Is My RL Algorithm a Good Tool?

What Evaluation Strategies Tell Us About Our Algorithms

Theresa Eimer

Goals of Evaluations

1. Support research contributions
2. Show gaps in our knowledge (e.g. theory - practice mismatches)
3. Provide a basis for transferring research into application
4. …

N. Enable future research progress

Current practice (In Empirical Online RL)

- ➔ Select algorithm(s) to evaluate
- ➔ Select meaningful environment(s) to evaluate on

- ➔ Set evaluation settings
- ➔ Set hyperparameters

- ➔ Perform several runs to account for randomness
- ➔ Compare mean reward or return over time

What Could “Better” Look Like?

- ➔ Select algorithm(s) to evaluate
- ➔ Select widespread community-driven benchmark for research question

What Could "Better" Look Like?

- ➔ Select algorithm(s) to evaluate
- ➔ Select widespread community-driven benchmark for research question

- ➔ Use benchmark evaluation settings
- ➔ Set hyperparameters using standardized process

What Could “Better” Look Like?

- ➔ Select algorithm(s) to evaluate
- ➔ Select widespread community-driven benchmark for research question

- ➔ Use benchmark evaluation settings
- ➔ Set hyperparameters using standardized process

- ➔ Perform several runs to account for randomness as prescribed by benchmark
- ➔ Compare benchmark metrics in statistical test

But: What Specifically Do We Want To Know?

But: What Specifically Do We Want To Know?

Application specialist: “I want to know which algorithm solves my exact task setting“

But: What Specifically Do We Want To Know?

Application specialist: “I want to know which algorithm solves my exact task setting“

RL algorithm researcher: “I want to know if the mechanics of this algorithm are good enough to solve RL problems generally“

But: What Specifically Do We Want To Know?

Application specialist: “I want to know which algorithm solves my exact task setting“

RL algorithm researcher: “I want to know if the mechanics of this algorithm are good enough to solve RL problems generally“

AGI enthusiast: “I want to know if this algorithm can solve all problems at once“

But: What Specifically Do We Want To Know?

Application specialist: “I want to know which algorithm solves my exact task setting”

Translation: evaluation environment is fixed, algorithm can be freely specified, often budget is limited

Evaluation should show:

But: What Specifically Do We Want To Know?

Application specialist: “I want to know which algorithm solves my exact task setting”

Translation: evaluation environment is fixed, algorithm can be freely specified, often budget is limited

Evaluation should show:

➔ Algorithm & all settings transfer between related problems

Zero-shot transfer of algorithm & training settings within domain

➔ Algorithm & settings can be very efficiently adapted to changes in the setting

Tunability values for very low-budget tuning

Excursion: Tunability [Probst et al. 2019]

Definition: “difference between the risk of an overall reference configuration and the risk of the best possible configuration on that dataset”

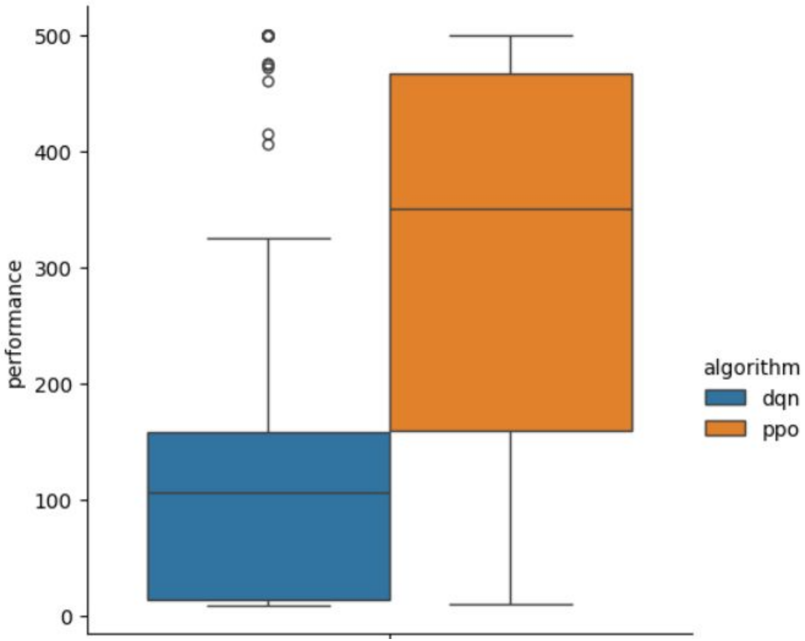
Excursion: Tunability [Probst et al. 2019]

Definition: “difference between the risk of an overall reference configuration and the risk of the best possible configuration on that dataset”

How well can the algorithm be adapted to different settings?

Example: PPO & DQN on CartPole

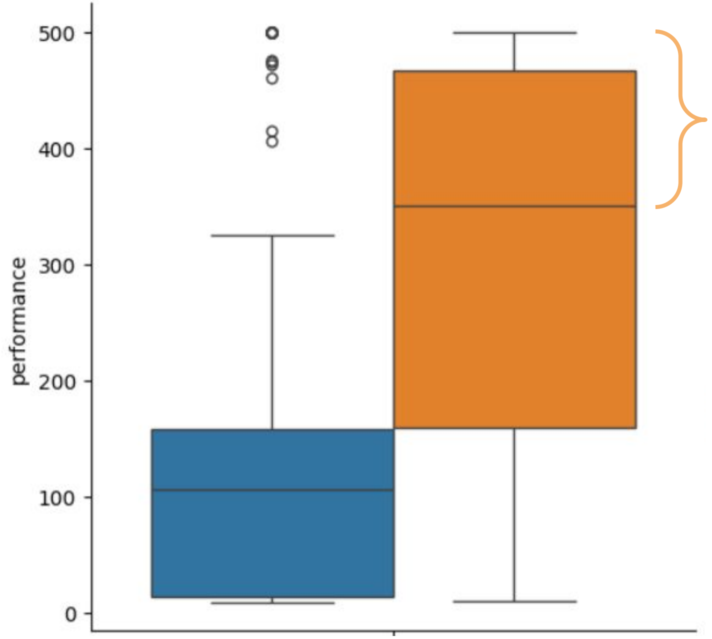
HPO Runs



- ➔ Done with ARLBench [Becktepe & Dierkes et al. 2024]
- ➔ Tuning via Hypersweeper with SMAC [Lindauer et al. 2022]
- ➔ Budget: 32 full runs
- ➔ 1 run per configuration

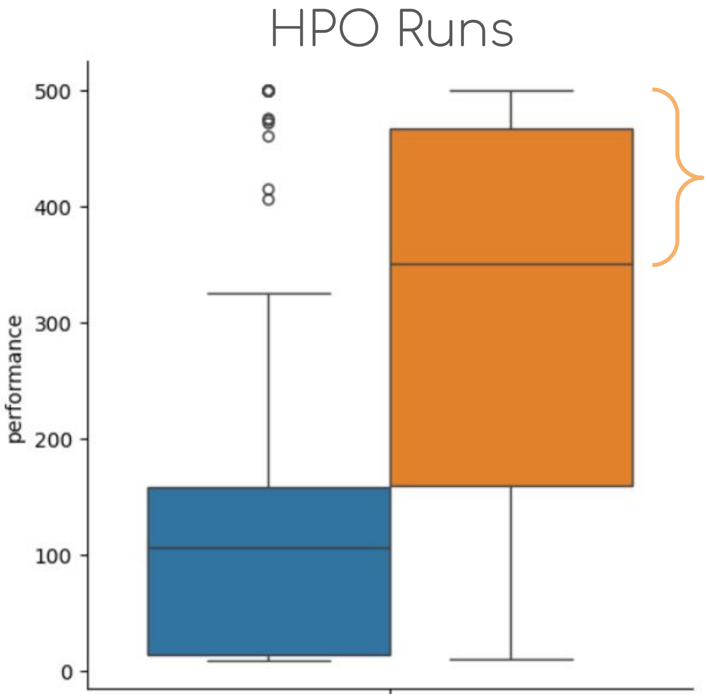
Excursion: Tunability [Probst et al. 2019]

HPO Runs



Tunability of PPO
compared to mean performance

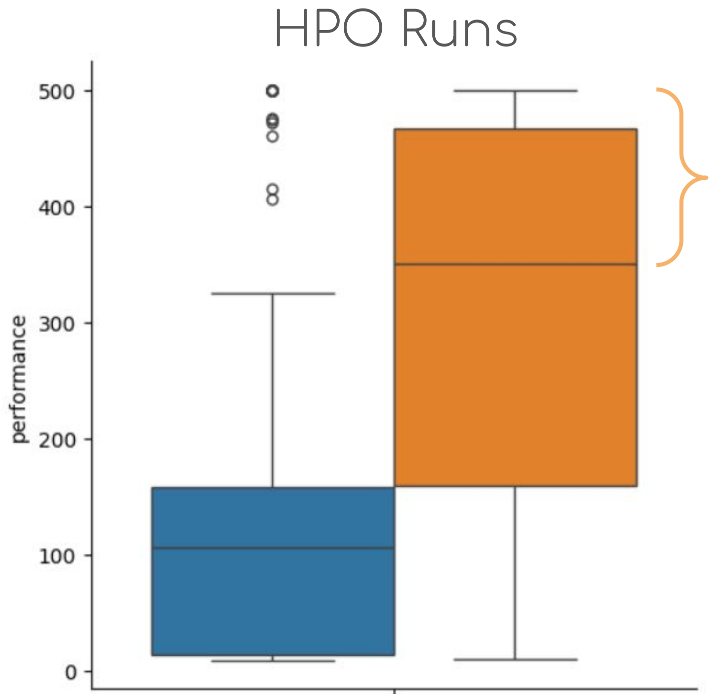
Excursion: Tunability [Probst et al. 2019]



Tunability of PPO
compared to mean performance

Tunability of DQN
compared to mean performance

Excursion: Tunability [Probst et al. 2019]

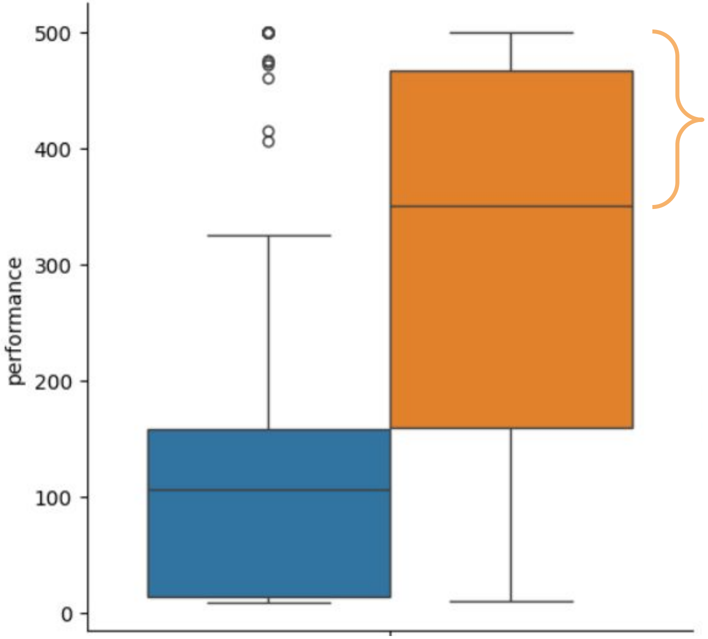


Low Tunability

Option 1: Algorithm is naturally good everywhere and doesn't need to be adapted

Excursion: Tunability [Probst et al. 2019]

HPO Runs

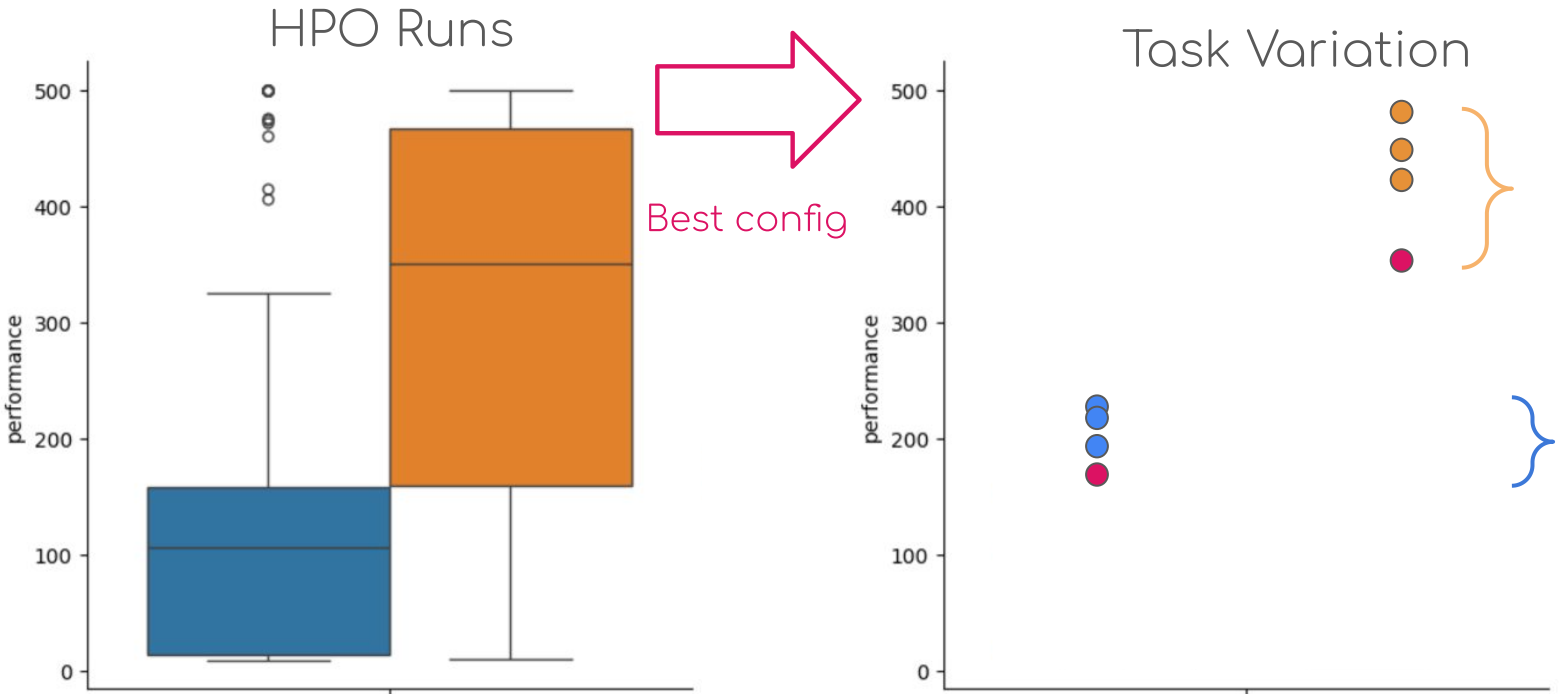


Low Tunability

Option 1: Algorithm is naturally good everywhere and doesn’t need to be adapted

Option 2: Algorithm is naturally okay everywhere but can’t easily be adapted

Excursion: Tunability [Probst et al. 2019]

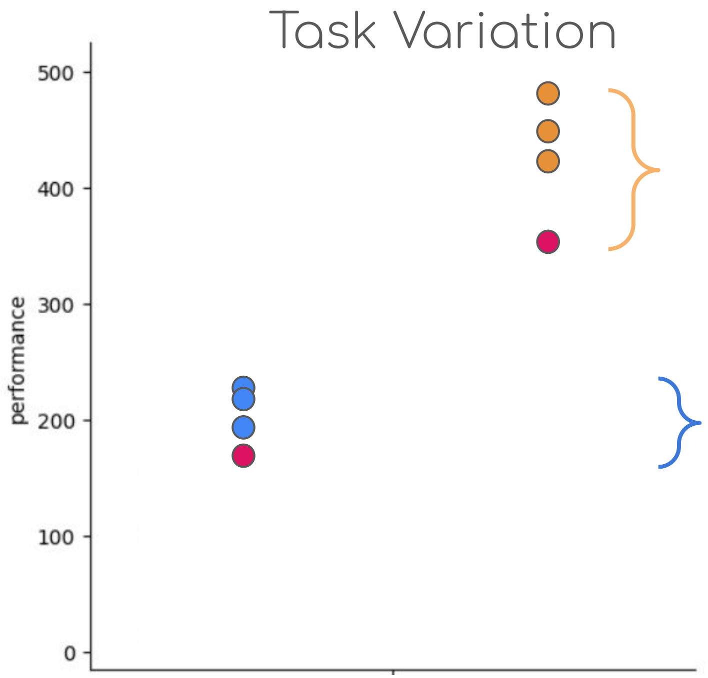


Excursion: Tunability [Probst et al. 2019]

Possible takeaways:

- ➔ Finding good HPs is easier in PPO
- ➔ On a task variation, it is faster to improve PPO

Application specialist: "PPO seems to be a better out of the box and I can get more out of it with only a few changes."



What do we actually want to know?

RL algorithm researcher: “I want to know if the mechanics of this algorithm are good enough to solve RL problems generally”

Translation: algorithm is fixed, environment choice should support research question, reasonable experimentation budget

What do we actually want to know?

RL algorithm researcher: "I want to know if the mechanics of this algorithm are good enough to solve RL problems generally"

Translation: algorithm is fixed, environment choice should support research question, reasonable experimentation budget

Evaluation should show:

- ➔ We can make the algorithm work on any setting within its scope (efficiently)
- ➔ Algorithm & all settings should transfer to some degree between tasks
- ➔ Algorithm behavior is good in key metrics important to research question

What do we actually want to know?

RL algorithm researcher: “I want to know if the mechanics of this algorithm are good enough to solve RL problems generally”

Translation: algorithm is fixed, environment choice should support research question, reasonable experimentation budget

Evaluation should show:

➔ We can make the algorithm work on any setting within its scope (efficiently)
High tunability on any setting, **efficient to tune** compared to other algorithms

➔ Algorithm & all settings should transfer to some degree between tasks

➔ Algorithm behavior is good in key metrics important to research question

What do we actually want to know?

RL algorithm researcher: "I want to know if the mechanics of this algorithm are good enough to solve RL problems generally"

Translation: algorithm is fixed, environment choice should support research question, reasonable experimentation budget

Evaluation should show:

➔ We can make the algorithm work on any setting within its scope (efficiently)
High tunability on any setting, **efficient to tune** compared to other algorithms
➔ Algorithm & all settings should transfer to some degree between tasks
Zero-shot transfer to some variations in setup & hyperparameters
➔ Algorithm behavior is good in key metrics important to research question

What do we actually want to know?

RL algorithm researcher: "I want to know if the mechanics of this algorithm are good enough to solve RL problems generally"

Translation: algorithm is fixed, environment choice should support research question, reasonable experimentation budget

Evaluation should show:

- ➔ We can make the algorithm work on any setting within its scope (efficiently)
  High tunability on any setting, **efficient to tune** compared to other algorithms
- ➔ Algorithm & all settings should transfer to some degree between tasks
  Zero-shot transfer to some variations in setup & hyperparameters
- ➔ Algorithm behavior is good in key metrics important to research question
  Standard evaluation metrics like exploration coverage

Excursion: Tuning Efficiency

What does it mean if an algorithm is efficient to tune?

Excursion: Tuning Efficiency

What does it mean if an algorithm is efficient to tune?

➔ Few evaluations are enough to point to the optimum
➔ Random sampling likely yields good configurations
➔ It’s clear early on if a configuration is good

Excursion: Tuning Efficiency

What does it mean if an algorithm is efficient to tune?

➔ Easily searchable, predictable performance landscape
➔ Random sampling likely yields good configurations
➔ It’s clear early on if a configuration is good

Excursion: Tuning Efficiency

What does it mean if an algorithm is efficient to tune?

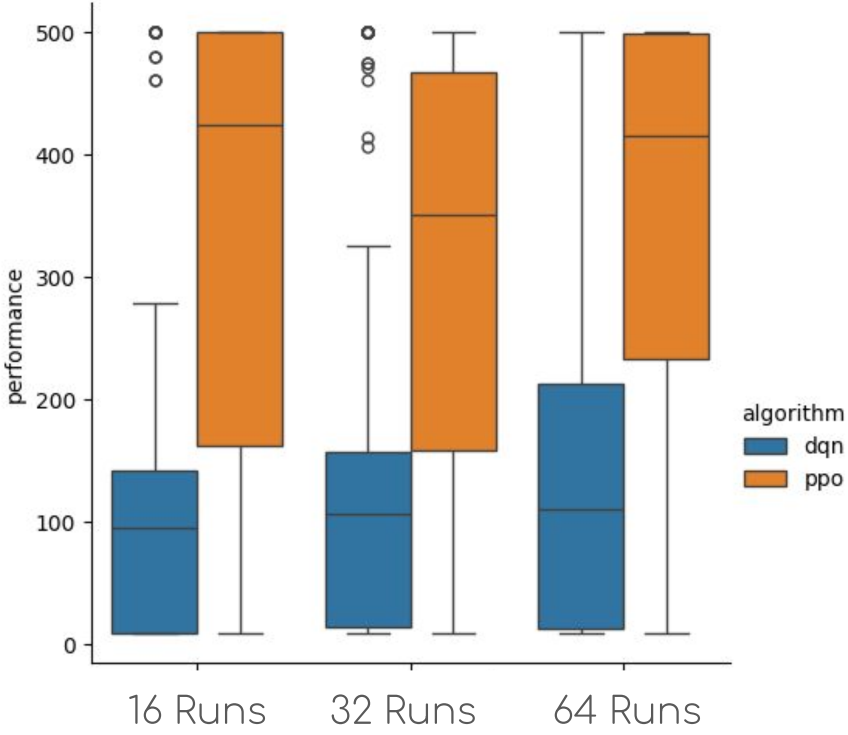
- ➔ Easily searchable, predictable performance landscape
- ➔ Much of the total configuration space has good performance
- ➔ It's clear early on if a configuration is good

Excursion: Tuning Efficiency

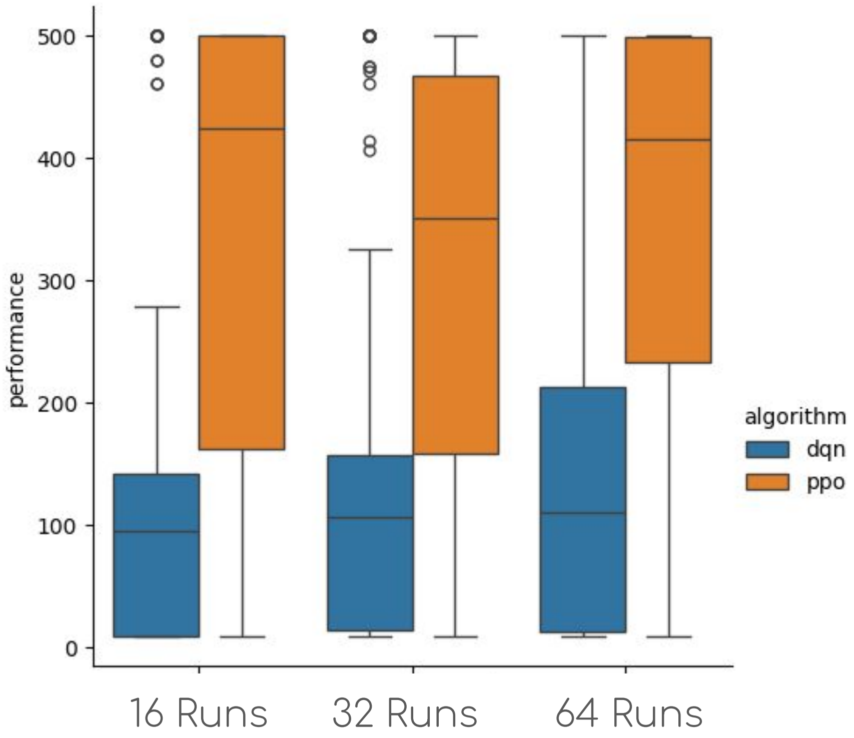
What does it mean if an algorithm is efficient to tune?

➔ Easily searchable, predictable performance landscape
➔ Much of the total configuration space has good performance
➔ High budget correlation

Example: PPO & DQN on CartPole



Example: PPO & DQN on CartPole

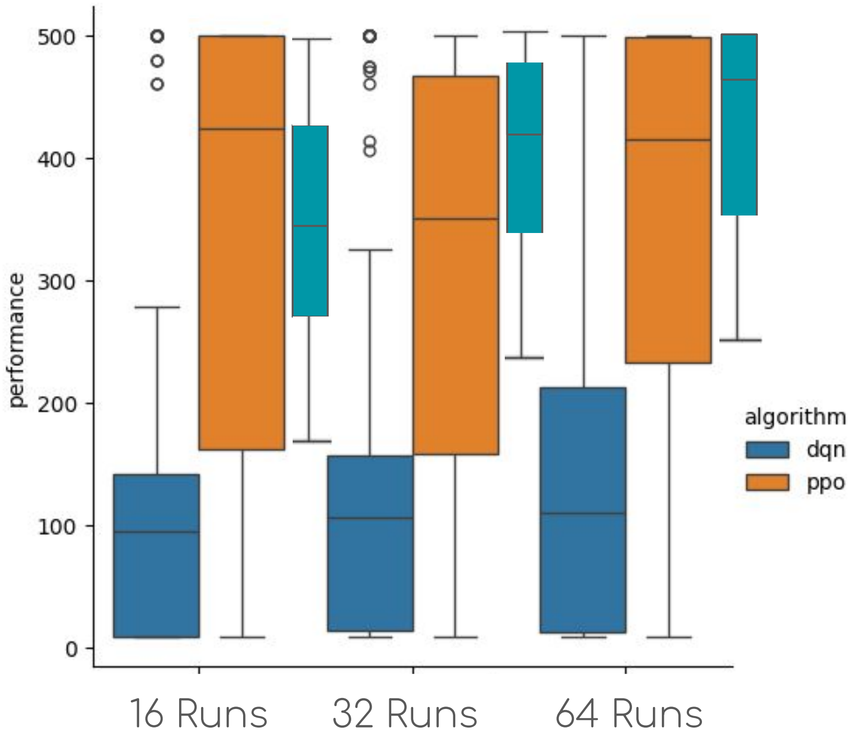


Possible takeaways:

- ➔ DQN responds predictably to search & more tuning effort
- ➔ PPO clearly has better average performance

RL algorithm researcher: "DQN is the more adaptable algorithm, but has poor default performance. If I can help it auto-adapt, I can get the best of both worlds."

Example: PPO & DQN on CartPole



Possible takeaways:

➔ DQN responds predictably to search & more tuning effort
➔ PPO clearly has better average performance

RL algorithm researcher: "DQN is the more adaptable algorithm, but has poor default performance. If I can help it auto-adapt, I can get the best of both worlds."

What do we actually want to know?

AGI enthusiast: “I want to know if this algorithm can solve all problems at once“

Translation: algorithm flexible, no environment limits, budget is no issue

What do we actually want to know?

AGI enthusiast: "I want to know if this algorithm can solve all problems at once"

Translation: algorithm flexible, no environment limits, budget is no issue

Evaluation should show:

➔ Algorithm & all settings should transfer well to any setting

➔ Algorithm should perform well anywhere with a single hyperparameter configuration

What do we actually want to know?

AGI enthusiast: “I want to know if this algorithm can solve all problems at once”

Translation: algorithm flexible, no environment limits, budget is no issue

Evaluation should show:

➔ Algorithm & all settings should transfer well to any setting
High zero-shot transfer of policy & algorithm
➔ Algorithm should perform well anywhere with a single hyperparameter configuration

What do we actually want to know?

AGI enthusiast: “I want to know if this algorithm can solve all problems at once”

Translation: algorithm flexible, no environment limits, budget is no issue

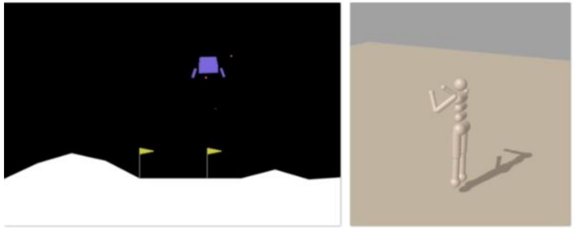
Evaluation should show:

➔ Algorithm & all settings should transfer well to any setting
High zero-shot transfer of policy & algorithm

➔ Algorithm should perform well anywhere with a single hyperparameter configuration
Good tuning outcomes in the **Algorithm Configuration** Setting

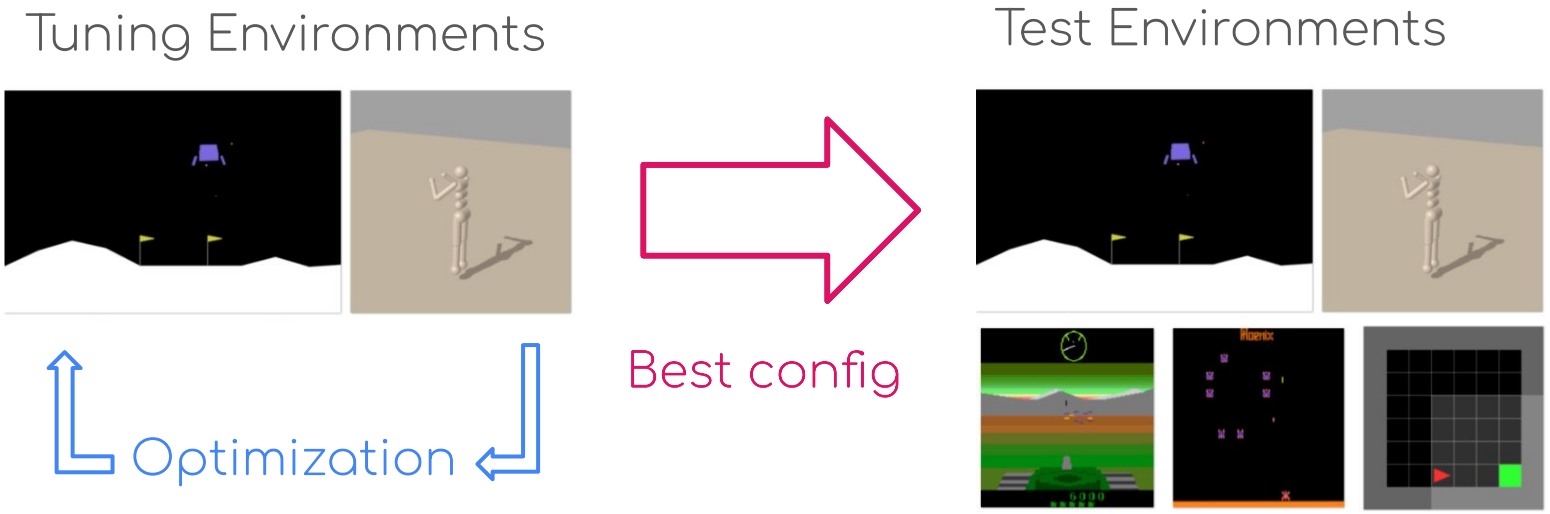
Excursion: Algorithm Configuration [Schede et al. 2022]

Tuning Environments



Optimization

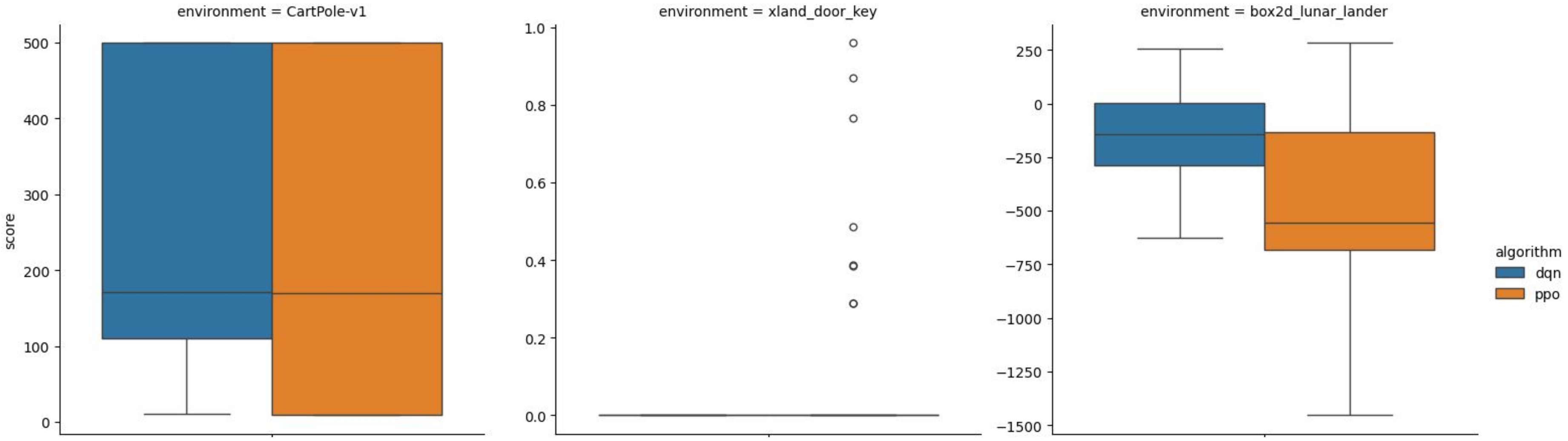
Excursion: Algorithm Configuration [Schede et al. 2022]



Example: PPO & DQN on CartPole

Tuned Environment

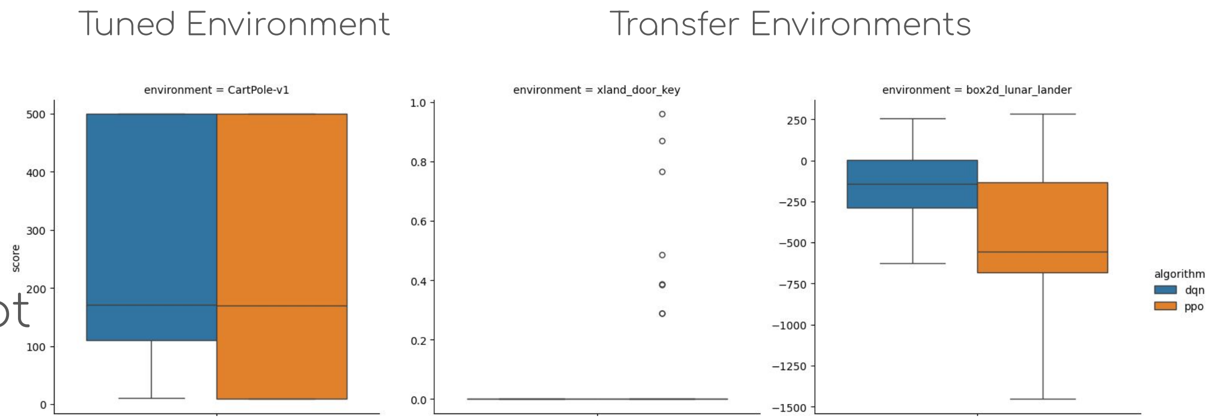
Transfer Environments



Example: PPO & DQN on CartPole

Possible takeaways:

- ➔ Both algorithms struggle in transfer
- ➔ Tuning on a single environment might not be enough



AGI enthusiast: “Currently none of these two algorithms are useful for me. I will have to try more extensive tuning or find an alternative.”

Framing 1: Complete Algorithms [Jordan et al. 2020]

Idea: All settings are considered part of the algorithm.

Framing 1: Complete Algorithms [Jordan et al. 2020]

Idea: All settings are considered part of the algorithm.

Algorithm 1	Algorithm 2
DQN	DQN
ResNet-50	3-layer MLP
lr =0.01	lr=0.005

...

Framing 1: Complete Algorithms [Jordan et al. 2020]

Idea: All settings are considered part of the algorithm.

Algorithm 1	Algorithm 2
DQN	DQN
ResNet-50	ResNet-50
Random HPO	HPO by BBO

...

Framing 1: Complete Algorithms [Jordan et al. 2020]

Idea: All settings are considered part of the algorithm.

Problems:

➔ Infinite amount of individual algorithms
➔ Not all algorithms vary in RL mechanics
➔ Difference being only e.g. network architecture or HPO can be interesting, but isn't always

Framing 1: Complete Algorithms [Jordan et al. 2020]

Idea: All settings are considered part of the algorithm.

Problems:

➔ Infinite amount of individual algorithms
➔ Not all algorithms vary in RL mechanics
➔ Difference being only e.g. network architecture or HPO can be interesting, but isn't always

Advantage: Setting is now an essential part of the comparison

Framing 2: Randomization [Bouthillier et al. 2019]

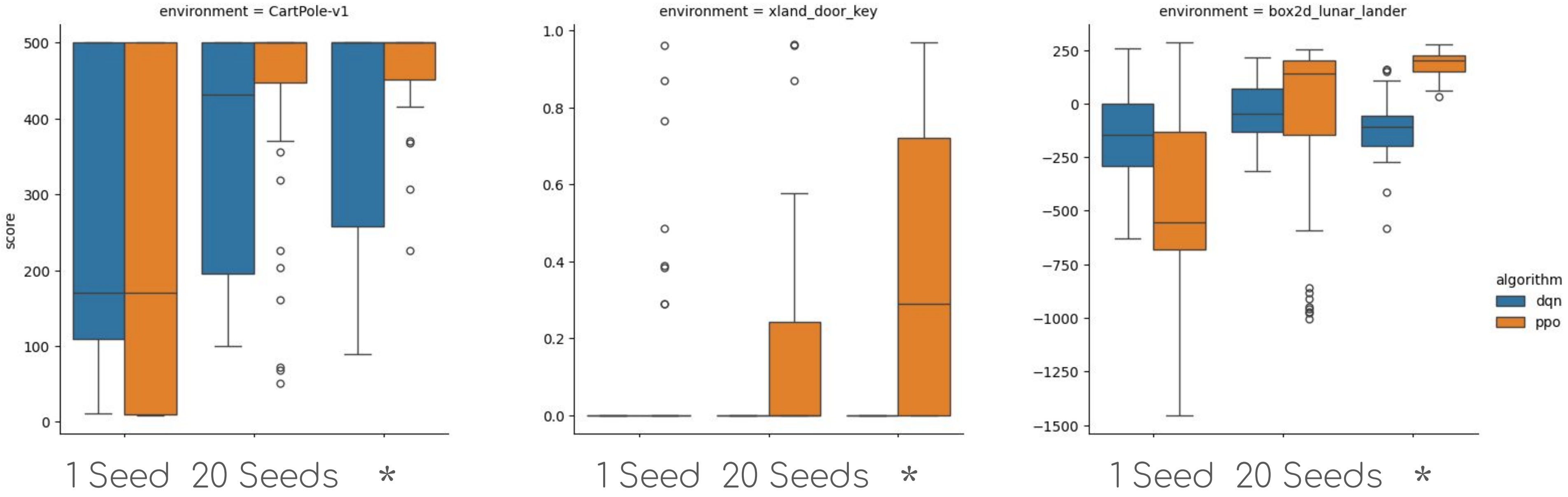
Idea: Randomize settings to lower standard error

Framing 2: Randomization [Bouthillier et al. 2019]

Idea: Randomize settings to lower standard error

Standard	More Random
Random Seeds	Random Seeds
	Random Networks
	Random n_envs
	...

Example: PPO & DQN on CartPole



* Tuned across 20 Runs with randomly sampled seed, n_envs, hidden size & activation function

Framing 2: Randomization [Bouthillier et al. 2019]

Idea: Randomize settings to lower standard error

Problems:

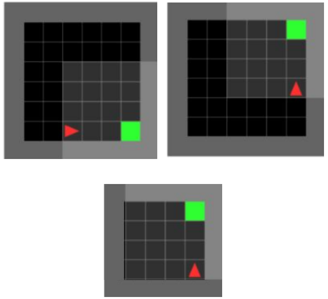
➔ Randomizing relevant factors can cause higher variance

Framing 2: Randomization [Bouthillier et al. 2019]

Idea: Randomize settings to lower standard error

Problems:

➔ Randomizing relevant factors can cause higher variance
➔ Randomizing the environment can cause results to be extremely hard to interpret



?

So What Is The Best Framing?

So What Is The Best Framing?

It doesn't exist!

So What Is The Best Framing?

It doesn't exist!

- ➔ Evaluation priorities should fit the research goals
- ➔ Exact setting and metrics depend on these priorities
- ➔ Standardized evaluation settings, HPO or metrics restrict expressiveness of our experiments

But What About Reproducibility?

But What About Reproducibility?

Bouthillier et al. 2019 distinguish between:

- ➔ Rerunning the code gives the same result
- ➔ Restaging the experiment approximately gives the same result
- ➔ The spirit of the result holds across similar experiments

But What About Reproducibility?

Bouthillier et al. 2019 distinguish between:

- ➔ Rerunning the code gives the same result
- ➔ Restaging the experiment approximately gives the same result
- ➔ The spirit of the result holds across similar experiments

More expressive evaluations make the spirit of the results clearer

So What Evaluation Should I Use Now?

So What Evaluation Should I Use Now?

- ➔ Use existing protocols as templates

So What Evaluation Should I Use Now?

- ➔ Use existing protocols as templates

  Best practices

  Standards for benchmarks

  New research on evaluation practices

So What Evaluation Should I Use Now?

- ➔ Use existing protocols as templates
- ➔ Consider non-standard metrics that support your goals

So What Evaluation Should I Use Now?

- ➔ Use existing protocols as templates
- ➔ Consider non-standard metrics that support your goals

Computational efficiency (e.g. wallclock time)

Generalizability across settings (e.g. random network architectures)

HPO metrics (e.g. tunability)

So What Evaluation Should I Use Now?

- ➔ Use existing protocols as templates
- ➔ Consider non-standard metrics that support your goals
- ➔ Show what sets your algorithm apart beyond just reward curves

Explicitly target a specific audience

Openly show Tradeoffs

Don’t be afraid of making a contribution to a specific area rather than a very general improvement