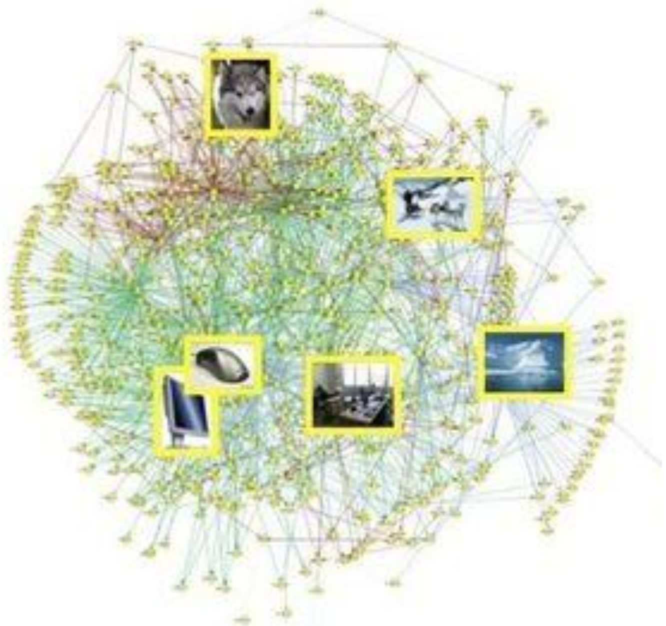
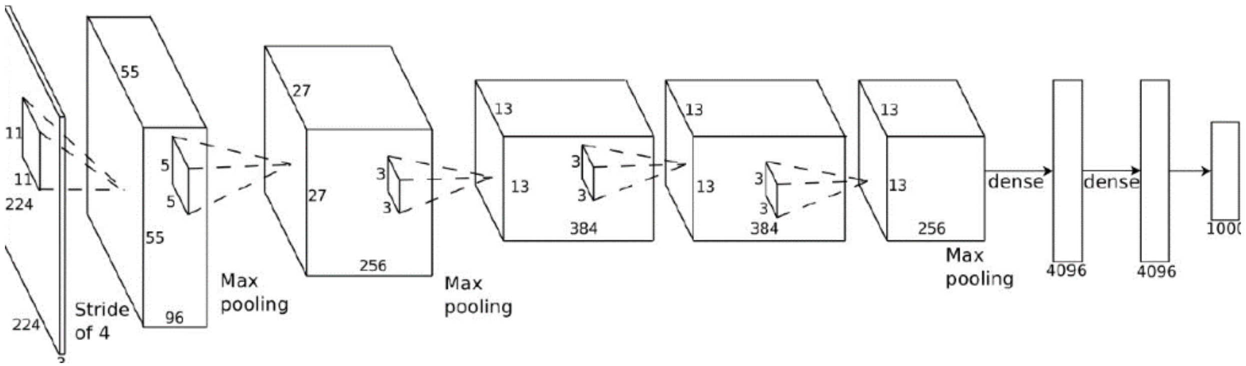
Offline Deep Reinforcement Learning Algorithms

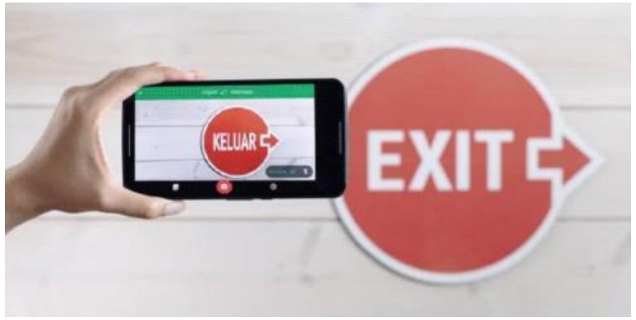
Sergey Levine

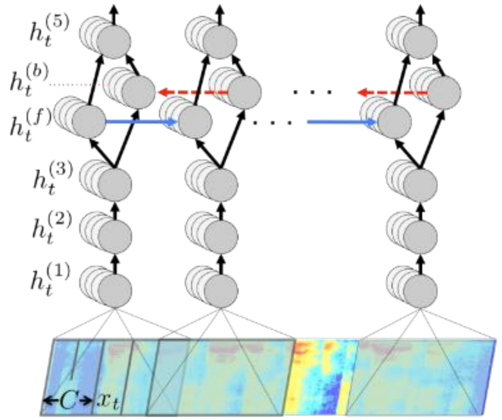
UC Berkeley

What makes modern machine learning work?

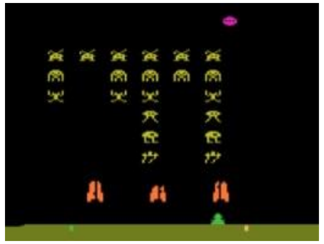




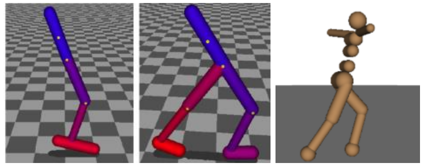




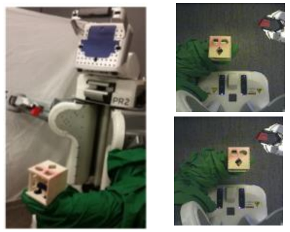
What about reinforcement learning?



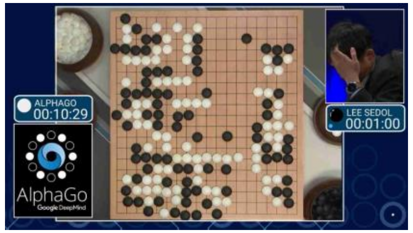
Mnih et al. '13

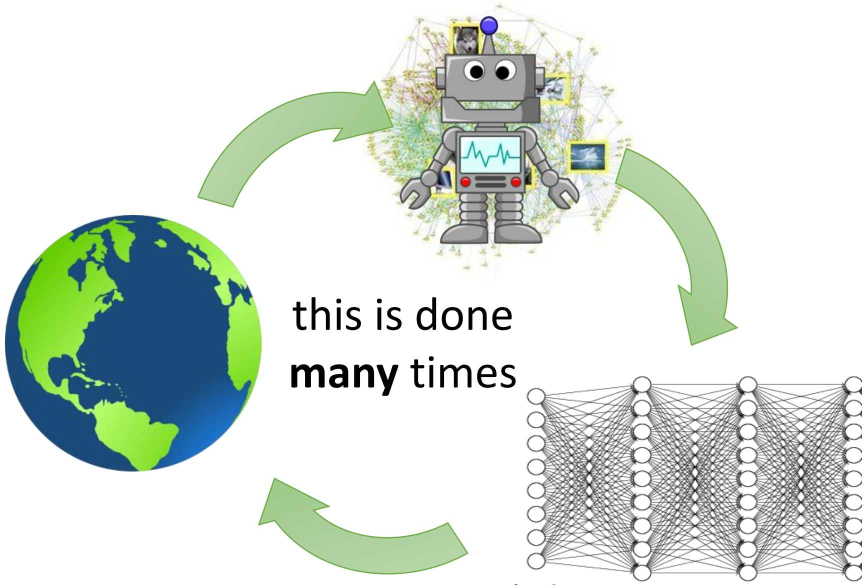


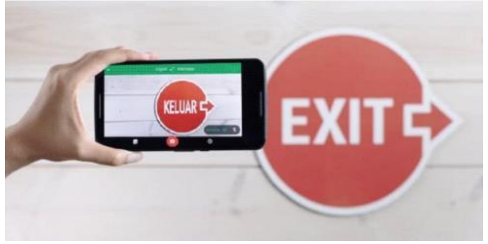
Schulman et al. '14 & '15

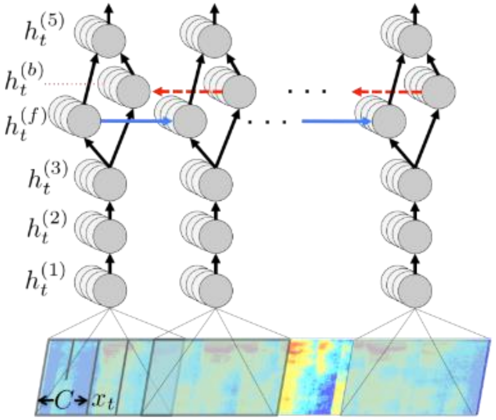


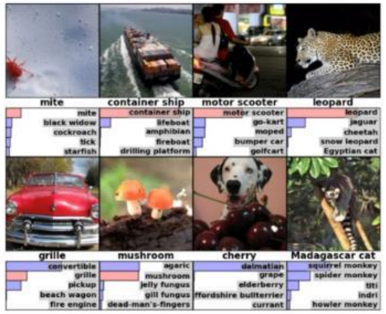
Levine*, Finn*, et al. '16







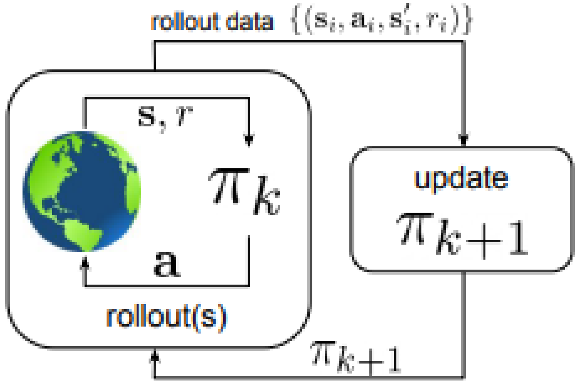




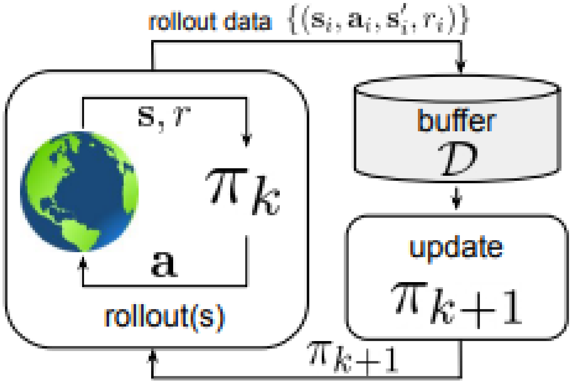
enormous gulf

Can we develop **data-driven** RL methods?

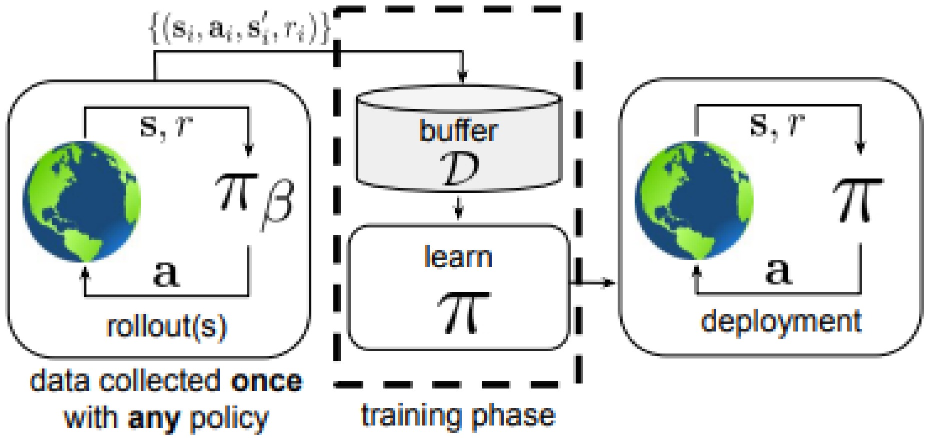
on-policy RL

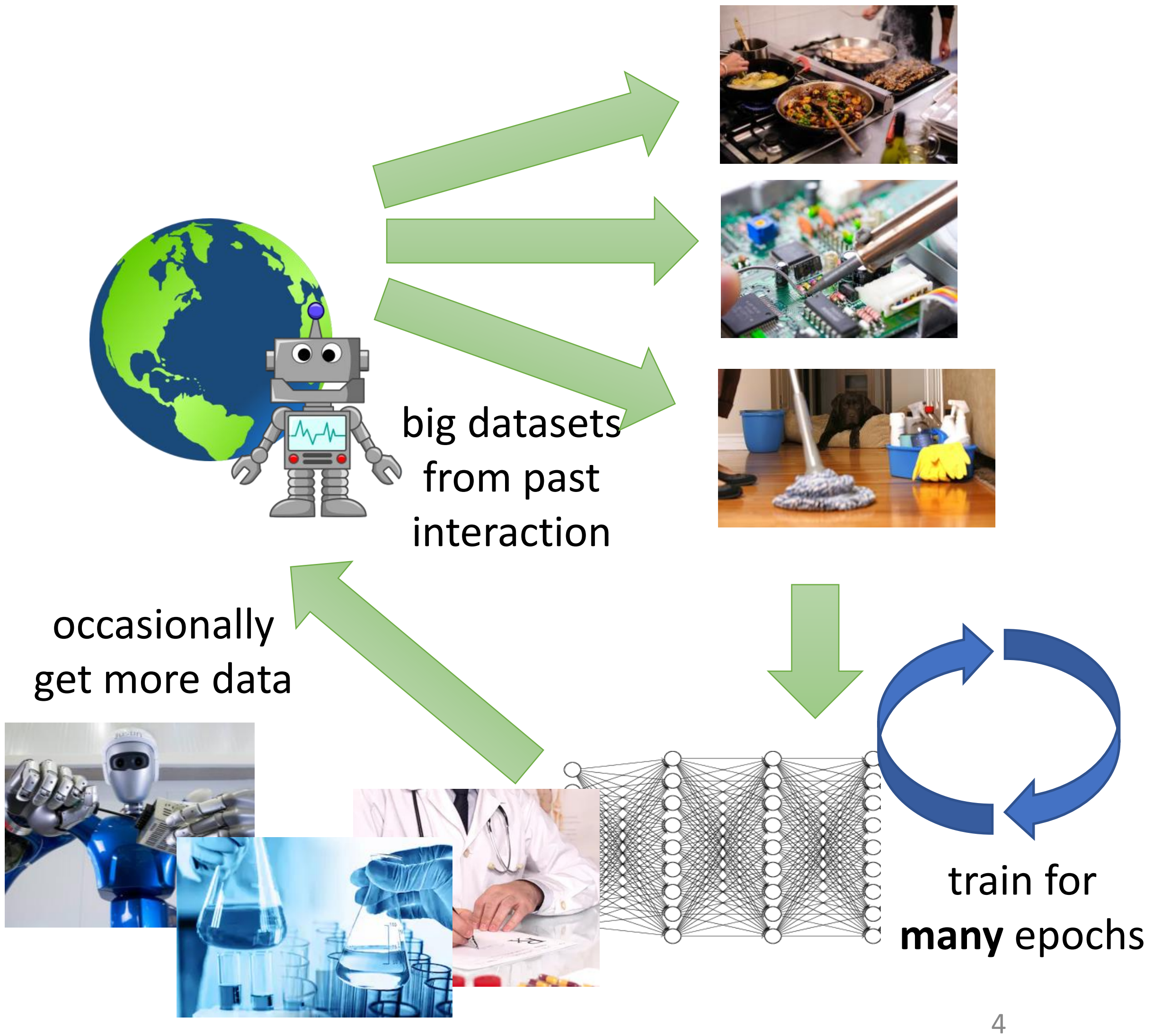


off-policy RL

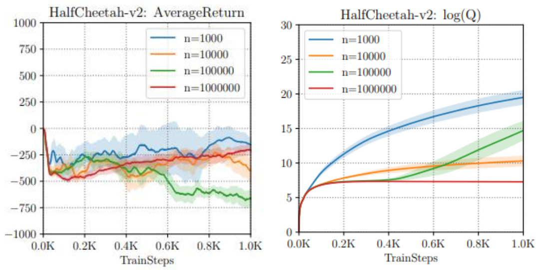


offline reinforcement learning

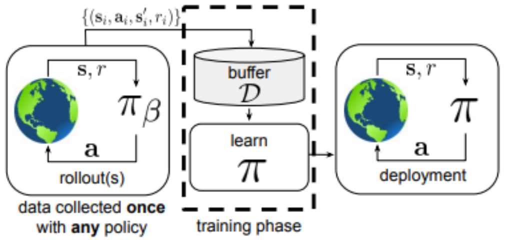




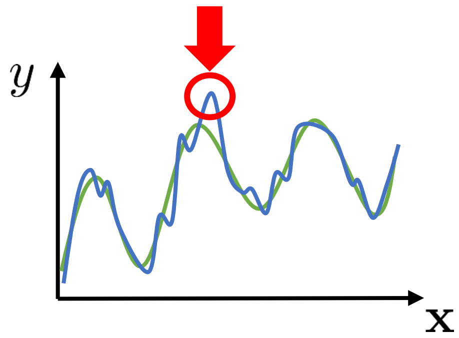
Levine, Kumar, Tucker, Fu. **Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems.** '20



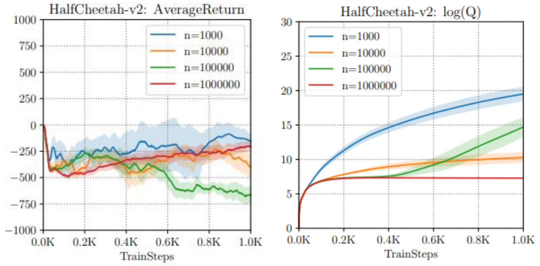
Why is offline RL difficult?



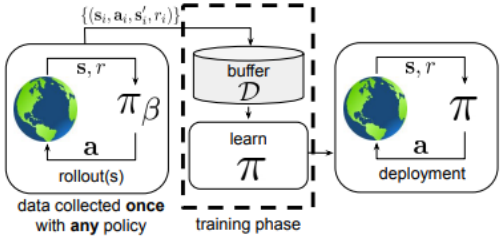
How do we design offline RL algorithms?



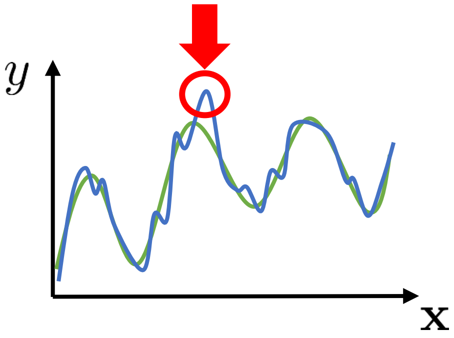
Conservative Q-Learning



Why is offline RL difficult?

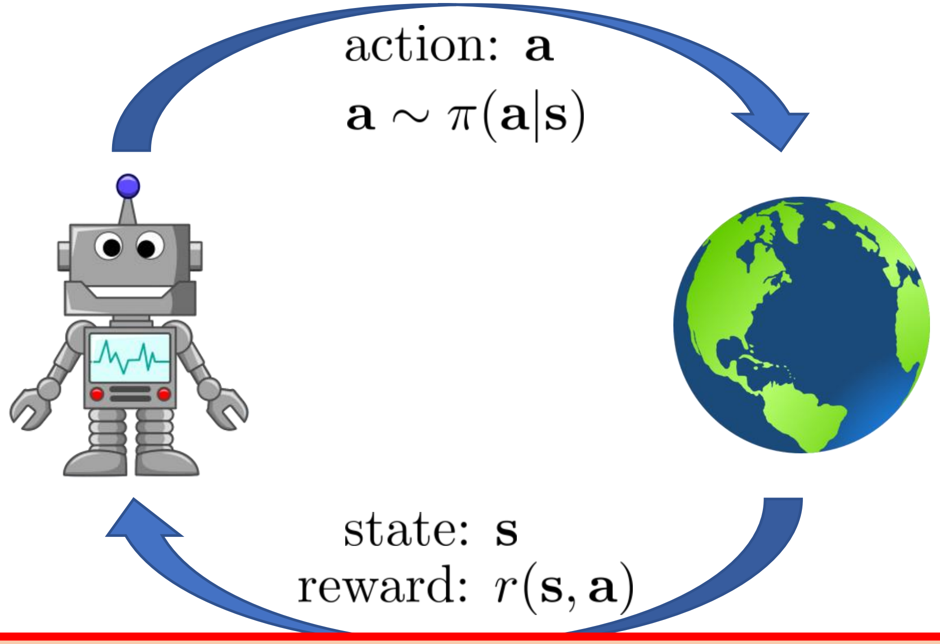


How do we design offline RL algorithms?



Conservative Q-Learning

Off-policy RL: a quick primer



This talk focuses entirely on **approximate dynamic programming** methods, but there are other methods too!

RL objective: $\max_\pi \sum_{t=1}^{T} E_{\mathbf{s}_t, \mathbf{a}_t \sim \pi}[r(\mathbf{s}_t, \mathbf{a}_t)]$

Q-function: $Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\mathbf{s}_{t'}, \mathbf{a}_{t'} \sim \pi}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t]$

$$\pi(\mathbf{a}|\mathbf{s}) = 1 \text{ if } \mathbf{a} = \arg\max_{\mathbf{a}} Q^\pi(\mathbf{s}, \mathbf{a})$$

$$Q^\star(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \max_{\mathbf{a}'} Q^\star(\mathbf{s}', \mathbf{a}')$$

enforce this equation at all states!

$$\text{minimize } \textstyle\sum_i (Q(\mathbf{s}_i, \mathbf{a}_i) - [r(\mathbf{s}_i, \mathbf{a}_i) + \max_{\mathbf{a}_i'} Q(\mathbf{s}_i', \mathbf{a}_i')])^2$$

$$\text{minimize } \textstyle\sum_i (Q(\mathbf{s}_i, \mathbf{a}_i) - y_i)^2$$

Off-policy RL: a quick primer

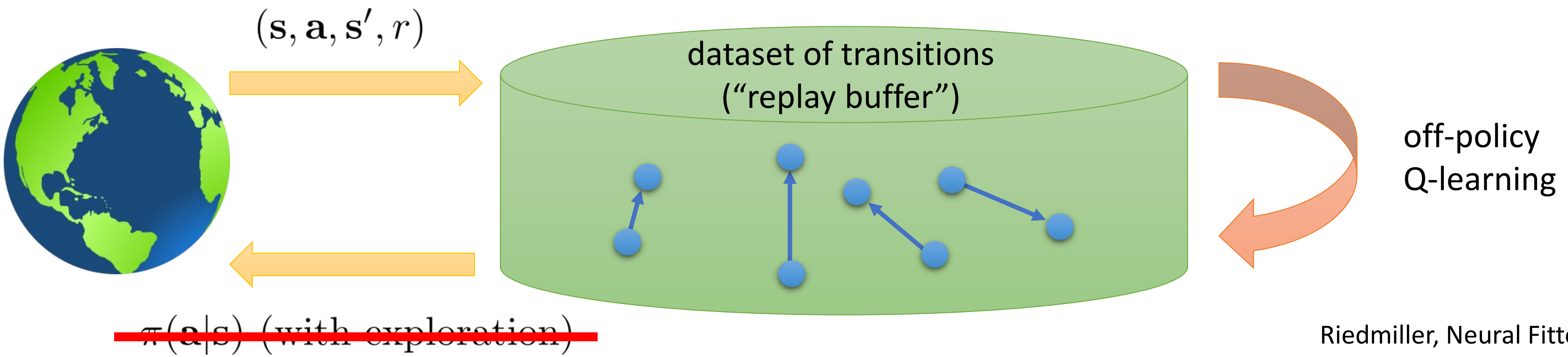
$$Q(\mathbf{s},\mathbf{a}) \leftarrow r(\mathbf{s},\mathbf{a}) + \max_{\mathbf{a}'} Q(\mathbf{s}',\mathbf{a}')$$ ⟵ don’t need on-policy data for this!

off-policy Q-learning:

1. collect dataset $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$ using some policy, add it to $\mathcal{B}$

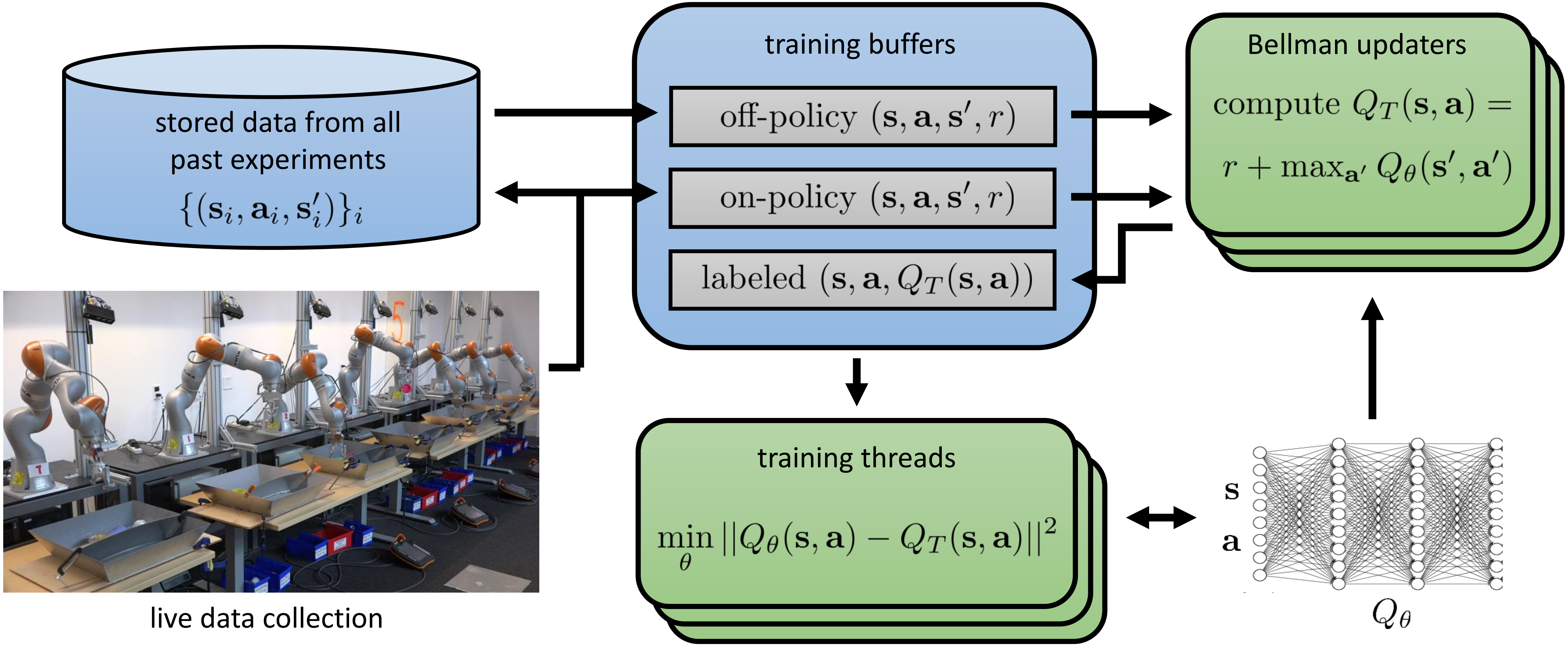
$K\times$

2. sample a batch $(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)$ from $\mathcal{B}$
3. minimize $\sum_i (Q(\mathbf{s}_i, \mathbf{a}_i) - [r(\mathbf{s}_i, \mathbf{a}_i) + \max_{\mathbf{a}'_i} Q(\mathbf{s}'_i, \mathbf{a}'_i)])^2$



See, e.g.
Riedmiller, Neural Fitted Q-Iteration ‘05
Ernst et al., Tree-Based Batch Mode RL ‘05

Does it work?



Kalashnikov, Irpan, Pastor, Ibarz, Herzong, Jang, Quillen, Holly, Kalakrishnan, Vanhoucke, Levine. **QT-Opt: Scalable Deep Reinforcement Learning of Vision-Based Robotic Manipulation Skills**

Does it work?

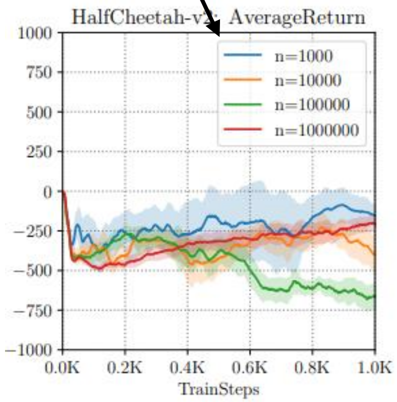
Method	Dataset	Success	Failure
Offline QT-Opt	580k offline	87%	13%
Finetuned QT-Opt	580k offline + 28k online	**96%**	**4%**

Kalashnikov, Irpan, Pastor, Ibarz, Herzong, Jang, Quillen, Holly, Kalakrishnan, Vanhoucke, Levine. **QT-Opt: Scalable Deep Reinforcement Learning of Vision-Based Robotic Manipulation Skills**

What's the problem?

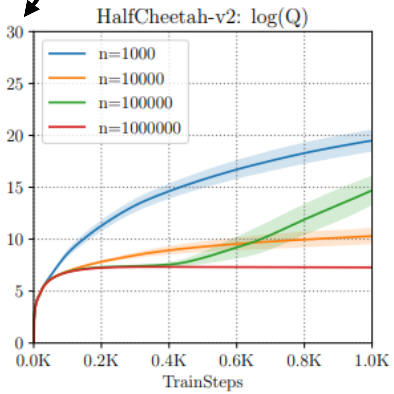
Hypothesis 1: Overfitting

amount of data



how well it does

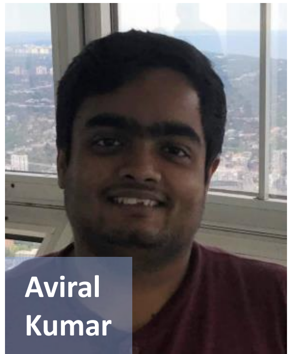
log scale (massive overestimation)

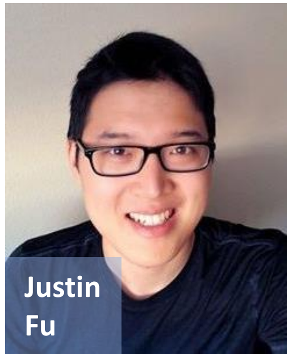


how well it *thinks* it does (Q-values)

Hypothesis 2: Training data is not good

Usually not the case: behavioral cloning of best data does better!





Kumar, Fu, Tucker, Levine. **Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction.** NeurIPS '19

Distribution shift in a nutshell

Example empirical risk minimization (ERM) problem:

$$\theta \leftarrow \arg\min_\theta E_{\mathbf{x} \sim p(\mathbf{x}), y \sim p(y|\mathbf{x})} \left[(f_\theta(\mathbf{x}) - y)^2 \right]$$

given some $\mathbf{x}^\star$, is $f_\theta(\mathbf{x}^\star)$ correct?

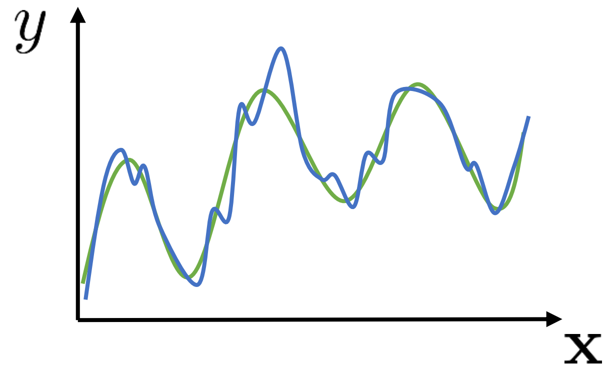
$E_{\mathbf{x} \sim p(\mathbf{x}), y \sim p(y|\mathbf{x})} \left[(f_\theta(\mathbf{x}) - y)^2 \right]$ is low

$E_{\mathbf{x} \sim \bar{p}(\mathbf{x}), y \sim p(y|\mathbf{x})} \left[(f_\theta(\mathbf{x}) - y)^2 \right]$ is not, for general $\bar{p}(\mathbf{x}) \neq p(\mathbf{x})$

what if $\mathbf{x}^\star \sim p(\mathbf{x})$? not necessarily...

usually we are not worried – neural nets generalize well!

what if we pick $\mathbf{x}^\star \leftarrow \arg\max_{\mathbf{x}} f_\theta(\mathbf{x})$?



Kumar, Fu, Tucker, Levine. **Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction.** NeurIPS ‘19

Where do we suffer from distribution shift?

~~$Q(\mathbf{s},\mathbf{a}) \leftarrow r(\mathbf{s},\mathbf{a}) + \max_{\mathbf{a}'} Q(\mathbf{s}',\mathbf{a}')$~~

$$Q(\mathbf{s},\mathbf{a}) \leftarrow \underbrace{r(\mathbf{s},\mathbf{a}) + E_{\mathbf{a}'\sim\pi_{\text{new}}}[Q(\mathbf{s}',\mathbf{a}')]}_{y(\mathbf{s},\mathbf{a})}$$

what is the objective?

$$\min_Q E_{(\mathbf{s},\mathbf{a})\sim\pi_\beta(\mathbf{s},\mathbf{a})}\left[(Q(\mathbf{s},\mathbf{a}) - y(\mathbf{s},\mathbf{a}))^2\right]$$

behavior policy

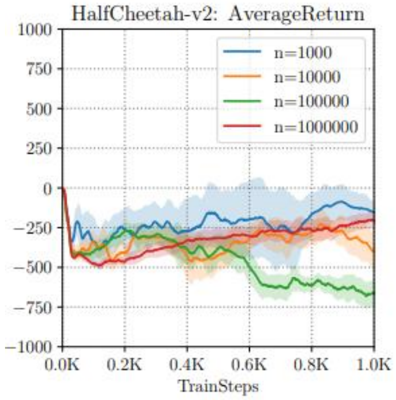
target value

expect good accuracy when $\pi_\beta(\mathbf{a}|\mathbf{s}) = \pi_{\text{new}}(\mathbf{a}|\mathbf{s})$

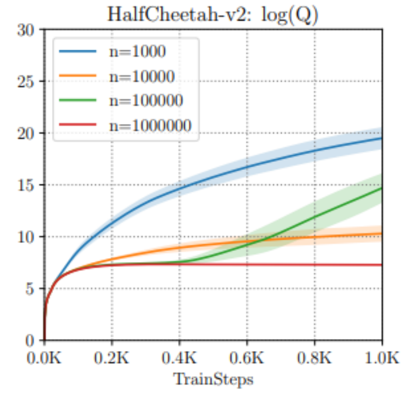
how often does *that* happen?

even *worse*: $\pi_{\text{new}} = \arg\max_\pi E_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})]$

(what if we pick $\mathbf{x}^\star \leftarrow \arg\max_{\mathbf{x}} f_\theta(\mathbf{x})$?)

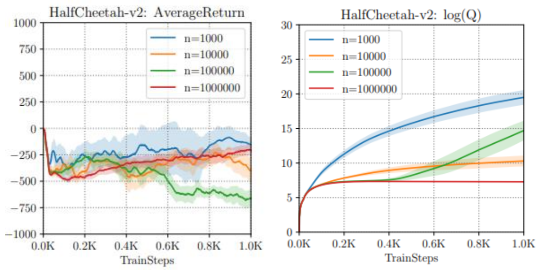


how well it does

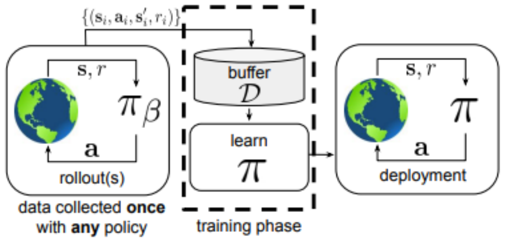


how well it *thinks* it does (Q-values)

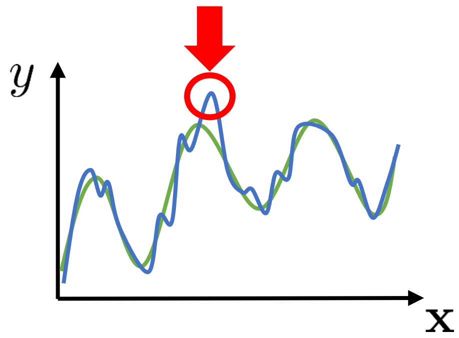
Kumar, Fu, Tucker, Levine. **Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction.** NeurIPS ‘19



Why is offline RL difficult?



How do we design offline RL algorithms?



Conservative Q-Learning

How do prior methods address this?

$$Q(\mathbf{s},\mathbf{a}) \leftarrow r(\mathbf{s},\mathbf{a}) + E_{\mathbf{a}'\sim\pi_{\text{new}}}[Q(\mathbf{s}',\mathbf{a}')]$$

$$\pi_{\text{new}}(\mathbf{a}|\mathbf{s}) = \arg\max_{\pi} E_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] \text{ s.t. } D_{\text{KL}}(\pi\|\pi_\beta) \leq \epsilon$$

This solves distribution shift, right?

No more erroneous values?

can partially mitigate with **support** constraint (see Kumar et al. ‘19 “BEAR”)

Issue 1: This might be **way** too conservative

Issue 2: Estimating the behavior policy is difficult

“policy constraint” method

very old idea (but it had no single name?)

Todorov et al. [passive dynamics in linearly-solvable MDPs]

Kappen et al. [KL-divergence control, etc.]

trust regions, covariant policy gradients, natural policy gradients, etc.

used in some form in recent papers:

Fox et al. ‘15 (“Taming the Noise…”)

Fujimoto et al. ‘18 (“Off Policy…”)

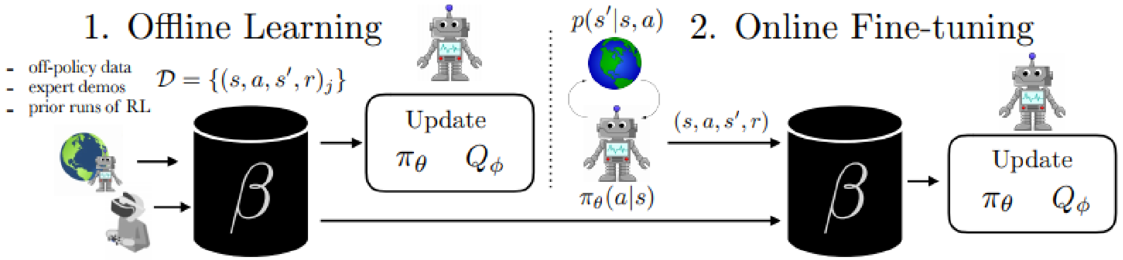
Jaques et al. ‘19 (“Way Off Policy…”)

Kumar et al. ‘19 (“Stabilizing…”)

Wu et al. ‘19 (“Behavior Regularized…”)

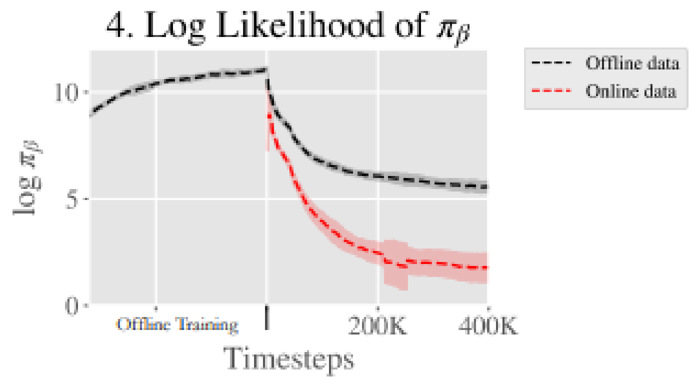
Levine, Kumar, Tucker, Fu. **Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems.** ‘20

How bad is it?



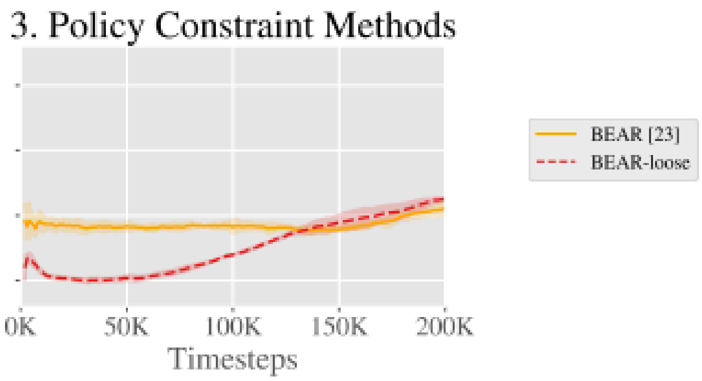
Issue 2: Estimating the behavior policy is difficult

Experiment: online finetuning from offline initialization



offline training

online training



online training

see also:
Ghasemipour et al., EMaQ: Expected-Max Q-Learning Operator for Simple Yet Effective Offline and Online RL, '20

- More **powerful behavior policy** models lead to **improvement**, implying behavior policy modeling is a **major bottleneck**

Nair, Dalal, Gupta, Levine. **Accelerating Online Reinforcement Learning with Offline Datasets.** '20

Avoiding behavior policies with **implicit** constraints

$$\pi_{\text{new}}(\mathbf{a}|\mathbf{s}) = \arg\max_{\pi} E_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] \text{ s.t. } D_{\text{KL}}(\pi\|\pi_\beta) \leq \epsilon$$

$$\pi^\star(\mathbf{a}|\mathbf{s}) = \frac{1}{Z(\mathbf{s})}\pi_\beta(\mathbf{a}|\mathbf{s})\exp\left(\frac{1}{\lambda}A^\pi(\mathbf{s},\mathbf{a})\right)$$

straightforward to show via duality

See also:
Peters et al. (REPS)
Rawlik et al. (“psi-learning”)
…many follow-ups

approximate via **weighted** max likelihood!

$$\pi_{\text{new}}(\mathbf{a}|\mathbf{s}) = \arg\max_{\pi} E_{(\mathbf{s},\mathbf{a})\sim\pi_\beta}\left[\log\pi(\mathbf{a}|\mathbf{s})\frac{1}{Z(\mathbf{s})}\exp\left(\frac{1}{\lambda}A^{\pi_{\text{old}}}(\mathbf{s},\mathbf{a})\right)\right]$$

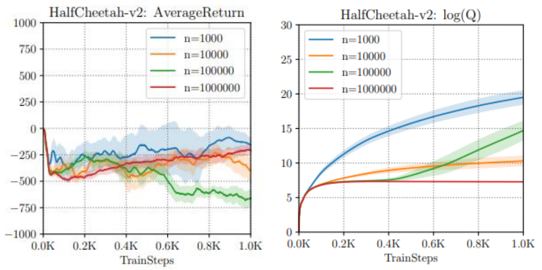
samples from dataset
$\mathbf{a} \sim \pi_\beta(\mathbf{a}|\mathbf{s})$

critic can be used to give us this

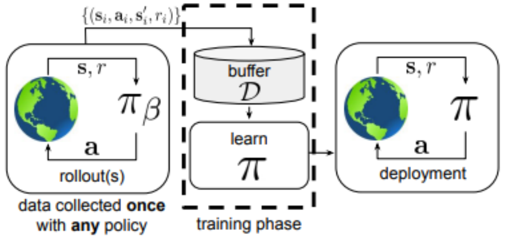
but maybe we can solve the overestimation problem at the **root**?

Peng*, Kumar*, Levine. **Advantage-Weighted Regression.** ‘19

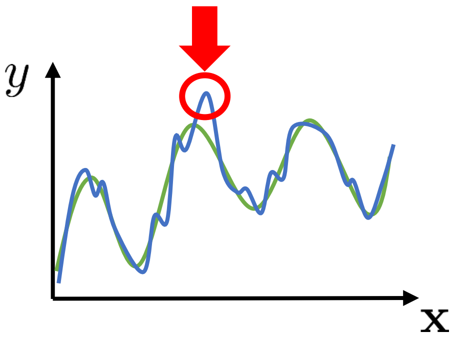
Nair, Dalal, Gupta, Levine. **Accelerating Online Reinforcement Learning with Offline Datasets.** ‘20



Why is offline RL difficult?

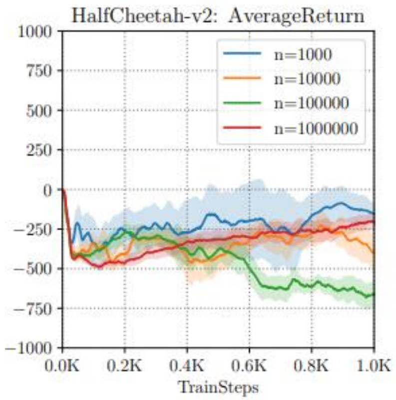


How do we design offline RL algorithms?

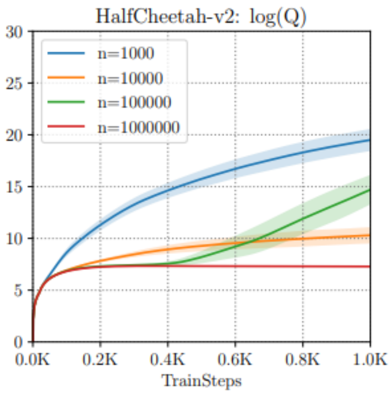


Conservative Q-Learning

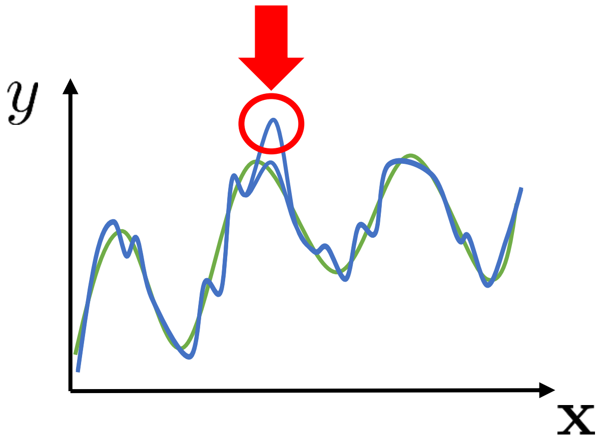
What about those Q-value errors?



how well it does



how well it *thinks* it does (Q-values)



$$\hat{Q}^\pi = \arg\min_Q \max_\mu \alpha E_{\mathbf{s}\sim D,\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})]$$ term to push down big Q-values

regular objective $$+E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[(Q(\mathbf{s},\mathbf{a}) - (r(\mathbf{s},\mathbf{a}) + E_\pi[Q(\mathbf{s}',\mathbf{a}')]))^2\right]$$

can show that $\hat{Q}^\pi \leq Q^\pi$ for large enough α

true Q-function

Learning with Q-function lower bounds

Algorithm:

1. Learn $\hat{Q}^\pi$ for current π such that $\hat{Q}^\pi \leq Q^\pi$
2. $\pi \leftarrow \arg\max_{\pi_{\text{new}}} E_{\pi_{\text{new}}}[\hat{Q}^\pi]$

A *better* bound:

<u>always</u> pushes Q-values down

push <u>up</u> on (**s**, **a**) samples in data

$$\hat{Q}^\pi = \arg\min_Q \max_\mu \alpha E_{\mathbf{s}\sim D, \mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \alpha E_{(\mathbf{s},\mathbf{a})\sim D}[Q(\mathbf{s},\mathbf{a})]$$
$$+ E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[(Q(\mathbf{s},\mathbf{a}) - (r(\mathbf{s},\mathbf{a}) + E_\pi[Q(\mathbf{s}',\mathbf{a}')]))^2\right]$$

no longer guaranteed that $\hat{Q}^\pi(\mathbf{s},\mathbf{a}) \leq Q^\pi(\mathbf{s},\mathbf{a})$ *for all* $(\mathbf{s},\mathbf{a})$

but guaranteed that $E_{\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^\pi(\mathbf{s},\mathbf{a})] \leq E_{\pi(\mathbf{a}|\mathbf{s})}[Q^\pi(\mathbf{s},\mathbf{a})]$ *for all* $\mathbf{s} \in D$

Kumar, Zhou, Tucker, Levine. **Conservative Q-Learning for Offline Reinforcement Learning.** ‘20

The conservative Q-learning (CQL) bound

minimize the **big** Q-values

maximize Q-values of state-action pairs in data

$$\hat{Q}^{\pi}_{\text{CQL}} = \arg\min_{Q}\max_{\mu} \alpha E_{\mathbf{s}\sim D,\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \alpha E_{(\mathbf{s},\mathbf{a})\sim D}[Q(\mathbf{s},\mathbf{a})] + \frac{1}{2} E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[(Q(\mathbf{s},\mathbf{a}) - (r(\mathbf{s},\mathbf{a}) + E_{\pi}[Q(\mathbf{s}',\mathbf{a}')]))^2\right]$$

Theorem 3.2 (Equation 2 results in a tighter lower bound). *The value of the policy under the Q-function from Equation 2, $\hat{V}^{\pi}(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}(\mathbf{s},\mathbf{a})]$, lower-bounds the true value of the policy obtained via exact policy evaluation, $V^{\pi}(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[Q^{\pi}(\mathbf{s},\mathbf{a})]$, when $\mu = \pi$, according to:*

$$\forall \mathbf{s},\ \ \hat{V}^{\pi}(\mathbf{s}) \leq V^{\pi}(\mathbf{s}) - \underbrace{\alpha\left(I - \gamma P^{\pi}\right)^{-1} \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}\left[\frac{\pi(\mathbf{a}|\mathbf{s})}{\hat{\pi}_{\beta}(\mathbf{a}|\mathbf{s})} - 1\right](\mathbf{s})}_{\text{pessimism due to regularizer}} + \underbrace{\left(I - \gamma P^{\pi}\right)^{-1} \frac{C_{r,T,\delta} R_{\max}}{(1-\gamma)}}_{\text{accounts for sampling error}}.$$

concentration constant

Kumar, Zhou, Tucker, Levine. **Conservative Q-Learning for Offline Reinforcement Learning.** '20

Does the bound hold in practice?

Underestimation vs. overestimation

$$E[\hat{Q}(\mathbf{s}, \mathbf{a})] - E[Q(\mathbf{s}, \mathbf{a})]$$

← from Monte Carlo estimation

Task Name	CQL($\mathcal{H}$)	CQL (Eqn. 1)	Ensemble(2)	Ens.(4)	Ens.(10)	Ens.(20)	BEAR
hopper-medium-expert	**-43.20**	-151.36	3.71e6	2.93e6	0.32e6	24.05e3	65.93
hopper-mixed	**-10.93**	-22.87	15.00e6	59.93e3	8.92e3	2.47e3	1399.46
hopper-medium	**-7.48**	-156.70	26.03e12	437.57e6	1.12e12	885e3	4.32

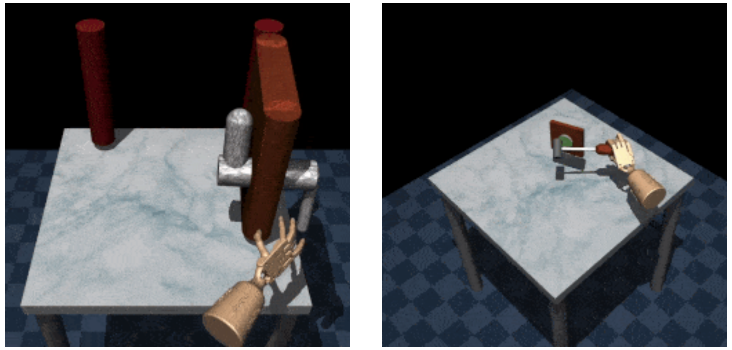
all prior methods have positive errors = wild optimism

CQL **always** has negative errors = pessimism

Kumar, Zhou, Tucker, Levine. **Conservative Q-Learning for Offline Reinforcement Learning.** ‘20

D4RL: Datasets for Data-Driven Deep RL

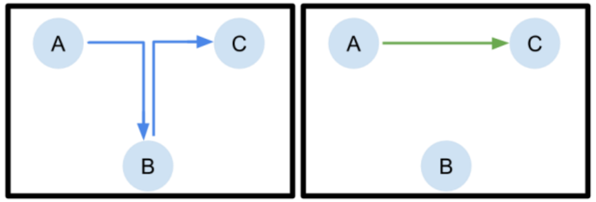
What are some important principles to keep in mind?



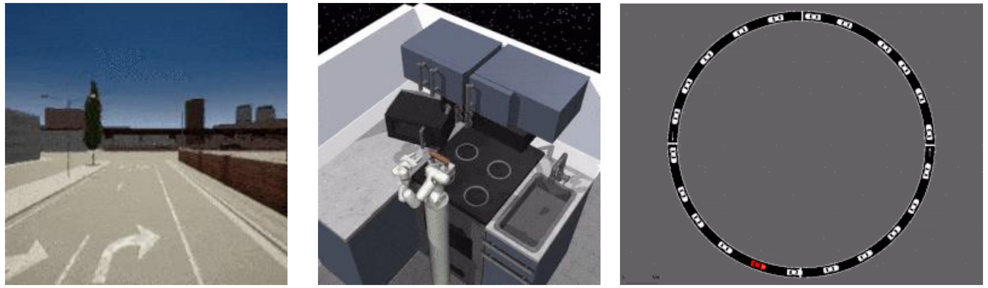
simulation & human data from Rajeswaran et al.

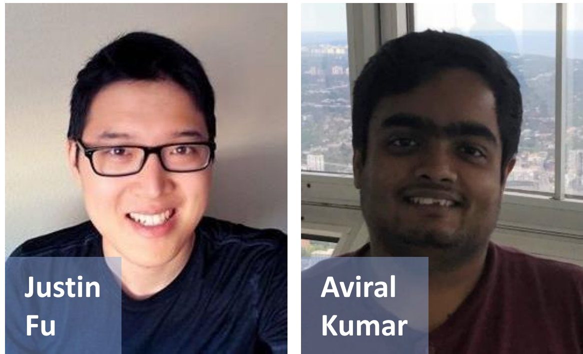
Data from non-RL policies, including data from humans

Stitching: data where dynamic programming can find much better solutions



Realistic tasks





Fu, Kumar, Nachum Tucker, Levine. **D4RL: Datasets for Data-Driven Deep Reinforcement Learning.** ‘20

How does CQL compare?

"**1%**" dataset from Agarwal et al.

Task Name	QR-DQN	REM	CQL($\mathcal{H}$)
Pong (1%)	-13.8	-6.9	**19.3**
Breakout	7.9	11.0	**61.1**
Q*bert	383.6	[illegible]	**[illegible]4012.0**
Seaquest	672.9	[illegible]	**[illegible]79.4**
Asterix*	166.3	386.5	**[illegible]592.4**

1.5 – 6x better

baseline: just clone the data

Domain	Task Name	BC	SAC	BEAR	BRAC-p	BRAC-v	CQL($\mathcal{H}$)	CQL(ρ)
AntMaze	antmaze-umaze	65.0	0.0	**73.0**	50.0	70.0	**74.0**	**73.5**
	antmaze-umaze-diverse	55.0	0.0	61.0	40.0	70.0	**84.0**	61.0
	antmaze-medium-play	0.0	0.0	0.0	0.0	0.0	**61.2**	4.6
	antmaze-medi[illegible]	[illegible]	[illegible]	[illegible]	0.[illegible]	[illegible]	**53.7**	5.1
	antmaze-large[illegible]	[illegible]	[illegible]	[illegible]	0.0	[illegible]	**15.8**	3.2
	antmaze-large[illegible]	[illegible]	[illegible]	[illegible]	0.0	0.0	**14.9**	2.3
Adroit	pen-human	[illegible]	[illegible]	[illegible]	8.1	0.6	37.5	**55.8**
	hammer-huma[illegible]	[illegible]	[illegible]	[illegible]	0.3	[illegible]	**4.4**	2.1
	door-human	[illegible]	[illegible]	[illegible]	-0.3	[illegible]	**9.9**	**9.1**
	relocate-huma[illegible]	[illegible]	[illegible]	[illegible]	-0.3	-0.3	0.20	**0.35**
	pen-cloned	[illegible]	[illegible]	[illegible]	1.6	-2.5	39.2	40.3
	hammer-cloned	0.8	0.2	0.3	0.3	[illegible]	2.1	**5.7**
	door-cloned	[illegible]	[illegible]	[illegible]	-0.1	[illegible]	0.4	**3.5**
	relocate-clone[illegible]	[illegible]	[illegible]	[illegible]	-0.3	-0.3	**-0.1**	**-0.1**
Kitchen	kitchen-comp[illegible]	[illegible]	[illegible]	[illegible]	0.0	[illegible]	**43.8**	31.3
	kitchen-partia[illegible]	[illegible]	[illegible]	[illegible]	0.0	[illegible]	**49.8**	**50.1**
	kitchen-undirected	47.5	2.5	47.2	0.0	[illegible]	**51.0**	**52.4**

nothing works on the harder mazes?

nothing beats behavioral cloning?

"**infinitely**" better

1.5-3x better

up to 5x better

1.1 – 1.3x better

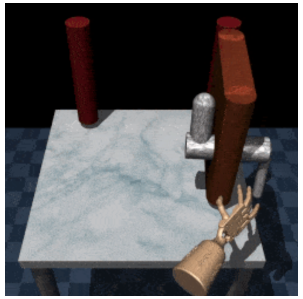
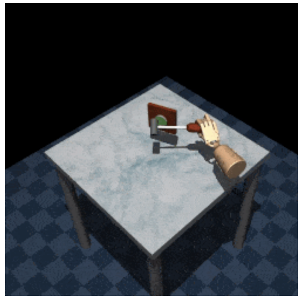
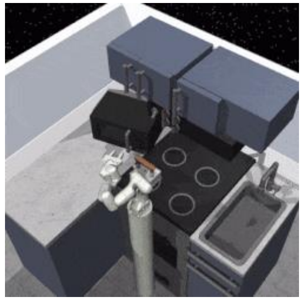
CQL seems to work pretty well on many tasks!

And we seem to know *why* it works!

But there is still plenty of room for improvement…

Kumar, Zhou, Tucker, Levine. **Conservative Q-Learning for Offline Reinforcement Learning.** '20




- Offline RL is quite difficult, but has **enormous promise**, and initial results suggest it can be **extremely powerful**

- Effective (dynamic programming) offline RL methods can be implemented by imposing **constraints** on the policy, perhaps implicitly

- Learning a lower bound Q-function (i.e., conservative Q-learning) can **substantially** improve offline RL performance

$$\hat{Q}^\pi_{\text{CQL}} = \arg\min_Q \max_\mu \alpha E_{\mathbf{s}\sim D,\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \alpha E_{(\mathbf{s},\mathbf{a})\sim D}[Q(\mathbf{s},\mathbf{a})] + \frac{1}{2}E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - (r(\mathbf{s},\mathbf{a}) + E_\pi[Q(\mathbf{s}',\mathbf{a}')])\right)^2\right]$$

Kumar, Fu, Tucker, Levine. **Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction.** NeurIPS ‘19

Nair, Dalal, Gupta, Levine. **Accelerating Online Reinforcement Learning with Offline Datasets.** ‘20

Kumar, Zhou, Tucker, Levine. **Conservative Q-Learning for Offline Reinforcement Learning.** ‘20

Fu, Kumar, Nachum Tucker, Levine. **D4RL: Datasets for Data-Driven Deep Reinforcement Learning.** ‘20