Offline Reinforcement Learning: Part 2

CS 224R

Course reminders

- Homework 2 due **tonight.**
- Homework 3 out today.
- Project proposal feedback coming out soon.

Announcements

- Moving two office hours (Dilip, Ansh) from in-person to hybrid

The plan for today

Offline RL: Part 2

1. Recap
2. Revisiting imitation learning for offline RL
   a. Weighted imitation learning
   b. Conditional imitation
3. Offline evaluation & hyperparameter tuning
4. Applications

Part of homework 3!

Key learning goals:

- two approaches for offline RL (+ when they work & don't work!)
- important considerations for tuning offline RL methods

Recap: Offline RL, data constraints, conservativeness

Why offline RL? Online data is expensive. *Reusing offline data* is good!

Key challenge: Overestimating Q-values because of shift between π_β and π_θ

- can **explicitly constrain** to the data by modeling π_β
  + fairly intuitive - often too conservative in practice
- implicitly constrain to data by **penalizing Q-values**
  + simple + can work well in practice - need to tune alpha

Trajectory stitching allows offline RL methods to improve over imitation.

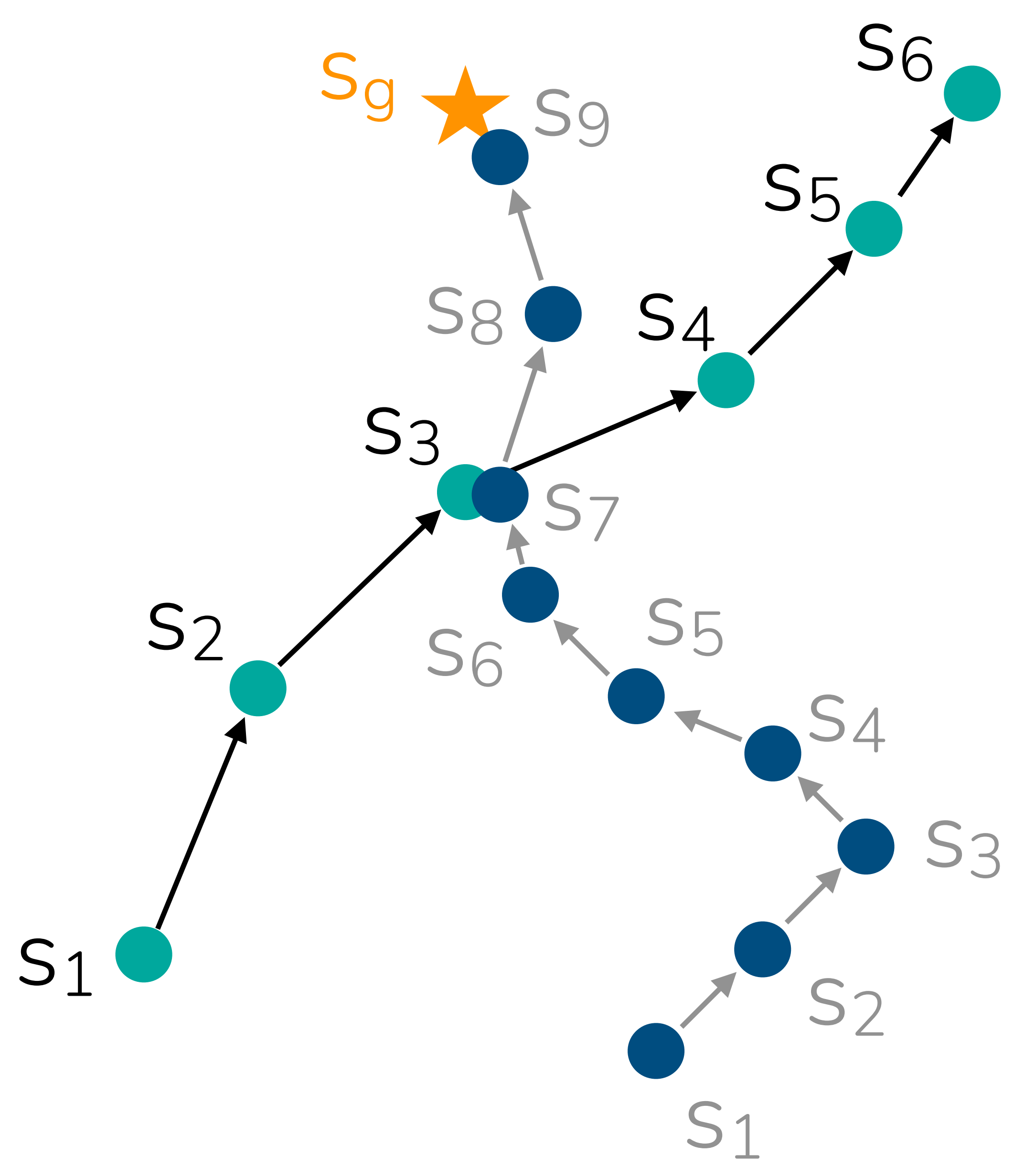
Recap: Why offline RL versus imitation learning?

(**Recall**: Imitation methods can't outperform the expert.)

Offline data may not be optimal!

—> Offline RL can leverage reward information to outperform behavior policy.

—> Good offline RL methods can ***stitch*** together good behaviors.



s_1 -> s_3 is good behavior

s_7 -> s_9 is good behavior

Offline RL methods can learn a policy that goes from s_1 to s_9!

Other ways to leverage reward information in imitation?

If we have reward labels: imitate only the good trajectories?

Filtered behavior cloning:

1. Rank trajectories by return $r(\tau) = \sum_{(\mathbf{s}_t, \mathbf{a}_t) \in \tau} r(\mathbf{s}_t, \mathbf{a}_t)$
2. Filter dataset to include top k% of data $\tilde{D} : \{\tau \,|\, r(\tau) > \eta\}$
3. Imitate filtered dataset: $\max_{\pi} \sum_{(\mathbf{s}, \mathbf{a}) \in \tilde{D}} \log \pi(\mathbf{a} \,|\, \mathbf{s})$

A very primitive approach to using reward information.

Therefore, a **good baseline** to test against!

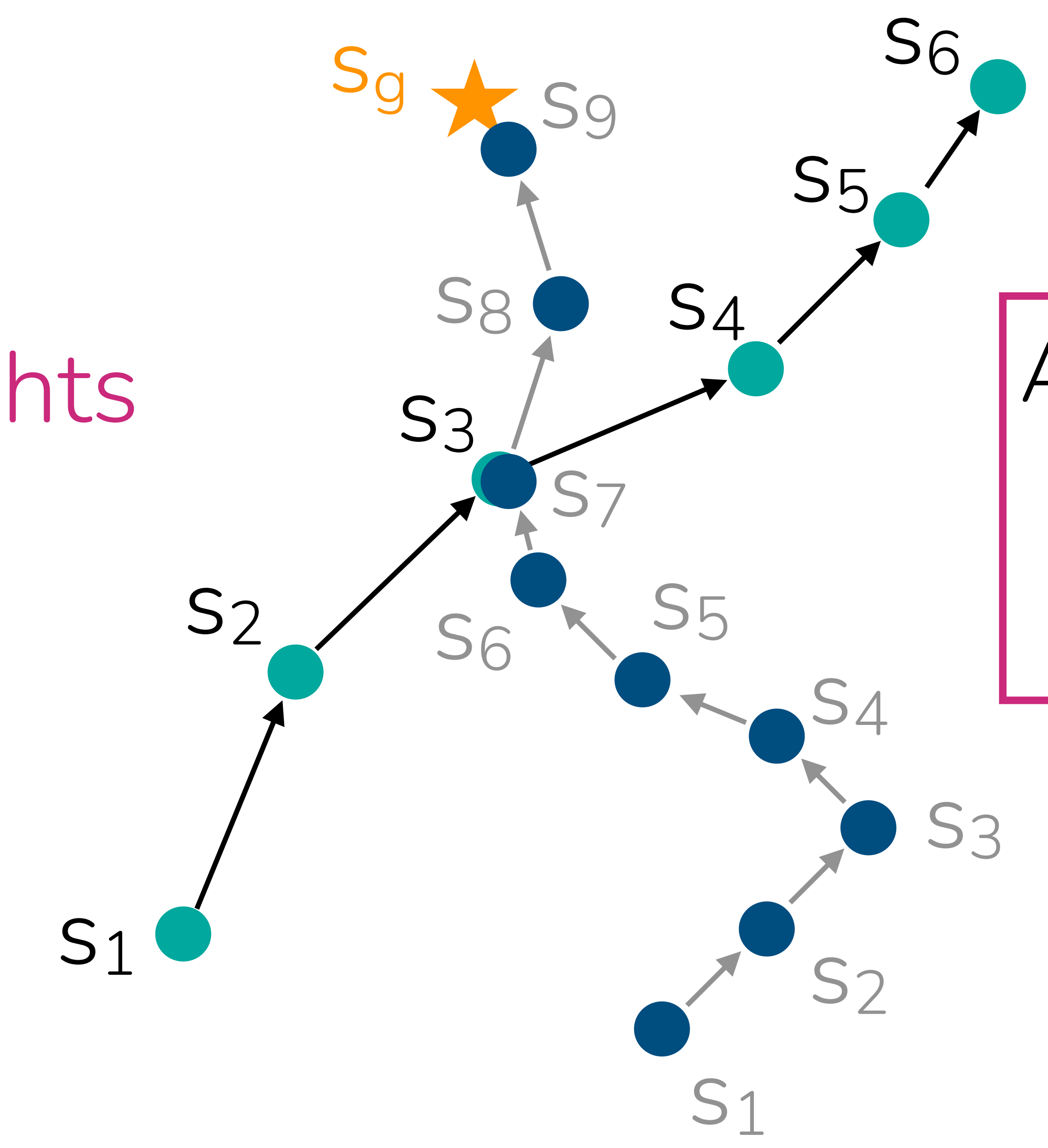
Better way to do weighted imitation learning?

Could we weight each transition depending on **how good the action is**?

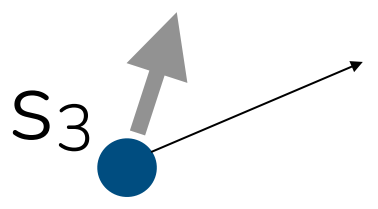
How do you measure how good an action is? Recall: advantage function A

$$\boxed{A^{\pi}(\mathbf{s}_t, \mathbf{a}_t) = Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t) - V^{\pi}(\mathbf{s}_t)\text{: how much better } \mathbf{a}_t \text{ is}}$$

$$\theta \leftarrow \underbrace{\arg\max_{\theta} E_{\mathbf{s},\mathbf{a}\sim D} \left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\right.}_{\text{standard imitation learning}} \underbrace{\left.\exp(A(\mathbf{s},\mathbf{a}))\right]}_{\text{with advantage weights}}$$



Adv. weights for s_3



Aside: Can show that advantage-weighted objective approximates KL-constrained objective.

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

See Peters et al. (REPS), Rawlik et al. (“psi-learning”)

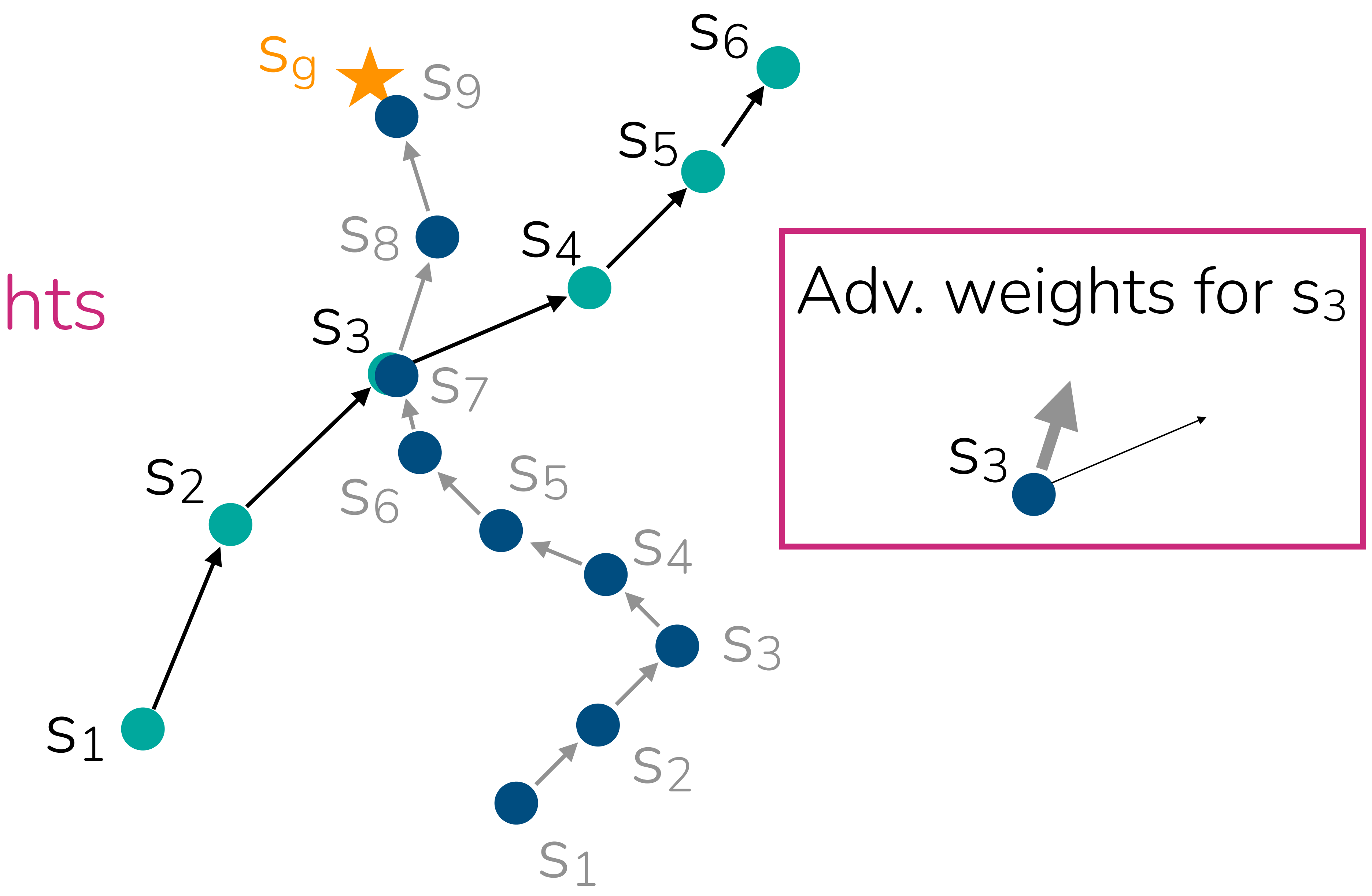
Better way to do weighted imitation learning?

Could we weight each transition depending on **how good the action is**?

How do you measure how good an action is? Recall: advantage function A

$$\boxed{A^\pi(\mathbf{s}_t, \mathbf{a}_t) = Q^\pi(\mathbf{s}_t, \mathbf{a}_t) - V^\pi(\mathbf{s}_t)\text{: how much better } \mathbf{a}_t \text{ is}}$$

$$\theta \leftarrow \underbrace{\arg\max_\theta E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\right.}_{\text{standard imitation learning}}\underbrace{\left.\exp(A(\mathbf{s},\mathbf{a}))\right]}_{\text{with advantage weights}}$$



Advantage of which policy? We'll use A^{π_β} for now.

Key question: How to estimate the advantage function?

Advantage-weighted regression

Could we weight each transition depending on **how good the action is**?

$$\theta \leftarrow \underbrace{\arg\max_{\theta} E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\right.}_{\text{standard imitation learning}}\underbrace{\left.\exp(A(\mathbf{s},\mathbf{a}))\right]}_{\text{with advantage weights}}$$

Key question: How to estimate the advantage function?

Estimate $V^{\pi_\beta}(s)$ with Monte Carlo, $\min_V E_{(\mathbf{s},\mathbf{a})\sim D}\left[\left(R_{\mathbf{s},\mathbf{a}} - V(\mathbf{s})\right)^2\right]$

Approximate $\hat{A}^{\pi_\beta}(\mathbf{s},\mathbf{a}) = R_{\mathbf{s},\mathbf{a}} - V(\mathbf{s})$

empirical return

Peng, Kumar, Zhang, Levine. Advantage-Weighted Regression. '19

Advantage-weighted regression

Full AWR algorithm

1. Fit value function: $\hat{V}^{\pi_\beta}(s) \leftarrow \arg\min_V E_{(\mathbf{s},\mathbf{a})\sim D}\left[\left(R_{\mathbf{s},\mathbf{a}} - V(\mathbf{s})\right)^2\right]$

2. Train policy: $\hat{\pi} \leftarrow \arg\max_\pi E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi(\mathbf{a}\,|\,\mathbf{s})\exp\left(\frac{1}{\alpha}\left(R_{\mathbf{s},\mathbf{a}} - \hat{V}^{\pi_\beta}(\mathbf{s})\right)\right)\right]$

hyperparameter

+ Simple

+ Avoids querying or training on any OOD actions!

- Monte Carlo estimation is noisy

- $\hat{A}^{\pi_\beta}$ assumes weaker policy than $\hat{A}^{\pi_\theta}$

Peng, Kumar, Zhang, Levine. Advantage-Weighted Regression. '19

Advantage-weighted regression

Estimate advantage function with TD updates instead of Monte Carlo?

1. Estimate Q^π-function: $\min_Q E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma E_{\mathbf{a}'\sim\pi(\cdot|\mathbf{s})}[Q(\mathbf{s}',\mathbf{a}')]\right)\right)^2\right]$

2. Estimate advantage as: $\hat{A}^\pi(\mathbf{s},\mathbf{a}) = \hat{Q}^\pi(\mathbf{s},\mathbf{a}) - E_{\bar{\mathbf{a}}\sim\pi(\cdot|\mathbf{s})}[\hat{Q}^\pi(\mathbf{s},\bar{\mathbf{a}})]$

3. Update policy as before: $\hat{\pi} \leftarrow \arg\max_\pi E_{\underline{\mathbf{s},\mathbf{a}\sim D}}\left[\log \pi(\mathbf{a}\,|\,\mathbf{s}) \exp\left(\frac{1}{\alpha}\hat{A}^\pi(\mathbf{s},\mathbf{a})\right)\right]$

"advantage weighted actor critic"

+ Policy still only trained on actions in data.

+ Temporal difference updates instead of Monte Carlo.

What might go wrong?

- Possibly querying OOD actions!

Nair, Gupta, Dalal, Levine. AWAC. '20

Wang et al. Critic Regularized Regression. NeurIPS '20

Can we do better?

Want to estimate advantages using TD updates, without querying Q on OOD actions.

AWAC: Estimate Q-function: $\min_{Q} E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma E_{\substack{\cancel{\mathbf{a}'\sim\pi(\cdot|\mathbf{s})} \\ \mathbf{a}'\sim D}}[Q(\mathbf{s}',\mathbf{a}')]\right)\right)^2\right]$

"SARSA algorithm"

Can we do better?

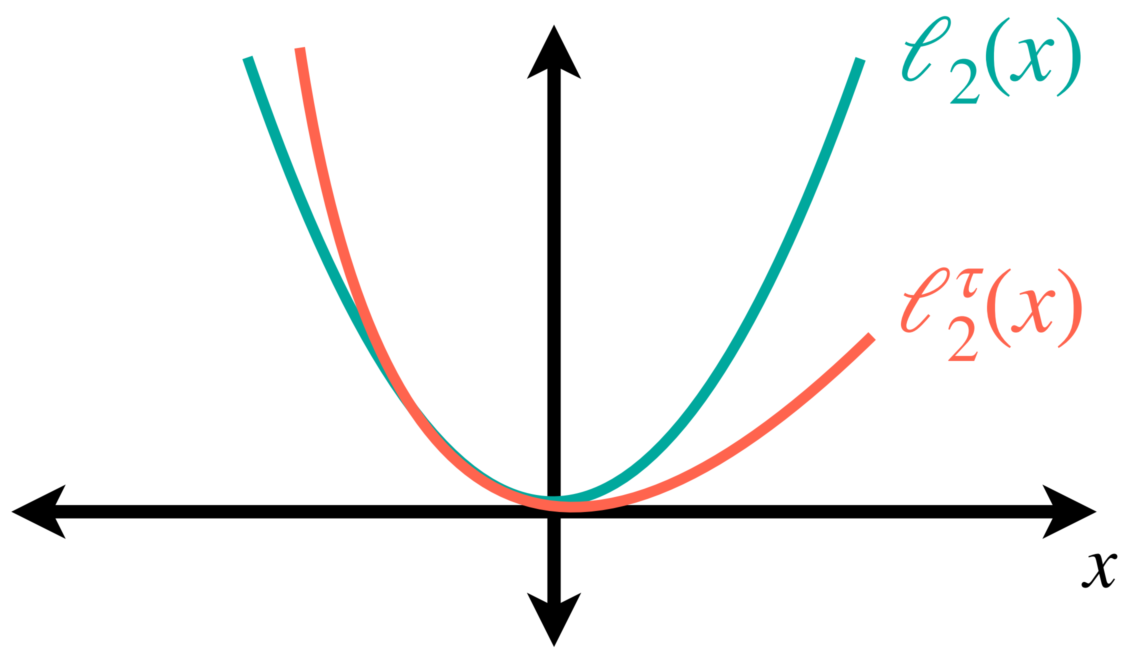
Want to estimate advantages using TD updates, without querying Q on OOD actions.

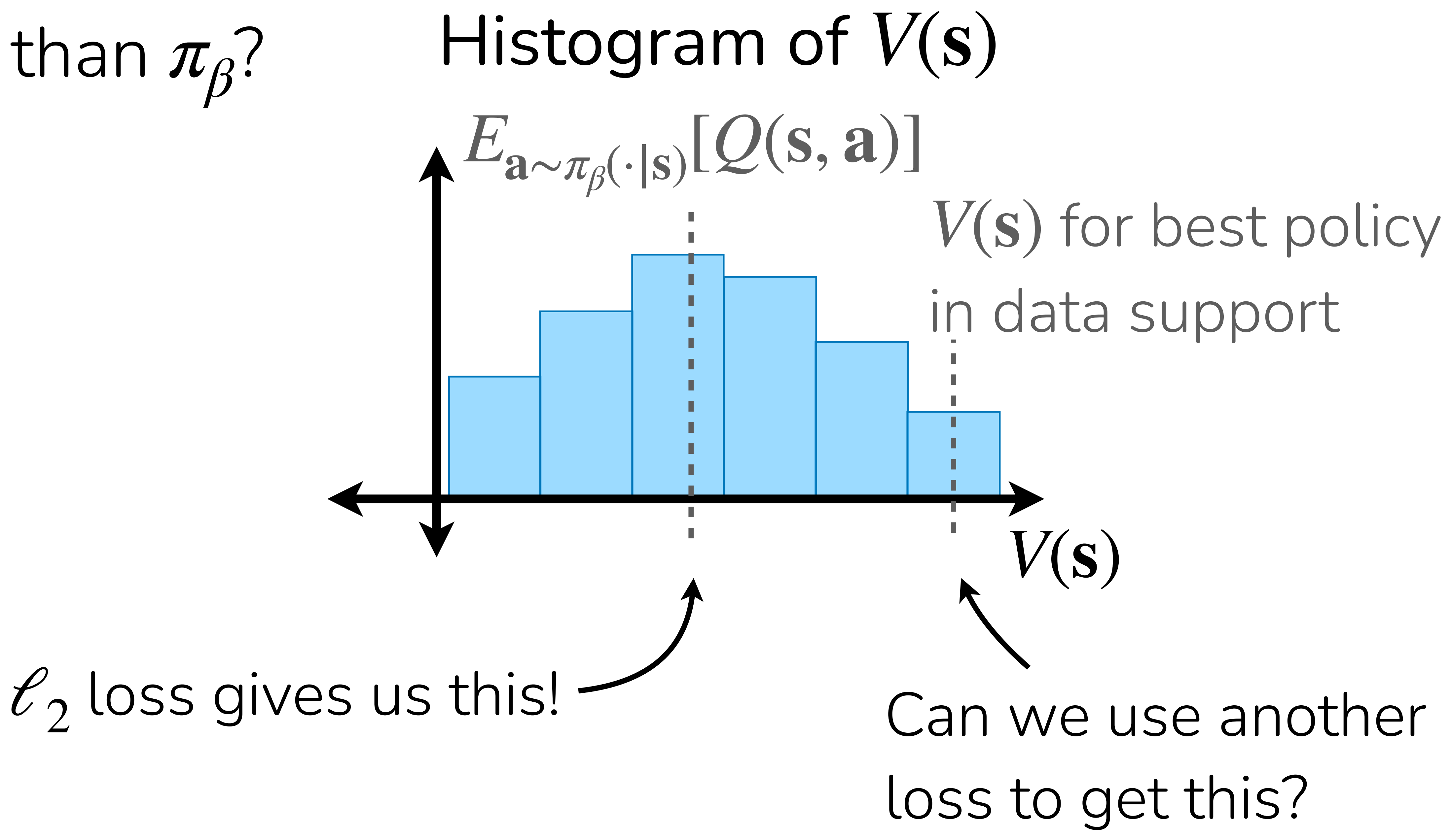
SARSA update: $\hat{Q}^{\pi_\beta} \leftarrow \arg\min_Q E_{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{a}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma \underline{Q(\mathbf{s}',\mathbf{a}')}\right)\right)^2\right]$

a sample of $V^{\pi_\beta}(\mathbf{s}')$

Can we estimate Q for a policy that is better than π_β?

Idea: Use an asymmetric loss function



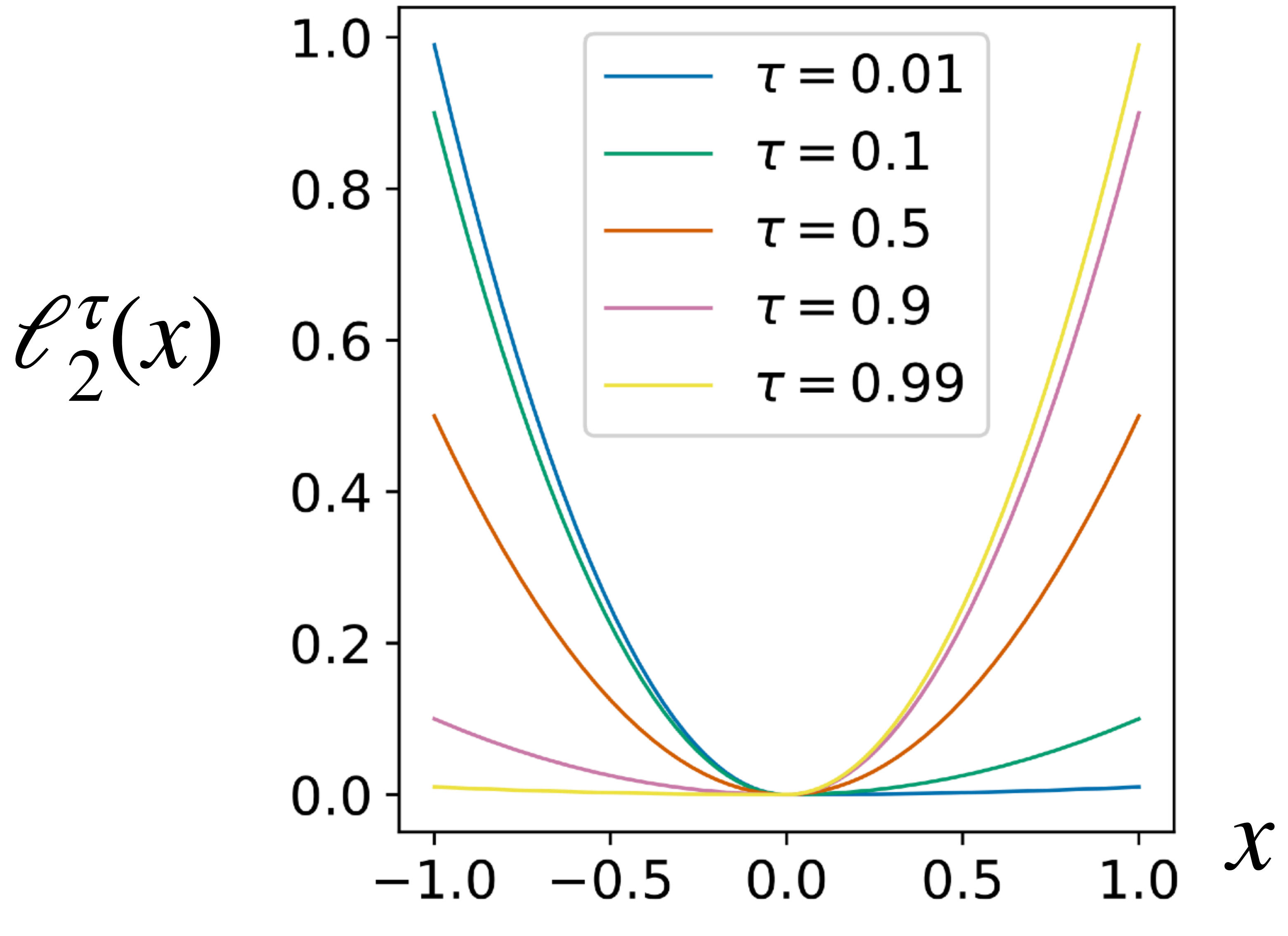


Aside: Expectile regression

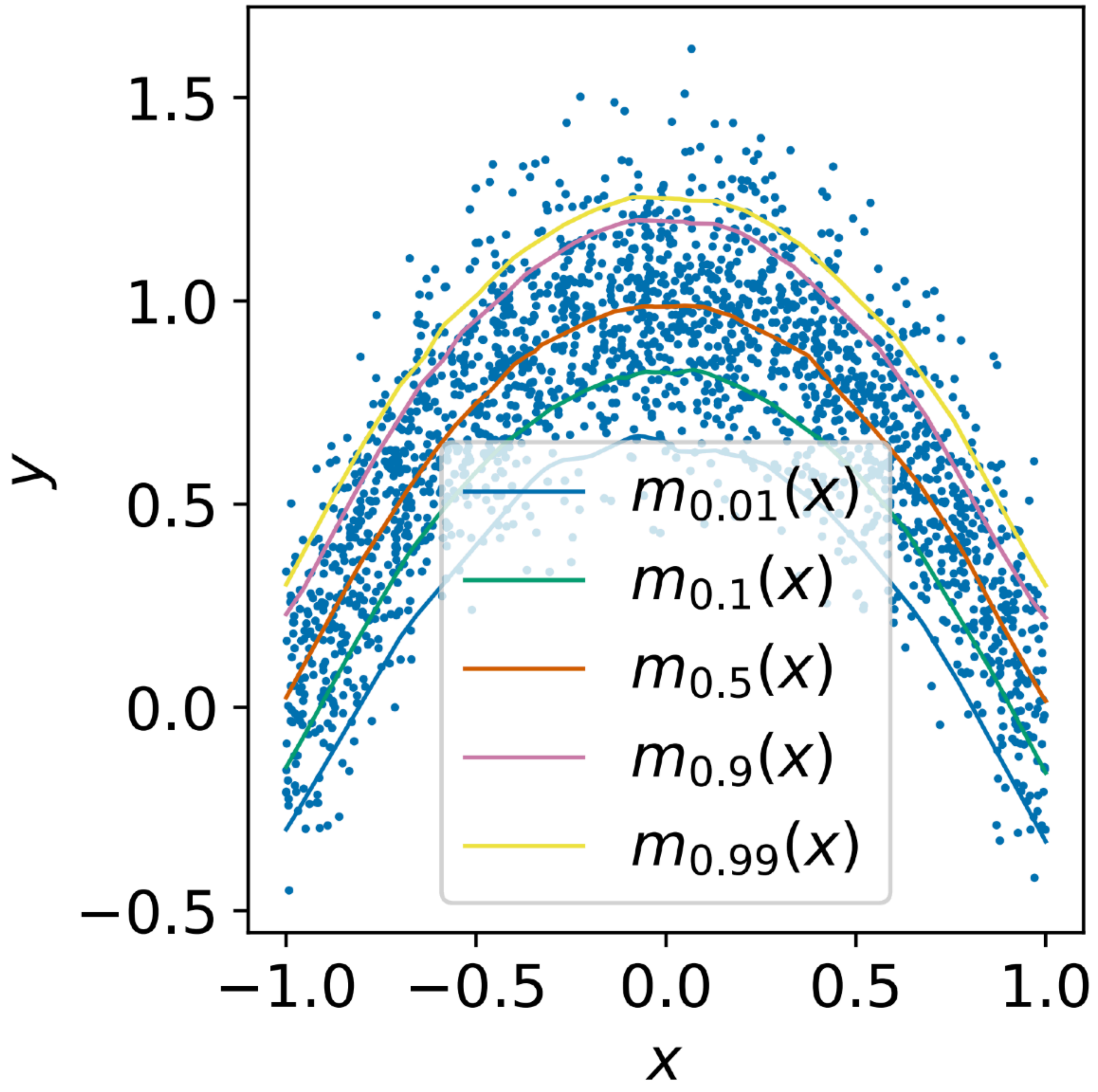
Instead of getting the mean of a random variable, can we get a higher or lower expectile?

Expectile regression loss:

$$\ell_2^\tau(x) = \begin{cases} (1-\tau)x^2 & \text{if } x < 0 \\ \tau x^2 & \text{otherwise} \end{cases}$$



Example with a 2D random variable



Kostrikov, Nair, Levine. Implicit Q-Learning. ICLR '22

Can we do better?

Want to estimate advantages using TD updates, without querying Q on OOD actions.

Full algorithm

Fit V with expectile loss: $\hat{V}(\mathbf{s}) \leftarrow \arg\min_V E_{(\mathbf{s},\mathbf{a})\sim D}\left[\ell_2^\tau\left(V(\mathbf{s}) - \hat{Q}(\mathbf{s},\mathbf{a})\right)\right]$ using small $\tau < 0.5$

Update Q with typical MSE loss: $\hat{Q}(\mathbf{s},\mathbf{a}) \leftarrow \arg\min_Q E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma\hat{V}(\mathbf{s}')\right)\right)^2\right]$

Extract policy with AWR: $\hat{\pi} \leftarrow \arg\max_\pi E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi(\mathbf{a}\,|\,\mathbf{s})\exp\left(\frac{1}{\alpha}\left(\hat{Q}(\mathbf{s},\mathbf{a}) - \hat{V}(\mathbf{s})\right)\right)\right]$

+ Never need to query OOD actions!

+ Policy (still) only trained on actions in data.

+ Decoupling actor & critic training —> computationally fast

policy improvement is implicit

-> implicit Q-learning (IQL)

You will implement IQL in homework 3!

Kostrikov, Nair, Levine. Implicit Q-Learning. ICLR '22

The plan for today

Offline RL: Part 2

1. Recap
2. Revisiting imitation learning for offline RL
    a. Weighted imitation learning
    b. Conditional imitation
3. Offline evaluation & hyperparameter tuning
4. Applications

Revisiting Filtered Behavior Cloning

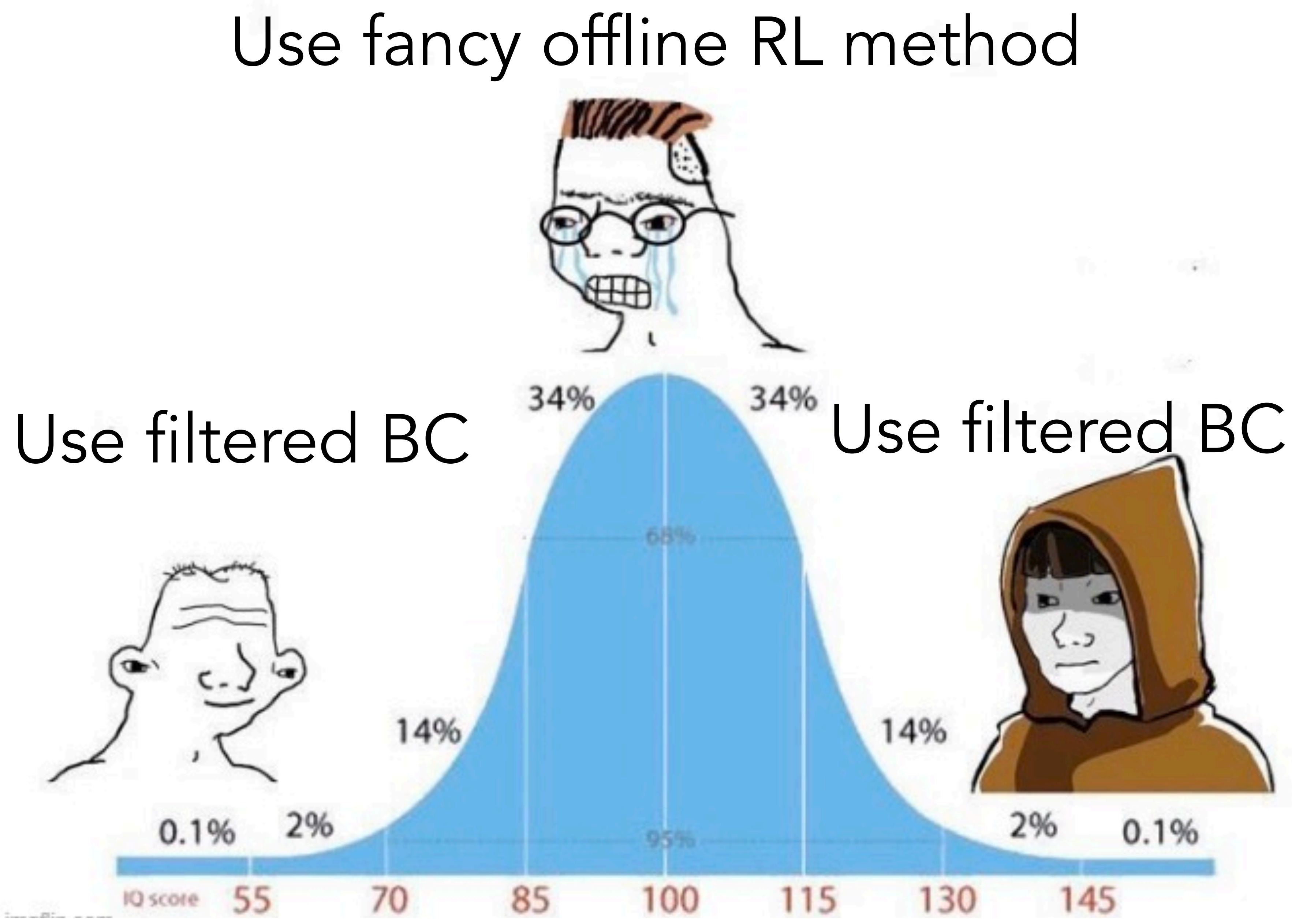
If we have reward labels: imitate only the good trajectories?

Filtered behavior cloning:

1. Rank trajectories by return $r(\tau) = \sum_{(\mathbf{s}_t,\mathbf{a}_t)\in\tau} r(\mathbf{s}_t, \mathbf{a}_t)$

2. Filter dataset to include top k% of data $\tilde{D} : \{\tau \,|\, r(\tau) > \eta\}$

3. Imitate filtered dataset: $\max_{\pi} \sum_{(\mathbf{s},\mathbf{a})\in\tilde{D}} \log \pi(\mathbf{a}\,|\,\mathbf{s})$



A very primitive approach to using reward information.

For some datasets, filtered BC can actually work really well!

What if we feel bad about discarding data?

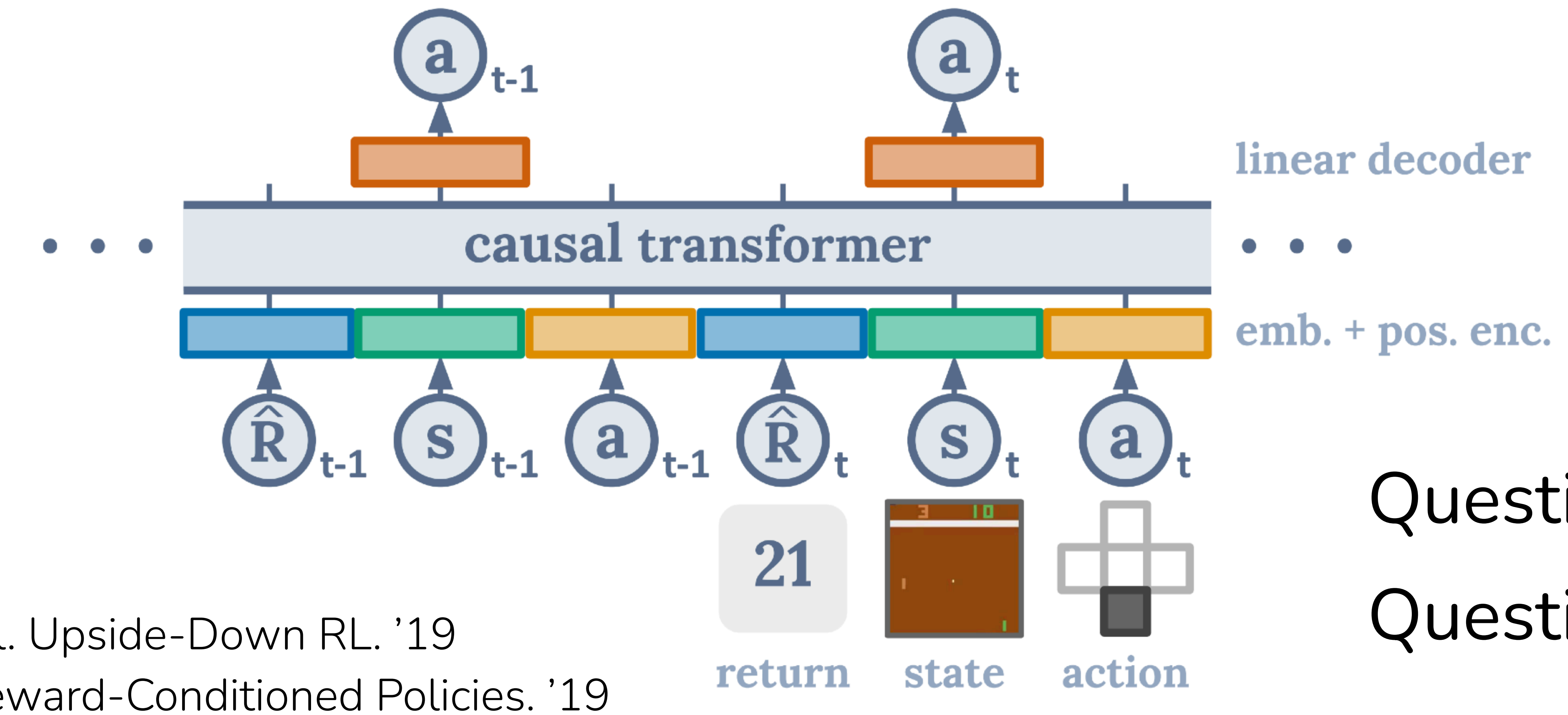
Return-conditioned policies

1. Imitate entire dataset: $\max_{\pi} \sum_{(\mathbf{s},\mathbf{a}) \in D} \log \pi(\mathbf{a} \mid \mathbf{s}, \underline{R_{\mathbf{s},\mathbf{a}}})$

Condition policy on (empirical) return to go.

- Policy will learn to mimic good and poor behaviors (and everything in between!)
- Pass in high return at test time
- Can use a sequence model:

Referred to as: upside-down RL, reward-conditioned policies, decision transformers



Question: Can this approach do data stitching?

Question: When would a sequence model be helpful?

Srivastava et al. Upside-Down RL. '19
Kumar et al. Reward-Conditioned Policies. '19
Chen*, Lu* et al. Decision Transformer. '21

The plan for today

Offline RL: Part 2

1. Recap
2. Revisiting imitation learning for offline RL
    a. Weighted imitation learning
    b. Conditional imitation
3. **Offline evaluation & hyperparameter tuning**
4. Applications

Hyperparameter tuning for offline RL

Train policy π_θ using offline dataset D. True objective: $\max_\theta \sum_t \mathbb{E}_{\mathbf{s}_t \sim d^{\pi_\theta}(\cdot), \mathbf{a}_t \sim \pi_\theta(\cdot|\mathbf{s}_t)} \left[r(\mathbf{s}_t, \mathbf{a}_t) \right]$

How good is the policy π_θ? Is policy π_{θ_1} better than policy π_{θ_2}? "offline policy evaluation"

There's no general, reliable way to evaluate offline. 😢 Also true for imitation learning!

Strategies:

- Roll-out policy in real world
  - \+ accurate - can be expensive, risky ~ no longer purely offline (consider using online data!)
- Evaluate in high-fidelity simulator or model
  - \+ might be good enough for comparing policies - developing simulator is hard
- Sometimes can use heuristics
  - \+ easy & cheap - not reliable, general-purpose

Hyperparameter tuning for offline RL

How good is the policy π_θ? **Is policy π_{θ_1} better than policy π_{θ_2}?** "offline policy evaluation"

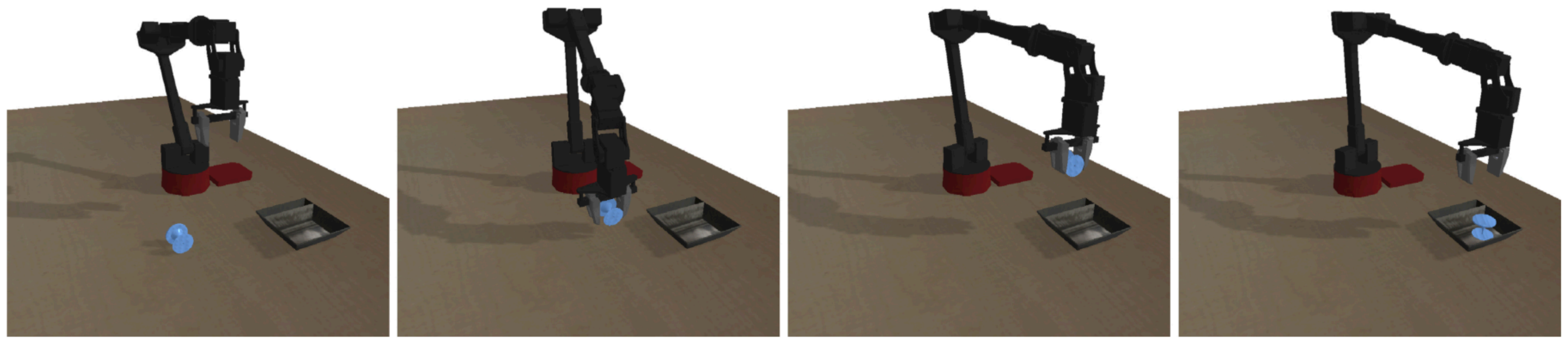
Strategies:

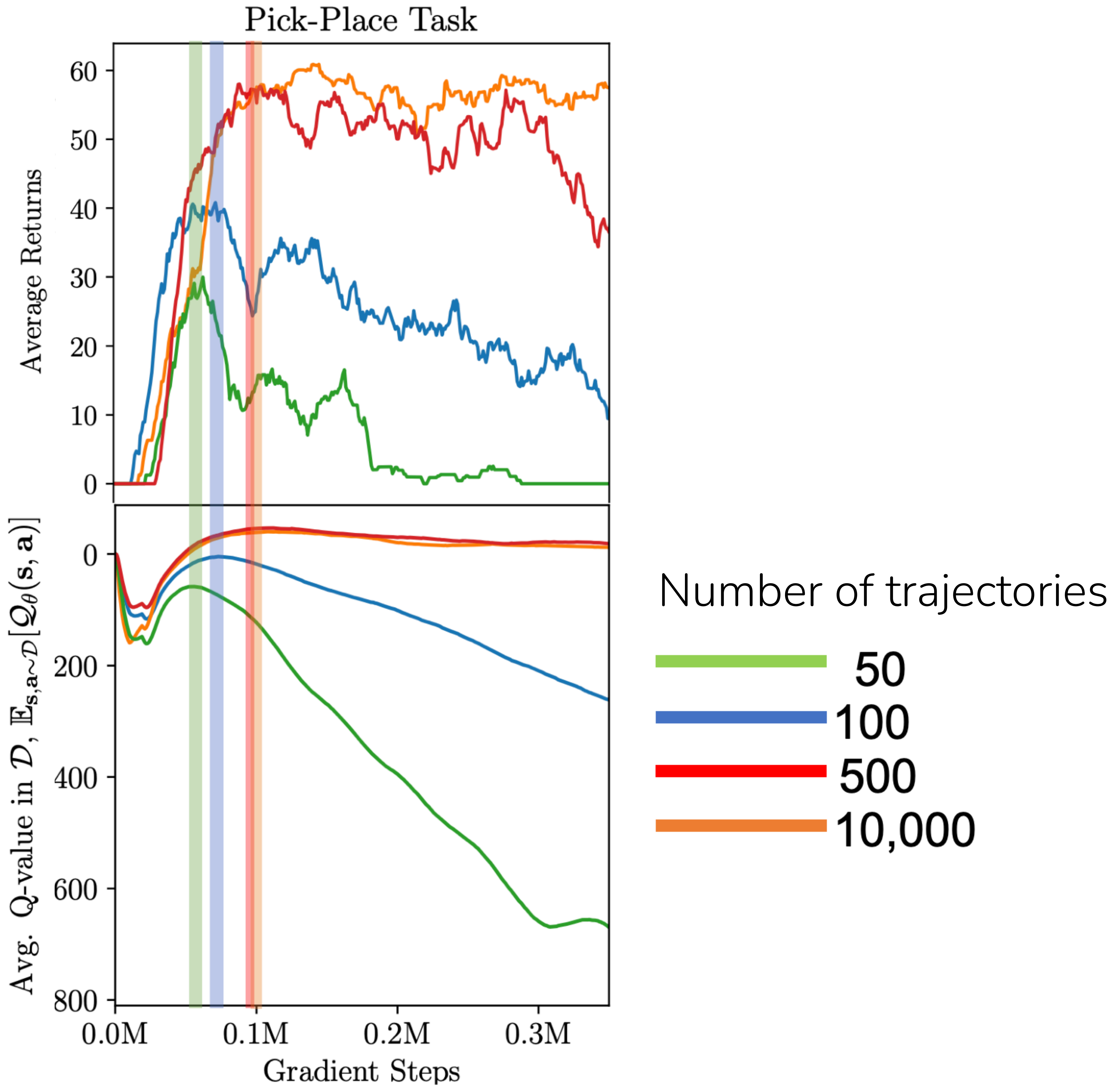
- Sometimes can use heuristics

    + easy & cheap - not reliable, general-purpose

Example heuristic for early stopping with CQL:

Look at peak average Q-value before decline





Kumar*, Singh*, Tian, Finn, Levine. A Workflow for Offline Model-Free Robotic Reinforcement Learning. CoRL '21

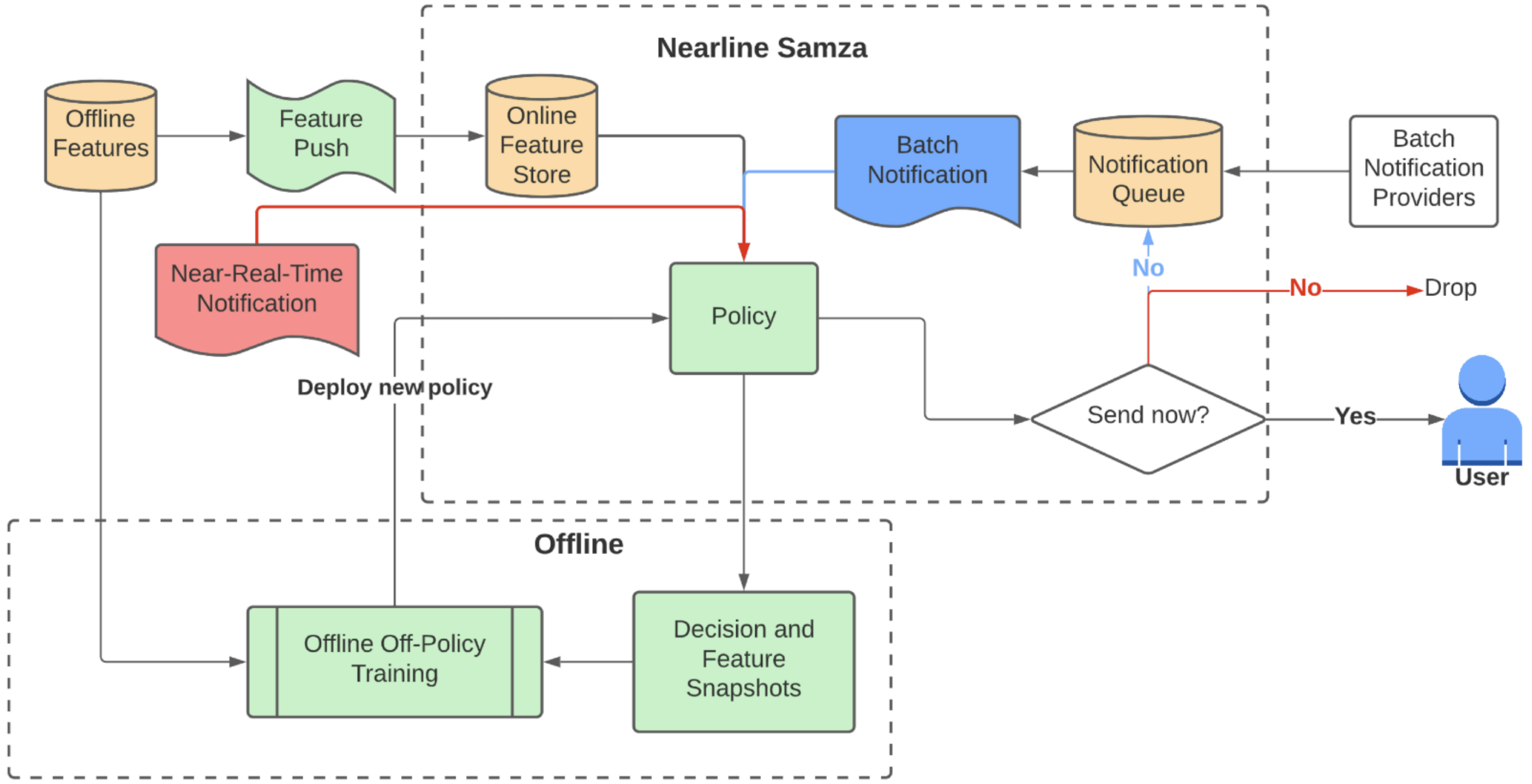
The plan for today

Offline RL: Part 2

1. Recap
2. Revisiting imitation learning for offline RL
   a. Weighted imitation learning
   b. Conditional imitation
3. Offline evaluation & hyperparameter tuning
4. **Applications**

Some example applications

Optimizing policy for sending notifications to users on LinkedIn



WAU: weekly active users
Volume: total # of notifications

CTR: click-through-rate of notifications

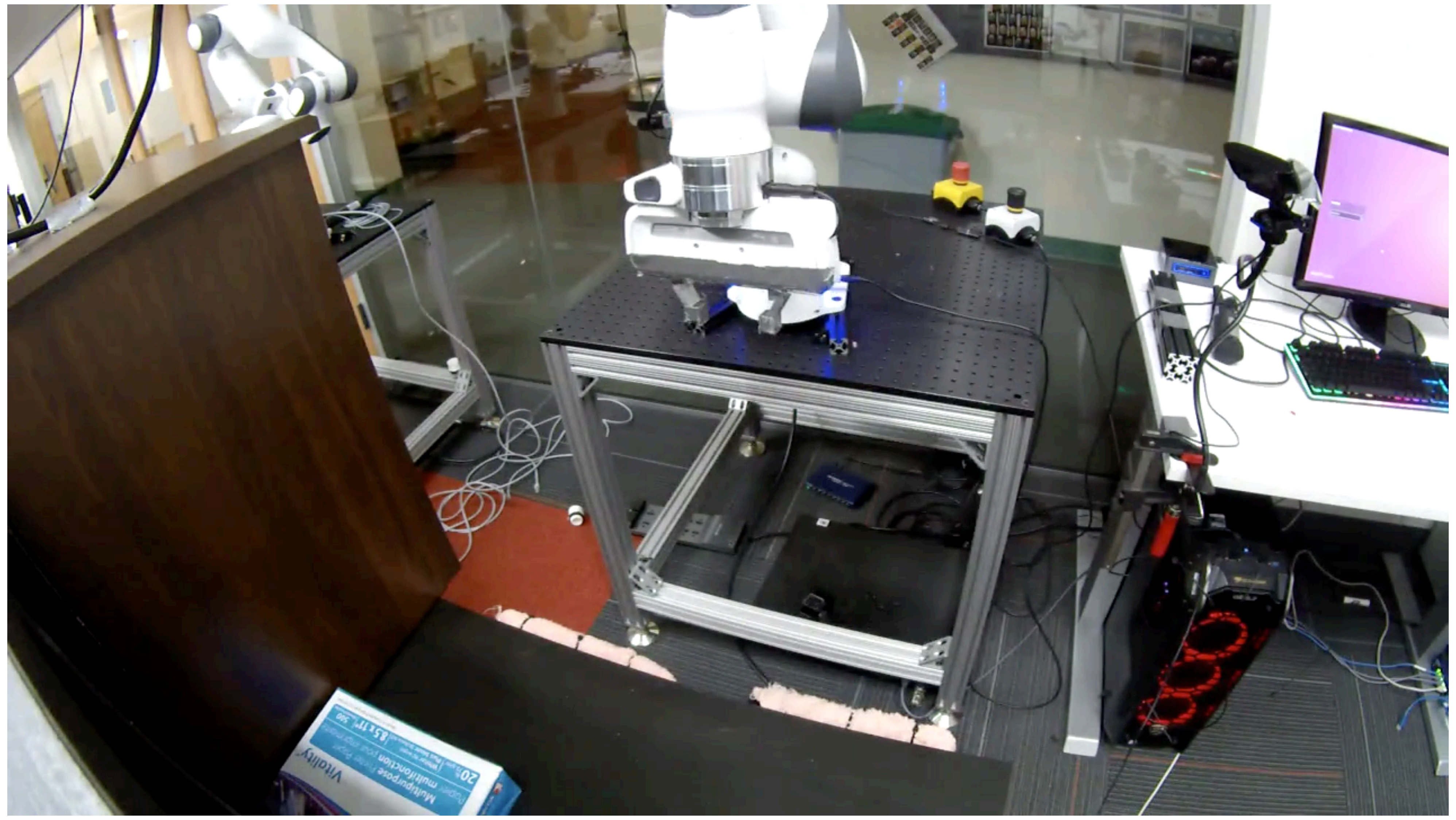
Metric	DDQN vs. Baseline	DDQN + CQL vs. Baseline
Sessions	not stat sig	+ 0.24%
WAU	-0.69%	+ 0.18%
Volume	+7.72%	-1.73%
CTR	-7.79%	+2.26%

Table 1: Online A/B test results for DDQN with and without CQL

Prabhakar, Yuan, Yang, Sun, Muralidharan. Multi-Objective Optimization of Notifications Using Offline RL. '22

Some example applications

Annie Chen, Alex Nam, Suraj Nair develop algorithm for scalably collecting robot data.



Chen*, Nam*, Nair*, Finn. Batch Exploration with Examples for Scalable Robotic RL, ICRA/RA-L '21

Rafael Rafailov reuses **same dataset** to train a policy with new offline RL method

1. Label 200 images as drawer open vs. closed.
2. Train classifier
   (for a reward signal)
3. Run offline RL with LOMPO.
   (precursor to COMBO)

ground truth video

predicted video

Rafailov*, Yu*, Rajeswaran, Finn. Offline RL from Images with Latent Space Models, L4DC '21

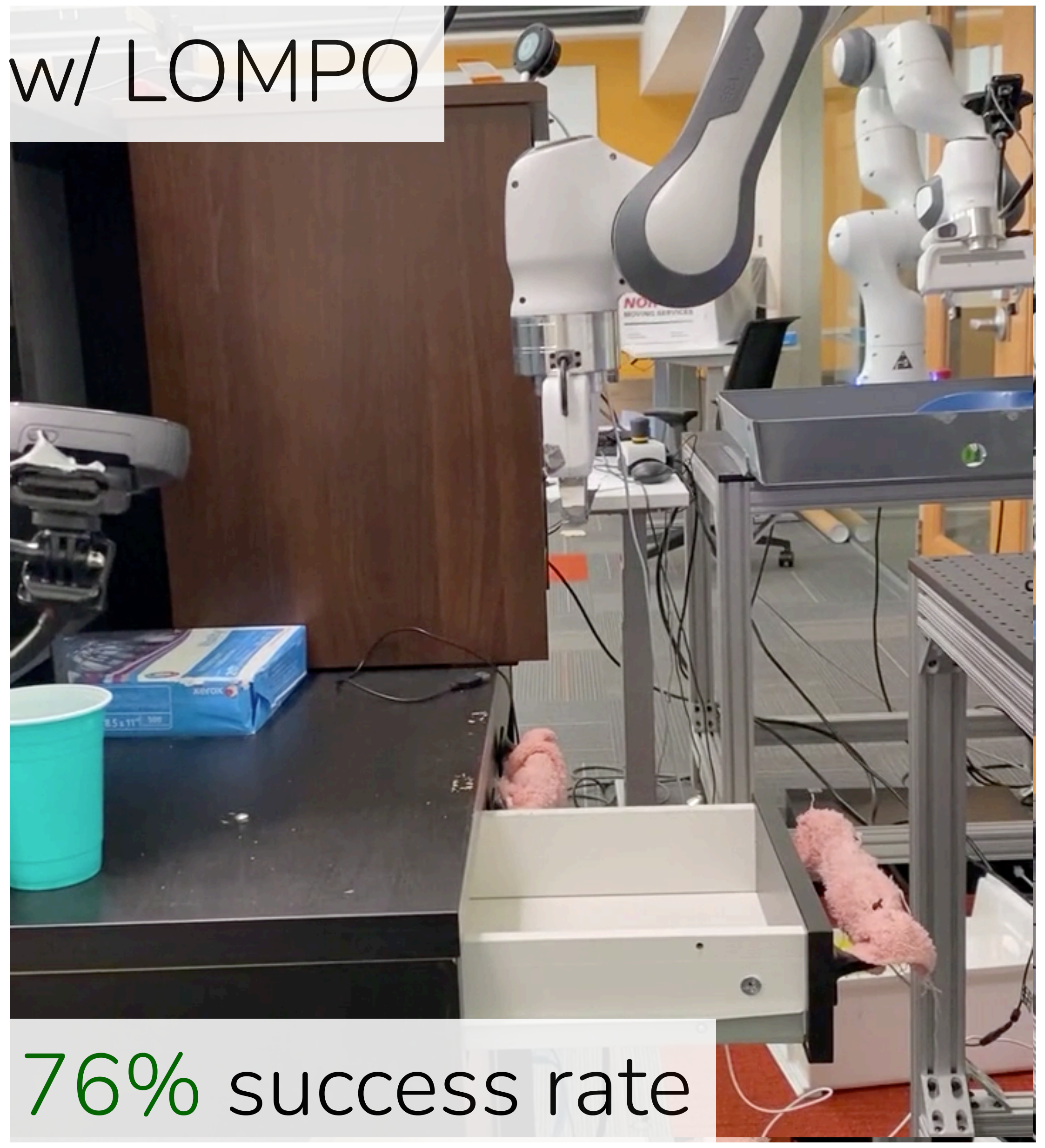
Some example applications

Annie Chen, Alex Nam, Suraj Nair develop algorithm for scalably collecting robot data.

Chen*, Nam*, Nair*, Finn. Batch Exploration with Examples for Scalable Robotic RL, ICRA/RA-L '21

Rafael Rafailov reuses **same dataset** to train a policy with new offline RL method

1. Label 200 images as drawer open vs. closed.
2. Train classifier (for a reward signal)
3. Run offline RL with LOMPO. (precursor to COMBO)



Rafailov*, Yu*, Rajeswaran, Finn. Offline RL from Images with Latent Space Models, L4DC '21

Which offline RL algorithm to use?

If you only want to train offline:

- **Filtered behavior cloning**: Good first approach to using offline data.
- **Implicit Q-learning**: Can stitch data & explicitly constrained to data support
- **Conservative Q-learning**: Just one hyperparameter

If you want offline pre-training + online fine-tuning:

- **Implicit Q-learning**: Seems most performant.

If you have a good way to train a dynamics model:

- **COMBO**: Similar to CQL, but benefits from learned model

Note: Still an active area of research!

The plan for today

Offline RL: Part 2

1. Recap
2. Revisiting imitation learning for offline RL
   a. Weighted imitation learning
   b. Conditional imitation

   } Part of homework 3!
3. Offline evaluation & hyperparameter tuning
4. Applications

Key learning goals:

- two approaches for offline RL (+ when they work & don't work!)
- important considerations for tuning offline RL methods

Course reminders

- Homework 2 due **tonight.**
- Homework 3 out today.
- Project proposal feedback coming out soon.

Announcements

- Moving two office hours (Dilip, Ansh) from in-person to hybrid