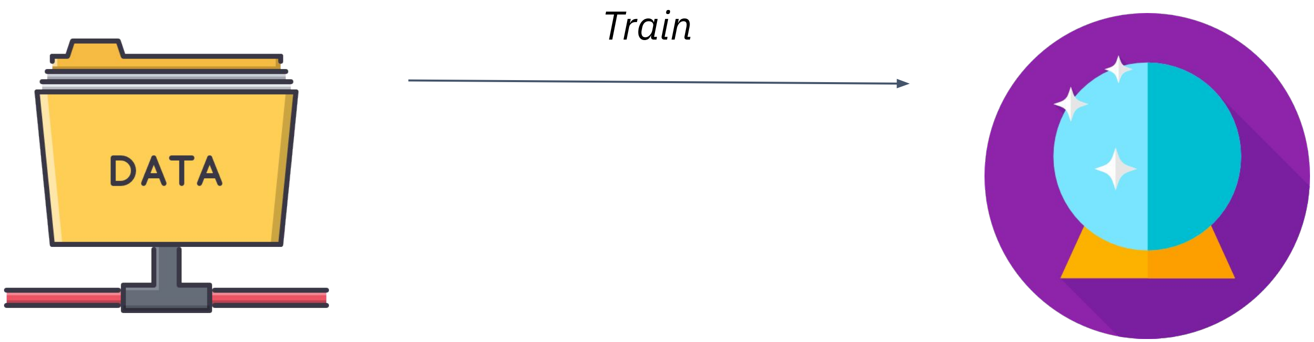
Challenges in Hyperparameter Optimization for Reinforcement Learning

Understanding AutoRL Through an AutoML Lens

Why Reinforcement Learning?

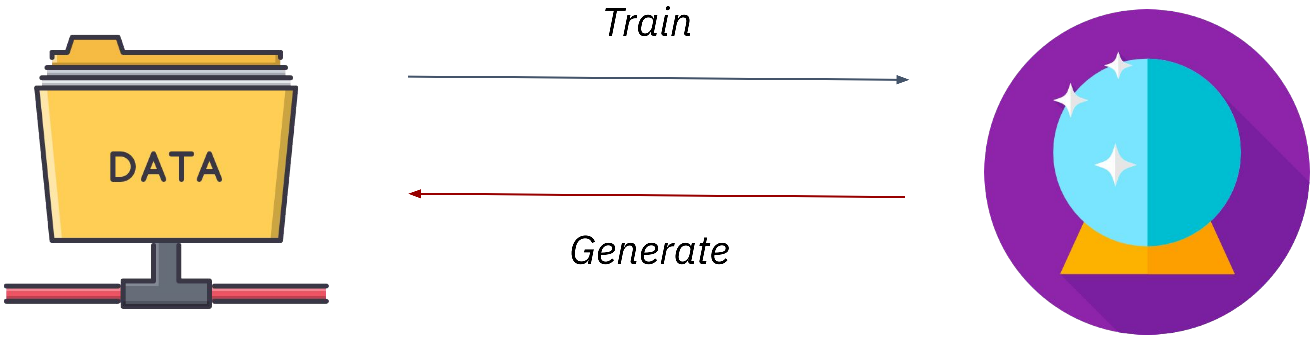
Supervised Learning



Icons: Flaticon.com

Why Reinforcement Learning?

Reinforcement Learning

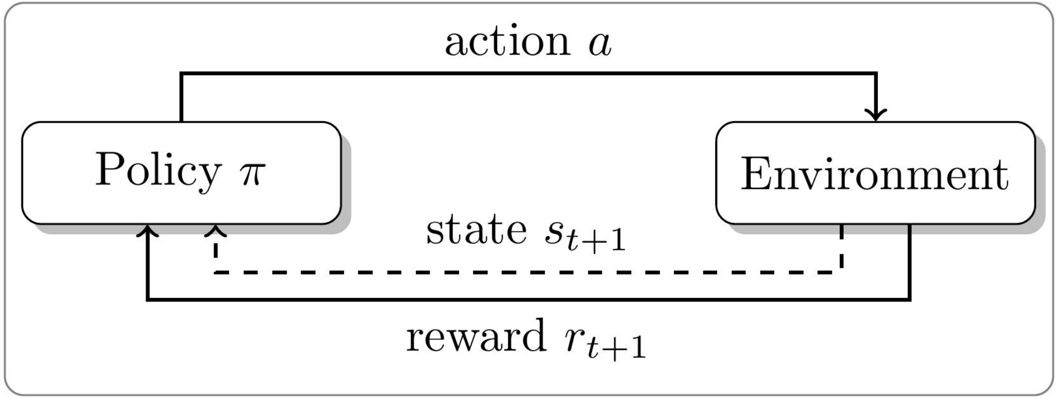


Icons: Flaticon.com

Why Reinforcement Learning?

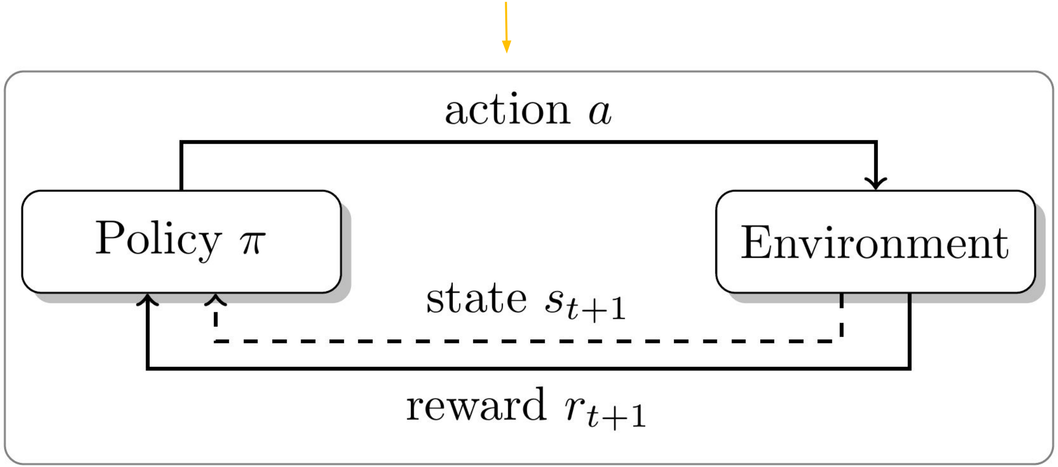
- High impact of AutoML methods & tools
- Ideal testbed for dynamic configuration
- Understanding data generation in RL can improve efficiency for data intensive tasks like NLP

The Reinforcement Learning Loop



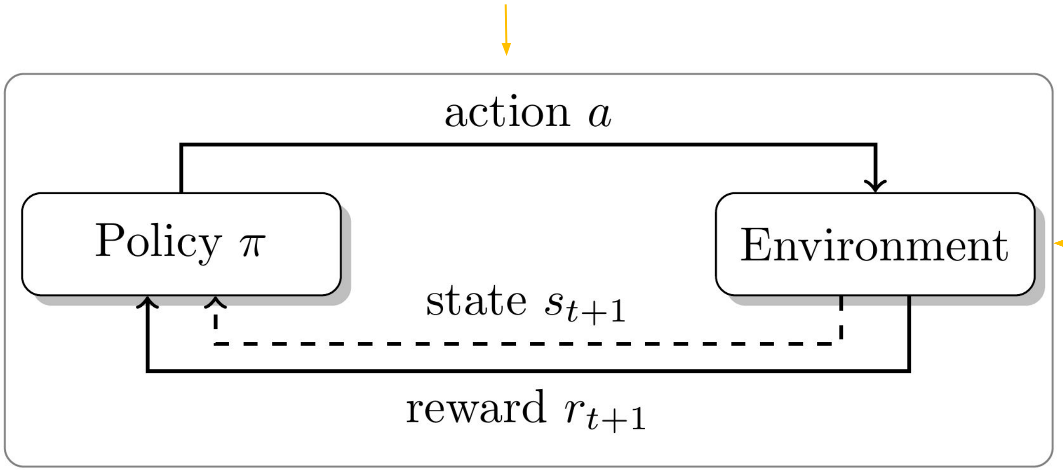
The Reinforcement Learning Loop

Exploration HPs



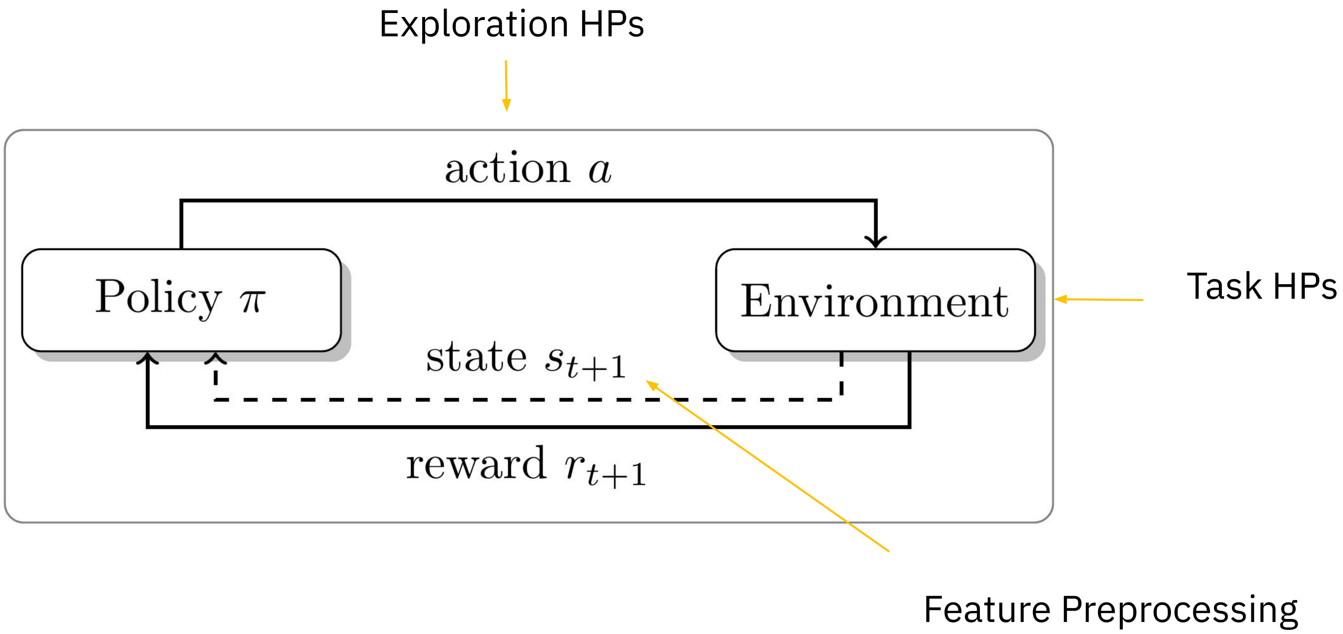
The Reinforcement Learning Loop

Exploration HPs



Task HPs

The Reinforcement Learning Loop



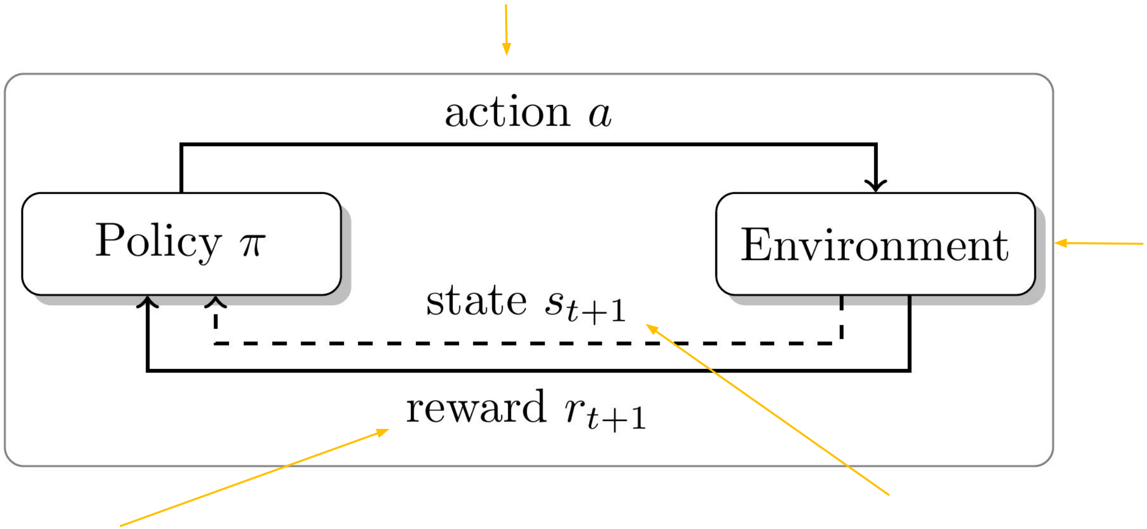
Exploration HPs

Task HPs

Feature Preprocessing

The Reinforcement Learning Loop

Exploration HPs

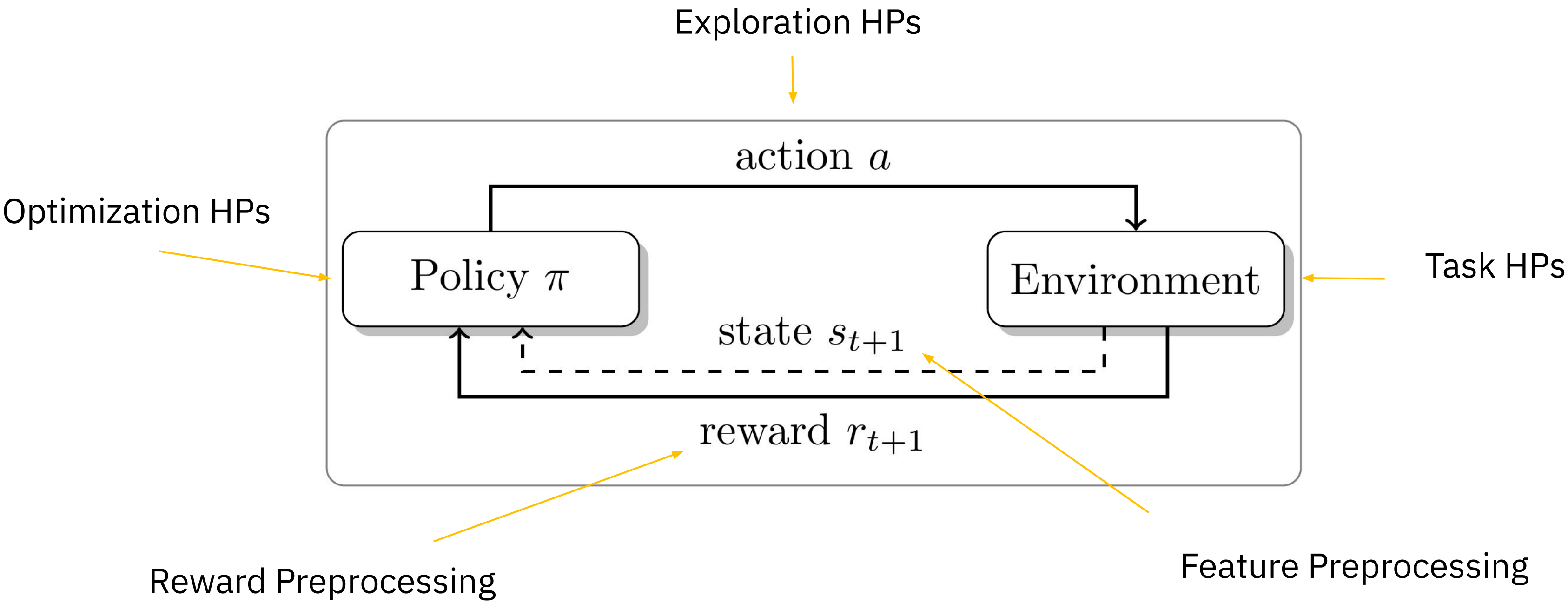


Task HPs

Reward Preprocessing

Feature Preprocessing

The Reinforcement Learning Loop



Exploration HPs

Optimization HPs

Task HPs

action a

Policy π

Environment

state s_{t+1}

reward r_{t+1}

Reward Preprocessing

Feature Preprocessing

Why Is AutoRL Challenging?

- Several components need to work together [Parker-Holder et al. ‘22]
- Data needs change during training [Klink et al. ‘20, Jiang et al. ‘21]
- Data distribution changes during training
- Instability in data generation and instability in training compound
- Optimal values of HPs change during the runtime [Mohan et al. ‘23]
- Meta-Learning components is often important

HPO in RL Currently

- HPO is often necessary to apply existing algorithms

  But: grid search is most common [Badia et al. ‘20, Hambro et al. ‘22]
- HPO methods tailored to RL exist [Jaderberg et al. ‘17, Wan et al. ‘22]

  But: no established user base, no established HPO settings
- On-the-fly HP adaption is gaining traction [O'Donoghue ‘23]

  But: few insights into where, when and why this works

HPO in RL Currently

- HPO is often necessary to apply existing algorithms

  But: grid search is most common [Badia et al. ‘20, Hambro et al. ‘22]
- HPO methods tailored to RL exist [Jaderberg et al. ‘17, Wan et al. ‘22]

  But: no established user base, no established HPO settings
- On-the-fly HP adaption is gaining traction [O'Donoghue ‘23]

  But: few insights into where, when and why this works

⇨ few insights, little adoption, little awareness of HPO best practices in the RL community

Hyperparameters in Reinforcement Learning and How to Tune Them

[Eimer et al. ICML'23]

LUH|AI

LUH|AI

Meta

Analyzing The HPO Landscape of RL

Most important questions:

- How important is HPO in RL?
- How dependent on the task are RL HPs?
- What are the best ways to tune HPs in RL?
- What’s missing in current HPO methods?

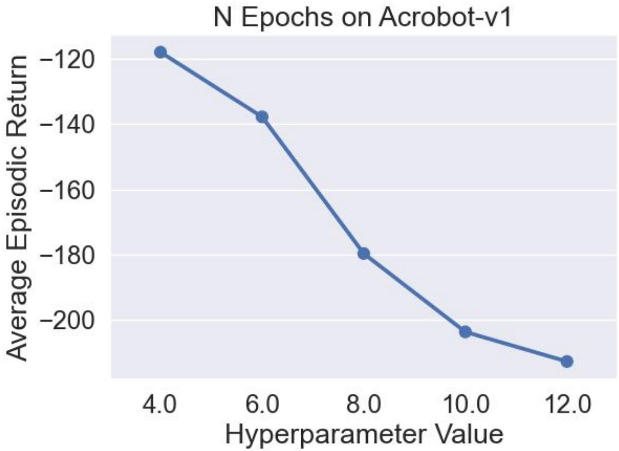
Analyzing The HPO Landscape of RL

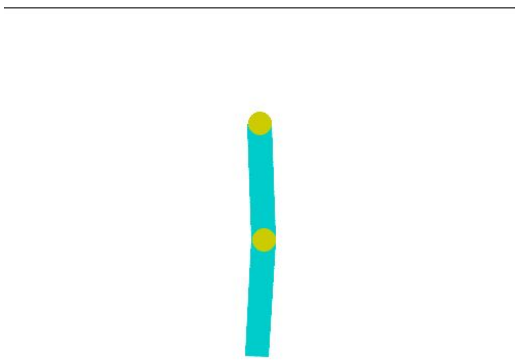
Methods:

- Hyperparameter sweeps for 128 algorithm/environment/HP combinations
- Minimal budget tuning experiments with PB2 [Parker-Holder et al. ‘20], DEHB [Awad et al. ‘21] and Random Search (RS) for 10 target algorithm runs
- Small budget tuning experiments on state-of-the-art environments with 64 target algorithm runs

Insight 1: Many HPs Are Relevant

Across all settings, **only 8 combinations** of algorithm/environment/HP show the worst HP value being within the standard deviation of the best one



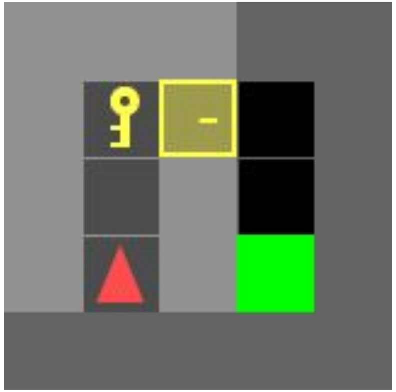


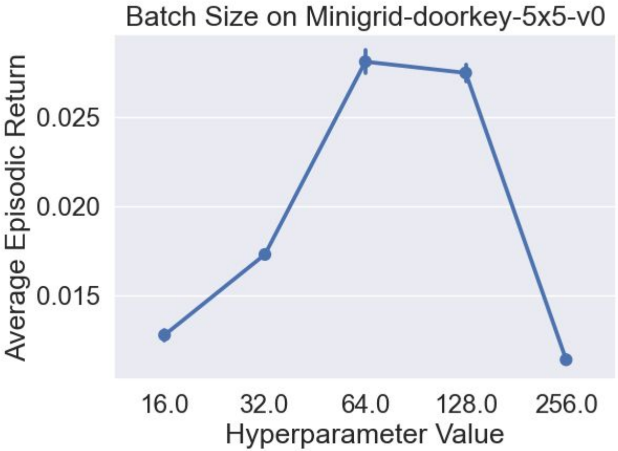
Left: PPO’s number of epochs on Acrobot.

Right: Acrobot example.

Insight 1: Many HPs Are Relevant

Across all settings, **only 8 combinations** of algorithm/environment/HP show the worst HP value being within the standard deviation of the best one



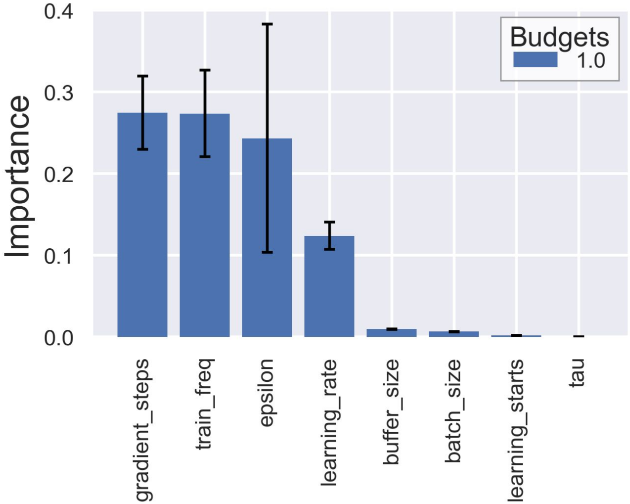


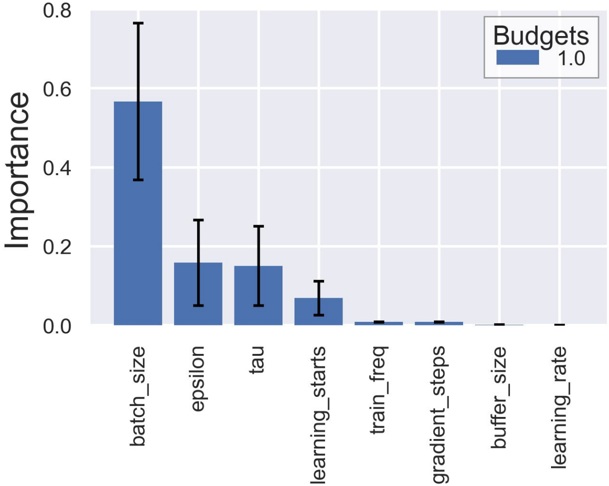
Left: MiniGrid DoorKey example.

Right: DQN’s batch size on MiniGrid DoorKey 5x5.

Insight 2: HP Importance Is Task Dependent

- HP importance computed by fANOVA [Hutter et al. '14]
- 1-4 main important HPs per algorithm and environment
- Ordering differs significantly between environments



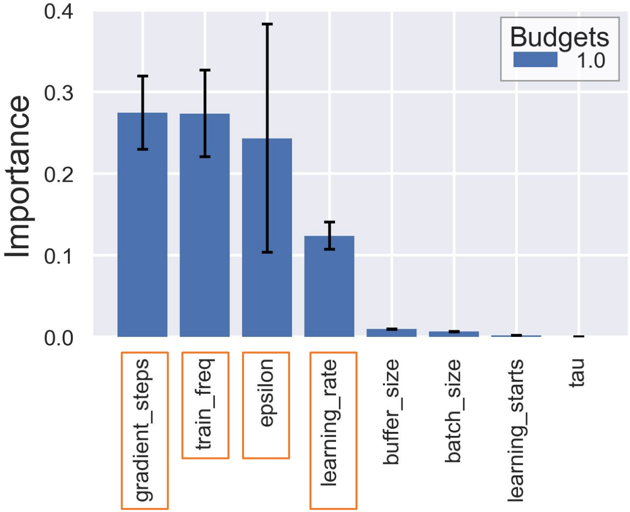


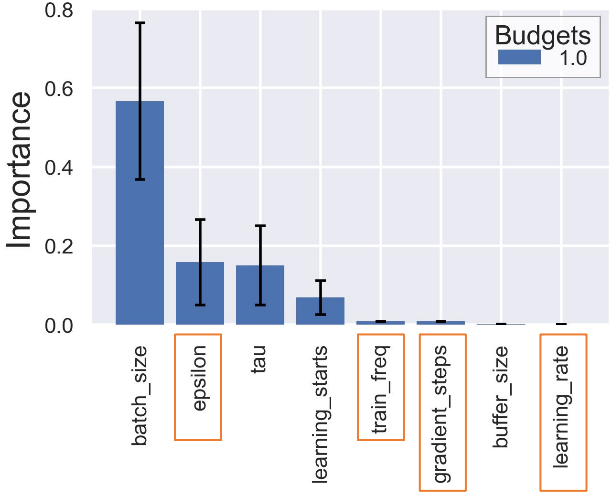
Left: DQN HP importance on Acrobot.

Right: DQN HP importance on MiniGrid 5x5.

Insight 2: HP Importance Is Task Dependent

- HP importance computed by fANOVA [Hutter et al. ‘14]
- 1-4 main important HPs per algorithm and environment
- Ordering differs significantly between environments



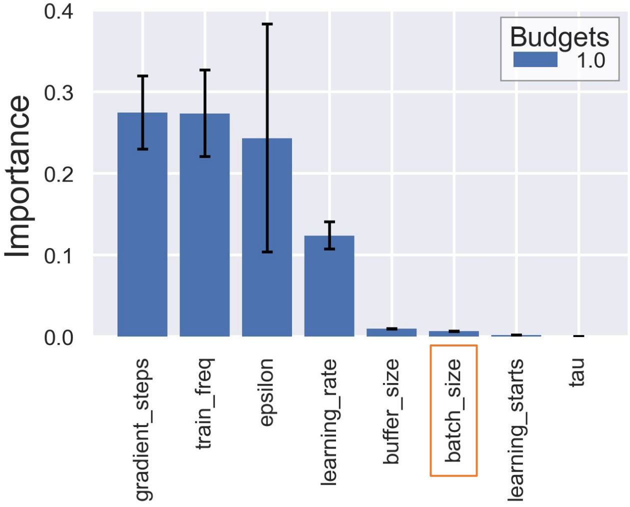


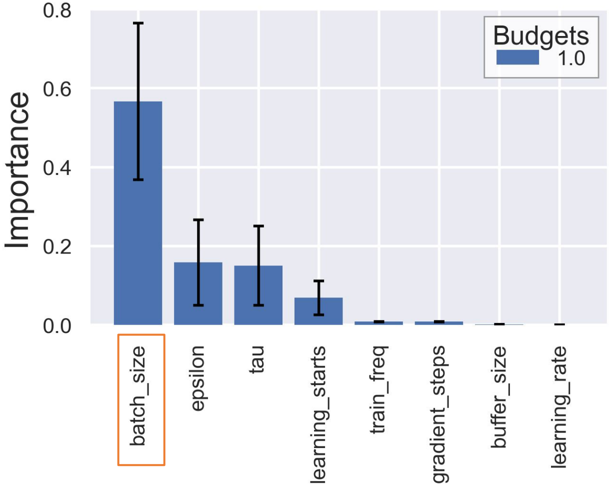
Left: DQN HP importance on Acrobot.

Right: DQN HP importance on MiniGrid 5x5.

Insight 2: HP Importance Is Task Dependent

- HP importance computed by fANOVA [Hutter et al. ‘14]
- 1-4 main important HPs per algorithm and environment
- Ordering differs significantly between environments



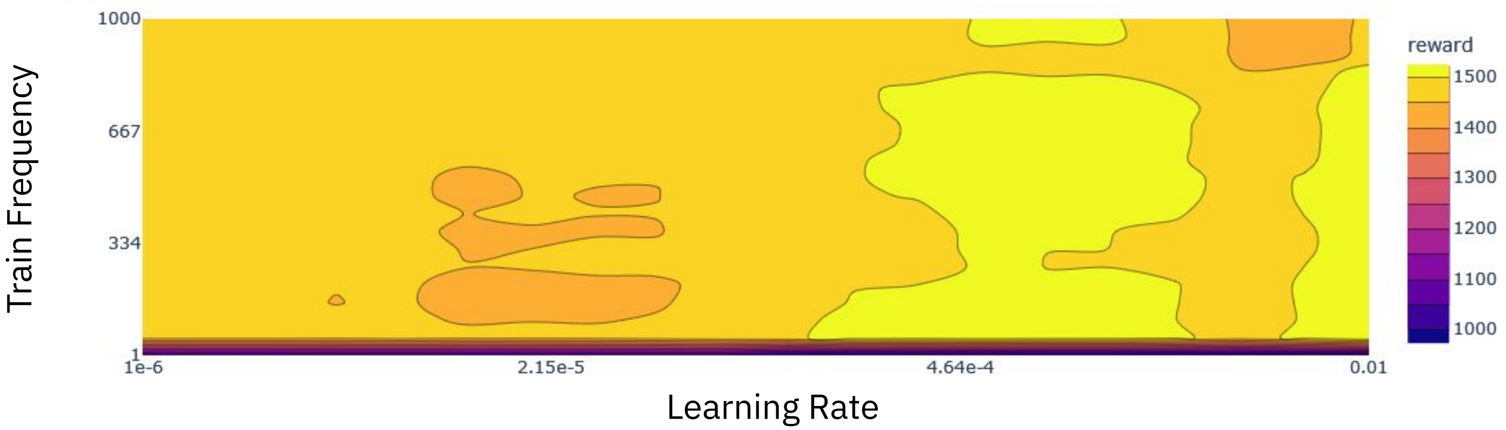


Left: DQN HP importance on Acrobot.

Right: DQN HP importance on MiniGrid 5x5.

Insight 3: HPs Are Benign

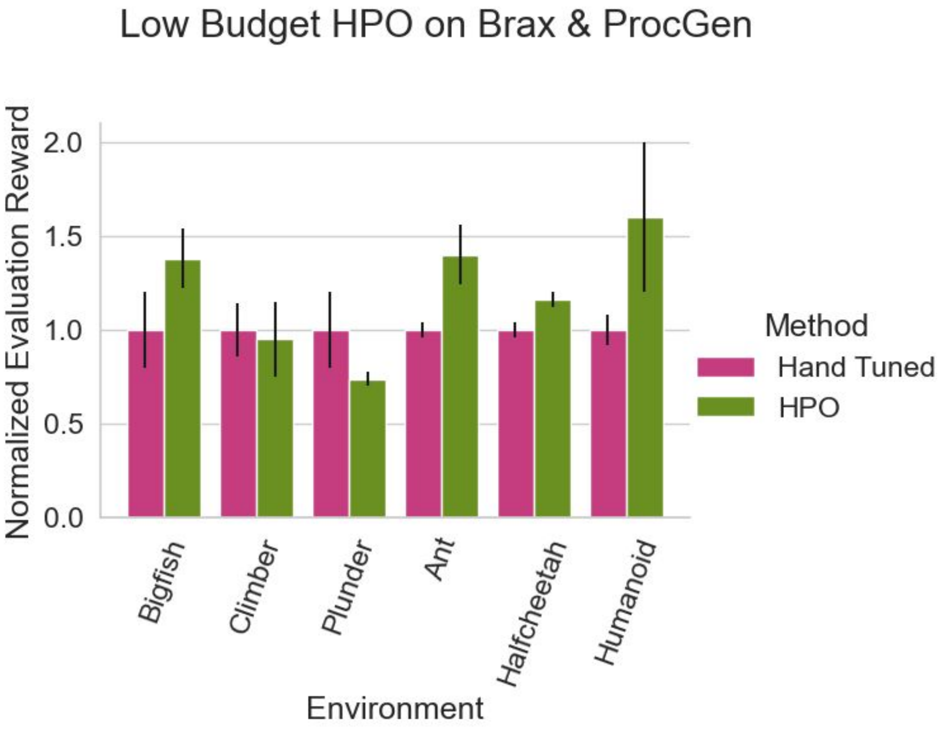
- PDPs [Moosbauer et al. ‘21] of selected environments show few interaction effects between HPs
- The HP ranges where agents perform well is fairly wide



PDP: Train Frequency and Learning Rate of SAC on Pendulum

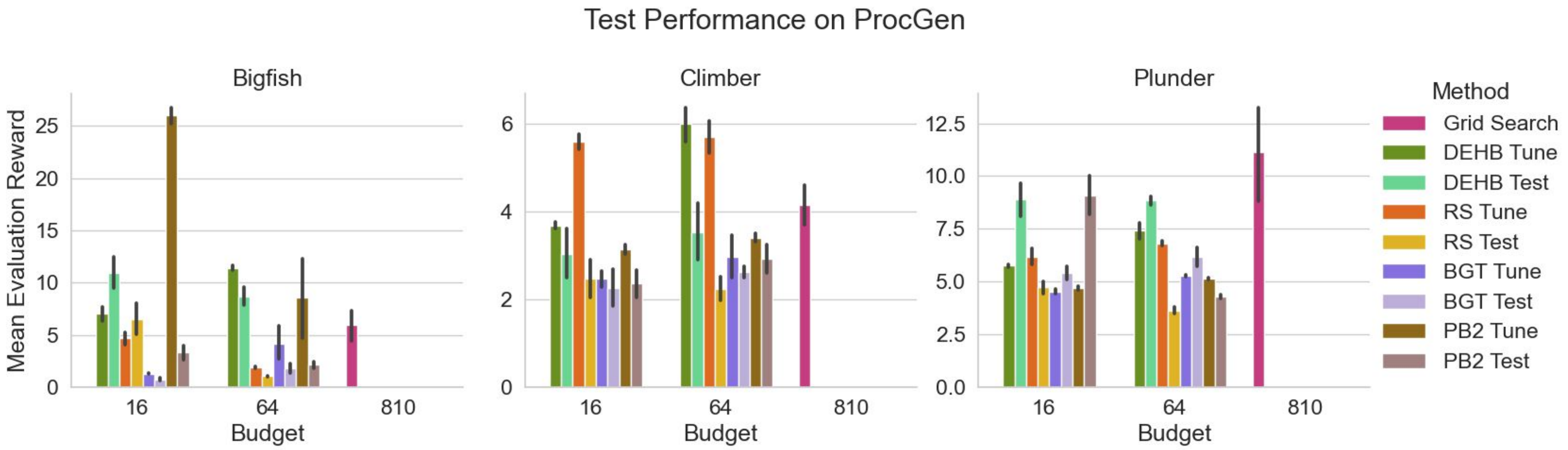
Insight 4: Small Budgets Are Enough

- **With only 10 runs:** improvements in performance compared to large sweeps on the tuning seeds
- **With 64 runs:** overall improvement over hand tuned default settings with 10x more budget using DEHB (see Figure)
- **But…**



Insight 5: Overfitting Is A Problem

- Up to 8x worse performance on test seed, even when tuning across multiple seeds
- PB2's overfitting is environment dependent - thus possibly due to its dynamic nature



What does that mean for tuning RL?

Most important questions:

- How important is HPO in RL?
- How dependent on the task are RL HPs?
- What are the best ways to tune HPs in RL?
- What’s missing in current HPO methods?

What does that mean for tuning RL?

Most important questions:

- How important is HPO in RL?
  Very important due to many relevant HPs for any given task
- How dependent on the task are RL HPs?
  Strong dependence across the board
- What are the best ways to tune HPs in RL?
  Established HPO methods work well, RL-specific ones have failure cases
- What’s missing in current HPO methods?
  ?

Hyperparameters in Contexual Reinforcement Learning are Highly Situational

[Eimer et al. EcoRL@NeurIPS'21]

LUH|AI

LUH|AI

LUH|AI

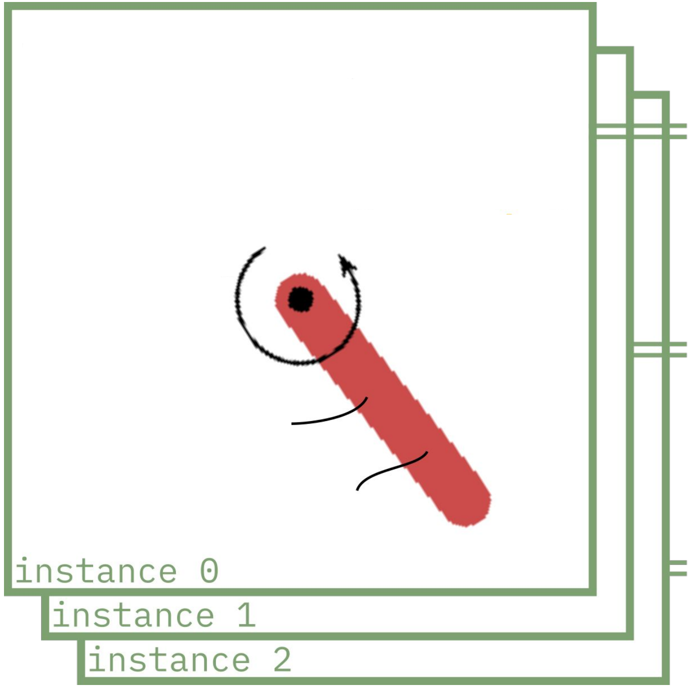
Cranking up the Difficulty: cRL

- In cRL an RL agent needs to solve a range of tasks
- Increased learning challenge due to more diverse data
- Increased data generation challenge due to larger state space

This difficulty increase transfers to the HPO task

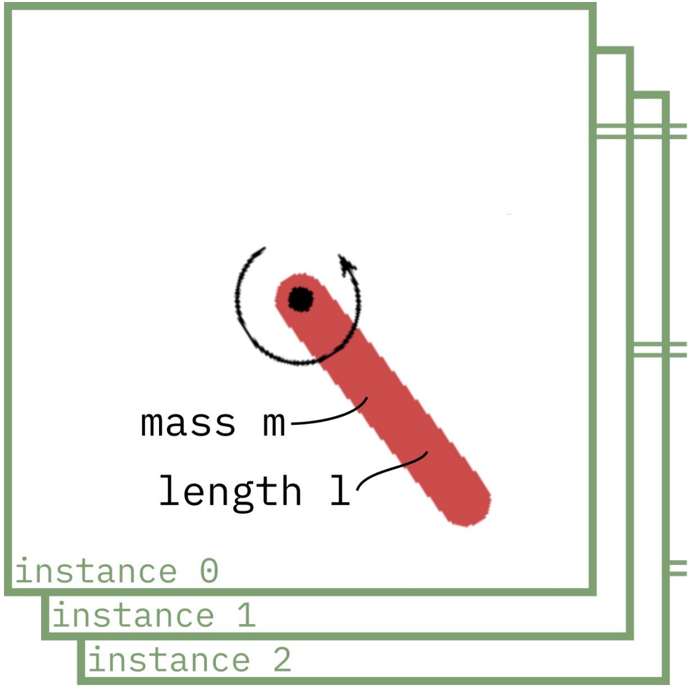
Cranking up the Difficulty: cRL

Example cRL Task: Pendulum instances from CARL [Benjamins et al. ‘23]



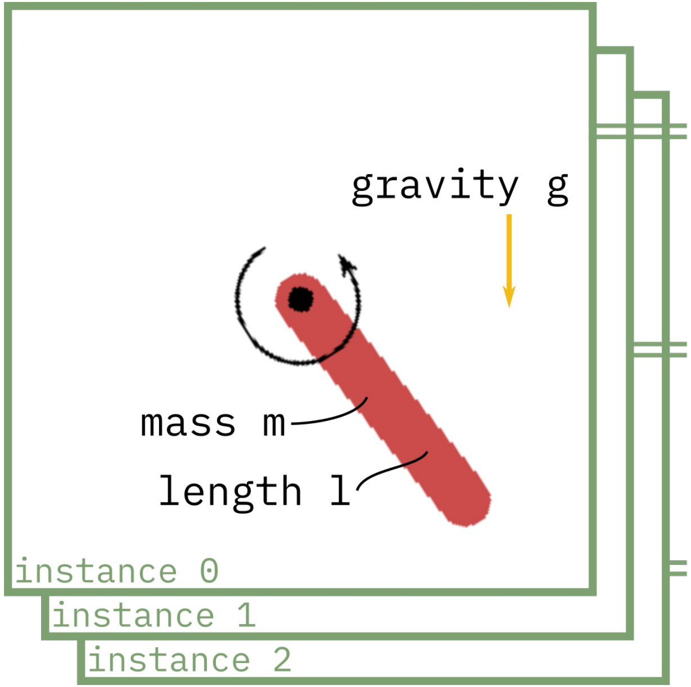
Cranking up the Difficulty: cRL

Example cRL Task: Pendulum instances from CARL [Benjamins et al. ‘23]



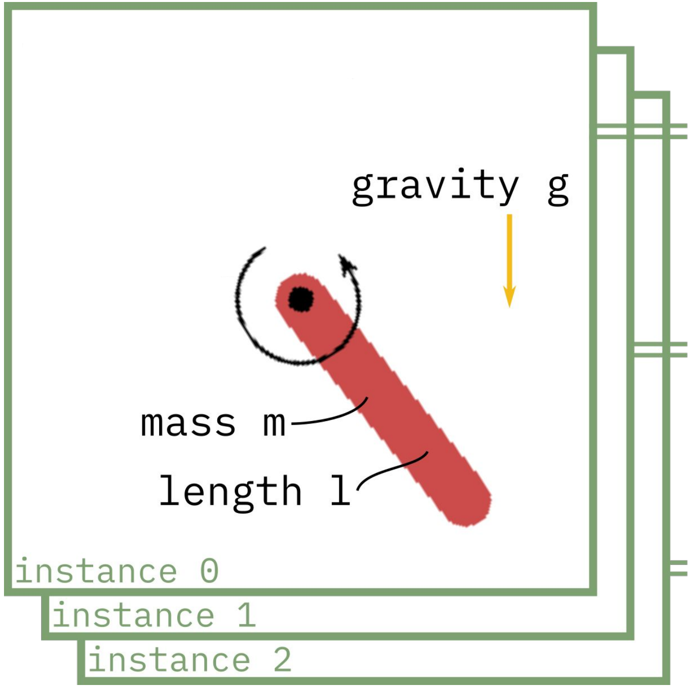
Cranking up the Difficulty: cRL

Example cRL Task: Pendulum instances from CARL [Benjamins et al. '23]



Cranking up the Difficulty: cRL

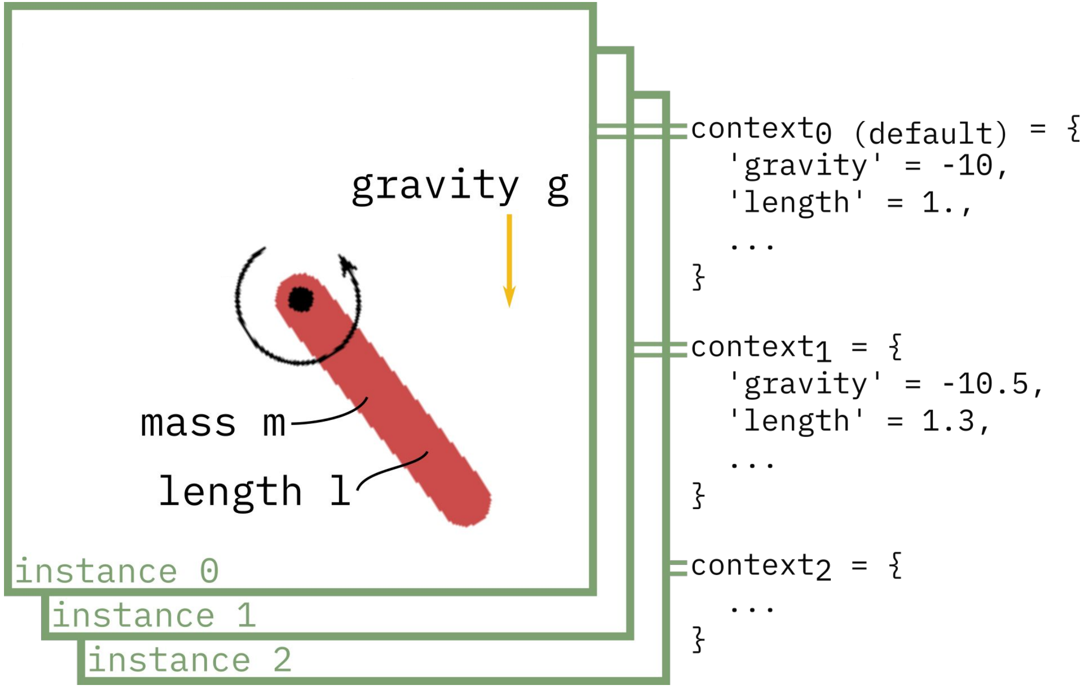
Example cRL Task: Pendulum instances from CARL [Benjamins et al. ‘23]



```
context0 (default) = {
   'gravity' = -10,
   'length' = 1.,
   ...
}
```

Cranking up the Difficulty: cRL

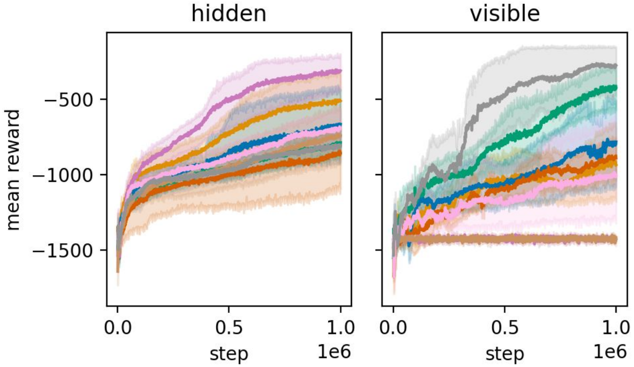
Example cRL Task: Pendulum instances from CARL [Benjamins et al. ‘23]

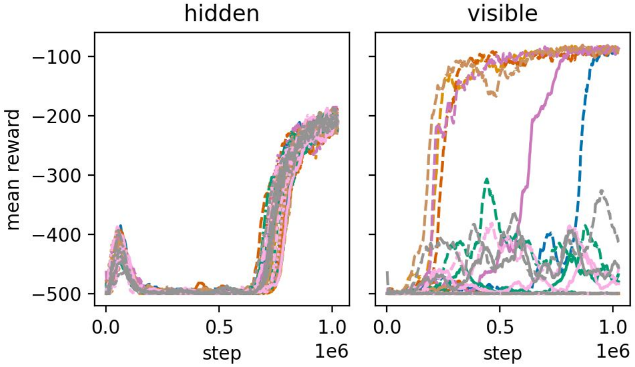


```
context0 (default) = {
   'gravity' = -10,
   'length' = 1.,
   ...
}

context1 = {
   'gravity' = -10.5,
   'length' = 1.3,
   ...
}

context2 = {
   ...
}
```

Cranking up the Difficulty: cRL

- Tuning a PPO agent explicitly asked to generalize (visible) and an agent not shown that the task is varied (hidden)
- We use PB2 and track performance of each configuration over time
- Explicitly optimizing for generalization makes it harder to find stable and well-performing configurations





Left: Population members over time on Pendulum across 5 seeds.

Right: Population members over time on Acrobot. Colors indicate configurations, patterns seeds.

Closing the Gap: Dynamic Configuration

- Optimal HP values change over time [Mohan et al. ‘23]
- Tasks and task spaces can also vary during training
- Dynamic HPO has shown promising results for RL already [Wan et al. ‘22]

Closing the Gap: Dynamic Configuration

- Optimal HP values change over time [Mohan et al. ‘23]
- Tasks and task spaces can also vary during training
- Dynamic HPO has shown promising results for RL already [Wan et al. ‘22]

However:

- Dynamic HPO methods tend to overfit [Zhang et al. ‘21]
- They are not as efficient as established HPO methods yet
- The HPs of dynamic HPO methods are not well understood yet

Ingredient 1: Fidelities

- Few insights into how performance on lower fidelities in RL translates to higher ones:
    - How do HPs on easier tasks correspond to more difficult ones?
    - Can we use low runtimes to predict performance on longer ones?
- Fidelities offer significant performance increases in AutoML [Li et al. '16]
- Information about fidelities is crucial for dynamic configuration

Ingredient 2: Task Features

- In AutoML, dataset features give insights into how HPs generalize
- In RL: currently only very high-level human made task descriptions
- Measuring task similarity is thus hard
- Possible direction: interpretable tasks via explicit task descriptions [Benjamins et al. '23]
- Task features can enable meta-learning HPs and AutoRL methods targeting generalization, e.g. curriculum learning

Ingredient 3: Better Usability of AutoRL

- RL has been expensive from a runtime perspective
- Recently: huge advances is efficiency
- Example:
  - My own PyTorch CartPole: ~30 minutes
  - PureJax [Lu ‘23] CartPole: ~20 seconds
- Standardized benchmarks are still in their infancy, however, limiting comparisons and accessibility [Shala et al. ‘22]

AutoRL for AutoML Methods

- Establishing HPO methods and best practices from the AutoML community can make RL research more efficient and effective immediately
- RL is an ideal testbed for dynamic configuration:
    - Dynamic HP analyses during training
    - HPO tools for on-the-fly adaption
    - Learnt dynamic configuration [Adriaensen et al. ‘22]

AutoML Ideas for AutoRL

- Adopting more HPO ideas like multi-fidelity optimization could drastically improve the efficiency of RL-specific HPO methods
- Grey-Boxing RL through e.g. meaningful task features can enable better optimization for generalization tasks
- Meta-learning based AutoRL approaches can build upon these efficiency and information gains, e.g. in learnt HPs, curriculum learning or exploration

Get In Touch!

LUH|AI

@The_Eimer

LUH-AI

@luh-ai

For more information, check out our paper website, blog post and GitHub repository.

Funded by:

European Research Council
Established by the European Commission

Backup Slides

Formal Definition of AutoRL

AutoRL Problem:

$$\max_{\zeta} f(\zeta, \theta^*) \quad \text{s.t.} \quad \theta^* \in \arg\max_{\theta} J(\theta; \zeta),$$

For inner RL loop:

$$\max_{\theta} J(\theta; \zeta) \quad \text{where} \quad J(\theta; \zeta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t \geq 0} \gamma^t r_t \right] ,$$

More Results Tuning cRL

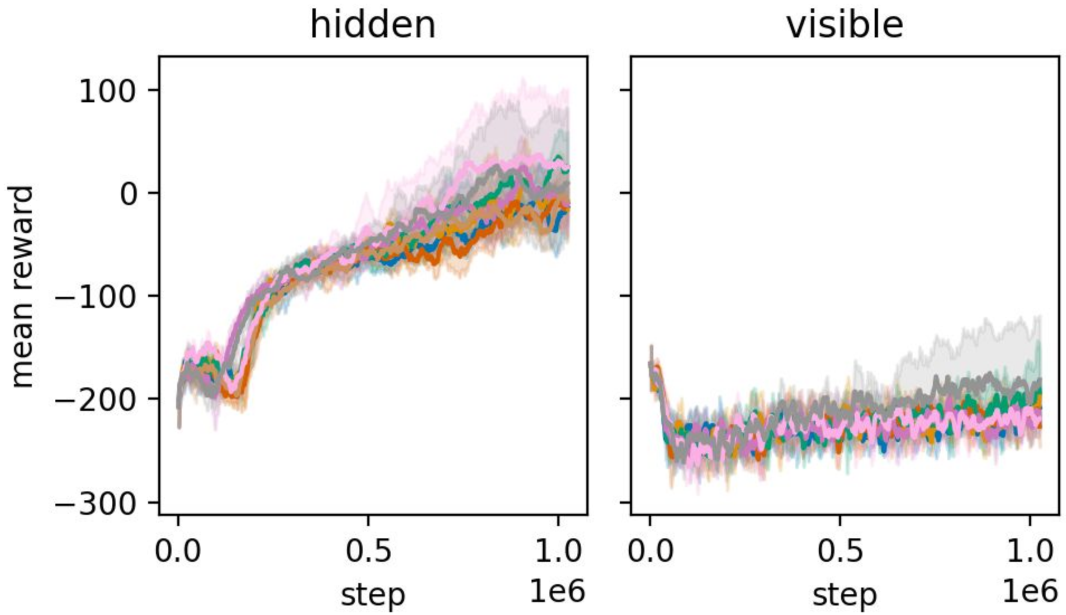


Figure: PB2 on LunarLander