No Data Retention Offline-to-Online RL

DongHu Kim

Roadmap

Offline RL (CQL)

Offline-to-Online RL (CalQL)

Offline data + Online RL (RLPD)

No Data Retention Off-to-On RL (WSRL)

tl;dr

WSRL = RLPD – Offline Data + Pretrained Policy Rollout

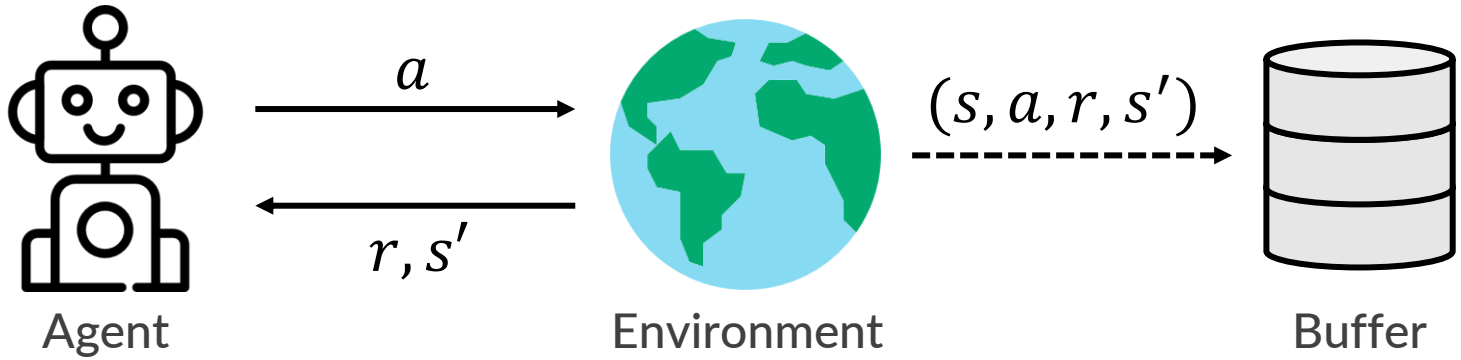
Method	Buffer Initialization	Technique
SimBa	Random action rollout	LayerNorm + Residual + RSNorm
RLPD	Offline data*	RR4 + LayerNorm + REDQ
WSRL	Pre-trained policy rollout	RR4 + LayerNorm + REDQ

*REDQ = CDQ + 10 heads

*Actual usage = 50% Offline + 50% Online

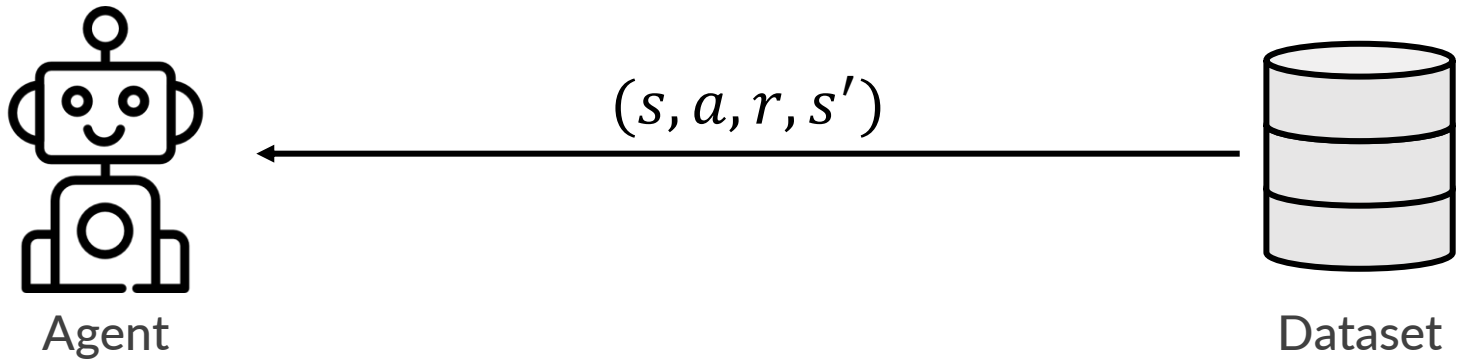
Online RL

- Agent learns from transitions, which are made live by interacting with the environment.



Offline RL

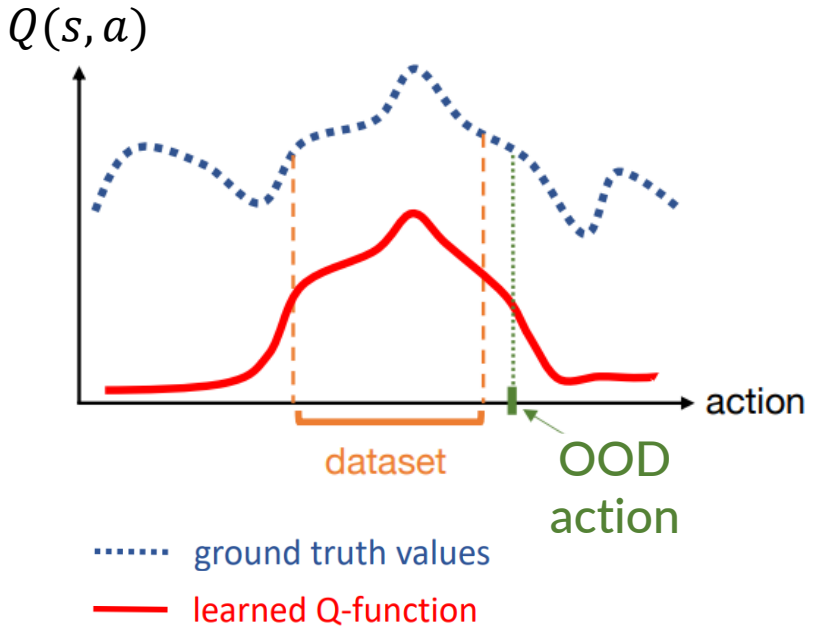
- Agent learns from transitions, which are already made in a form of fixed dataset.
  - via expert policies (from online RL),
  - via teleoperation,
  - etc.



Offline RL

- Problem: Overestimation bias
  - Can be easily fixed in online RL (e.g., Double-Q, CDQ).
  - Impossible in offline RL because agent cannot interact with the environment.
- Solution: Conservative RL objective **(CQL)**
  - Minimize the value of actions that are outside the dataset.

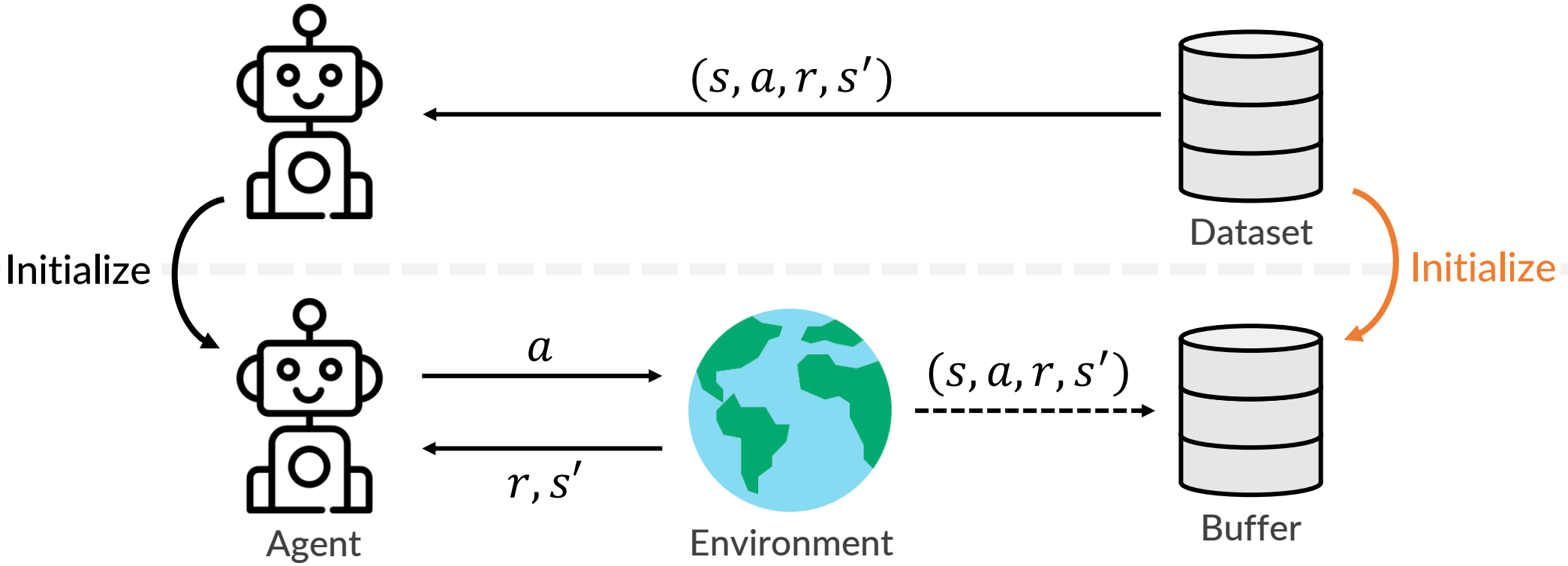
$$\min_\theta \alpha \underbrace{\left(\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[Q_\theta(s,a)\right] - \mathbb{E}_{s,a\sim\mathcal{D}}\left[Q_\theta(s,a)\right]\right)}_{\text{Conservatism objective}} + \underbrace{\frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - \mathcal{B}^\pi\bar{Q}(s,a)\right)^2\right]}_{\text{Bellman objective}}$$



[1] Conservative Q-Learning for Offline Reinforcement Learning., Kumar et al.

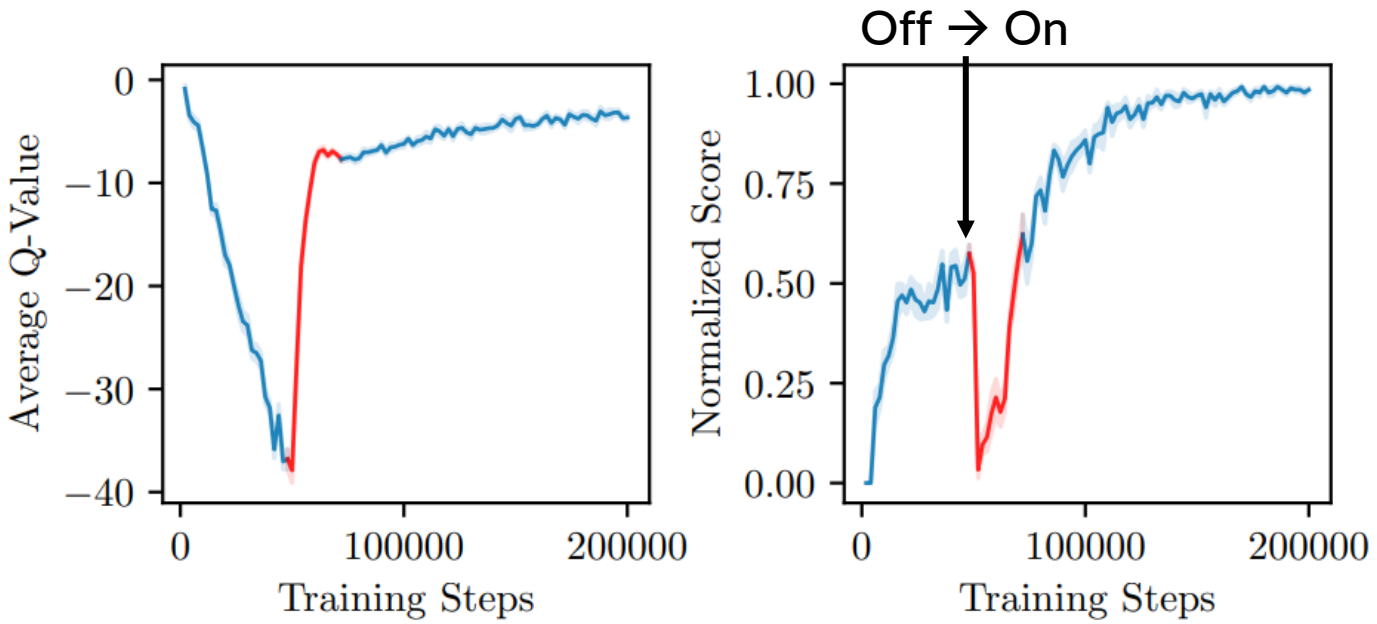
Offline-to-Online RL

- Offline RL = Training on (somewhat) large dataset = Pre-training
  - Then, Fine-tuning = Domain-specific training = Online RL
  - Online RL fine-tuning starts by filling the replay buffer with offline data.



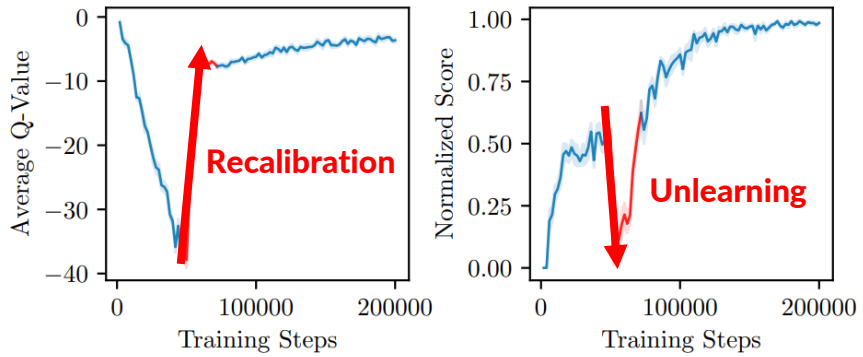
Offline-to-Online RL

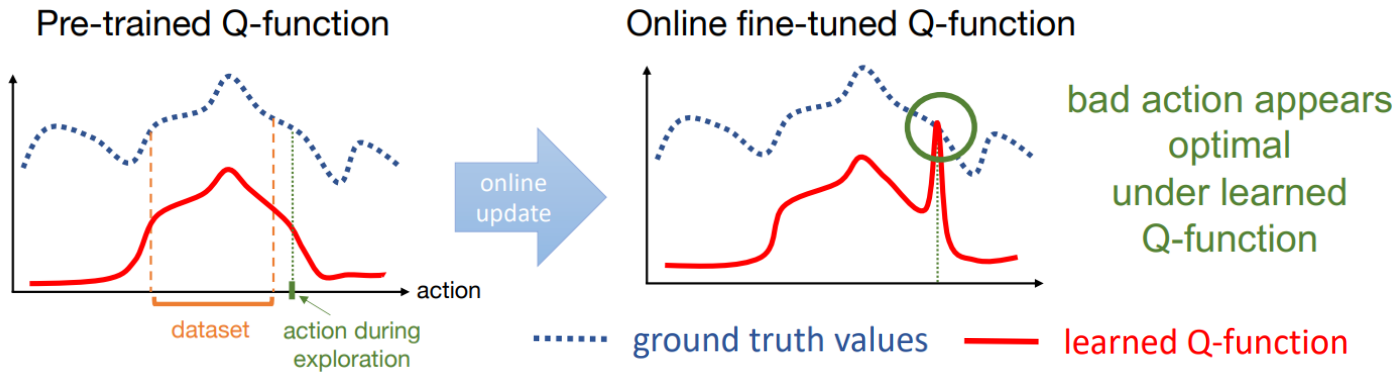
- Pathology: Initial unlearning (performance drop) at the start of online RL



Offline-to-Online RL

- Problem: Over-conservatism of Offline RL
  - The learned values are too small and must be ‘recalibrated’ to true Q values during Online RL.
  - In the process, bad actions may appear optimal.



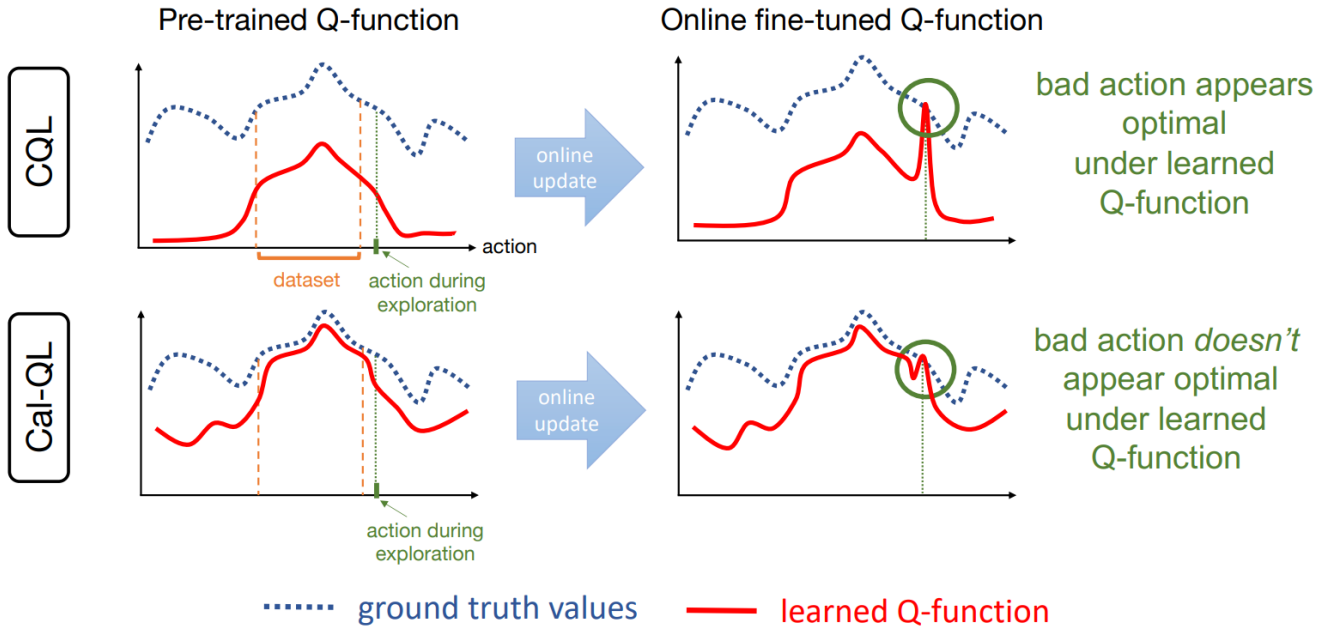


Offline-to-Online RL

- Solution: Less conservatism (CalQL)
  - Learn minimal Q-values, but not below a reasonable bound (e.g., V-value).

$$\min_\theta \alpha \left(\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[\max\left(Q_\theta(s,a), V^\mu(s)\right)\right] - \mathbb{E}_{s,a\sim\mathcal{D}}\left[Q_\theta(s,a)\right] \right) + \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - \mathcal{B}^\pi\bar{Q}(s,a)\right)^2\right]$$

Calibrated conservatism objective — Bellman objective



[1] Cal-QL: Calibrated Offline RL Pre-Training for Efficient Online Fine-Tuning., Nakamoto et al.

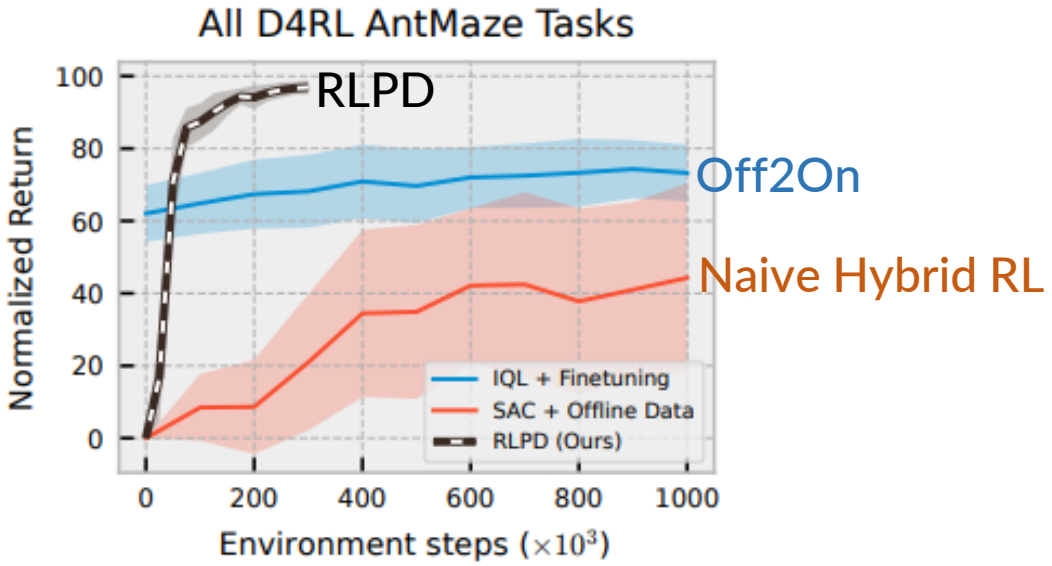
Offline data + Online RL (Hybrid RL)

- Only online RL, but sample from both replay buffer and offline dataset.
  - e.g., Training batch = 50% from online buffer + 50% from offline dataset (RLPD)

Offline data + Online RL (Hybrid RL)

- Offline dataset provides high quality samples in the early-stage.
  - With some techniques, we can leverage this prior to achieve good performance fast.

1. **Replay ratio 4**, for sample efficiency (i.e., to learn fast).
2. **LayerNorm**, to mitigate overestimation.
3. **REDQ**, to mitigate overestimation.



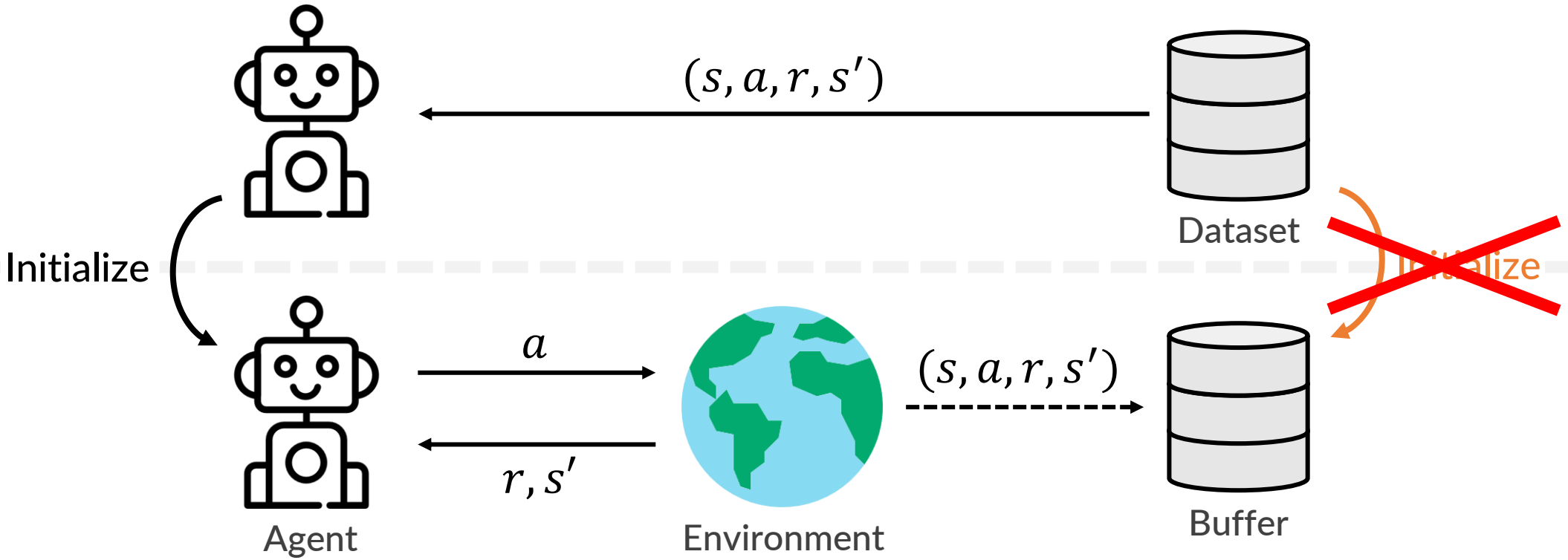
[1] Efficient Online Reinforcement Learning with Offline Data., Ball et al.

Summary (so far)

- Offline RL (CQL)
  - Overestimation bias → Conservative objective function
- Off-to-On RL (CalQL)
  - Over-conservative values → Calibrated objective function
- Hybrid RL (RLPD)
  - RR4 + LayerNorm + REDQ

No Data Retention Off-to-On RL

- OfflineRL = Pre-training, OnlineRL = Fine-tuning
  - What kind of fine-tuning keeps the pre-training dataset?!



No Data Retention Off-to-On RL

- Problem: Off-to-on RL does not work without offline data.
  - If this doesn't work, what's the point of pre-training even?

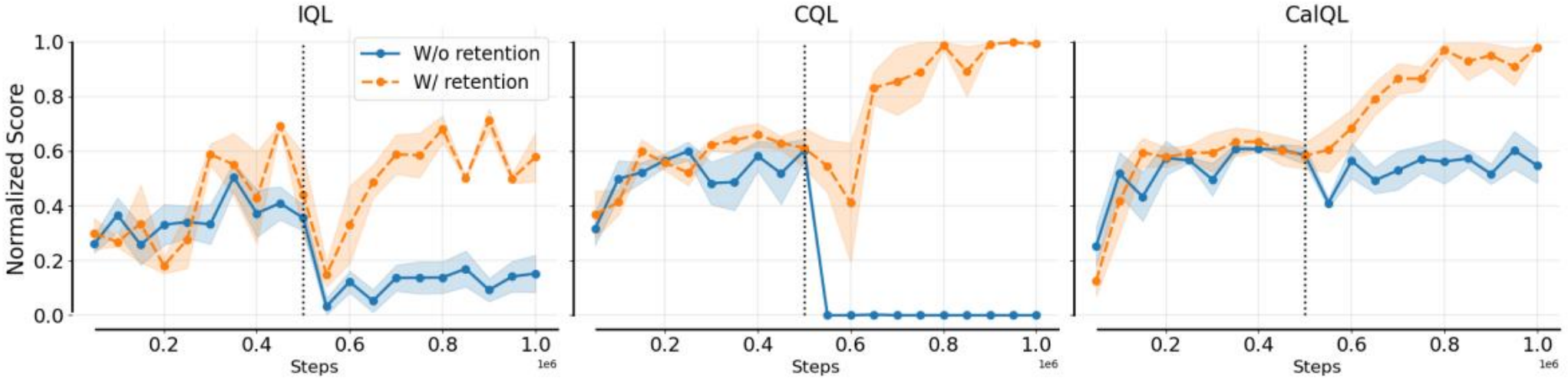


Figure 2: In no-retention fine-tuning, IQL, CQL, and CalQL all fail to fine-tune on `kitchen-partial`. In contrast, when continually training on offline data during fine-tuning, these algorithms work as intended. Vertical dotted line indicates the separation between pre-training and fine-tuning.

No Data Retention Off-to-On RL

- Problem: Off-to-on RL does not work **with** offline data.
  - Again, why pre-train when learning from scratch does way better?

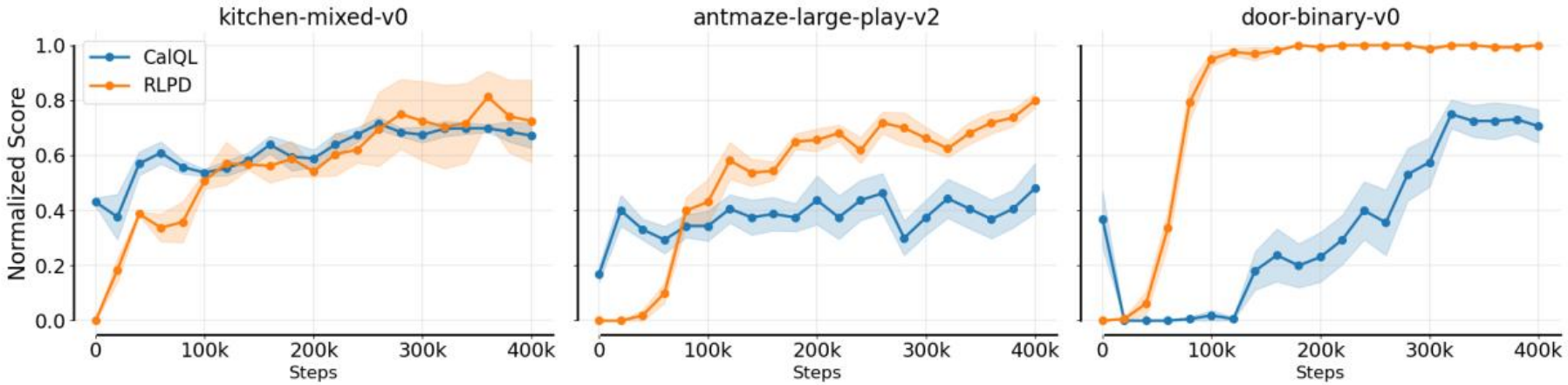
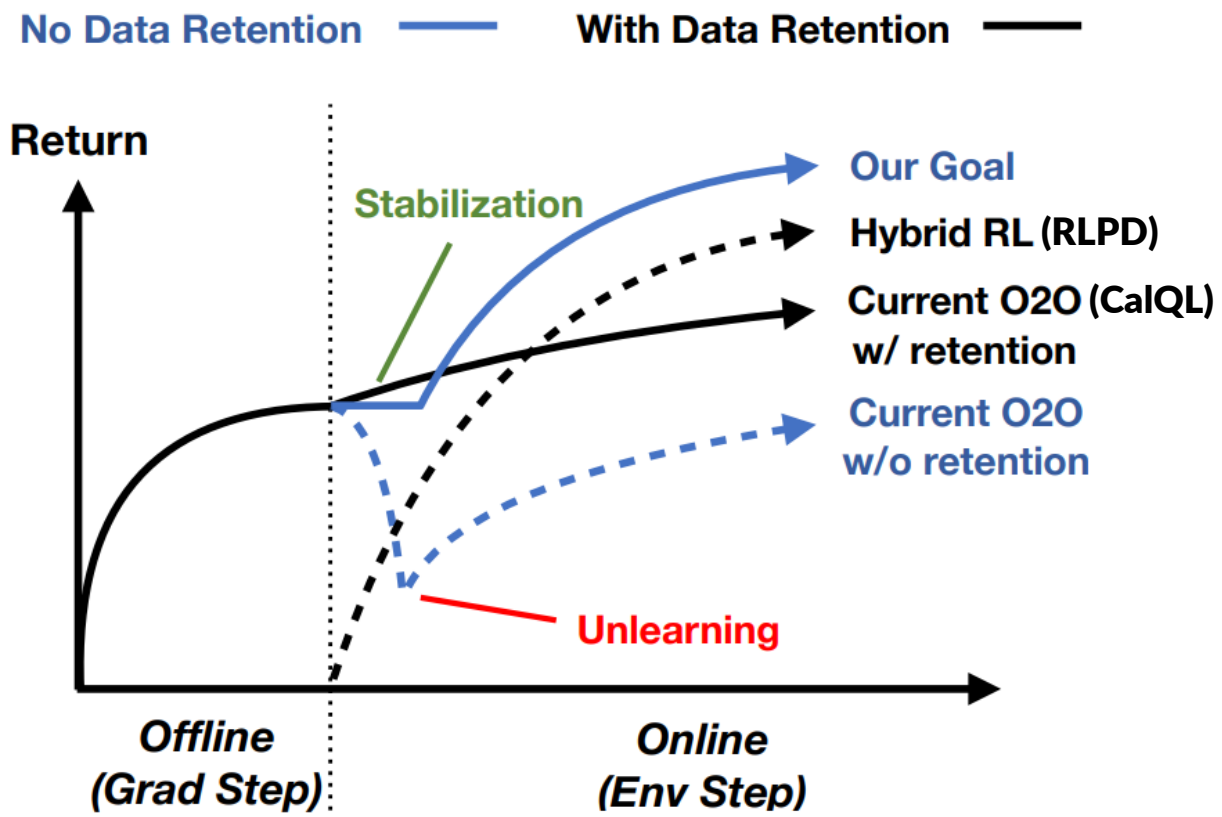


Figure 6: Retaining offline data is not efficient, and is outperformed by online RL methods like RLPD on three different environments. RLPD starts from scratch, and CalQL starts from pre-trained initialization at step 0.

No Data Retention Off-to-On RL

- Goal: Make pretrain-then-finetune great again.



Finding The Culprit

- In CalQL, the over-conservative values were suspected to be the issue of off-to-on RL.
  - However, we saw that CalQL also fails to leverage pre-trained knowledge.

- An underlying problem: **Data distribution shift between offline and online data**
  - This can explain both failures with and without offline data.

Distribution Shift - Without Offline Data

- Main observations
  - TD Loss is stable for online data (d) but diverges for offline data (b,c).
  - (1) Offline and online data distribution shift exists.
  - (2) Catastrophic forgetting on offline data occurs (a devastation in sparse reward envs (a)).

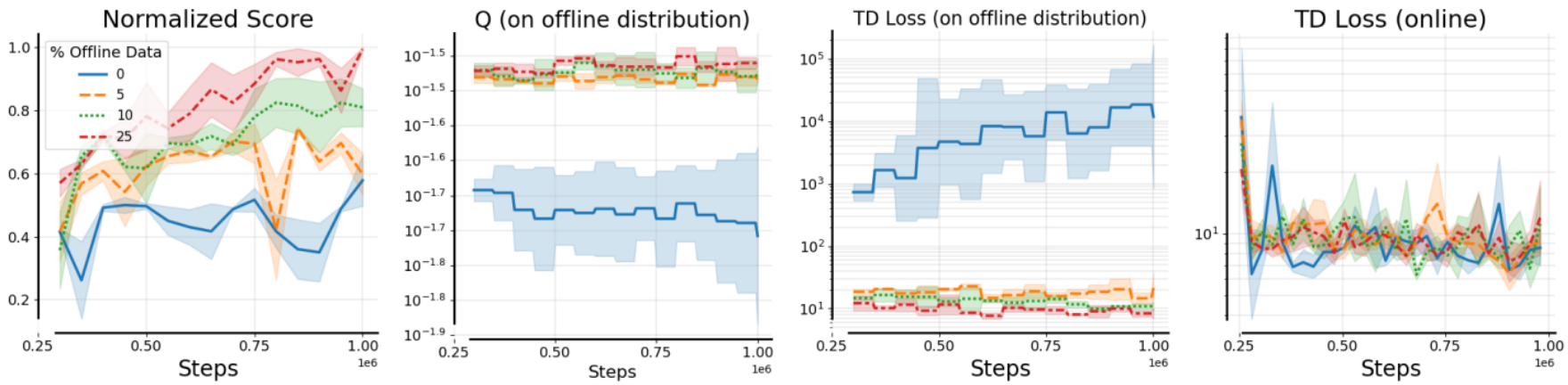
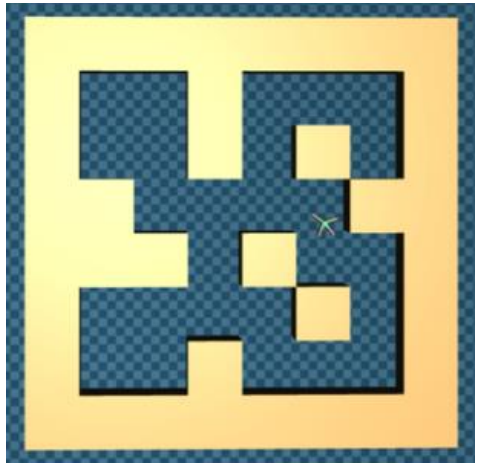


Figure 3: When offline data is removed (to different extents) during fine-tuning, performance drops (subfigure a) because the Q-function fit on offline dataset distribution diverges (subfigure b, c), even though the Q-function can fit the online distribution (subfigure d). This plot shows fine-tuning CalQL on `kitchen-partial` with 0/5/10/25% offline data in each update batch. We have similar findings with IQL and CQL.

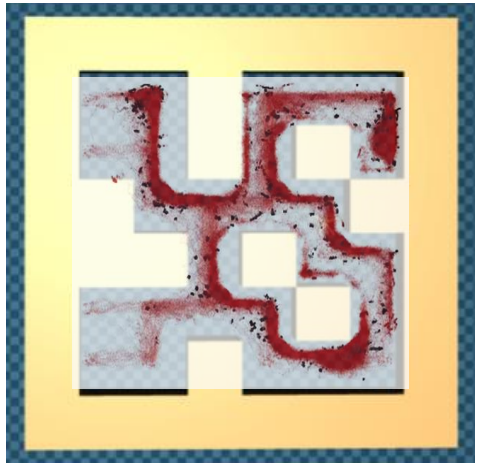
Distribution Shift - With Offline Data

- (Personal interpretation) In off-to-on, we **need** to forget some pre-trained knowledge.
  - Take Antmaze for example, where only a portion of offline data becomes relevant in online RL.
  - Although less severely, these irrelevant data will still hold the agent from learning fast.

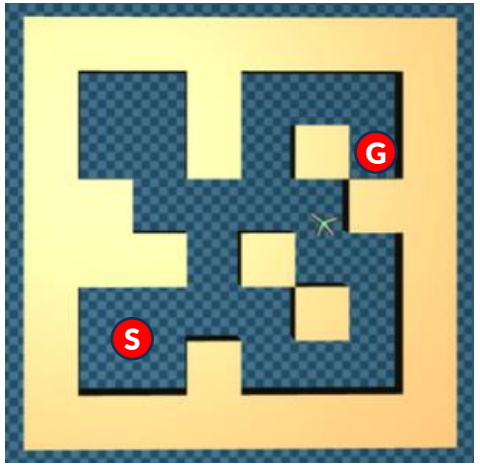
Antmaze-medium

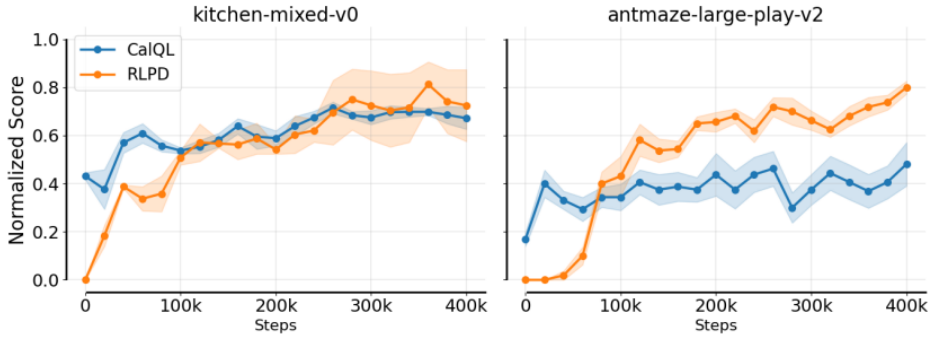


Offline data



Online RL



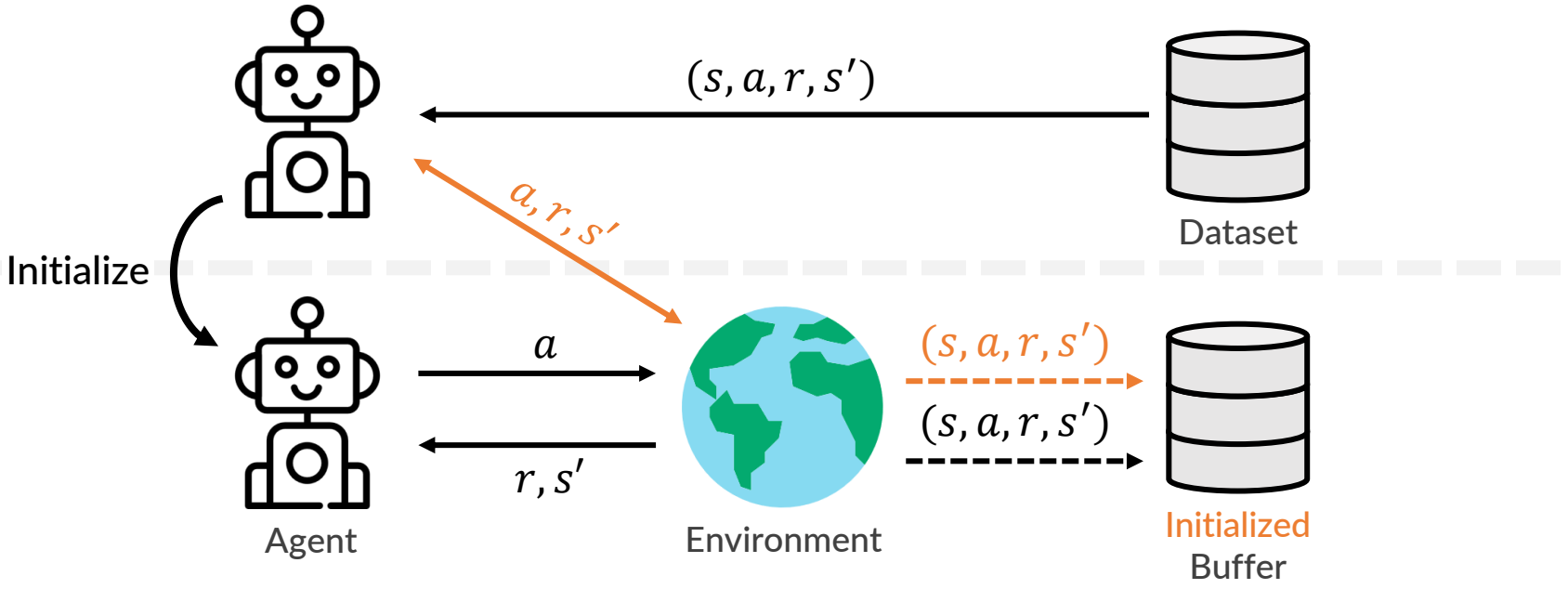


Warm Start RL (WSRL)

- So far:
  - Offline data is required to prevent unlearning and catastrophic forgetting.
  - But offline data has irrelevant samples, which hurt performance.
- Solution: Create relevant samples using the pre-trained policy!
  - Will mitigate catastrophic forgetting (because they are created by the pre-trained policy).
  - Will create relevant samples (because they are created in the fine-tuning environment).

Warm Start RL (WSRL)

- Method
  1. Pre-train with any offline RL algorithm (e.g., CalQL)
  2. Initialize the replay buffer by rollouts of pre-trained (frozen) policy.
  3. Fine-tune the pre-trained policy with any online RL algorithm (e.g., RLPD)



Result

- No data retention finally works

(RLPD w/o offline data)

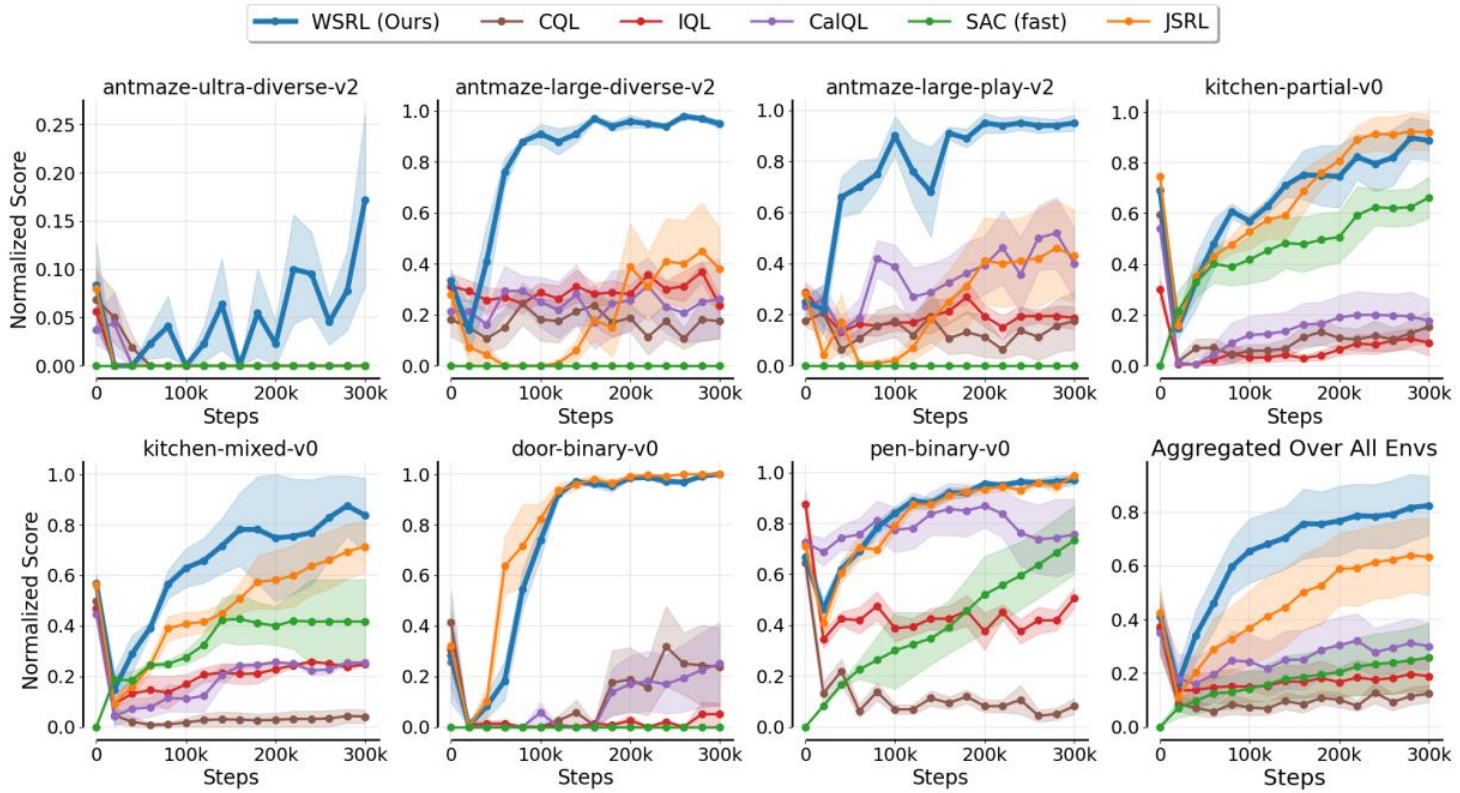


Figure 7: In **no-retention fine-tuning**, WSRL fine-tunes efficiently and greatly outperforms all previous algorithms, which often fail to recover from an initial dip in performance. JSRL, the closest baseline, uses a data-collection technique similar to warmup.

Result

- Even outperforms data retention algorithms!

(CalQL + RLPD)

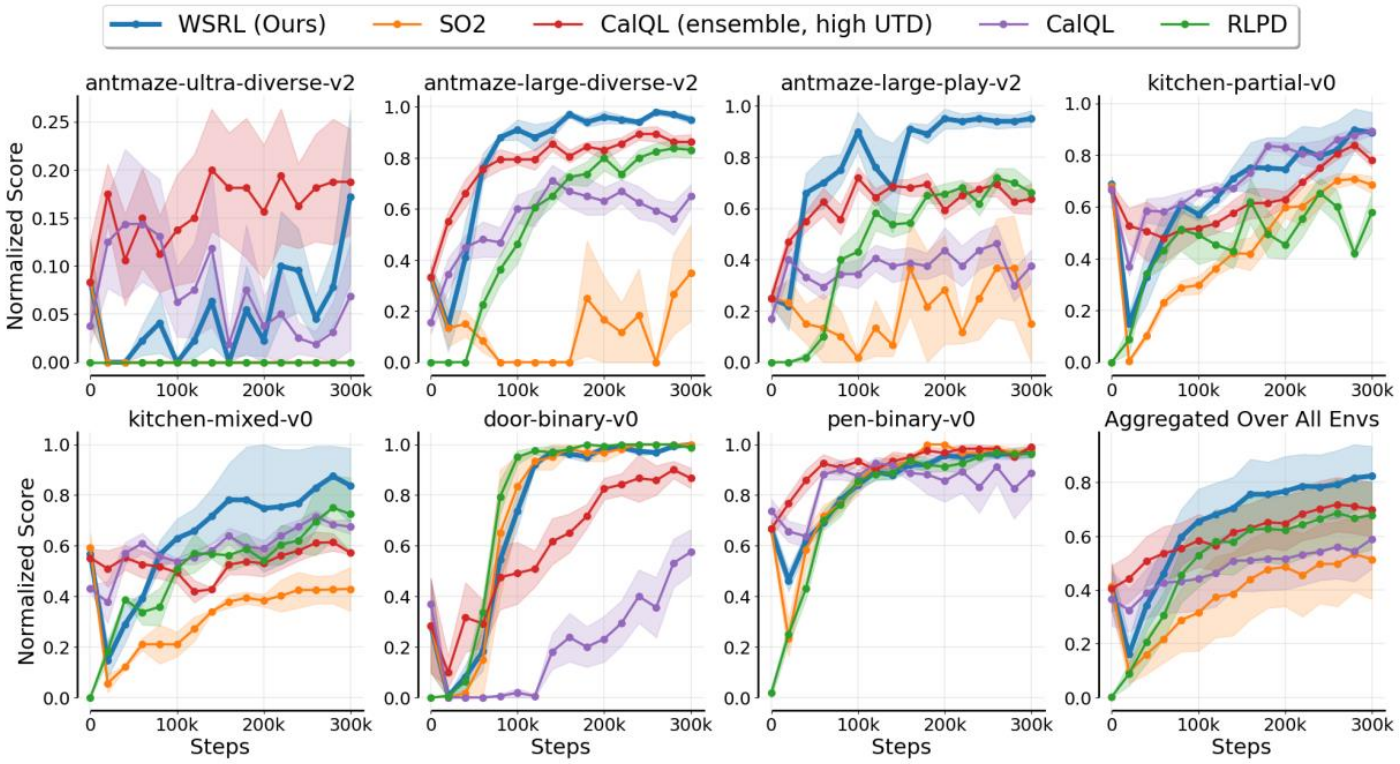
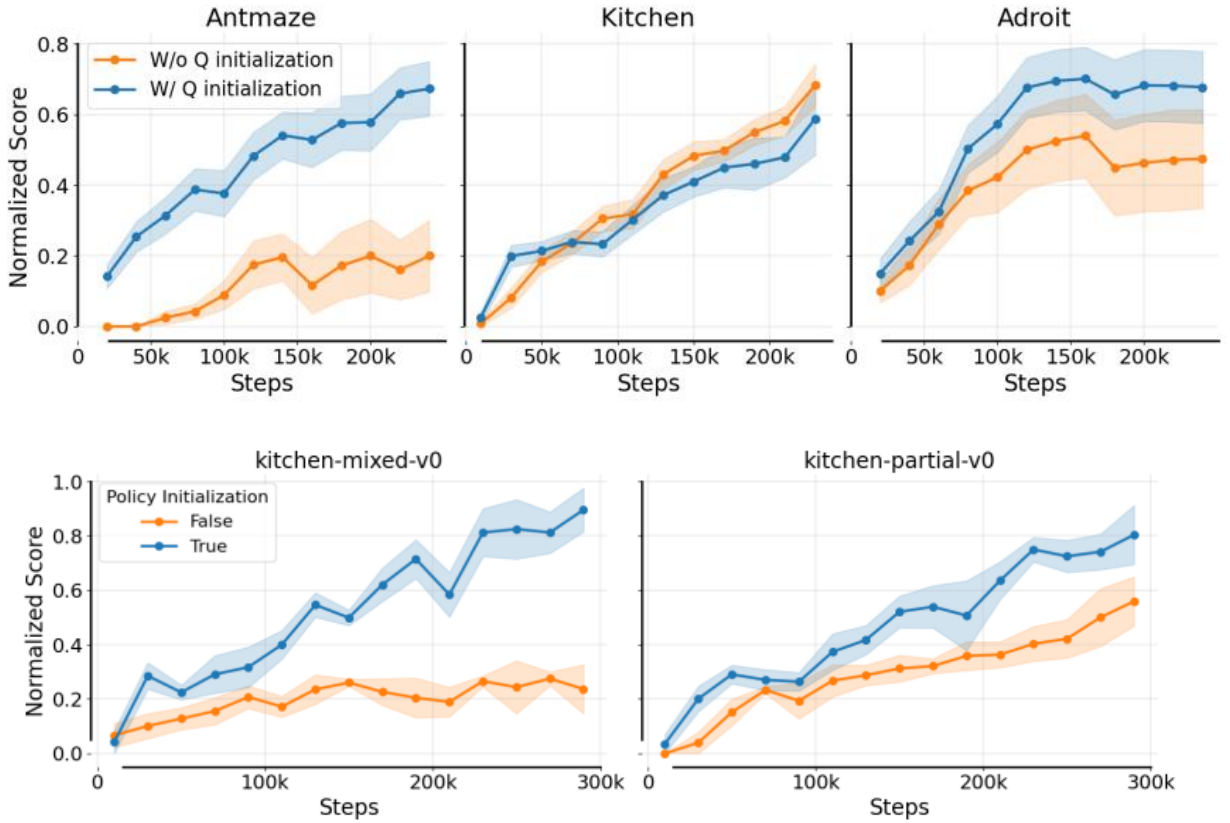


Figure 9: Compared to methods that ***do* retain offline data** online, WSRL, perhaps surprisingly, is still able to fine-tune faster or competitively despite not retaining any offline data. This in particular highlights benefits of warmup.

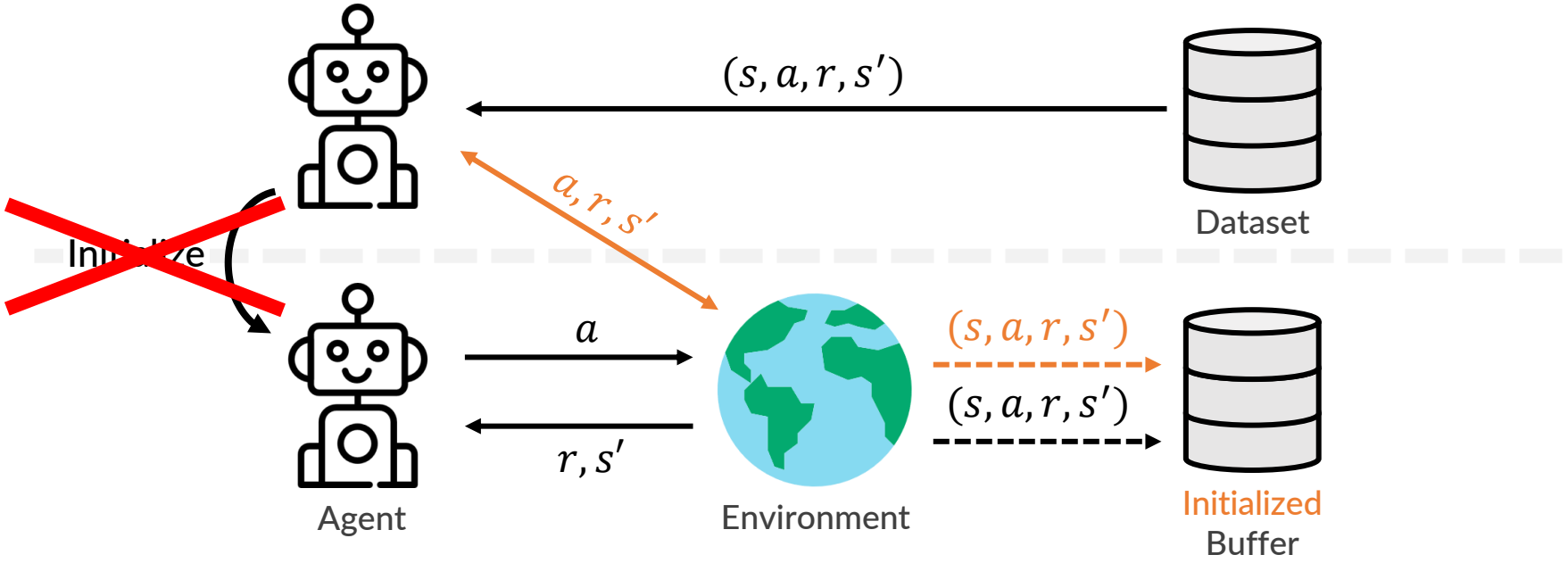
Comparison With JSRL

- Ablation on whether to load the pre-trained parameters on actor, critic, or both.
  - Unsurprisingly, both are crucial.



Comparison With JSRL

- Jump Start RL (JSRL)
  - Quite similar to WSRL, except that the agent is learned from scratch...
  - So... what if we initialize the agent with pre-trained model?



Comparison With JSRL

- …well that's a weird comparison.

they can definitely be applied or repurposed to our setting. **JSRL** [51] uses a pre-trained policy to roll in for some number of steps during each episode and then rolls out with the current policy. The online policy is trained from scratch with both the roll-in and rollout experience. To improve JSRL's competitiveness, we also initialize the online policy with the pre-trained policy and run it with high UTD. Offline RL methods

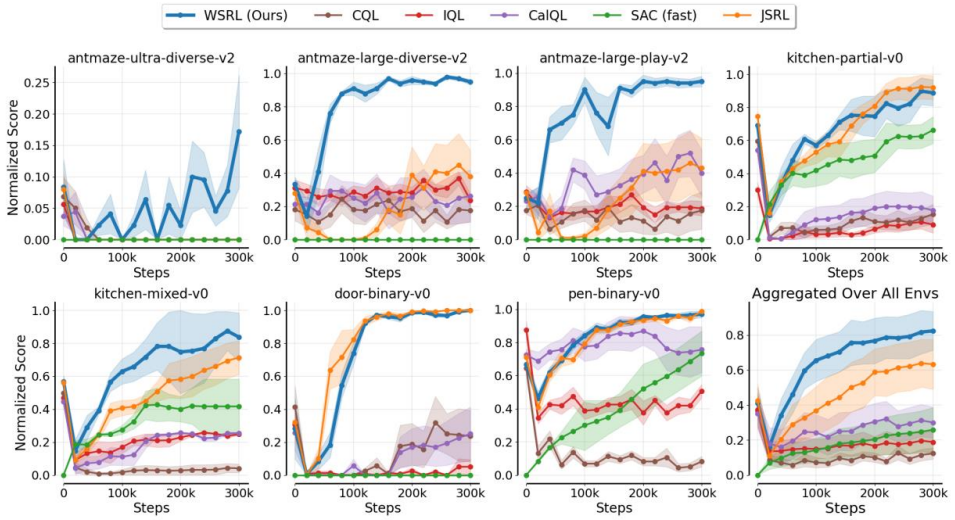


Figure 7: In **no-retention fine-tuning**, WSRL fine-tunes efficiently and greatly outperforms all previous algorithms, which often fail to recover from an initial dip in performance. JSRL, the closest baseline, uses a data-collection technique similar to warmup.

Summary

1. Offline data should NOT be retained during online RL, as it loses the meaning of pre-training.
2. One major issue of off-to-on RL is the distribution shift between offline and online data.
3. WSRL: Get middle-ground samples (between off and on) by rolling out pre-trained policy.