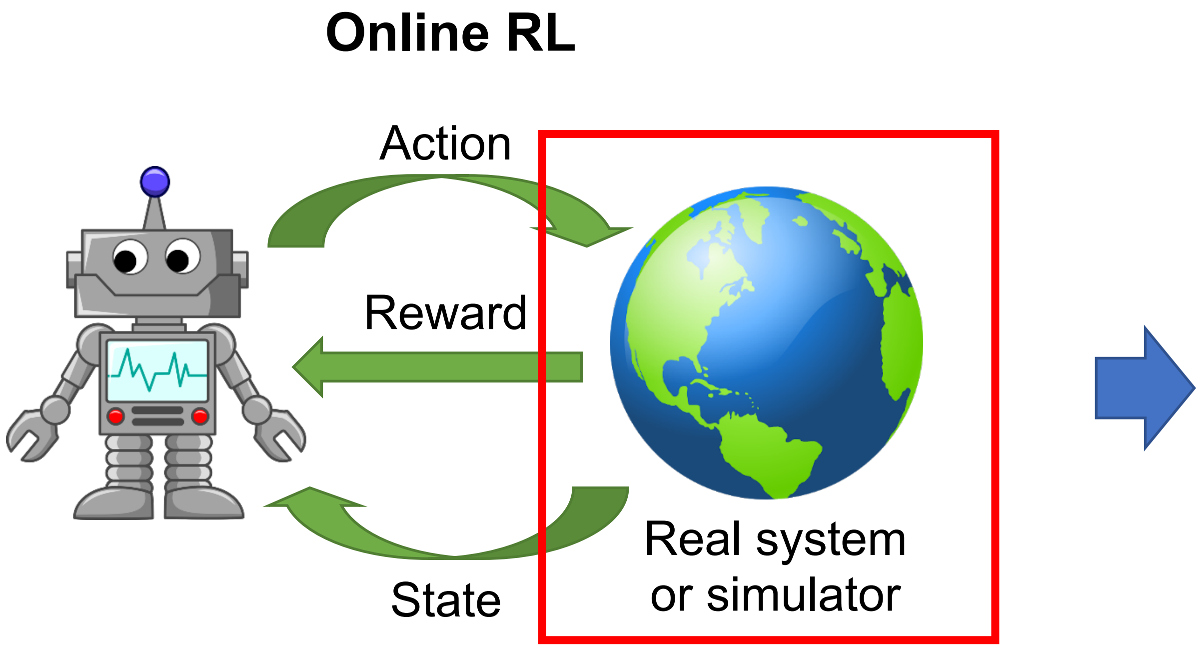
Discriminator-Weighted Offline Imitation Learning from Suboptimal Demonstrations

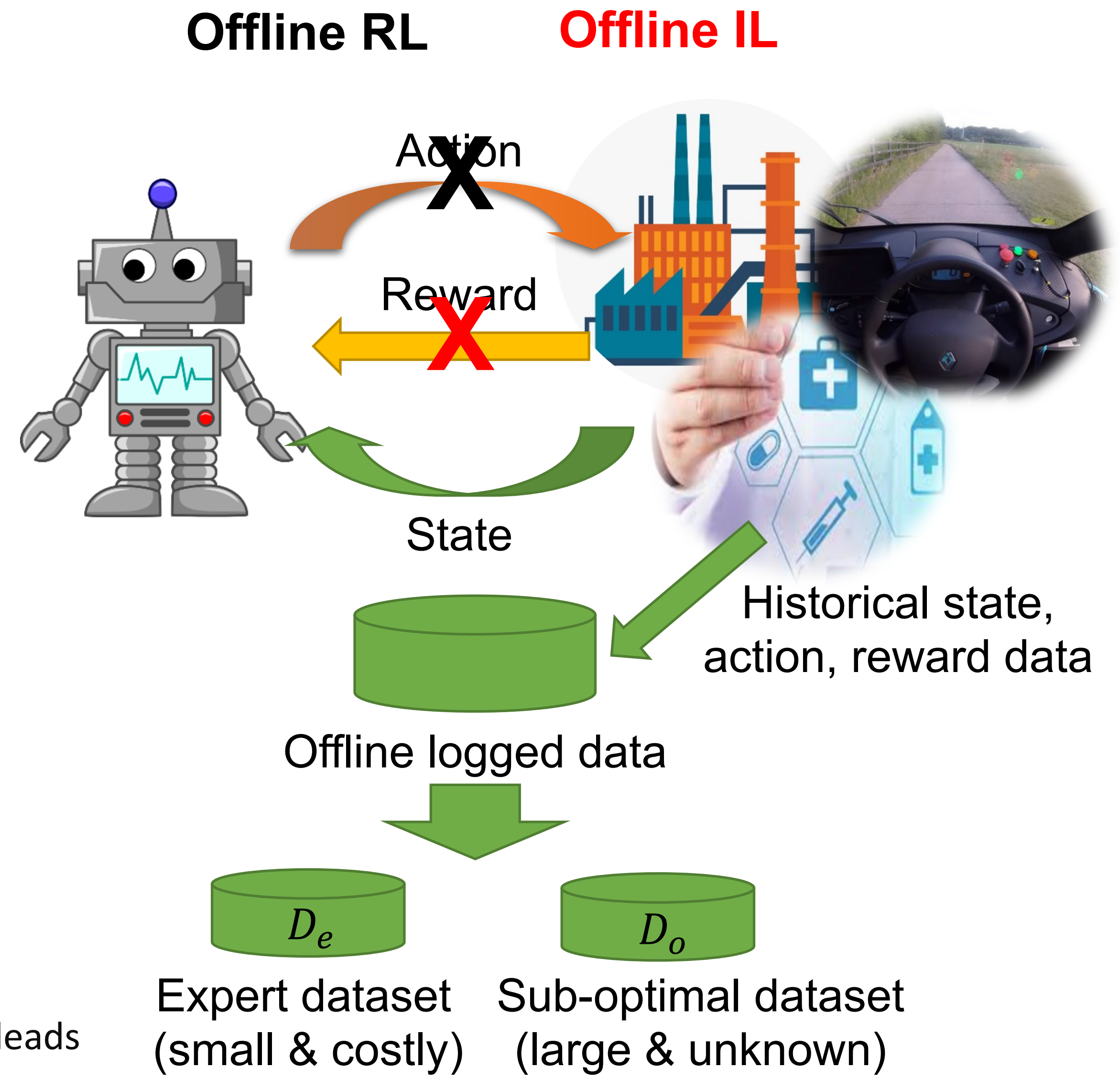
Haoran Xu[1], Xianyuan Zhan[2], Honglei Yin[1] & Huilin Qin[1]

[1] JD Intelligent Cities Research, JD Technology
[2] Institute of AI Industry Research (AIR), Tsinghua University

2022/7/20

From offline RL to offline IL





In most real-world applications:

- We may not know or hard to obtain a reward function
- Obtain optimal expert data is costly, dataset is small
- Could have a large sub-optimal dataset, composition is unknown

Technical difficulties:

- BC policy generalize poorly if only trained on limited expert data
- Naïvely involving suboptimal data in imitation learning generally leads to worse performance

What about current IL methods that can work offline?

—They cannot deal with additional suboptimal data!

Behavior Cloning

$$\min_{\pi} J_{\mathrm{BC}}(\pi) := -\frac{1}{N}\sum_{k=1}^{N} \log \pi(a_k|s_k).$$

Known to suffer from compounding errors.

ValueDICE&IQ-Learn

$$-D_{\mathrm{KL}}\left(d^{\pi} || d^{\exp}\right) = \min_{x:\mathcal{S}\times\mathcal{A}\to\mathbb{R}} \log \mathbb{E}_{(s,a)\sim d^{\exp}}[e^{x(s,a)}] - \mathbb{E}_{(s,a)\sim d^{\pi}}[x(s,a)].$$

Minimizing the KL-divergence between the policy state- action occupancies and the expert.

Behavioral cloning from noisy demonstrations (BCND)

$$\arg\max_{\theta} \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s) \cdot \pi_{\theta_{old}}(a|s)].$$

Use a learned ensemble policy to reweight policy learning. However, assume expert data occupy majority part.

- Kostrikov et al, Imitation Learning via Off-Policy Distribution Matching, ICLR 2020
- Garg et al, IQ-Learn: Inverse soft-Q Learning for Imitation, NeurIPS 2021
- Sasaki et al, Behavioral Cloning from Noisy Demonstrations, ICLR 2021

Two underlying tasks

BC task:

$$\min_{\pi} \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e} [-\log \pi(a|s)]$$

Expert dataset D_e
(small & costly)

Sub-optimal dataset D_o
(large & unknown)

Discriminating task:

$$\min_{d} \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e} [-\log d(s,a)] + \boxed{\mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o} [-\log(1-d(s,a))]}$$

May also contain expert data

We can do better via positive-unlabeled (PU) learning

$$\min_{d} \eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e} [-\log d(s,a)] + \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o} [-\log(1-d(s,a))] - \eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e} [-\log(1-d(s,a))]$$

Still not good enough: both tasks lacks enough information to improve their performance

Proportion of positive samples

Balance the impact of the second term

Learn two tasks cooperatively

Add information from BC task to discriminator learning objective:

$$\min_{d} \eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}[-\log d(s,a,\log\pi(a|s))] + \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o}[-\log(1-d(s,a,\log\pi(a|s)))]$$
$$-\eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}[-\log(1-d(s,a,\log\pi(a|s)))] .$$

What benefit would this change bring to us?

- Loss $\mathcal{L}_d$ is now functional of BC policy π
- We are interested to see how the variation of π impact $\mathcal{L}_d$
- To more robustly learn d, we make π challenge d

  → Learning d through minimizing $\mathcal{L}_d$ becomes harder

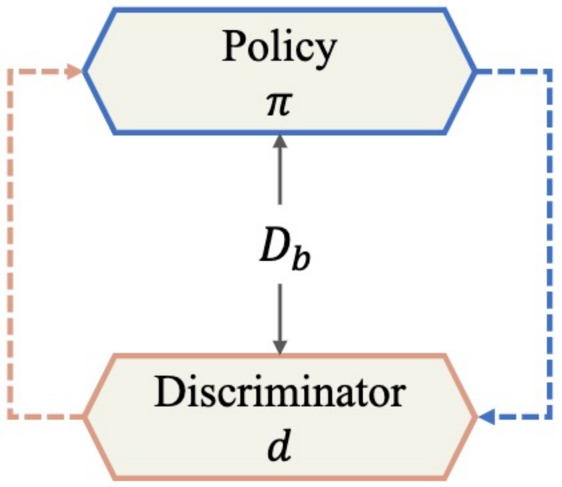
  → Find maxima of $\mathcal{L}_d$'s functional $J(\pi)$
- Add $\mathcal{L}_w$ to the BC task:

Theorem 3.1. *Assume $\mathcal{L}_d(d, \log\pi)$ is twice continously differentiable with respect to d, and d is continuously differentiable with respect to $\log\pi$. With a given discriminator d, then a relaxed neccessary condition for $\mathcal{L}_d(d, \log\pi)$ attains its maxima with respect to π is to require a corrective loss term $\mathcal{L}_w$ is minimized by π, where $\mathcal{L}_w$ is given as follows:*

$$\mathcal{L}_w = \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}\left[\log\pi(a|s)\cdot\left(\frac{\eta}{d}+\frac{\eta}{1-d}\right)\right] - \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o}\left[\log\pi(a|s)\cdot\frac{1}{1-d}\right]$$

$$\min_{\pi} \alpha \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}[-\log\pi(a|s)] - \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}\left[-\log\pi(a|s)\cdot\frac{\eta}{d(1-d)}\right] + \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o}\left[-\log\pi(a|s)\cdot\frac{1}{1-d}\right]$$

DWBC



New BC Task

$$L_\pi = \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_b}[-\log\pi(a|s)\cdot f(d(s,a,\log\pi))]$$

New Discriminating Task

$$L_d = \eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}[-\log d(s,a,\log\pi)] + \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_o}[-\log(1-d(s,a,\log\pi))] \\ -\eta \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_e}[-\log(1-d(s,a,\log\pi))]$$

Generalized BC objective:

$$\min_\pi \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}_b}[-\log\pi(a|s)\cdot f(s,a)]$$

Can learn with noisy expert data

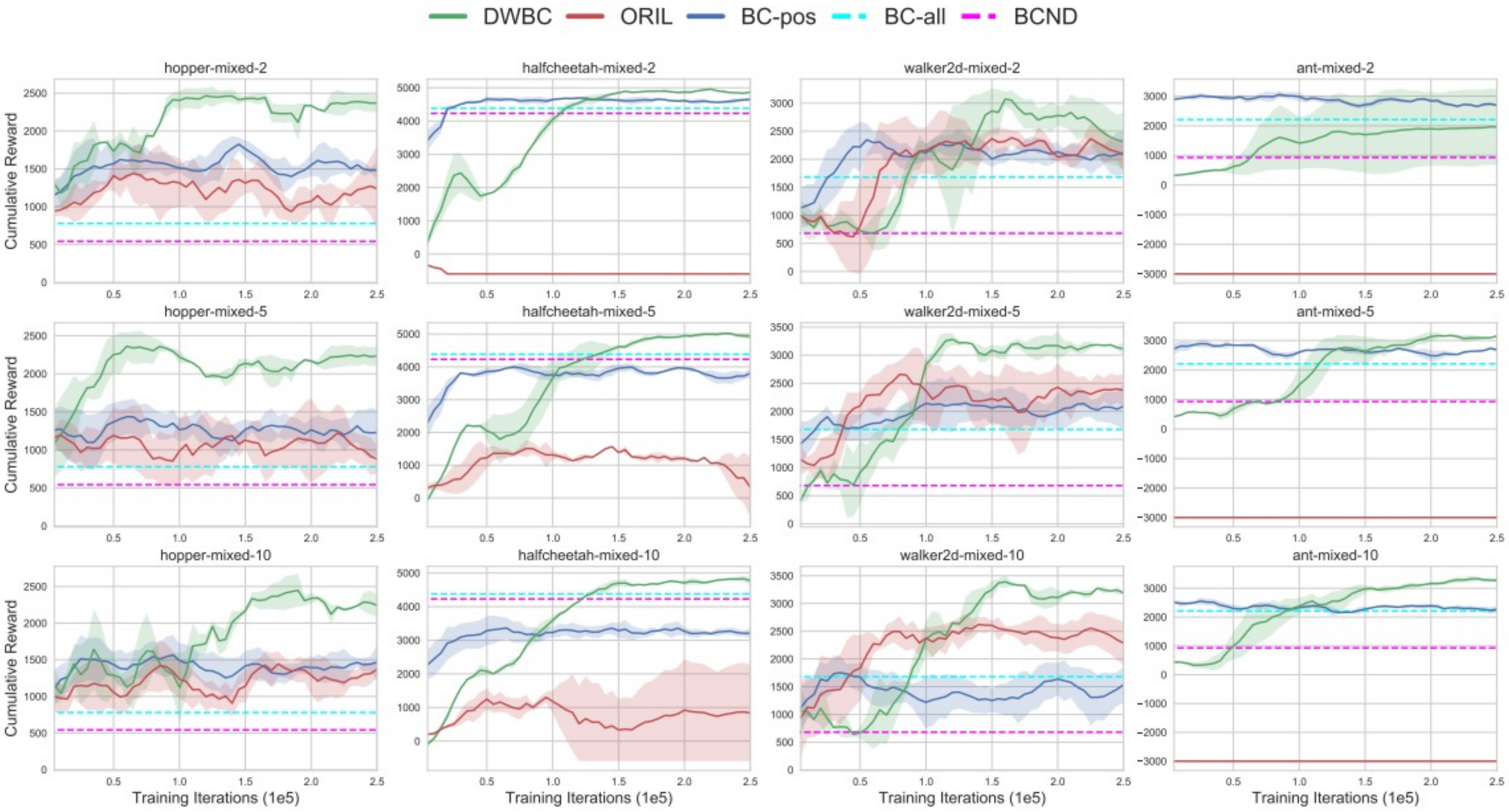
$$\text{Behavioral cloning weights} = \begin{cases} \alpha - \eta/d(1-d), & (s,a)\in\mathcal{D}_e \\ 1/(1-d), & (s,a)\in\mathcal{D}_o \end{cases}.$$

Boost loss for optimal data in sub-optimal dataset

d clip to [0.1,0.9]

Experiments

Performance of DWBC compared with other baseline algorithms



Baselines **(adding the comparision with recent work DemoDICE in the arxiv version):**

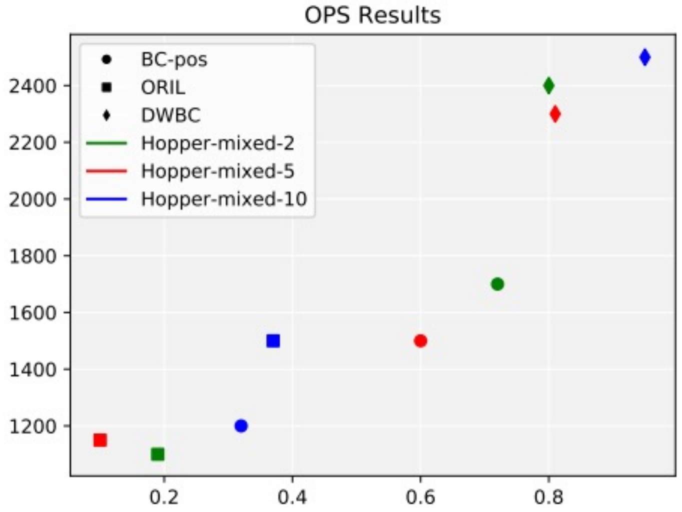
- BC-pos: BC on expert data only.
- BC-all: BC on all data.
- BCND-all: Behavioral Cloning from Noisy Demonstrations (Sasaki & Yamashina, 2021) on all data.
- ORIL: Offline Reinforced Imitation Learning (Zolna et al. 2020) on D_e and D_o.

Dataset statistics

Dataset-X	$\#\mathcal{D}_e$	$\#\mathcal{D}_o$
Ant_mixed-2	46,646	254,869
Ant_mixed-5	19,209	282,306
Ant_mixed-10	9,866	29,1649
Hopper_mixed-2	96,222	303,737
Hopper_mixed-5	39,590	360,369
Hopper_mixed-10	19,176	380,783
Halfcheetah_mixed-2	19,980	181,818
Halfcheetah_mixed-5	7,992	193,806
Halfcheetah_mixed-10	3,996	197,802
Walker2d_mixed-2	74,857	226,050
Walker2d_mixed-5	31,010	269,897
Walker2d_mixed-10	15,569	285,338

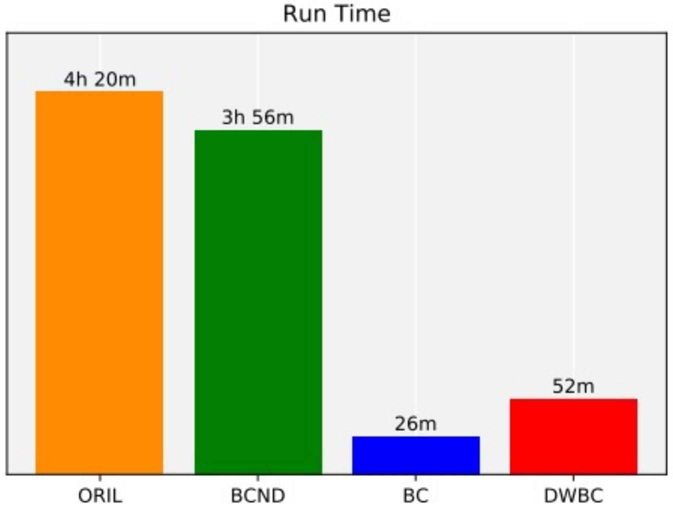
Additional Advantage

Byproduct: offline policy selection by the discriminator



- The learned discriminator provides a new means for difficult offline policy selection
  a) Learn a discriminator using DWBC
  b) Use data from D_e and $\log \pi$ (the testing policy) as input, compute average output value
  c) Average output score indicates rank of the policy

Computation Time



- Computationally cheap! Only slightly more than original BC
- Simple to implement: only need to learn an additional discriminator through supervised learning
- No inverse RL or online/offline RL or policy ensemble

Thanks !