When to Trust Your Simulator: Dynamics-Aware Hybrid Offline-and-Online Reinforcement Learning

Reinforcement Learning - Paper presentation

Enrico Loda, Simone Dario

Department of Computer Science
Master Degree in Artificial Intelligence
University of Verona

August 28, 2025

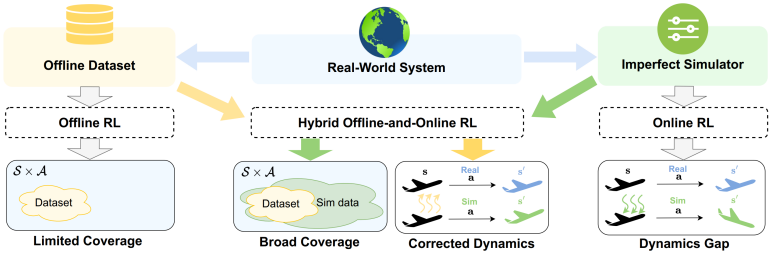
Overview

1. **Introduction**
2. **Background**
   - 2.1 Markov Decision Process
   - 2.2 Actor Critic
   - 2.3 Offline RL via Value Regularization
   - 2.4 RL Using Simulators with Dynamics Gap
3. **Hybrid Offline-and-Online RL**
   - 3.1 Incorporating Offline Data in Online Learning
   - 3.2 Adaptive Value Regularization on High Dynamics-Gap Samples
   - 3.4 Fixing Bellman Error due to Dynamics Gap
   - 3.5 Practical Implementation & Pseudocode
   - 3.6 Interpretation of Dynamics-Aware Policy Evaluation
4. **Experiments & Results**
   - 4.1 Sim-based Experiments
   - 4.2 Real-based Experiments
   - 4.3 Ablation Study
5. **Conclusion**
6. **Code**

Introduction

In recent years, RL has struggled to achieve practical applicability in real-world scenarios. Online RL methods often rely on simulators with simplified dynamics, which creates severe sim-to-real gaps when deploying policies to real systems. On the other hand, offline RL approaches require extremely large datasets.

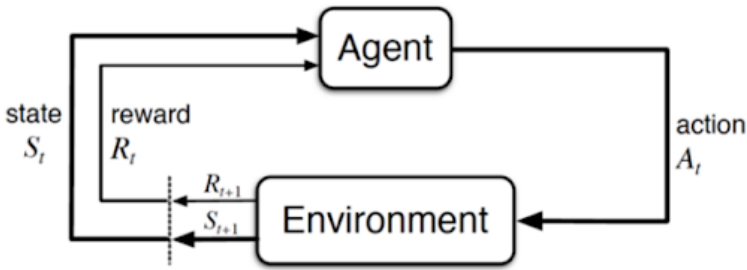
To address these limitations, the paper introduces **H2O: Dynamics-Aware Hybrid Offline-and-Online Reinforcement Learning**, that combines offline datasets collected from real environments with online rollouts from simulators, and introduces a novel dynamics-aware policy evaluation.



Section 2

Background

Markov Decision Process



$$\mathcal{M} := (\mathcal{S}, \mathcal{A}, r, P_M, \rho, \gamma)$$

Actor Critic

Actor-critic framework: *policy evaluation* and *policy improvement*.
Policy evaluation:

$$\hat{Q} \leftarrow \arg\min_{Q} \mathbb{E}_{s,a,s' \sim \mathcal{U}} \Big[\big(Q(s,a) - \hat{\mathcal{B}}^{\pi} \hat{Q}(s,a) \big)^2 \Big], \tag{1}$$

where the Bellman operator is given by

$$\hat{\mathcal{B}}^{\pi} \hat{Q}(s,a) = r(s,a) + \gamma \, \mathbb{E}_{a' \sim \hat{\pi}(\cdot|s')} \Big[\hat{Q}(s',a') \Big].$$

Policy improvement:

$$\hat{\pi} \leftarrow \arg\max_{\pi} \mathbb{E}_{s,a \sim \mathcal{U}} \big[\hat{Q}(s,a) \big], \tag{2}$$

where $\mathcal{U}$ denotes either the replay buffer $\mathcal{B}$ generated by previous policy interactions (online RL) or a fixed dataset $\mathcal{D}$ (offline RL).
The paper uses a SAC (Soft Actor Critic) framework, which augments the objective with an entropy regularization term that encourages exploration and stochasticity in the policy.

Offline Reinforcement Learning via Value Regularization

In the offline setting, performing the standard Bellman update can result in overestimation over Q-values due to the distributional shift.
Solution: Conservative Q-Learning, regularize the value function on *Out of Distribution (OOD)* actions.

$$\min_{Q} \max_{\mu} \alpha \left[\mathbb{E}_{s \sim D,\, a \sim \mu(\cdot|s)} \left[Q(s,a) \right] - \mathbb{E}_{(s,a) \sim D} \left[Q(s,a) \right] + \mathcal{E} \left[Q,\ \hat{\mathcal{B}}^{\pi} \hat{Q} \right] \right].$$

The algorithm learns a policy $\hat{\pi}$ upon a lower-bounded Q-function, which additionally pushes down the Q-values on actions induced by $\mu(a \mid s)$ (some sampling distribution) and pulls up the Q-values on trustworthy offline data.
This value regularization framework has the potential to handle data from two sources. For this reason it is used in the paper to perform simultaneous offline and online policy learning using a similar value regularization recipe.

Reinforcement Learning Using Simulators with Dynamics Gap

Dealing with dynamics gaps in simulators is challenging:

- *System identification*
- *Domain Randomization* (DR)
- *Dynamics Adaptation* (DARC)

However, none of these approaches provides a structured framework that simultaneously leverages offline data and online learning to address the dynamics gap effectively. For this reason, the paper introduces a novel approach to addressing the dynamics gap.

Section 3

Hybrid Offline-and-Online RL

Incorporating Offline Data in Online Learning

The authors introduce a **dynamics-aware policy evaluation** scheme that penalize the **Q-values** of simulated samples with high **dynamics gaps**.

$$\min_{Q} \max_{d^{\phi}} \beta \left[\mathbb{E}_{\mathbf{s},\mathbf{a} \sim d^{\phi}(\mathbf{s},\mathbf{a})} \left[Q(\mathbf{s},\mathbf{a}) \right] - \mathbb{E}_{\mathbf{s},\mathbf{a} \sim \mathcal{D}} \left[Q(\mathbf{s},\mathbf{a}) \right] + \mathcal{R}(d^{\phi}) \right] + \tilde{\mathcal{E}} \left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q} \right)$$

- β is a positive scaling parameter.
- $d^{\phi}(\mathbf{s},\mathbf{a})$ is a state-action sampling distribution associated to high dynamics-gap samples.
- $\mathcal{R}(d^{\phi})$ is a regularization term for d^{ϕ} to enforce this designed behavior.
- $\tilde{\mathcal{E}}(Q, \hat{\mathcal{B}}^{\pi} \hat{Q})$ denotes the modified Bellman error of the mixed data from offline dataset $\mathcal{D}$ and the simulation rollout in online replay buffer $\mathcal{B}$. The data are generated by the real MDP $\mathcal{M}$ and the simulated MDP $\hat{\mathcal{M}}$, respectively.

Adaptive Value Regularization on High Dynamics-Gap Samples 1/3

Sim-to-real dynamics gaps can be **heterogeneous**, uniform value regularization may not be appropriate.
To achieve adaptive value regularization, $\mathcal{R}(d^{\phi})$ represents the Kullback-Leibler (KL) divergence between $d^{\phi}(s,a)$ and $\omega(\mathbf{s},\mathbf{a})$, which is the distribution that characterizes the dynamics gaps.
The inner maximization over $d^{\phi}(s,a)$ in the dynamics-aware policy evaluation now correspond to the following optimization problem:

$$\max_{d^{\phi}} \mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^{\phi}(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] - D_{KL}(d^{\phi}(\mathbf{s},\mathbf{a})\|\omega(\mathbf{s},\mathbf{a})) \quad \text{s.t.} \quad \sum_{\mathbf{s},\mathbf{a}} d^{\phi}(\mathbf{s},\mathbf{a}) = 1, \quad d^{\phi}(\mathbf{s},\mathbf{a}) \geq 0$$

Adaptive Value Regularization on High Dynamics-Gap Samples 2/3

Above optimization problem admits a closed-form solution:

$$d^{\phi}(\mathbf{s}, \mathbf{a}) \propto \omega(\mathbf{s}, \mathbf{a}) \exp(Q(\mathbf{s}, \mathbf{a}))$$

Plugging the result in the formula the following is obtained:

$$\min_{Q} \beta \left(\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp(Q(\mathbf{s}, \mathbf{a})) - \mathbb{E}_{\mathbf{s},\mathbf{a} \sim \mathcal{D}}[Q(\mathbf{s}, \mathbf{a})] \right) + \tilde{\mathcal{E}} \left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q} \right)$$

The first term now corresponds to a weighted soft-maximum of Q-values at any state-action pair. The question now becomes how to evaluate $\omega(s, a)$.

Adaptive Value Regularization on High Dynamics-Gap Samples 3/3

We can evaluate the dynamics gap as

$$u(s,a) := D_{KL}\big(P_{\hat{\mathcal{M}}}(s'|s,a) \,\big\|\, P_{\mathcal{M}}(s'|s,a)\big) = \mathbb{E}_{s'\sim P_{\hat{\mathcal{M}}}}\left[\log \frac{P_{\hat{\mathcal{M}}}(s'|s,a)}{P_{\mathcal{M}}(s'|s,a)}\right]$$

$\omega(s,a)$ can be represented as a normalized distribution of u.
According to Bayes' rule, the dynamic ratio can be seen as:

$$\frac{P_{\hat{\mathcal{M}}}(s'|s,a)}{P_{\mathcal{M}}(s'|s,a)} = \frac{p(s'|s,a,\text{sim})}{p(s'|s,a,\text{real})} = \frac{1-p(\text{real}|s,a,s')}{p(\text{real}|s,a,s')} \Big/ \frac{1-p(\text{real}|s,a)}{p(\text{real}|s,a)}$$

It is possible to approximate $p(\text{real}|s,a), p(\text{real}|s,a,s')$ with two discriminator $D_{\phi_{sas}}(\cdot|s,a,s')$ and $D_{\phi_{sa}}(\cdot|s,a)$, optimized with standard cross-entropy loss.

Fixing Bellman Error due to Dynamics Gap

Due to the existence of **dynamics gaps**, the next state s' from simulated dynamics $P_{\hat{\mathcal{M}}}$ might be wrong, resulting in a miscalculation of target Q-values.
Since it is not possible to have s' directly from the real dynamics $P_{\mathcal{M}}$ for any arbitrary state-action pair, the inverted dynamic ratio is used as an **importance sampling weight**:

$$\tilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^{\pi}\hat{Q}\right) = \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(\left(Q - \hat{\mathcal{B}}^{\pi}\hat{Q}\right)(s,a)\right)^2\right] + \\ + \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{B}}\left[\frac{P_{\mathcal{M}}(s' \mid s,a)}{P_{\hat{\mathcal{M}}}(s' \mid s,a)}\left(\left(Q - \hat{\mathcal{B}}^{\pi}\hat{Q}\right)(s,a)\right)^2\right]$$

Practical Implementation & Pseudocode

Algorithm 1 Dynamics-Aware Hybrid Offline-and-Online Reinforcement Learning (H2O)

Data: an offline dataset $\mathcal{D}$ from the real world, an imperfect simulator with biased dynamics $\widehat{\mathcal{M}}$

1: **Initialize:** critic network Q_θ, target network $Q_{\bar{\theta}}$, actor network π_ϕ, replay buffer $B = \emptyset$ for simulated transitions, discriminators $D_{\Phi_{sas}}(\cdot|s, a, s')$ and $D_{\Phi_{sa}}(\cdot|s, a)$

2: **for** step $t = 1, \ldots, T$ **do**

3: $\quad B \leftarrow B \cup \text{ROLLOUT}(\pi_\phi, \widehat{\mathcal{M}})$ ▷ Collect simulated data

4: $\quad \Phi_{sas} \leftarrow \Phi_{sas} - \eta \nabla_{\Phi_{sas}} \text{CROSSENTROPYLOSS}(B, \mathcal{D}, \Phi_{sas})$ ▷ Update $D_{\Phi_{sas}}$

5: $\quad \Phi_{sa} \leftarrow \Phi_{sa} - \eta \nabla_{\Phi_{sa}} \text{CROSSENTROPYLOSS}(B, \mathcal{D}, \Phi_{sa})$ ▷ Update $D_{\Phi_{sa}}$

6: $\quad \theta \leftarrow \theta - \eta_Q \nabla_\theta \left[\beta(\sum_{(s,a)\in batch} \log \omega(s, a) \exp(Q_\theta(s, a)) - \mathbb{E}_{s,a\sim\mathcal{D}}[Q_\theta(s, a)]) + \widetilde{\mathcal{E}}(Q_\theta, \hat{\mathcal{B}}^\pi Q_{\bar{\theta}}) \right]$ ▷ Eval

7: $\quad \phi \leftarrow \phi + \eta_\pi \nabla_\phi \left[\mathbb{E}_{s,a\sim\{\mathcal{D}\cup B\}}[Q_\theta(s, a) - \lambda \log \pi_\phi(a|s)] \right]$; ▷ Improv

8: $\quad$ **if** t % target_update_period $= 0$ **then**

9: $\quad\quad \bar{\theta} \leftarrow (1 - \tau)\bar{\theta} + \tau\theta$ ▷ Soft update periodically

10: $\quad$ **end if**

11: **end for**

Two practical relaxations are implemented:

- line 6: approximated using only a mini-batch of the state-action pairs in the buffer
- cannot sample next states s' from black-box simulator for the computation of $u(s, a)$, so the expected value is the average of N samples taken from a gaussian $\mathcal{N}(s', \Sigma_D)$

Interpretation of Dynamics-Aware Policy Evaluation 1/3

Considering that $\log \sum_{s,a} \omega(s,a) \exp Q(s,a) = \log \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(s,a)} \exp Q(s,a)$, it can be shown that:

$$\mathbb{E}_{s,a\sim\omega(s,a)}[Q(s,a)] \leq \log \mathbb{E}_{s,a\sim\omega(s,a)} \exp(Q(s,a)) \leq \mathbb{E}_{s,a\sim\omega(s,a)}[Q(s,a)] + \frac{\mathrm{Var}_{\omega}[\exp(Q(s,a))]}{2\exp(2Q_{\min})}$$

With this bound, $\mathbb{E}_{s,a\sim\omega(s,a)}[Q(s,a)]$ can be a reasonable approximation if $\gamma \to 1$ and the range of reward function is designed properly.

Interpretation of Dynamics-Aware Policy Evaluation 2/3

Consider the following approximated policy evaluation objective:

$$\min_{Q} \beta \left(\mathbb{E}_{s,a\sim\omega(s,a)}[Q(s,a)] \right) - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})] + \tilde{\mathcal{E}} \left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q} \right)$$

By assuming a tabular Q-function and applying an approximate dynamic programming approach, the Q-value update rule is derived as:

$$\hat{Q}^{k+1}(s,a) = (\hat{\mathcal{B}}^{\pi} \hat{Q}^{k})(s,a) - \beta \left[\frac{\omega(s,a) - d^{\pi_{\mathcal{D}}}_{\mathcal{M}}(s,a)}{d^{\pi_{\mathcal{D}}}_{\mathcal{M}}(s,a) + d^{\pi}_{\hat{\mathcal{M}}}(s,a)} \right]$$

- $d^{\pi_{\mathcal{D}}}_{\mathcal{M}}(s,a)$: State-action marginal distribution under the behavioral policy;
- $d^{\pi}_{\hat{\mathcal{M}}}(s,a)$: State-action marginal distribution under the learned policy.

Interpretation of Dynamics-Aware Policy Evaluation 3/3

Considering $\nu(s,a)$ as the second term in the RHS it can be seen as an adaptive reward adjustment term:

$$\hat{Q}^{k+1}(s,a) = (\hat{\mathcal{B}}^{\pi}\hat{Q}^{k})(s,a) - \beta\left[\frac{\omega(s,a) - d_{\mathcal{M}}^{\pi_{\mathcal{D}}}(s,a)}{d_{\mathcal{M}}^{\pi_{\mathcal{D}}}(s,a) + d_{\hat{\mathcal{M}}}^{\pi}(s,a)}\right] = (\hat{\mathcal{B}}^{\pi}\hat{Q}^{k})(s,a) + \nu(s,a)$$

- if $\omega(s,a) > d_{\mathcal{M}}^{\pi_{\mathcal{D}}}(s,a)$, then $\nu(s,a)$ serve as a reward penalty and either the state-action pair has large dynamics gap or it belongs to low data density areas;
- if $\omega(s,a) < d_{\mathcal{M}}^{\pi_{\mathcal{D}}}(s,a)$, then $\nu(s,a)$ serve as a reward boost and either the state-action pair has low dynamics gap or have more information from offline data.

Section 4

Experiments & Results

Sim-based Experiments setups

Simulation-based experiments: use the original MuJoCo-HalfCheetah task environments as the real environments, and introduce dynamics gaps upon it to create the simulated environments.

- Offline dataset: D4RL (Medium, Medium Replay, Medium Expert)
- The online training is performed on the modified simulation environment, and we evaluate the performance of the learned policy in the original unchanged MuJoCo environment in terms of the average return

The variations for each experiments are:

- Gravity 2x,
- Friction 0.3x,
- Joint Noise, random noise sampled from $\mathcal{N}(0, 1)$.

Results - sim-based

Table 1: Average returns for MuJoCo-HalfCheetah tasks. Results are averaged over 5 random seeds.

Dataset	Unreal dynamics	SAC(sim)	CQL(real)	DARC	DARC+	H2O
Medium	Gravity	4513±513	6066±73	5011±456	5706±440	**7085±416**
	Friction	2684±2646	6066±73	6113±104	6047±112	**6848±445**
	Joint Noise	4137±805	6066±73	5484±171	5314±520	**7212±236**
Medium Replay	Gravity	4513±513	5774±214	5105±460	4958±540	**6813±289**
	Friction	2684±2646	5774±214	5503±263	5288±100	**5928±896**
	Joint Noise	4137±805	5774±214	5137±225	5230±209	**6747±427**
Medium Expert	Gravity	4513±513	3748±892	**4759±353**	72±109	**4707±779**
	Friction	2684±2646	3748±892	**9038±1480**	7989±3999	6745±562
	Joint Noise	4137±805	3748±892	**5288±104**	733±767	**5280±1329**

Real-based Experiments setups

Real-World experiments: real wheel-legged robot that moves on a pair of wheels to keep it balanced.

- Robot state space: $\mathcal{S} = (\theta, \dot{\theta}, x, \dot{x})$, where θ denotes the forward tilt angle of the body, x is the robot's displacement, and $\dot{\theta}$ and $\dot{x}$ are the angular and linear velocities, respectively.
- Two tasks:
  1. **Standing still**: robot has to stand still and not move or fall down.
  2. **Moving straight**: control the robot to move forward at a target velocity v while keeping its balance.
- Offline dataset: 100,000 human-controlled transitions with different reward functions.

Experiments results - real-based

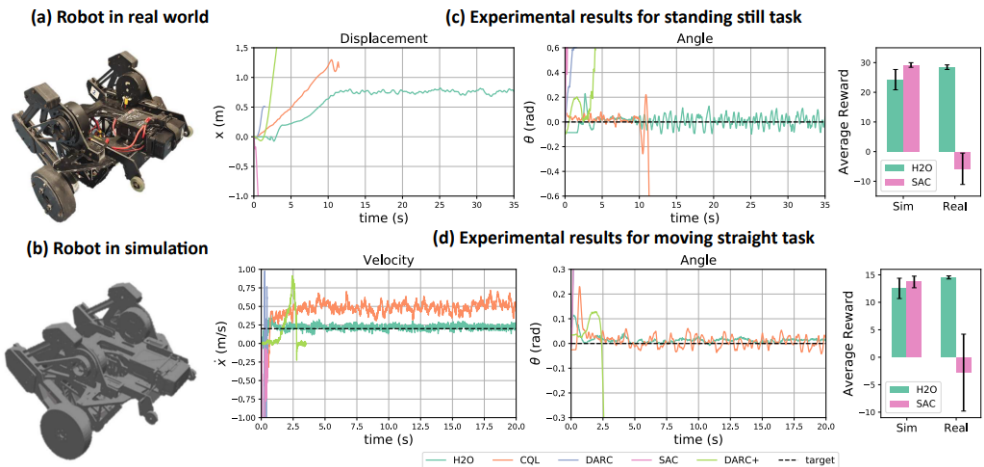


Figure 2: Real-world validation on a wheel-legged robot

Ablation Study

Table 2: Ablations on H2O. "-a" denotes no adaptive weights ω. "-dr" denotes no dynamics ratio to fix the Bellman error. "-reg" stands for no value regularization applied in the framework.

	H2O	H2O-a	H2O-dr	H2O-a-dr	H2O-reg	H2O-reg-dr	CQL	SAC
Adaptive ω	✓	✗	✓	✗	-	-	-	-
Uniform ω	✗	✓	✗	✓	-	-	-	-
Modified $\widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q}\right)$	✓	✓	✗	✗	✓	✗	-	-
Average Return	**6813±289**	6675±179	4721±196	5223±198	6501±147	5290±356	5774±214	4513±513

Section 5

Conclusion

Conclusion

- H2O aims to unify offline and online reinforcement learning while addressing the sim-to-real dynamics gap in imperfect simulators.
- It introduces a *dynamics-aware policy evaluation* scheme that penalizes Q-values of simulated samples with large dynamics gaps and boosts rewards in regions with low gaps and abundant offline data. This makes full use of both limited offline datasets and unrestricted simulated rollouts, allowing them to complement each other in policy learning.
- Extensive simulation and real-world experiments show that H2O outperforms existing cross-domain RL methods.

Section 6

Code

H2O Repository Structure

Main Folders

- `SimpleSAC`:
  - Core implementation of Hybrid SAC
  - Policy, Q-functions, replay buffer
  - Integration with discriminator and CQL
- `Network`:
  - Structure of the discriminators neural network
- `viskit`: logging and visualization tools
- `xml_path`: environment configuration files (MuJoCo XMLs)

Thank you for your attention!

Now on VS Code