\} + \lambda \|w\|^2 \tag{2}$$



(a) Hopper (b) Walker2d (c) HalfCheetah (d) Ant

Figure 1: Upper Envelopes trained on batches from different MuJoCo environments.

Select the best actions

Two approaches for selecting the best actions

- BAIL-ratio, for a fixed $x > 0$, choose all (s_i, a_i) pairs from the batch data such that:

$$G_i > xV(s_i)$$

- BAIL-difference, for a fixed $x > 0$, choose all (s_i, a_i) pairs from the batch data such that:

$$G_i \geq V(s_i) - x$$

Augmented and oracle returns

The problem:

- For data points appearing near the beginning of the episode, the finite-horizon return will closely approximate the (idealized) infinite-horizon return due to discounting;
- For a data point near the end of an episode, the finite horizon return can be inaccurate and should be augmented.

$$G_i = \sum_{t=i}^{T} \gamma^{t-i} r_t + \gamma^{T-i+1} \sum_{t=j}^{T} \gamma^{t-j} r_t$$

Pseudo-Code of BAIL

Algorithm 1 BAIL

Initialize upper envelope parameters ϕ, ϕ', policy parameters θ. Obtain batch data $\mathcal{B}$. Randomly split data into training set $\mathcal{B}_t$ and validation set $\mathcal{B}_v$ for the upper envelope.
Compute return G_i for each data point i in $\mathcal{B}$.
Obtain upper envelope by minimizing the loss $L^K(\phi)$:
for $j = 1, \ldots, J$ **do**
  Sample a mini-batch B from $\mathcal{B}$.
  Update ϕ using the gradient: $\nabla_\phi \sum_{i \in B} (V_\phi(s_i) - G_i)^2 \{\mathbb{1}_{(V_\phi(s_i) > G_i)} + K \mathbb{1}_{(V_\phi(s_i) < G_i)}\} + \lambda \|\phi\|^2$

  if time to do validation for the upper envelope **then**
    Compute validation loss on B_v
    Update ϕ and ϕ' according to the validation loss
  end if
end for
Select data point i if $G_i > x V_\phi(s_i)$, where x is such that $p\%$ of data in $\mathcal{B}$ are selected. Let $\mathcal{U}$ be the set of selected data points.
for $l = 1, \ldots, L$ **do**
  Sample a mini-batch U of data from $\mathcal{U}$.
  Update θ using the gradient: $\nabla_\theta \sum_{i \in U} (\pi_\theta(s_i) - a_i)^2$
end for

Pseudo-Code of Progressive BAIL

Algorithm 2 Progressive BAIL

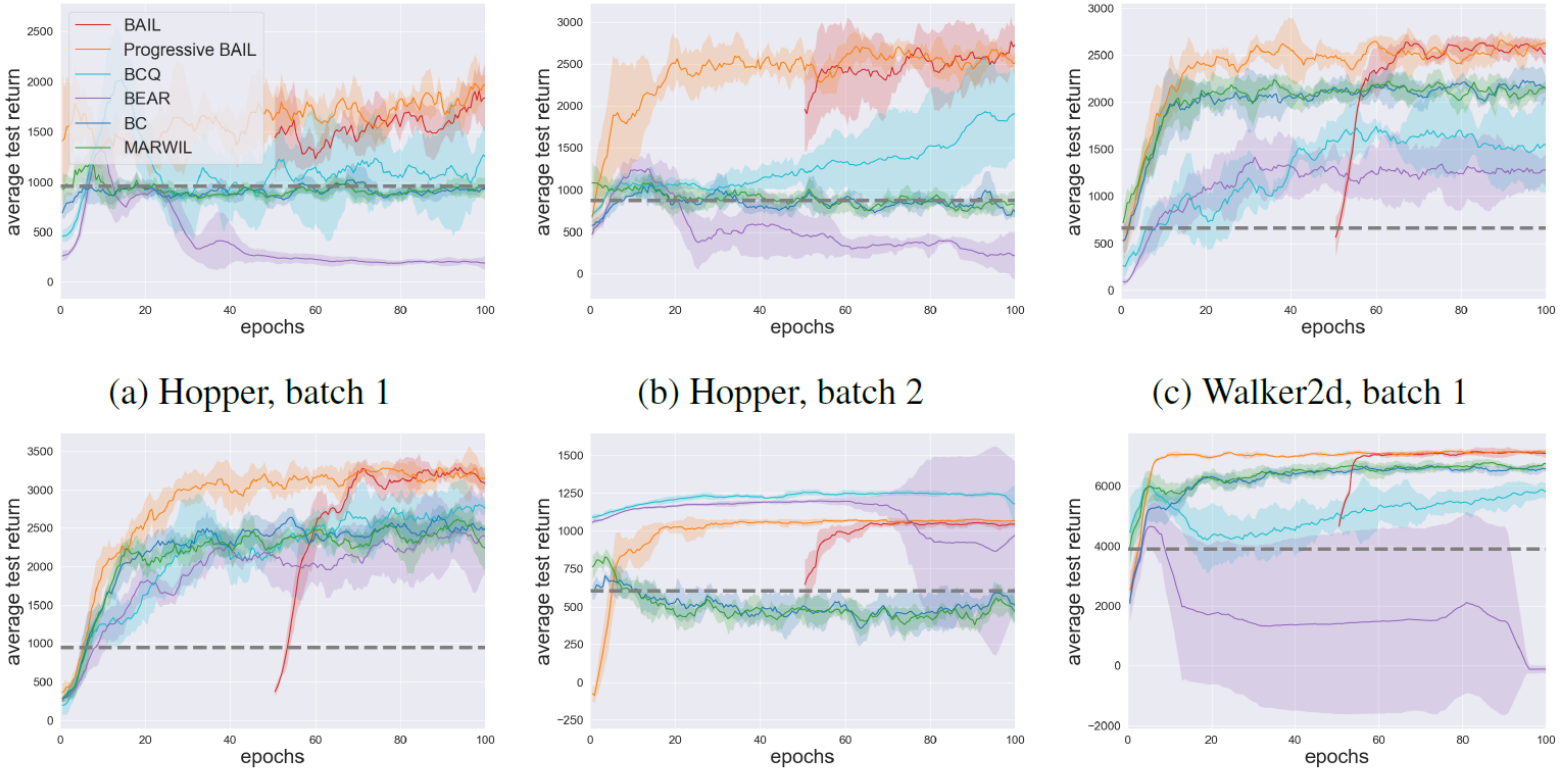
Initialize upper envelope parameters ϕ, ϕ', policy parameters θ.
Obtain batch data $\mathcal{B}$. Randomly split data into training set $\mathcal{B}_t$ and validation set $\mathcal{B}_v$ for the upper envelope.
Compute return G_i for each data point i in $\mathcal{B}$.
for $l = 1, \ldots, L$ **do**
  Sample a mini-batch of data B from the batch $\mathcal{B}_t$.
  Update ϕ using the gradient: $\nabla_\phi \sum_{i \in B_t} (V_\phi(s_i) - G_i)^2 \left\{ \mathbb{1}_{(V_\phi(s_i) > G_i)} + K \mathbb{1}_{(V_\phi(s_i) < G_i)} \right\} + \lambda \|\phi\|^2$
  if time to validate **then**
    Compute validation loss on B_v
    Update ϕ and ϕ' according to validation loss
  end if
  Select data point i if $G_i > x V_\phi(s_i)$, where x is such that $p\%$ of data in B are selected. Let U be the set of selected data points.
  Update θ using the gradient: $\nabla_\theta \sum_{i \in U} (\pi_\theta(s_i) - a_i)^2$
end for

Experiment

Table 1: Performance of five Batch DRL algorithms for 22 different training datasets.

Environment	BAIL	BCQ	BEAR	BC	MARWIL
$\sigma = 0.1$ Hopper B1	**2173 ± 291**	1219 ± 114	505 ± 285	626 ± 112	827 ± 220
$\sigma = 0.1$ Hopper B2	**2078 ± 180**	1178 ± 87	985 ± 3	579 ± 141	620 ± 336
$\sigma = 0.1$ Walker B1	**1125 ± 113**	576 ± 309	610 ± 212	514 ± 17	436 ± 24
$\sigma = 0.1$ Walker B2	**3141 ± 300**	2338 ± 388	2707 ± 425	1741 ± 239	1810 ± 200
$\sigma = 0.1$ HC B1	**5746 ± 29**	**5883 ± 43**	0 ± 0	**5546 ± 29**	**5573 ± 35**
$\sigma = 0.1$ HC B2	**7212 ± 43**	**7562 ± 31**	0 ± 0	6765 ± 108	**6828 ± 111**
$\sigma = 0.5$ Hopper B1	**2054 ± 158**	1145 ± 300	203 ± 42	919 ± 52	946 ± 103
$\sigma = 0.5$ Hopper B2	**2623 ± 282**	1823 ± 555	241 ± 239	694 ± 64	818 ± 112
$\sigma = 0.5$ Walker B1	**2522 ± 51**	1552 ± 455	1248 ± 181	2178 ± 178	2111 ± 52
$\sigma = 0.5$ Walker B2	**3115 ± 133**	2785 ± 123	2302 ± 630	2483 ± 94	2364 ± 228
$\sigma = 0.5$ HC B1	1055 ± 9	**1222 ± 38**	924 ± 579	570 ± 35	512 ± 43
$\sigma = 0.5$ HC B2	**7173 ± 120**	5807 ± 249	−114 ± 140	**6545 ± 171**	**6668 ± 93**
SAC Hopper B1	**3296 ± 105**	2681 ± 438	1000 ± 110	2853 ± 318	2897 ± 227
SAC Hopper B2	1831 ± 915	**2134 ± 917**	1139 ± 317	**2240 ± 367**	**2063 ± 168**
SAC Walker B1	**2455 ± 211**	**2408 ± 84**	−3 ± 5	1674 ± 277	1484 ± 140
SAC Walker B2	**4767 ± 130**	3794 ± 398	325 ± 75	2599 ± 145	2651 ± 268
SAC HC B1	**10143 ± 77**	8607 ± 473	7392 ± 257	8874 ± 221	9105 ± 90
SAC HC B2	**10772 ± 59**	**10106 ± 134**	7217 ± 273	9523 ± 164	9488 ± 136
SAC Ant B1	**4284 ± 64**	**4042 ± 113**	3452 ± 128	**3986 ± 112**	**4033 ± 130**
SAC Ant B2	**4946 ± 148**	**4640 ± 76**	3712 ± 236	**4618 ± 111**	**4589 ± 130**
SAC Humanoid B1	**3852 ± 430**	1411 ± 250	0 ± 0	543 ± 378	589 ± 121
SAC Humanoid B2	**3565 ± 153**	1221 ± 207	0 ± 0	1216 ± 826	1033 ± 257

Experiment

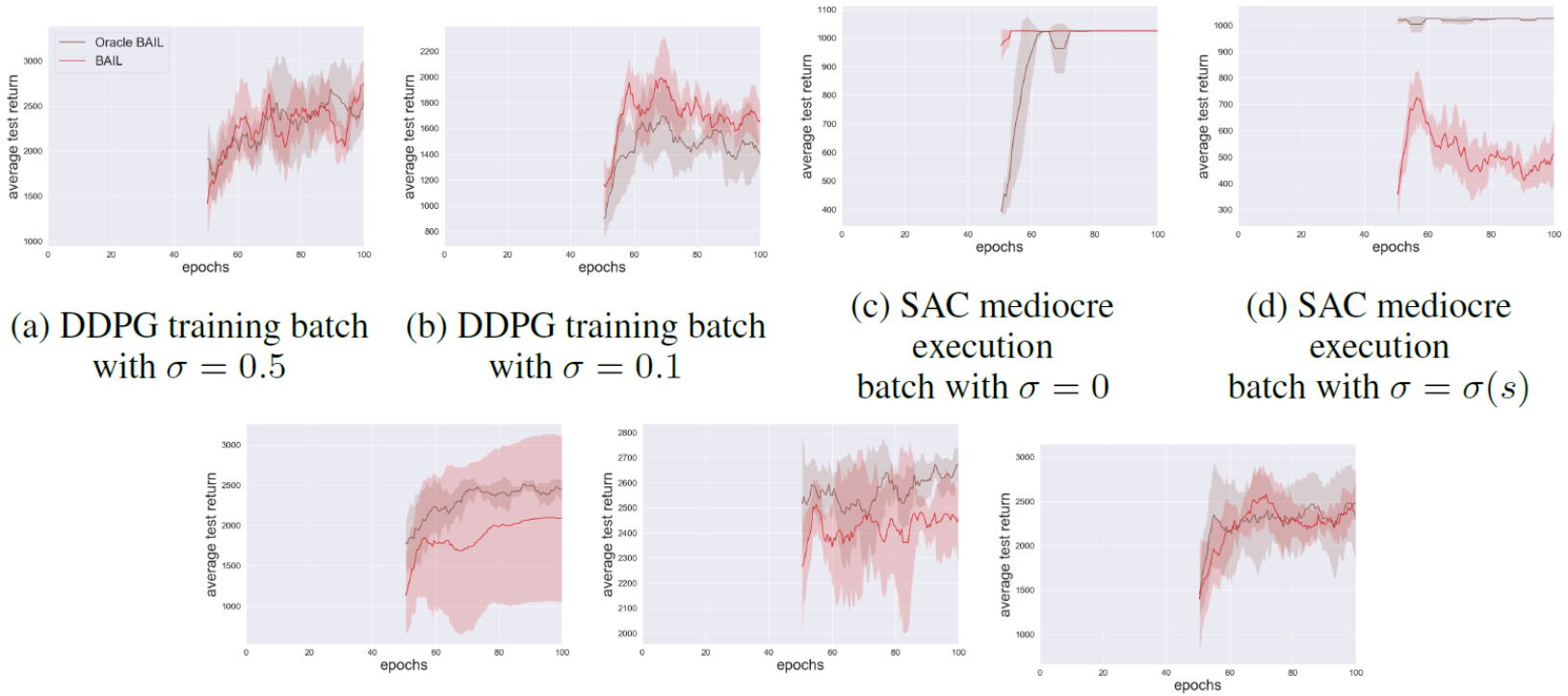


(a) Hopper, batch 1 (b) Hopper, batch 2 (c) Walker2d, batch 1

(d) Walker2d, batch 2 (e) HalfCheetah, batch 1 (f) HalfCheetah, batch 2

Figure 2: Learning curves using DDPG training batches with $\sigma = 0.5$.

Experiment



(a) DDPG training batch with $\sigma = 0.5$

(b) DDPG training batch with $\sigma = 0.1$

(c) SAC mediocre execution batch with $\sigma = 0$

(d) SAC mediocre execution batch with $\sigma = \sigma(s)$

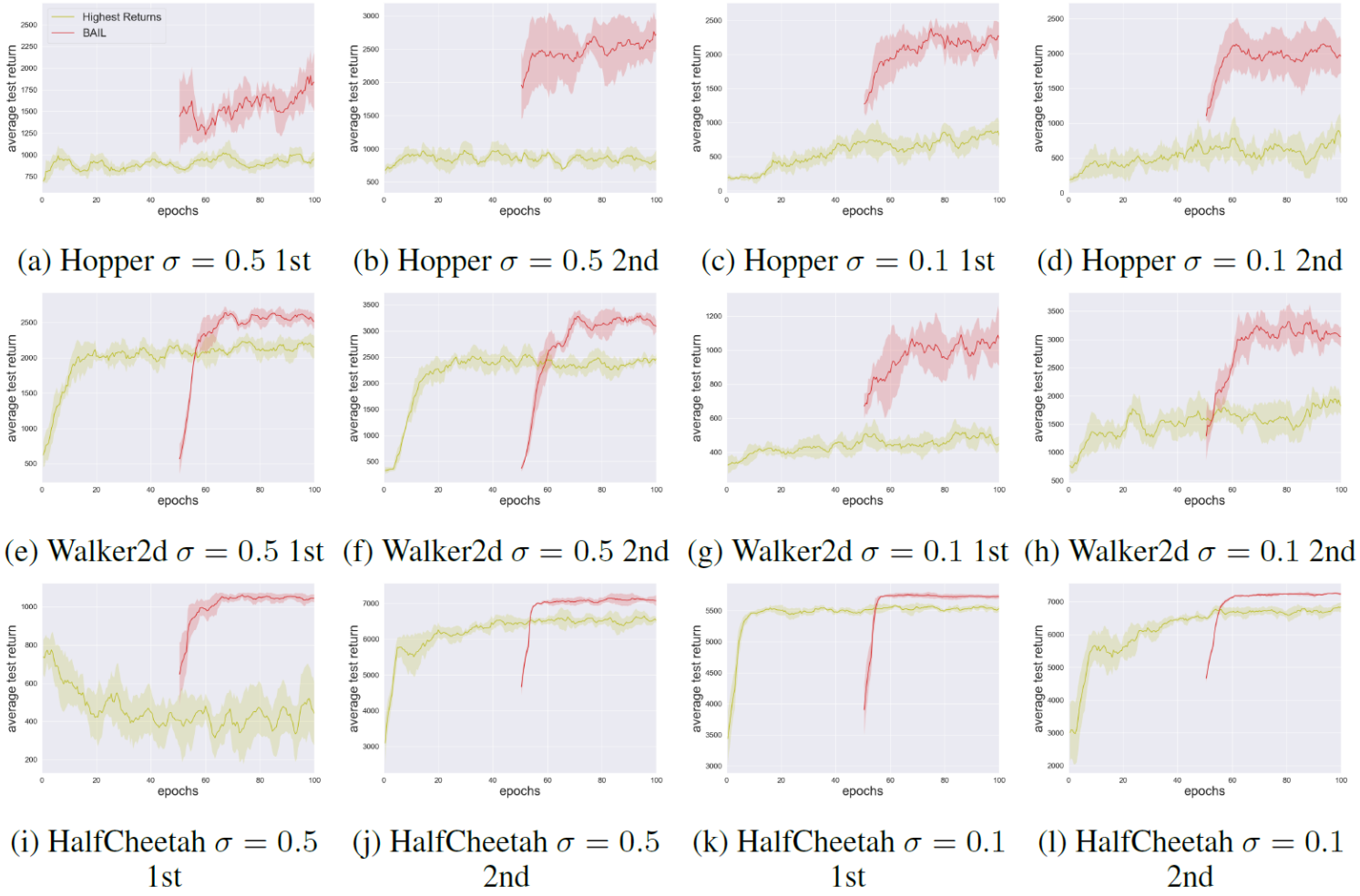
(e) SAC optimal execution batch with $\sigma = 0$

(f) SAC optimal execution batch with $\sigma = \sigma(s)$

(g) SAC training batch

Figure 4: Augmented Returns versus Oracle Performance. All learning curves are for the Hopper-v2 environment. The x-axis ranges from 50 to 100 epochs since this comparison involves only BAIL. The results show that the augmentation heuristic typically achieves oracle-level performance.

Experiment



(a) Hopper $\sigma = 0.5$ 1st (b) Hopper $\sigma = 0.5$ 2nd (c) Hopper $\sigma = 0.1$ 1st (d) Hopper $\sigma = 0.1$ 2nd

(e) Walker2d $\sigma = 0.5$ 1st (f) Walker2d $\sigma = 0.5$ 2nd (g) Walker2d $\sigma = 0.1$ 1st (h) Walker2d $\sigma = 0.1$ 2nd

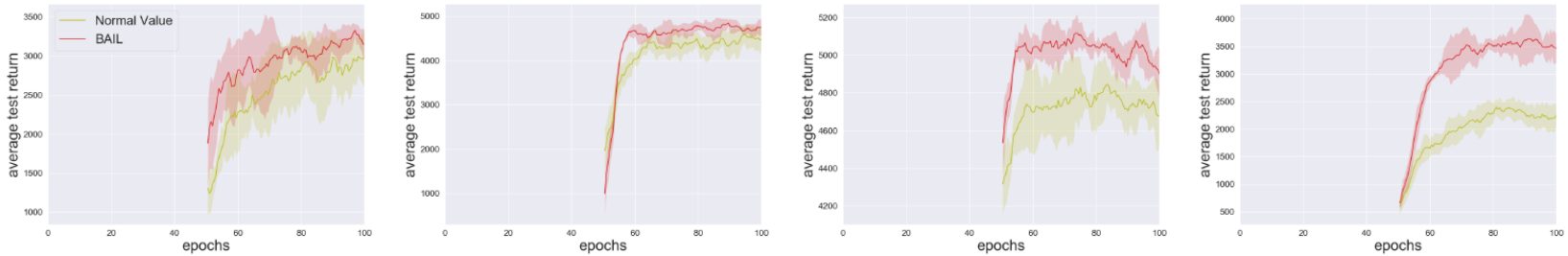
(i) HalfCheetah $\sigma = 0.5$ 1st (j) HalfCheetah $\sigma = 0.5$ 2nd (k) HalfCheetah $\sigma = 0.1$ 1st (l) HalfCheetah $\sigma = 0.1$ 2nd

Figure 5: Ablation study for data selection. The figure compares BAIL with the algorithm that simply chooses the state-action pairs with the highest returns (without using an upper envelope). The learning curves show that the upper envelope is critical components of BAIL.

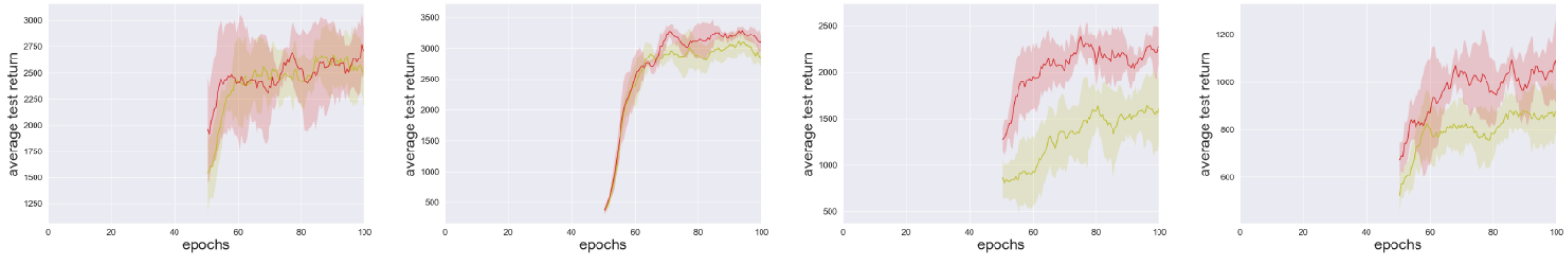
Experiment

D.3 Ablation study using standard regression instead of an upper envelope

Figure 6 compares BAIL with the more naive scheme of using standard regression in place of an upper envelope. The learning curves show that the upper envelope is a critical component of BAIL.



(a) Training SAC, Hopper (b) Training SAC, Walker2d (c) Training SAC, Ant (d) Training SAC, Humanoid



(e) Training DDPG $\sigma = 0.5$, Hopper (f) Training DDPG $\sigma = 0.5$, Walker2d (g) Training DDPG $\sigma = 0.1$, Hopper (h) Training DDPG $\sigma = 0.1$, Walker2d

THANKS