Term Project: Evaluation of Meta-Learned RL Optimizers

Analysis of **Optim4RL** and **OPEN** on Discrete and Continuous Domains

Presented by Arseniy Kan, Seong Ju Meang

AI51101 Optimization for AI

CONTENTS

Motivation

Motivation – Why Optimization in RL is Hard

Hyperparameter Constraint

Standard optimizers (Adam, RMSProp) require extensive hyperparameter tuning and/or rely on human expertise

Non-IID Data

Traditional optimizers designed for SL assume *Independent and Identically Distributed* data. However, in RL consists of locally correlated trajectories

Non-stationary Targets

Due to highly stochastic agent-environment interactions, the agent-gradients have high bias and variance,

Meta-Learned Optimizer

Instead of relying on human design, meta-learning trains optimizers to discover update rules from data. Although models like VeLO succeed in Supervised Learning, they perform poorly in RL, leaving effective RL meta-optimization an unsolved problem.

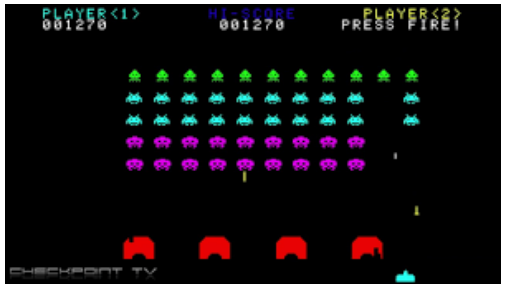
Project Scope

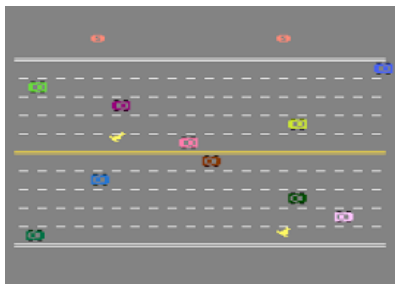
Conducted a comparative analysis between two approaches in meta-learned optimizers for RL

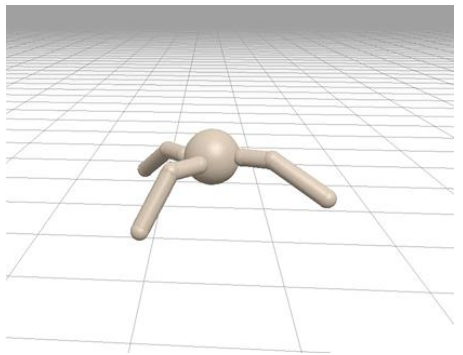
1. **Optim4RL:** *Learning to Optimize for Reinforcement Learning* (RLC August 2024)
2. **OPEN:** *Can Learned Optimization Make Reinforcement Learning Less Difficult?* (NeurIPS Dec 2024)

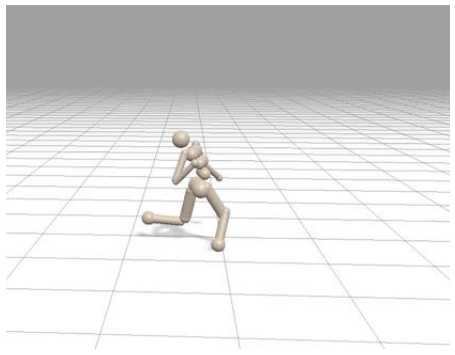
Evaluated against traditional optimizers such as Adam and RMSProp across a range of both discrete and continuous tasks

- Discrete: Atari-like environments in Gymnax (Breakout, Asterix, Space Invaders, Freeway).
- Continuous: Brax environments (Ant, Humanoid).







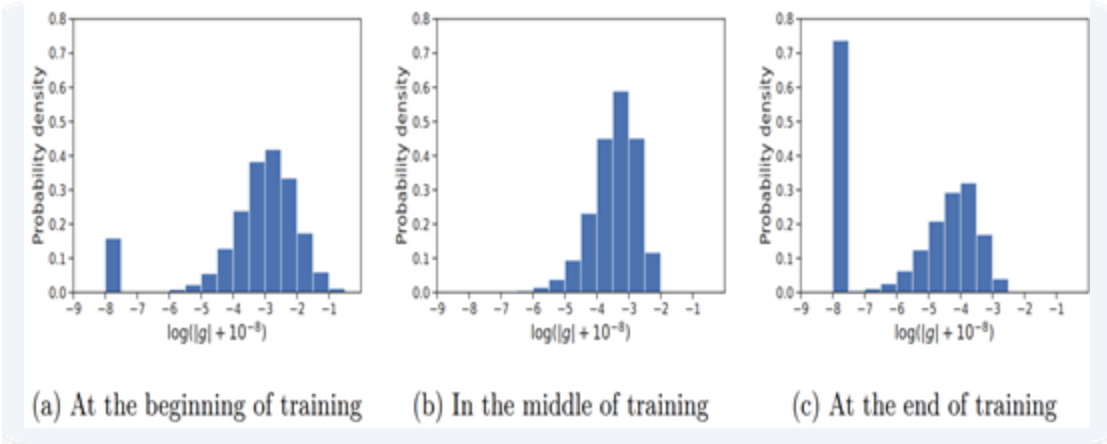


Background

Visualizing the Instability

Agent-gradient distributions in RL evolve significantly over orders of magnitude.

- Distributions shift from the beginning to the end of training.
- RL gradients are fundamentally non-IID.
- The visualization demonstrates the significant evolution of probability density over time.



(a) At the beginning of training (b) In the middle of training (c) At the end of training

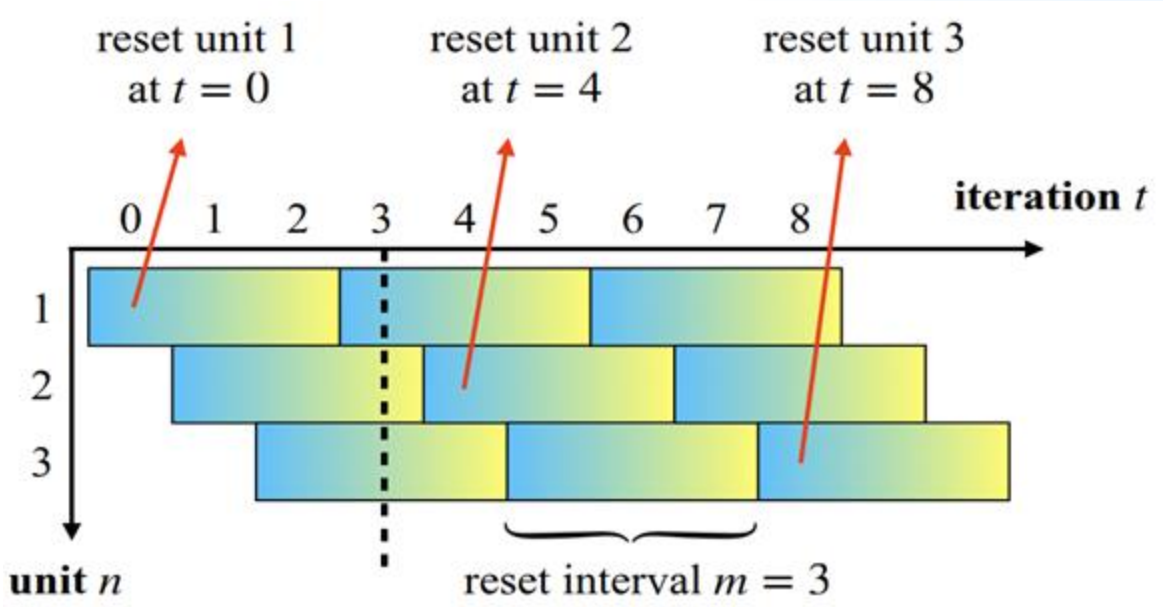
Optim4RL: Pipeline Training Strategy

Parallel Units

Train multiple agents (Policy/Value Nets) simultaneously in parallel environments to gather diverse experience.

Reset Interval

Periodically reset parameters to mix Early, Middle, and Late stage gradients.



(a) Pipeline training

Goal

Make the instantaneous gradient distribution closer to an IID-like distribution for the optimizer.

Ex :
unit 1: (Ant agent 1, task 1, optimizer state 1)
unit 2: (Ant agent 2, task 2, optimizer state 2)
unit 3: (Ant agent 3, task 3, optimizer state 3)

Optim4RL: Improving Inductive Bias

Learned Optimizers

Should automatically learn useful features, reducing reliance on human expert knowledge.

The Need

NNs are flexible but struggle with basic math like division. They need a structural hint (inductive bias).

Search Space

Constraining the search space to follow successful formulas (like Adam) prevents erratic update rules.

Optim4RL: Search space

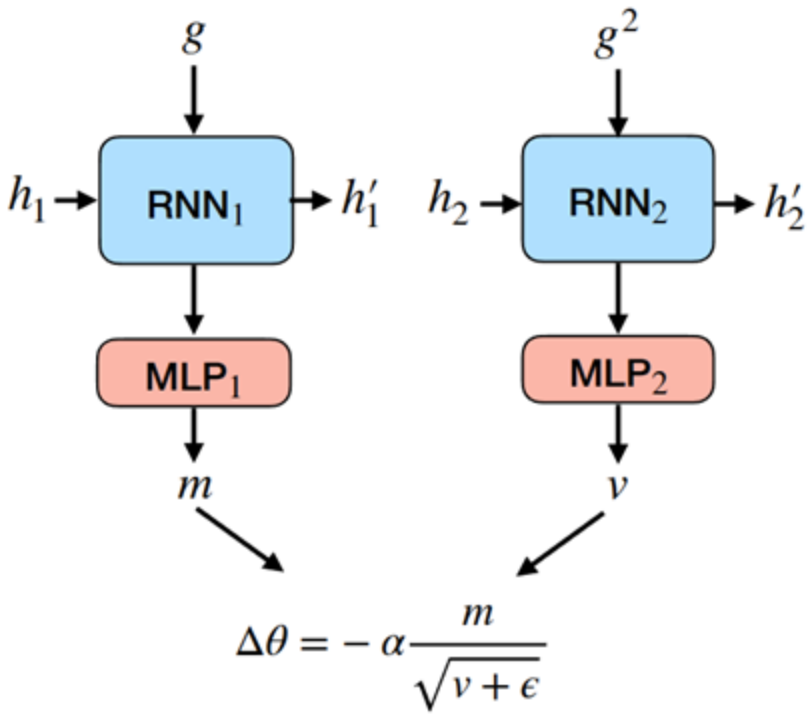
$$\Delta\theta = -\alpha \frac{m}{\sqrt{v+\epsilon}},$$

- if the hypothesis space that the optimizer need to learn is too vast, the model might learn erratic update rules, leading to poor performance
- the author solution is constrained it to follow the formula structure used by successful optimizer like Adam or RMSProp
- Constraining the search space to follow successful formulas (like Adam) prevents erratic update rules.

Optim4RL: RL-Specific Adaptation

- The RNNs that compute m and v can learn the temporal patterns of gradient changes.

- This allows them to generate sophisticated momentum values tailored to the unique characteristics of the RL environment (like non-iid gradients), giving them the potential to outperform a generic Adam optimizer.



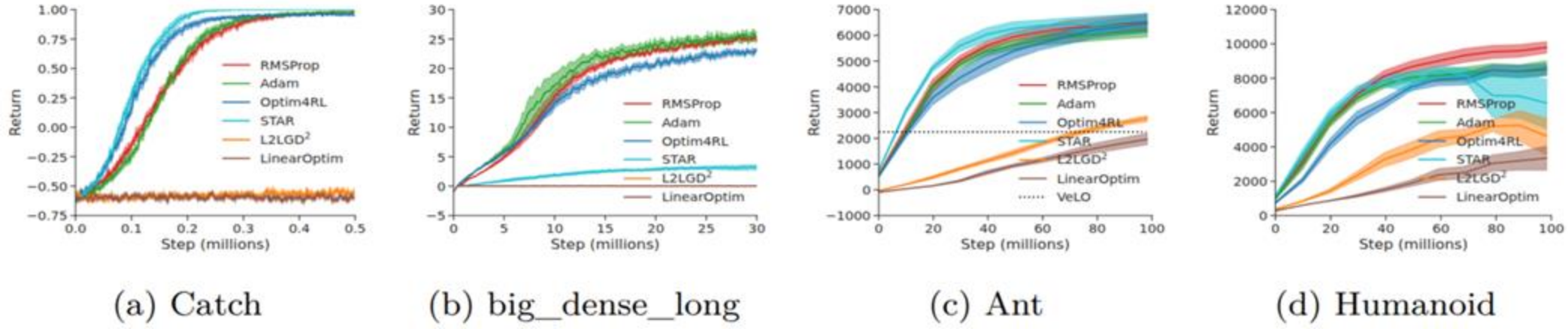
(b) The optimizer structure of Optim4RL

Optim4RL: Ablation Study: Linear optim

$$\Delta\theta = -\alpha(a \cdot g + b)$$

a,b from **dual RNN**

This ablation examines whether imposing constraints in an adaptive-optimizer-like manner actually helps



(a) Catch (b) big_dense_long (c) Ant (d) Humanoid

Optim4RL: Pipeline Training Effectiveness

Method \ Task	small_dense_long	big_dense_long	Ant	Humanoid
With Pipeline Training	32.22±0.52	23.10±0.31	6421±355	8440±364
W.o. Pipeline Training	30.64±0.69	22.47±0.51	5038±235	6557±1055

OPEN

- **L**earned **O**ptimization for **P**lasticity, **E**xploration and **N**on-stationarity.

- A meta-optimizer designed with **explicit input features** targeting these hurdles, trained directly inside RL environments.

OPEN Results: Single-Task Performance

MinAtari Benchmark

Tested on Breakout, Asterix, Space Invaders, Freeway.

- OPEN significantly **outperformed all baselines** (Adam, Lion, VeLO) in 3 out of 4 tasks.
- Massive performance gains in *Asterix* and *Breakout*.
- Validates the design choice of explicitly targeting RL difficulties.

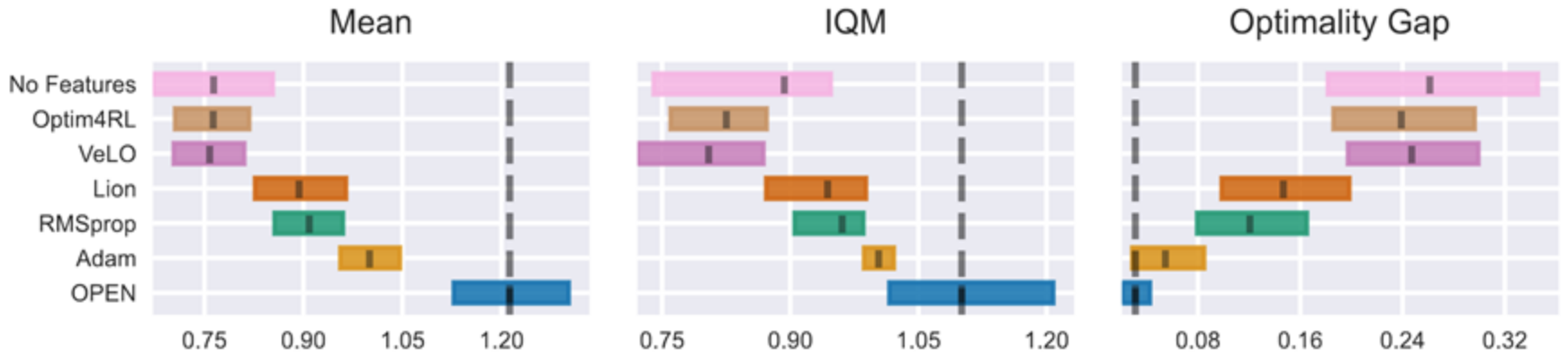
Brax Ant (Continuous)

A continuous control task where exploration is less critical.

- OPEN achieved parity with Adam and Lion.
- **Key Takeaway:** While designed for hard exploration, OPEN remains robust and stable in standard tasks, acting as a reliable "drop-in" replacement.

OPEN Results: Multi-Task Learning

OPEN was meta-trained on 4 distinct MinAtar environments **simultaneously**.



Key Finding 1: Dominance

OPEN was the **only** learned optimizer to achieve an average score higher than Adam (Baseline = 1.0) across the multi-task suite.

Key Finding 2: Failure of Others

Other learned optimizers (Optim4RL, VeLO) consistently performed **worse** than Adam in this setting, failing to generalize across task dynamics.

Conclusion: OPEN demonstrates potential as a "Foundation Optimizer" capable of internalizing update rules that work across diverse environments.

Comparison: Design Philosophy

Feature	Optim4RL (Previous Work)	OPEN (This Paper)
Input Features	**Minimalist:** Uses only gradients. Relies on RNN to learn representations implicitly.	**Explicit:** Inputs features for Non-stationarity, Plasticity, and Multi-timescale momentum.
Exploration	**Implicit:** Relies on policy entropy or action noise.	**Explicit Control:** Outputs learned noise (δ_i) to manage exploration actively.
Update Type	Deterministic updates.	Stochastic updates (Mean + Noise).

Comparison: Training & Stability

Optim4RL: Feasibility Proof

- **Method:** Meta-gradients (Bilevel optimization).
- **Issue:** High variance and non-i.i.d gradients lead to instability.
- **Scale:** Limited to smaller networks/horizons.
- **Role:** Proved learned optimizers *can* work in RL.

OPEN: Foundation Model

- **Method:** Evolution Strategies (ES) via JAX.
- **Advantage:** Gradient-free, stable, highly parallelizable.
- **Scale:** Foundation-level performance with ~1 GPU-month.
- **Role:** Proves learned optimizers can be *scalable tools*.

Summary

- **Targeted Design:** OPEN succeeds by explicitly encoding RL difficulties (Non-stationarity, Plasticity, Exploration).
- **Superior Generalization:** It is the only learned optimizer outperforming Adam in multi-task settings and transferring 0-shot to complex environments like Craftax.
- **Evolution over Optim4RL:** Moves beyond proof-of-concept by solving stability issues via ES and introducing explicit exploration control.

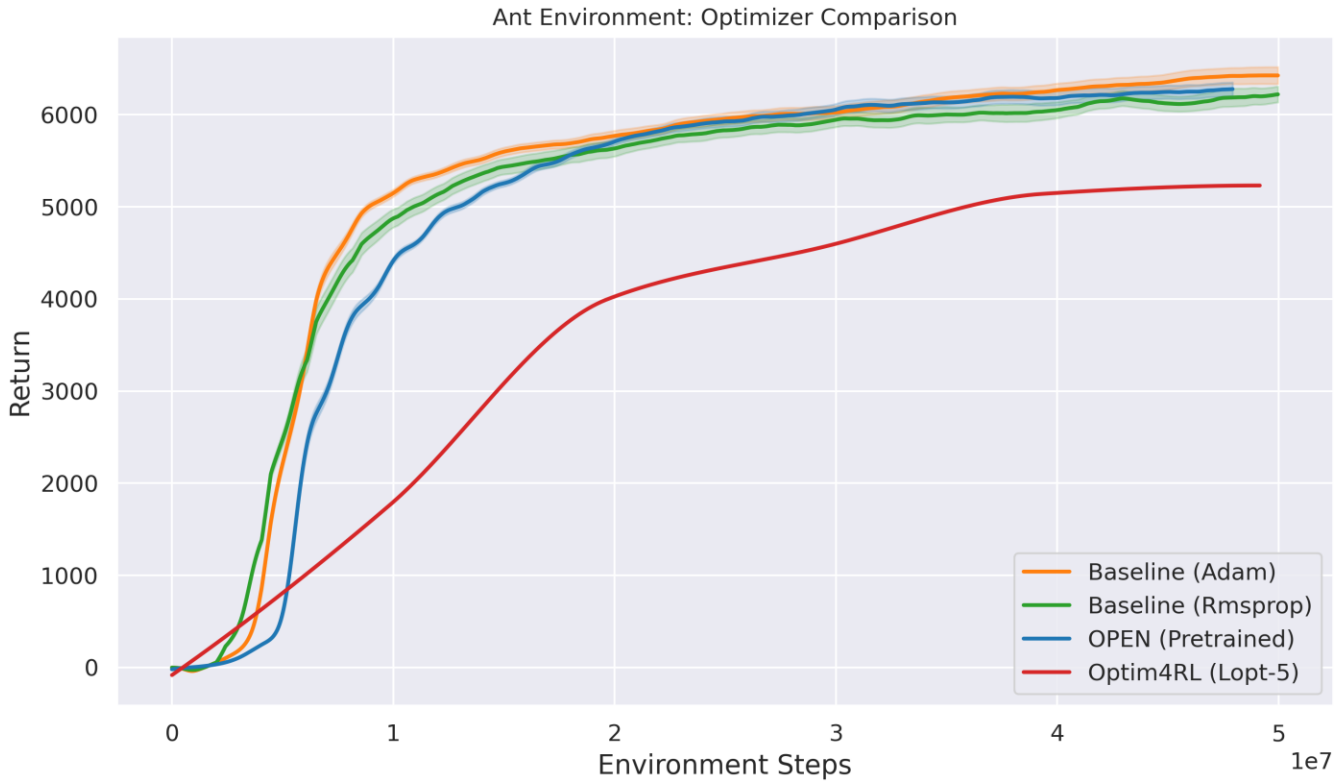
Term Project Results

Project Results - Overview

- **Reproduced experiments from both papers.** Compared 4 optimizers in Ant environment:
  - Adam (baseline)
  - RMSProp
  - OPEN
  - Optim4RL
- **Evaluated OPEN's Multi-Task Performance Claim:**
  - Checked Multi-OPEN performance vs. Adam, RMSProp in 4 Atari like games and 1 Brax Environment
- **Is it truly hyperparameter free?**
  - Optim4RL turned to be very dependent on meta hyperparameters.
- **Policy Collapse Experiment**
  - Optim4RL
  - PPO with Adam vs Tuned Adam (Ant)
  - PPO with optim4RL vs Optim4RL with L2 regularization(Ant)

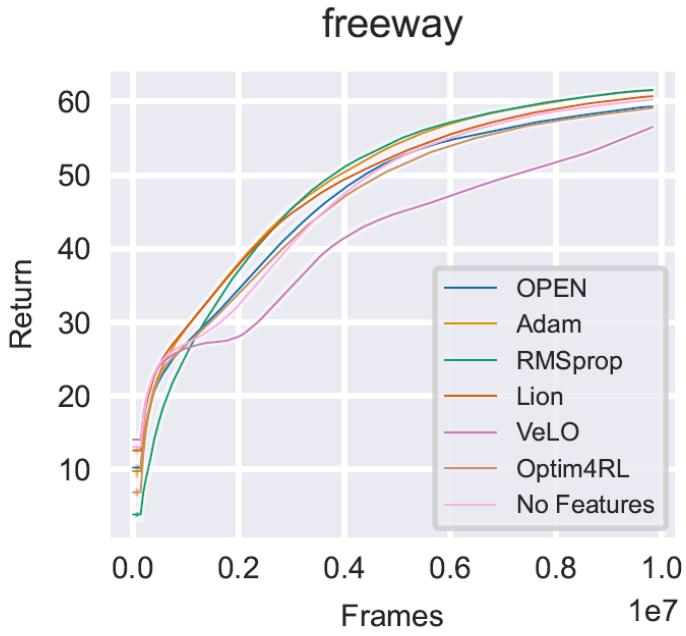
Project Results – Ant Environment Performance

- **Task:** Logistic Regression on **12 LIBSVM benchmark** problems.
- **Baseline:** Standard Adam with a fully grid-searched, hand-tuned learning rate.
- **Result:** D-Adapt Adam (green) **matched or exceeded** the performance of the hand-tuned Adam (gray) on all 12 datasets—with zero manual tuning.

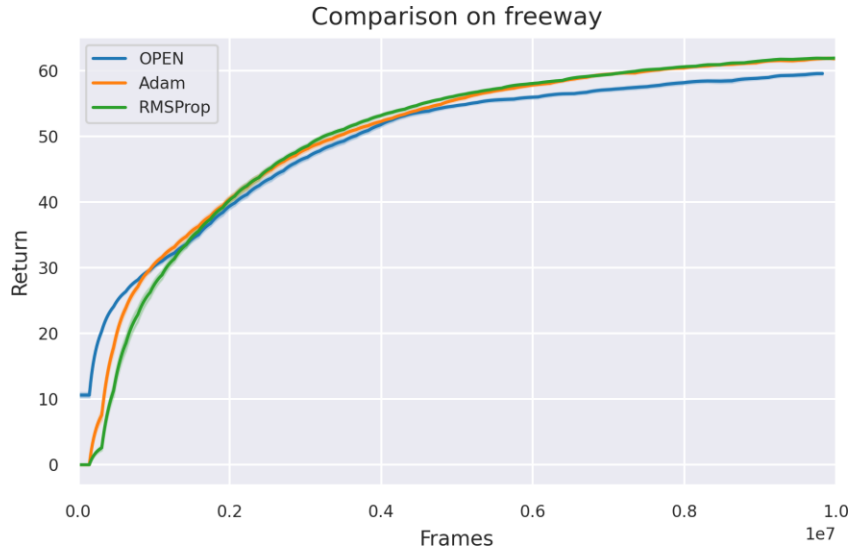


Project Results – Multi-task Performance

- **Experiment:** Tested OPEN pretrained on 3 Atari-like environments simultaneously.
- **Baseline:** Adam and RMSProp tuned for each specific task
- **Result (Freeway):** Our experiment validates the result from the paper. OPEN performs similarly to Adam and RMSprop



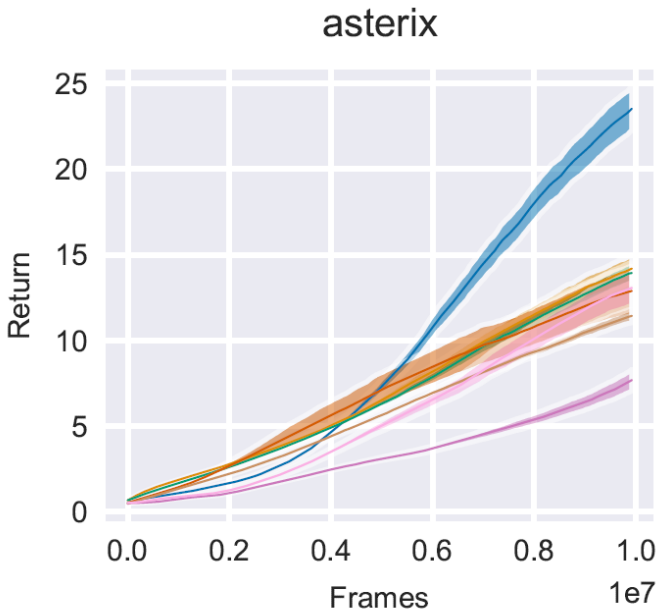
From the paper



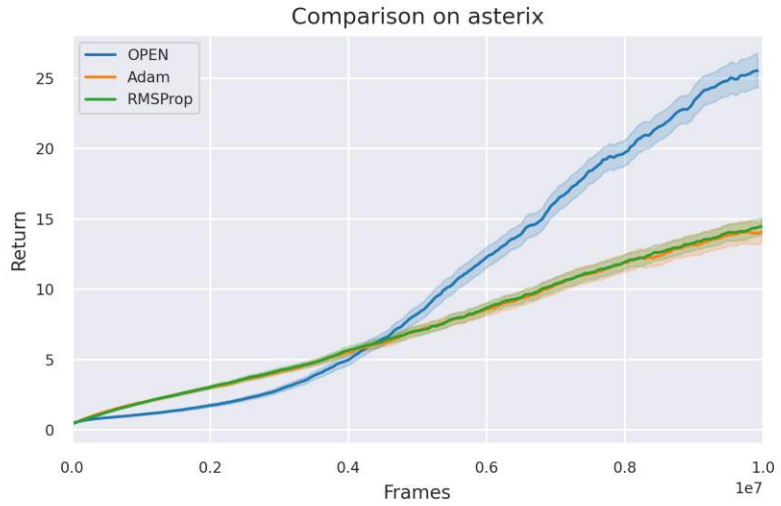
Our results

Project Results – Multi-task Performance

- **Experiment:** Tested OPEN pretrained on 3 Atari-like environments simultaneously.
- **Baseline:** Adam and RMSProp tuned for each specific task
- **Result (Asterix):** Our experiment validates the result from the paper. OPEN significantly outperforms baseline.



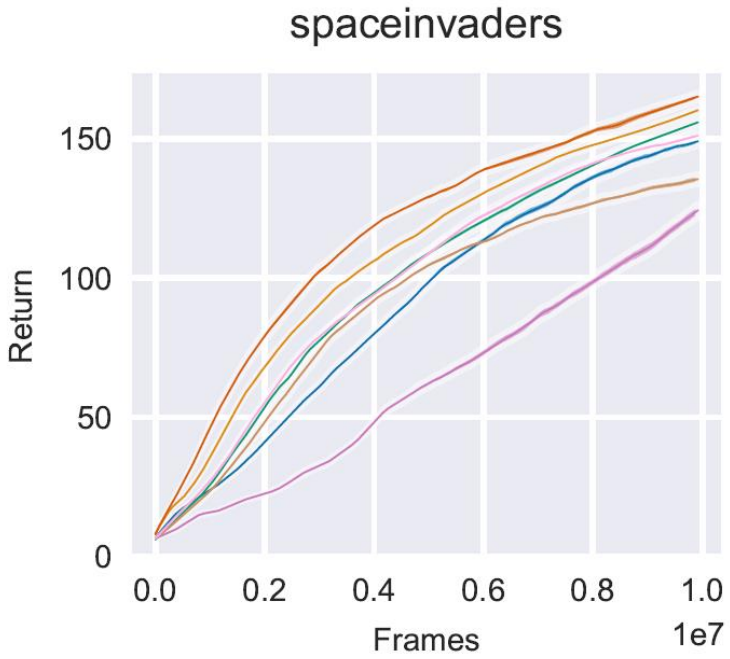
From the paper (OPEN is blue)



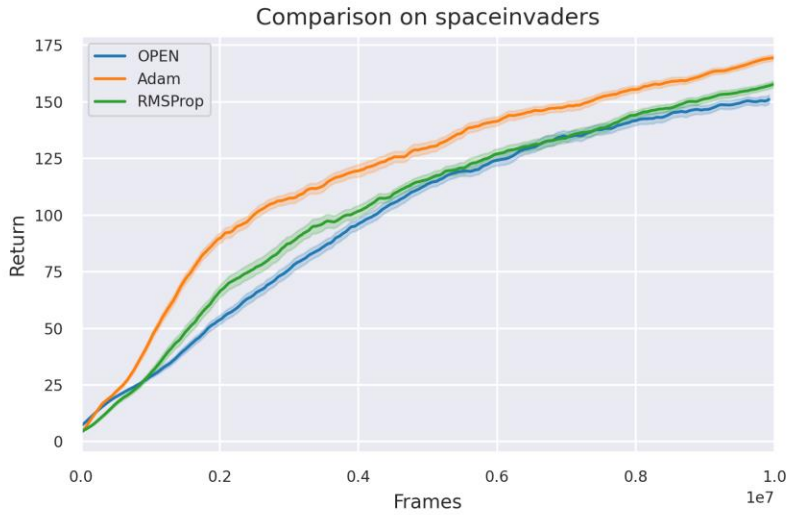
Our results

Project Results – Multi-task Performance

- **Experiment:** Tested OPEN pretrained on 3 Atari-like environments simultaneously.
- **Baseline:** Adam and RMSProp tuned for each specific task
- **Result (Space Invaders):** Our experiment validates the result from the paper. OPEN underperforms baseline.



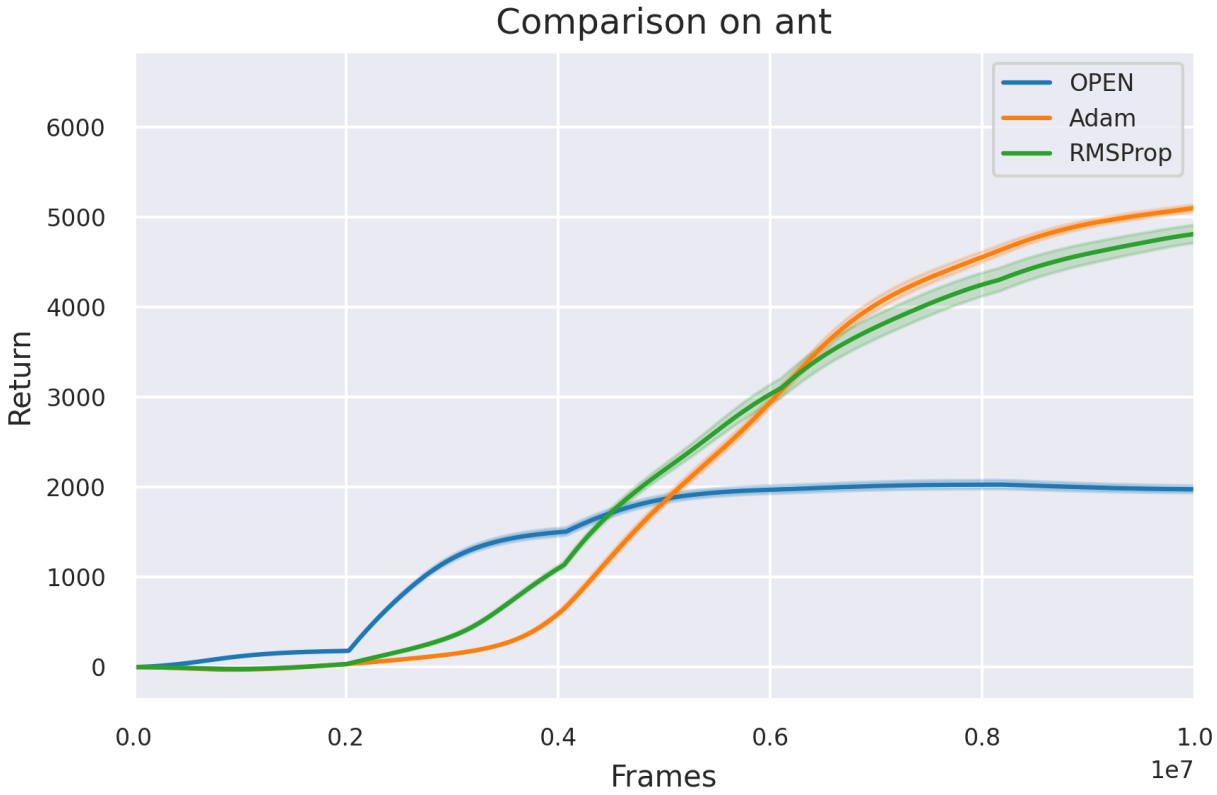
From the paper (OPEN is blue)



Our results

Project Results – Multi-task Performance (additional test)

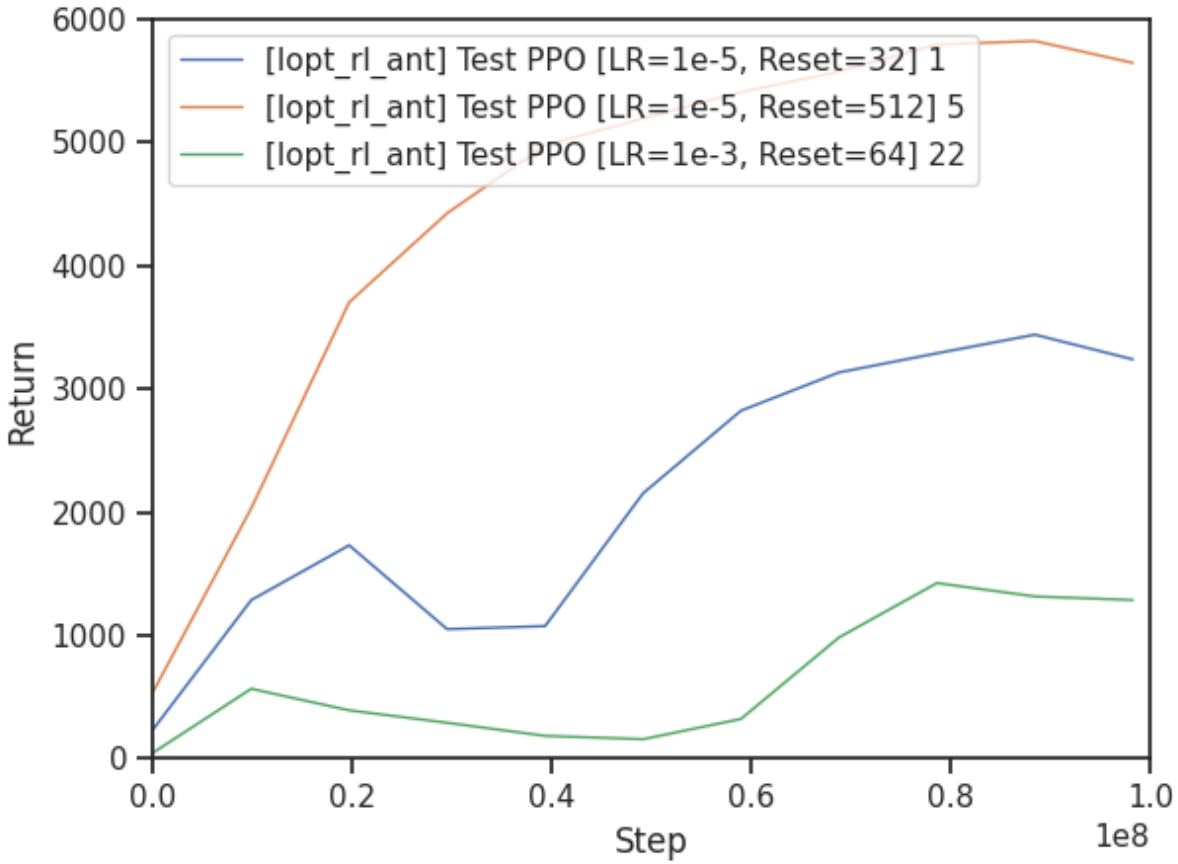
- **Experiment:** We also wanted to test how well would OPEN generalize to an out-of-distribution task like Ant
- **Baseline:** Adam and RMSProp tuned for Ant env
- **Result (Ant):** OPEN fails to generalize to a different domain task like continuous Brax environment.



Our results

Project Results – The "Meta-Tuning" Paradox

- **The Broken Promise:**
  - **Theory:** The core motivation of learned optimizers is to automate the optimization process and reduce the burden of manual hyperparameter tuning.
  - **Reality:** The tuning burden is not eliminated; it is merely **shifted** one level up to the *meta-training* phase.
- **Evidence: The "Hidden" Grid Search**
  - **Observation:** In reproducing **Optim4RL**, we found that the "tuning-free" optimizer actually relies on extensive meta-hyperparameter sweeps.
  - The authors' own configuration file includes a massive search space (5 X 5 X 5) involving meta-learning rates and reset intervals.
- **Sensitivity**
  - **Sensitivity Analysis:** As shown in the plot (Ant-v2 task), changing *only* the meta-hyperparameters results in **drastically** different outcomes

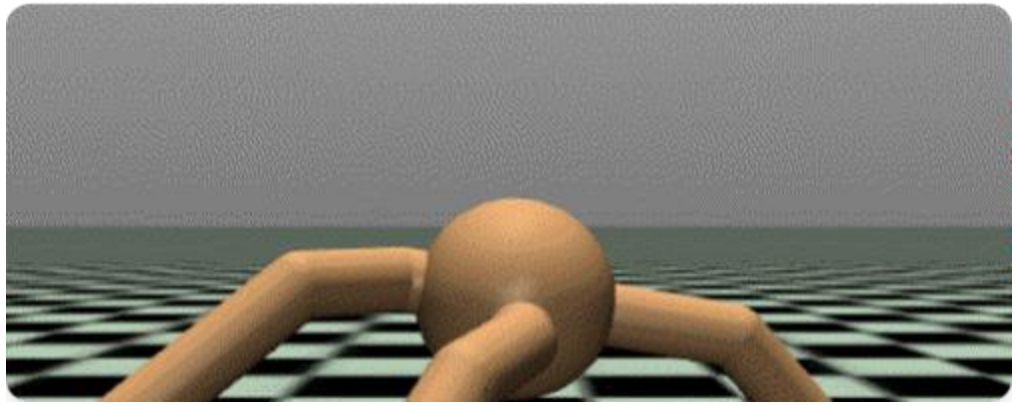


Experiment: Policy Collapse

Policy collapse : a phenomenon where a well learned policy collapses later and cannot recover

- **The Issue:** Default Adam settings are not appropriate for RL tasks. Asymmetric momentum (beta1 0.9, beta2 0.99) creates excessive updates, leading to policy collapse.
- **The Fix:** Setting beta1 == beta2 allows changes at similar speeds, mitigating policy collapse.
- **Regularization:** L2 regularization also has a significant effect in mitigating policy collapse phenomena.

Testing on Non-Stationary Tasks



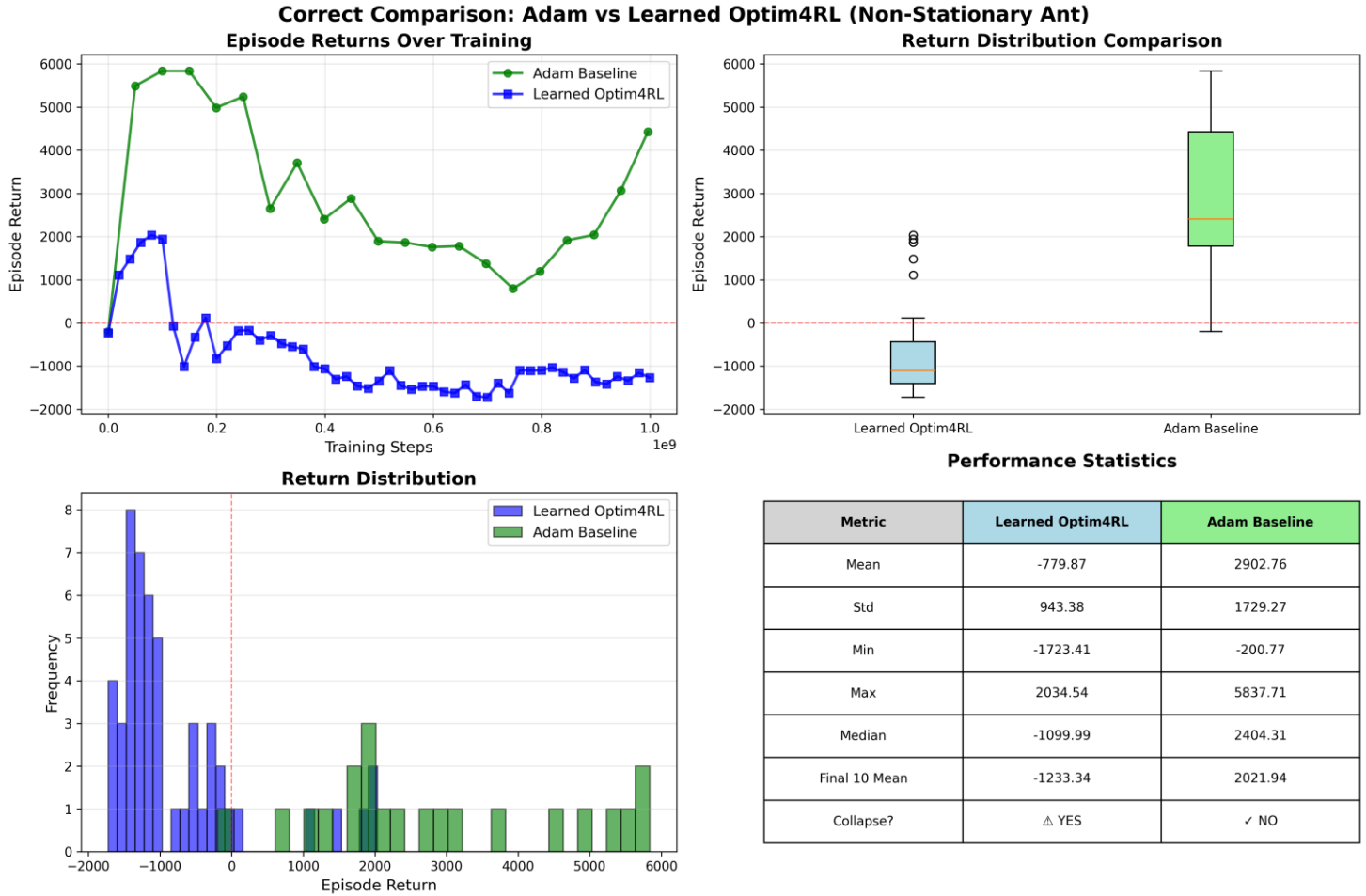
Ant Environment

Simulated actuator wear and ground friction changes to test robustness.

Humanoid Environment

Introduced slippery ground and muscle fatigue to simulate changing dynamics.

Experiment: PPO with Optim4RL vs default Adam (Ant task)

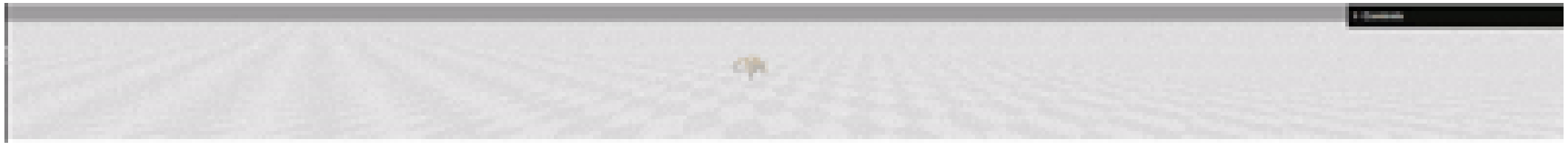


Performance Statistics

Metric	Learned Optim4RL	Adam Baseline
Mean	-779.87	2902.76
Std	943.38	1729.27
Min	-1723.41	-200.77
Max	2034.54	5837.71
Median	-1099.99	2404.31
Final 10 Mean	-1233.34	2021.94
Collapse?	⚠ YES	✓ NO

Experiment: PPO with Optim4RL vs default Adam (Ant task)

Optim4RL optimizer :Ant task



Default adam

Experiment: PPO Tunned Adam vs Optim4RL with L2

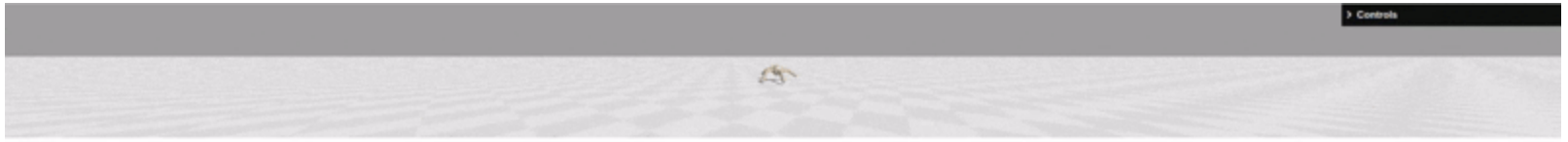
Mean Return	1774.12	-137.46	-1911.59
Std Dev	596.83	472.21	-124.62
Min Return	-87.12	-866.45	-779.32
Max Return	3016.49	1064.83	-1951.66
Median	1810.79	-335.22	-2146.01
Coef. of Var.	0.336	3.435	+3.099
Severity Score	0/15	13/15	+13
Policy Collapse?	✓ NO	⚠ YES	

Experiment: PPO Tunned Adam vs Optim4RL with L2

Optim4rl : L2 reguarization

Tunned Adam

Experiment: learned stationary task optimizer apply to non stationary task



Discussion

Discussion – OPEN vs. Optim4RL

- **Context:** OPEN (July 9, 2024) is published after Optim4RL (February 3, 2023) paper and partially builds on top of it.
- **Result:** OPEN outperforms Optim4RL across diverse tasks and both in single- and multi-task performance. SOTA in the narrow, evolving field of meta-learned optimizers for RL.
- **Reason:** Unlike Optim4RL, which relies on a rigid Adam-like structure to stabilize gradients , OPEN employs **learned stochasticity** and **specialized input features** (e.g., dormancy) to explicitly target RL pathologies like plasticity loss and exploration.

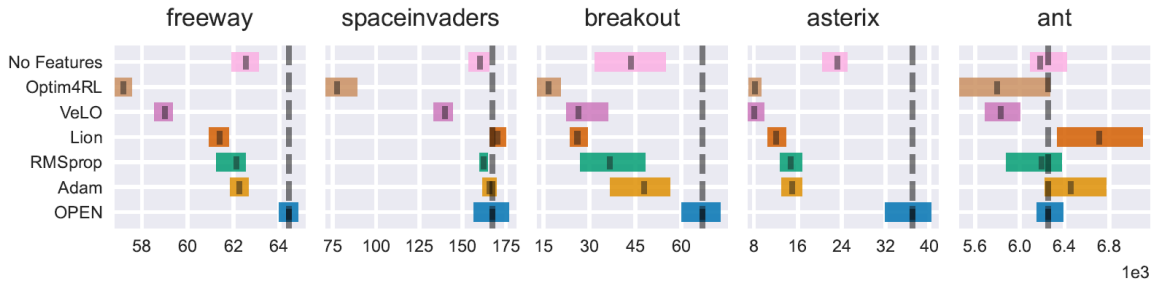
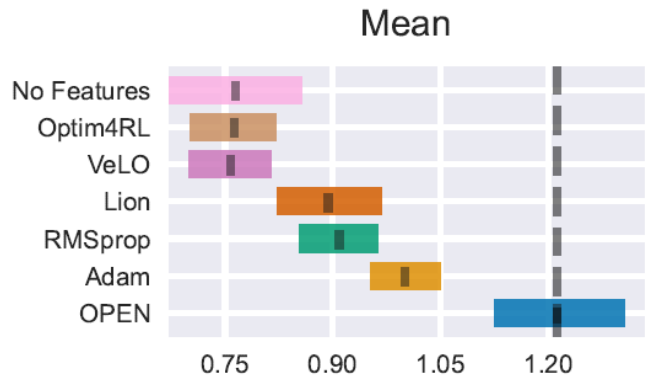


Figure 2: IQM of final returns for the five single-task training environments, evaluated over 16 random environment seeds. We plot 95% stratified bootstrap confidence intervals for each environment.

Single-task performance

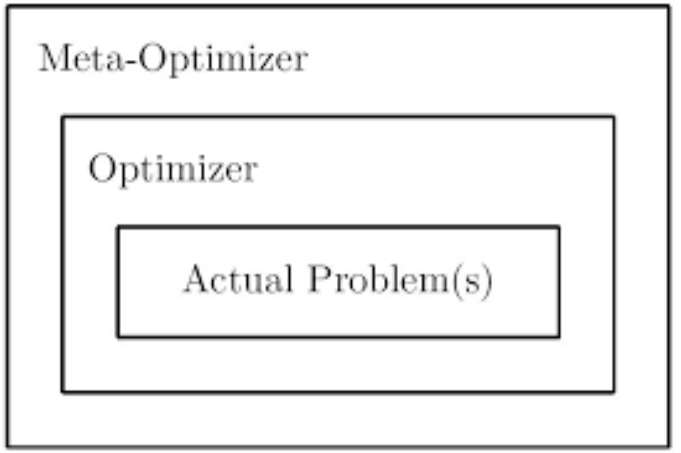


Multi-task performance

Conclusion

Summary

- **Performance disparity**
  1. Meta-learned optimizers consistently outperformed baselines in **discrete control tasks** (Atari/Gymnax).
  2. However, they struggled to show consistent gains over Adam in **continuous control tasks** (Brax), often matching or underperforming the baseline.



- **Barriers to Practical Adoption**
  1. **Computational Overhead:** Meta-training introduces significant computational cost compared to standard hyperparameter tuning.
  2. **The "Tuning Shift":** The burden of tuning is not eliminated but shifted to the **meta-level** (e.g., choosing meta-learning rates and schedules remains difficult). No Free Lunch?
  3. **Generalization Gap:** Learned update rules excel in in-distribution tasks but suffer performance degradation in out-of-distribution (OOD) environments.
  4. **Baseline Robustness:** Standard Adam remains a highly competitive and robust baseline, often converging with default or minimally tuned hyperparameters.

Conclusion: Meta-learned optimization is a promising frontier for addressing RL-specific pathologies like non-stationarity.

However, current methods struggle replace hand-designed optimizers like Adam and its variants.

Q&A Session

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

2 0 0 7

THANK YOU