MUNI

# Discovered Policy Optimisation

**Hugo Adamove**

IV125 Formela lab seminar

June 9, 2025

# Motivation

# Trust Region Learning and PPO

## Trust Region Learning (TRL) Update

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[A_{\pi_i}(s,a)\right] - C_{\pi_i} D_{\mathrm{KL}}(\pi_i, \pi'), \tag{1}$$

where

- $C_{\pi_i}$ is a constant dependent on $\pi_i$ only,
- $\Pi$ is the set of all possible policies.

# Trust Region Learning and PPO

## Trust Region Learning (TRL) Update

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[A_{\pi_i}(s,a)\right] - C_{\pi_i} D_{\mathrm{KL}}(\pi_i, \pi'), \tag{1}$$

where

- $C_{\pi_i}$ is a constant dependent on $\pi_i$ only,
- $\Pi$ is the set of all possible policies.

Schulman et al. [1] showed that this update guarantees monotonic improvement in the expected return $\eta$:

$$\eta(\pi_{i+1}) \geq \eta(\pi_i)$$

# Trust Region Learning and PPO

## Trust Region Learning (TRL) Update

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[A_{\pi_i}(s,a)\right] - C_{\pi_i} D_{\mathrm{KL}}(\pi_i, \pi'), \tag{1}$$

where

- $C_{\pi_i}$ is a constant dependent on $\pi_i$ only,
- $\Pi$ is the set of all possible policies.

Schulman et al. [1] showed that this update guarantees monotonic improvement in the expected return $\eta$:

$$\eta(\pi_{i+1}) \geq \eta(\pi_i)$$

## PPO Update

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[L_{\text{clip}}\right], \tag{2}$$

where

$$L_{\text{clip}} = \min\left(A_{\pi_i}(s,a)\, r(\pi'),\, A_{\pi_i}(s,a)\, \text{CLIP}\Big(r(\pi'), 1-\epsilon, 1+\epsilon\Big)\right).$$

# PPO's "Paradox"

- PPO stabilizes training and achieves SOTA performance across various tasks (Henderson et al., 2018; Berner et al., 2019) [2].

# PPO's "Paradox"

- PPO stabilizes training and achieves SOTA performance across various tasks (Henderson et al., 2018; Berner et al., 2019) [2].
- PPO's theoretical foundations remain limited to heuristic "proof by analogy" rather than a rigorous guarantee of monotonic improvement.

# PPO's "Paradox"

- PPO stabilizes training and achieves SOTA performance across various tasks (Henderson et al., 2018; Berner et al., 2019) [2].
- PPO's theoretical foundations remain limited to heuristic "proof by analogy" rather than a rigorous guarantee of monotonic improvement.
- Despite being based on TRL principles, PPO fails to constrain update size, violating core TRL principles (Wang et al., 2020; Engstrom et al., 2020) [3].

# PPO's "Paradox"

- PPO stabilizes training and achieves SOTA performance across various tasks (Henderson et al., 2018; Berner et al., 2019) [2].
- PPO's theoretical foundations remain limited to heuristic "proof by analogy" rather than a rigorous guarantee of monotonic improvement.
- Despite being based on TRL principles, PPO fails to constrain update size, violating core TRL principles (Wang et al., 2020; Engstrom et al., 2020) [3].
- How can PPO's empirical success be theoretically justified?

# Mirror Learning

# Mirror Learning

## Mirror Learning: A Unifying Framework of Policy Optimisation

***Jakub Grudzien, Christian A Schroeder De Witt, Jakob Foerster*** *Proceedings of the 39th International Conference on Machine Learning*, PMLR 162:7825-7844, 2022.

### Abstract

Modern deep reinforcement learning (RL) algorithms are motivated by either the general policy improvement (GPI) or trust-region learning (TRL) frameworks. However, algorithms that strictly respect these theoretical frameworks have proven unscalable. Surprisingly, the only known scalable algorithms violate the GPI/TRL assumptions, e.g. due to required regularisation or other heuristics. The current explanation of their empirical success is essentially "by analogy": they are deemed approximate adaptations of theoretically sound methods. Unfortunately, studies have shown that in practice these algorithms differ greatly from their conceptual ancestors. In contrast, in this paper, we introduce a novel theoretical framework, named Mirror Learning, which provides theoretical guarantees to a large class of algorithms, including TRPO and PPO. While the latter two exploit the flexibility of our framework, GPI and TRL fit in merely as pathologically restrictive corner cases thereof. This suggests that the empirical performance of state-of-the-art methods is a direct consequence of their theoretical properties, rather than of aforementioned approximate analogies. Mirror learning sets us free to boldly explore novel, theoretically sound RL algorithms, a thus far uncharted wonderland.

Figure: Mirror Learning paper [4].

# Mirror Learning (cont.)

## Mirror Learning Update

$$\pi_{i+1} = \arg \max_{\pi' \in \mathcal{N}(\pi_i)} \mathbb{E}_{s \sim \pi_i} \left[ A_{\pi_i}(s, a) \right] - \mathcal{D}_{\pi_i}(\pi'), \tag{3}$$

- $\mathcal{N}(\pi_i)$: The neighborhood of the current policy $\pi_i$, which defines the region around the policy in which the update is allowed.
- $\mathcal{D}_{\pi_i}(\pi')$: The drift function, which measures the "distance" or cost of updating the current policy $\pi_i$ to the new policy $\pi'$.

# Mirror Learning (cont.)

## The Fundamental Theorem of Mirror Learning

If the drift function satisfies the following conditions:

- **Non-negativity**: $\mathcal{D}_{\pi_i}(\pi) \geq 0$ for all policies $\pi$ and $\pi_i$,
- **Zero drift at current policy**: $\mathcal{D}_{\pi_i}(\pi_i) = 0$,
- **Zero gradient at current policy**: $\nabla_{\pi_i}\mathcal{D}_{\pi_i}(\pi_i) = 0$, the gradient of the drift function is zero when $\pi = \pi_i$, i.e., "gentle" for small changes.

Then the Mirror Learning algorithm attains the monotonic improvement property $\eta(\pi_{i+1}) \geq \eta(\pi_i)$ and as $i \to \infty$, $\eta(\pi_i) \to \eta(\pi^*)$. [4].

# Mirror Learning & GPI

## (Reminder) Mirror Learning Update

$$\pi_{i+1} = \arg \max_{\pi' \in \mathcal{N}(\pi_i)} \mathbb{E}_{s \sim \pi_i, a \sim \pi'} \left[ A_{\pi_i}(s, a) \right] - \mathcal{D}_{\pi_i}(\pi'),$$

## Generalized Policy Iteration

If we set $\mathcal{D}_{\pi_i}(\pi) \equiv 0$ and $\mathcal{N}(\pi_i) \equiv \Pi$ we obtain:

$$\pi_{i+1} = \arg \max_{\pi' \in \Pi} \mathbb{E}_{s \sim \pi_i, a \sim \pi'} \left[ A_{\pi_i}(s, a) \right] - 0,$$

# Mirror Learning & GPI

## (Reminder) Mirror Learning Update

$$\pi_{i+1} = \arg \max_{\pi' \in \mathcal{N}(\pi_i)} \mathbb{E}_{s \sim \pi_i, a \sim \pi'} \left[ A_{\pi_i}(s, a) \right] - \mathcal{D}_{\pi_i}(\pi'),$$

## Generalized Policy Iteration

If we set $\mathcal{D}_{\pi_i}(\pi) \equiv 0$ and $\mathcal{N}(\pi_i) \equiv \Pi$ we obtain:

$$\pi_{i+1} = \arg \max_{\pi' \in \Pi} \mathbb{E}_{s \sim \pi_i, a \sim \pi'} \left[ A_{\pi_i}(s, a) \right] - 0,$$

Which is equivalent to General Policy Iteration (GPI) update:

$$\pi_{i+1} = \arg \max_{\pi' \in \Pi} \mathbb{E}_{s \sim \pi_i, a \sim \pi'} \left[ Q_{\pi_i}(s, a) \right],$$

Since $A_{\pi_i}(s, a) = Q_{\pi_i}(s, a) - V_{\pi_i}(s)$

# Mirror Learning & PPO

Recall that PPO update is theoretically defined as:

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[ L_{\text{CLIP}} \right]$$

where

$$L_{\text{CLIP}} = \mathbb{E}_{a\sim\pi} \left[ \min \left( r(\pi') A_\pi(s,a),\ \text{CLIP}\Big(r(\pi'), 1 \pm \epsilon\Big) A_\pi(s,a) \right) \right].$$

# Mirror Learning & PPO

Recall that PPO update is theoretically defined as:

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[ L_{\text{CLIP}} \right]$$

where

$$L_{\text{CLIP}} = \mathbb{E}_{a\sim\pi} \left[ \min \left( r(\pi') A_\pi(s,a),\ \text{CLIP}\Big(r(\pi'), 1 \pm \epsilon\Big) A_\pi(s,a) \right) \right].$$

By using the trick of "adding and subtracting" and importance sampling $L_{CLIP} =$

$$\mathbb{E}_{a\sim\pi'} \Big[ A_\pi(s,a) \Big] - \mathbb{E}_{a\sim\pi} \left[ r(\pi') A_\pi(s,a) - \min \left( r(\pi') A_\pi(s,a),\ \text{clip}(r(\pi'), 1 \pm \epsilon) A_\pi(s,a) \right) \right].$$

# Mirror Learning & PPO

Recall that PPO update is theoretically defined as:

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[ L_{\text{CLIP}} \right]$$

where

$$L_{\text{CLIP}} = \mathbb{E}_{a\sim\pi} \left[ \min \left( r(\pi') A_\pi(s,a),\ \text{CLIP}\Big(r(\pi'), 1 \pm \epsilon\Big) A_\pi(s,a) \right) \right].$$

By using the trick of "adding and subtracting" and importance sampling $L_{CLIP} =$

$$\mathbb{E}_{a\sim\pi'} \Big[ A_\pi(s,a) \Big] - \mathbb{E}_{a\sim\pi} \left[ r(\pi') A_\pi(s,a) - \min \left( r(\pi') A_\pi(s,a),\ \text{clip}(r(\pi'), 1 \pm \epsilon) A_\pi(s,a) \right) \right].$$

Next, we focus on the expectation that is subtracted. First, we can replace the min operator with max, using the identity $\min f(x) = \max[-f(x)]$, as follows:

$$\mathbb{E}_{a\sim\pi} \left[ r(\pi') A_\pi(s,a) - \max \Big( -r(\pi') A_\pi(s,a),\ -\text{clip}(r(\pi'), 1 \pm \epsilon) A_\pi(s,a) \Big) \right].$$

# Mirror Learning & PPO (cont.)

$$\mathbb{E}_{a\sim\pi}\left[r(\pi')A_\pi(s,a) - \max\left(-r(\pi')A_\pi(s,a),\ -\text{clip}(r(\pi'), 1\pm\epsilon)A_\pi(s,a)\right)\right].$$

# Mirror Learning & PPO (cont.)

$$\mathbb{E}_{a\sim\pi}\left[r(\pi')A_\pi(s,a) - \max\left(-r(\pi')A_\pi(s,a),\ -\text{clip}(r(\pi'),1\pm\epsilon)A_\pi(s,a)\right)\right].$$

Now, we move $r(\pi)A_\pi(s,a)$ inside the max, and obtain:

$$\mathbb{E}_{a\sim\pi}\left[\max\left(0,\left[r(\pi') - \text{clip}(r(\pi'),1\pm\epsilon)\right]A_\pi(s,a)\right)\right].$$

# Mirror Learning & PPO (cont.)

$$\mathbb{E}_{a\sim\pi}\left[r(\pi')A_\pi(s,a) - \max\left(-r(\pi')A_\pi(s,a),\ -\text{clip}(r(\pi'),1\pm\epsilon)A_\pi(s,a)\right)\right].$$

Now, we move $r(\pi)A_\pi(s,a)$ inside the max, and obtain:

$$\mathbb{E}_{a\sim\pi}\left[\max\left(0,\left[r(\pi') - \text{clip}(r(\pi'),1\pm\epsilon)\right]A_\pi(s,a)\right)\right].$$

Finally, we can simplify this as:

$$\mathbb{E}_{a\sim\pi}\left[\text{ReLU}\left(\left[r(\pi') - \text{clip}(r(\pi'),1\pm\epsilon)\right]A_\pi(s,a)\right)\right].$$

# Mirror Learning & PPO (cont.)

## Mirror Learning view of PPO

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[A_{\pi_i}(s,a)\right] - \mathcal{D}^{\text{PPO}}_{\pi_i}(\pi'),$$

where

$$\mathcal{D}^{\text{PPO}}_{\pi_i} \equiv \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[\text{ReLU}\Big(\Big[r(\pi') - \text{clip}(r(\pi'), 1\pm\epsilon)\Big] A_\pi(s,a)\Big)\right].$$

# Mirror Learning & PPO (cont.)

## Mirror Learning view of PPO

$$\pi_{i+1} = \arg\max_{\pi' \in \Pi} \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[A_{\pi_i}(s,a)\right] - \mathcal{D}^{\text{PPO}}_{\pi_i}(\pi'),$$

where

$$\mathcal{D}^{\text{PPO}}_{\pi_i} \equiv \mathbb{E}_{s\sim\pi_i, a\sim\pi'} \left[ \text{ReLU}\Big( \Big[ r(\pi') - \text{clip}(r(\pi'), 1 \pm \epsilon) \Big] A_{\pi}(s,a) \Big) \right].$$

## Monotonic improvement of PPO

To show monotonic improvement property, it suffices to show that:

- **Non-negativity**: $D^{\text{PPO}}_{\pi_i}(\pi) \geq 0$ for all policies $\pi$ and $\pi_i$,
- **Zero drift at current policy**: $D^{\text{PPO}}_{\pi_i}(\pi_i) = 0$,
- **Zero gradient at current policy**: $\nabla_{\pi_i} \mathcal{D}_{\pi_i}(\pi_i) = 0$, the gradient of the drift function is zero when $\pi = \pi_i$, i.e., "gentle" for small changes.

# Mirror Learning Space

- **Mirror Learning reveals more algorithms**
  - An infinite space of theoretically sound mirror learning algorithms beyond PPO.
  - It is sufficient that proper conditions of the drift functions are satisfied.

- **Current algorithms are handcrafted**
  - PPO and others were handcrafted through intuition & trial and error, and may not be optimal for learning.

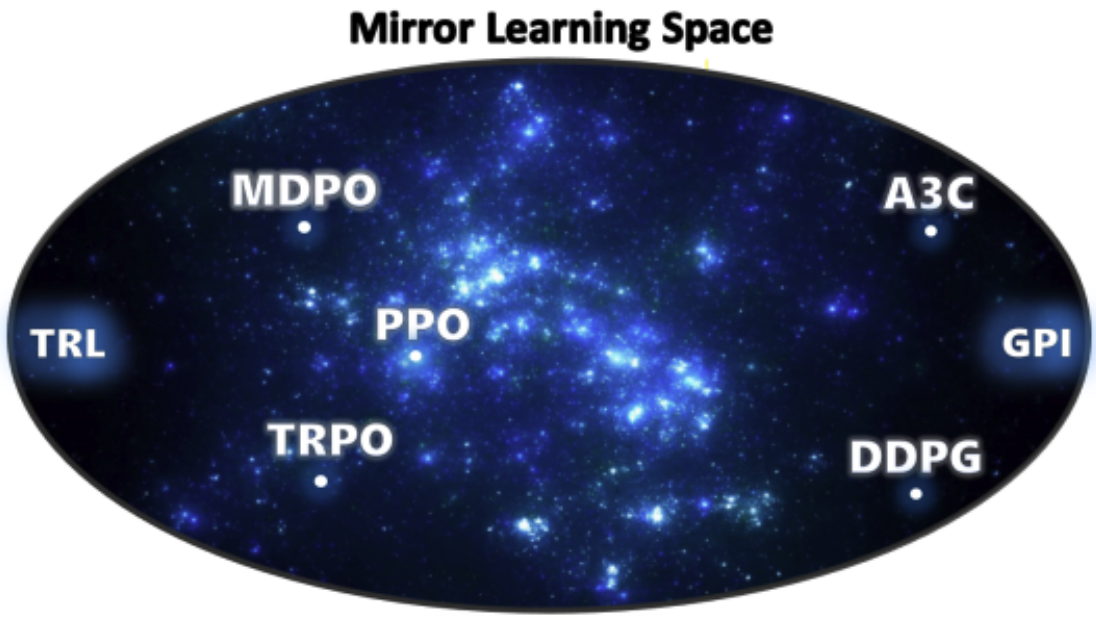



Figure: Taken from [4].

# Disovered Policy Optimisation

# Meta-Learning

- **"Learning to learn"**: Find the best learning algorithm for a distribution of tasks.

# Meta-Learning

- **"Learning to learn"**: Find the best learning algorithm for a distribution of tasks.
- **Outer loop**: Optimizes the learning algorithm.
- **Inner loop**: Optimizes the model parameters.

# Meta-Learning

- **"Learning to learn"**: Find the best learning algorithm for a distribution of tasks.
- **Outer loop**: Optimizes the learning algorithm.
- **Inner loop**: Optimizes the model parameters.

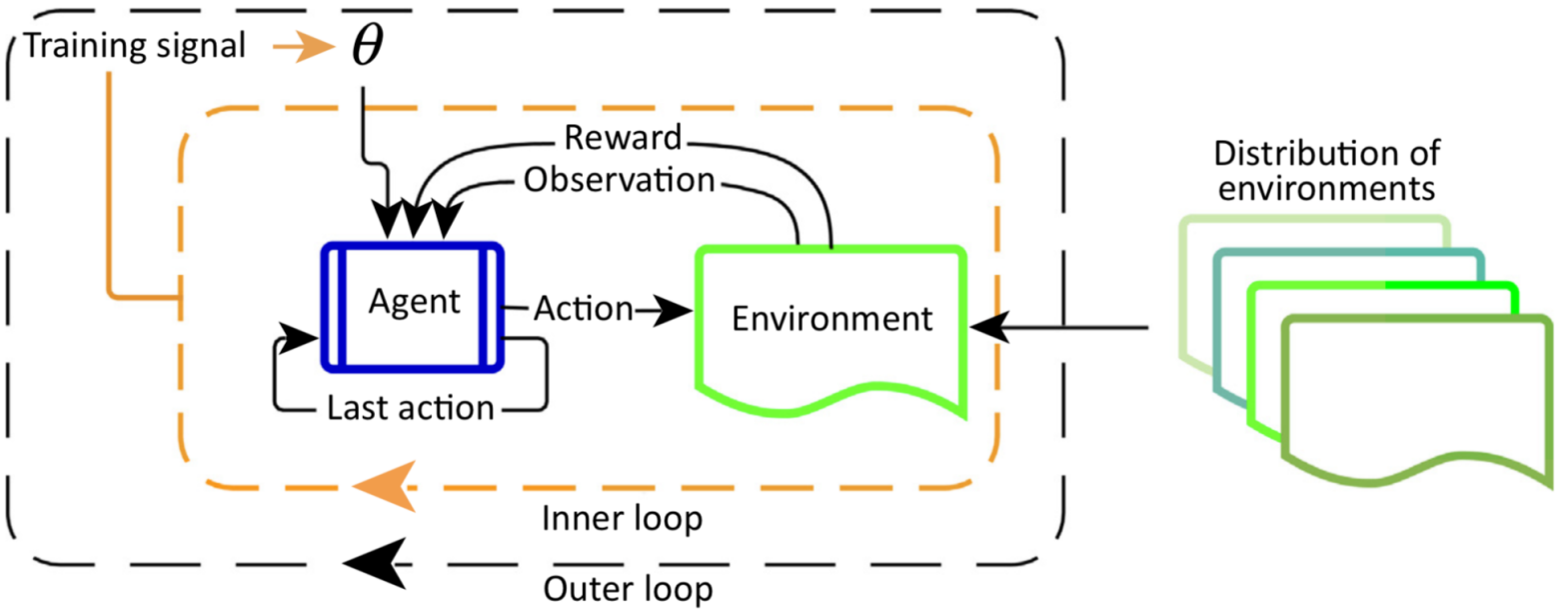



Figure: Meta-RL training loops [5].

# Mirror Learning + Meta-learning = DPO

## Discovered Policy Optimisation

**Chris Lu*** 
FLAIR, University of Oxford
christopher.lu@exeter.ox.ac.uk

**Jakub Grudzien Kuba*** †
BAIR, UC Berkeley
kuba@berkeley.edu

**Alistair Letcher**
aletcher.github.io
ahp.letcher@gmail.com

**Luke Metz**
Google Brain
luke.s.metz@gmail.com

**Christian Schroeder de Witt**
FLAIR, University of Oxford
cs@robots.ox.ac.uk

**Jakob Foerster**
FLAIR, University of Oxford
jakob.foerster@eng.ox.ac.uk

### Abstract

Tremendous progress has been made in reinforcement learning (RL) over the past decade. Most of these advancements came through the continual development of new algorithms, which were designed using a combination of mathematical derivations, intuitions, and experimentation. Such an approach of creating algorithms manually is limited by human understanding and ingenuity. In contrast, *meta-learning* provides a toolkit for automatic *machine learning* method optimisation, potentially addressing this flaw. However, black-box approaches which attempt to discover RL algorithms with minimal prior structure have thus far not outperformed existing hand-crafted algorithms. *Mirror Learning*, which includes RL algorithms, such as PPO, offers a potential middle-ground starting point: while every method in this framework comes with theoretical guarantees, components that differentiate them are subject to design. In this paper we explore the Mirror Learning space by meta-learning a "drift" function. We refer to the immediate result as *Learnt Policy Optimisation* (LPO). By analysing LPO we gain original insights into policy optimisation which we use to formulate a novel, closed-form RL algorithm, *Discovered Policy Optimisation* (DPO). Our experiments in *Brax* environments confirm state-of-the-art performance of LPO and DPO, as well as their transfer to unseen settings.

Figure: Discovered Policy Optimisation paper [6].

# Learnt Policy Optimization (LPO)

- Use meta-learning to learn the optimal[1] drift function $\mathcal{D}^*_{\pi_i}(\pi')$.

[1] Optimal w.r.t. the distribution of the tasks.

# Learnt Policy Optimization (LPO)

- Use meta-learning to learn the optimal[1] drift function $\mathcal{D}^*_{\pi_i}(\pi')$.
- The drift function is approximated using a neural network computing function $f$ with parameters $\phi$ and input $x$.

$$\mathcal{D}_{\pi_i}(\pi') \approx \mathbb{E}^{\pi_i}\left[f_\phi(x)\right]$$

[1] Optimal w.r.t. the distribution of the tasks.

# LPO Network

On input $f$ receives probability ratio between a candidate and the old policy, defined as: $r = \frac{\pi'(a|s)}{\pi(a|s)}$, the advantage: $A = A_\pi(s, a)$,

# LPO Network

On input $f$ receives probability ratio between a candidate and the old policy, defined as: $r = \frac{\pi'(a|s)}{\pi(a|s)}$, the advantage: $A = A_\pi(s, a)$,

To ease learning complicated mappings, apply non-linear transformations of the arguments (as shown in E), forming the following input:

$$x_{r,A} = \left[(1-r), (1-r)^2, (1-r)A, (1-r)^2 A, \log(r), \log(r)^2, \log(r)A, \log(r)^2 A\right]$$

# LPO Network

On input $f$ receives probability ratio between a candidate and the old policy, defined as: $r = \frac{\pi'(a|s)}{\pi(a|s)}$, the advantage: $A = A_\pi(s, a)$,

To ease learning complicated mappings, apply non-linear transformations of the arguments (as shown in E), forming the following input:

$$x_{r,A} = \left[(1-r), (1-r)^2, (1-r)A, (1-r)^2 A, \log(r), \log(r)^2, \log(r)A, \log(r)^2 A\right]$$

In order to guarantee that the neural network is a valid drift function, it must satisfy:

- $f_\phi(x_{r,A}) = 0$ and $\nabla_r f_\phi(x_{r,A}) = 0$ whenever $r = 1$ ($\pi = \pi'$),
- $f_\phi(x_{r,A}) \geq 0$ everywhere.

# LPO Network

On input $f$ receives probability ratio between a candidate and the old policy, defined as: $r = \frac{\pi'(a|s)}{\pi(a|s)}$, the advantage: $A = A_\pi(s, a)$,

To ease learning complicated mappings, apply non-linear transformations of the arguments (as shown in E), forming the following input:

$$x_{r,A} = \left[(1-r), (1-r)^2, (1-r)A, (1-r)^2 A, \log(r), \log(r)^2, \log(r)A, \log(r)^2 A\right]$$

In order to guarantee that the neural network is a valid drift function, it must satisfy:

- $f_\phi(x_{r,A}) = 0$ and $\nabla_r f_\phi(x_{r,A}) = 0$ whenever $r = 1$ ($\pi = \pi'$),
- $f_\phi(x_{r,A}) \geq 0$ everywhere.

In this model, $x_{r,A} = 0$ whenever $r = 1$, and the first condition is guaranteed by excluding bias terms from the network architecture.

# LPO Network

On input $f$ receives probability ratio between a candidate and the old policy, defined as: $r = \frac{\pi'(a|s)}{\pi(a|s)}$, the advantage: $A = A_\pi(s, a)$,

To ease learning complicated mappings, apply non-linear transformations of the arguments (as shown in E), forming the following input:

$$x_{r,A} = \left[(1-r), (1-r)^2, (1-r)A, (1-r)^2 A, \log(r), \log(r)^2, \log(r)A, \log(r)^2 A\right]$$

In order to guarantee that the neural network is a valid drift function, it must satisfy:

- $f_\phi(x_{r,A}) = 0$ and $\nabla_r f_\phi(x_{r,A}) = 0$ whenever $r = 1$ ($\pi = \pi'$),
- $f_\phi(x_{r,A}) \geq 0$ everywhere.

In this model, $x_{r,A} = 0$ whenever $r = 1$, and the first condition is guaranteed by excluding bias terms from the network architecture.
To meet the latter two conditions, we apply the ReLU activation at the last layer with a slight shift:

$$x \mapsto \text{ReLU}(x - \xi), \quad \xi = 10^{-6}$$

# LPO Training Loop

- **Inner loop**: Train the model by optimizing the model parameters $\theta$.
- **Outer loop**: Estimate the gradient and update the meta-parameters $\phi$ using Evolution Strategies (ES).
  - ES approximates the gradient by sampling random perturbations $\epsilon$ and evaluating the objective function at $\phi + \sigma\epsilon$.
  - This approach eliminates the need for back-propagation

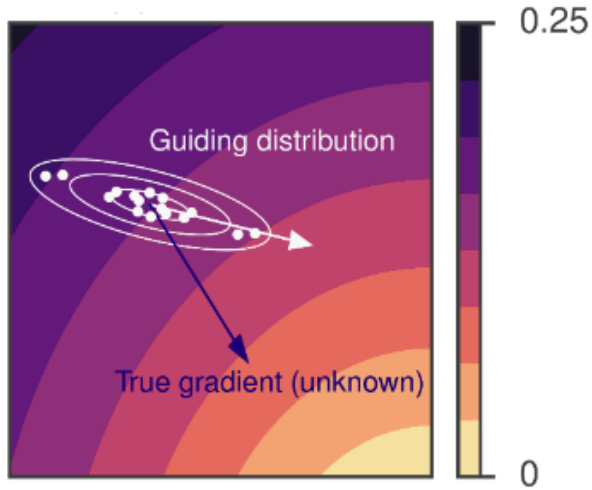



Figure: Gradient estimation using ES [7].

# LPO technical details

- Two variants trained
  - LPO-Zero: trained from scratch
  - LPO: adding PPO's drift at the last layer

$$f_\phi(x_{r,A}) = \text{ReLU}\left(\tilde{f}_\phi(x_{r,A}) - \xi + \text{ReLU}\left((r - \text{clip}(r, 1 \pm \epsilon)) \cdot A\right)\right)$$

    where $\tilde{f}_\phi(x_{r,A})$ is the output of the last hidden layer of the drift network.
- Drift function neural network with 1 hidden layer and 128 hidden units
- Default hyperparameters for PPO
- 4 V100 GPUs
- 700 outer iterations $\approx$ 2 days of compute

# LPO findings

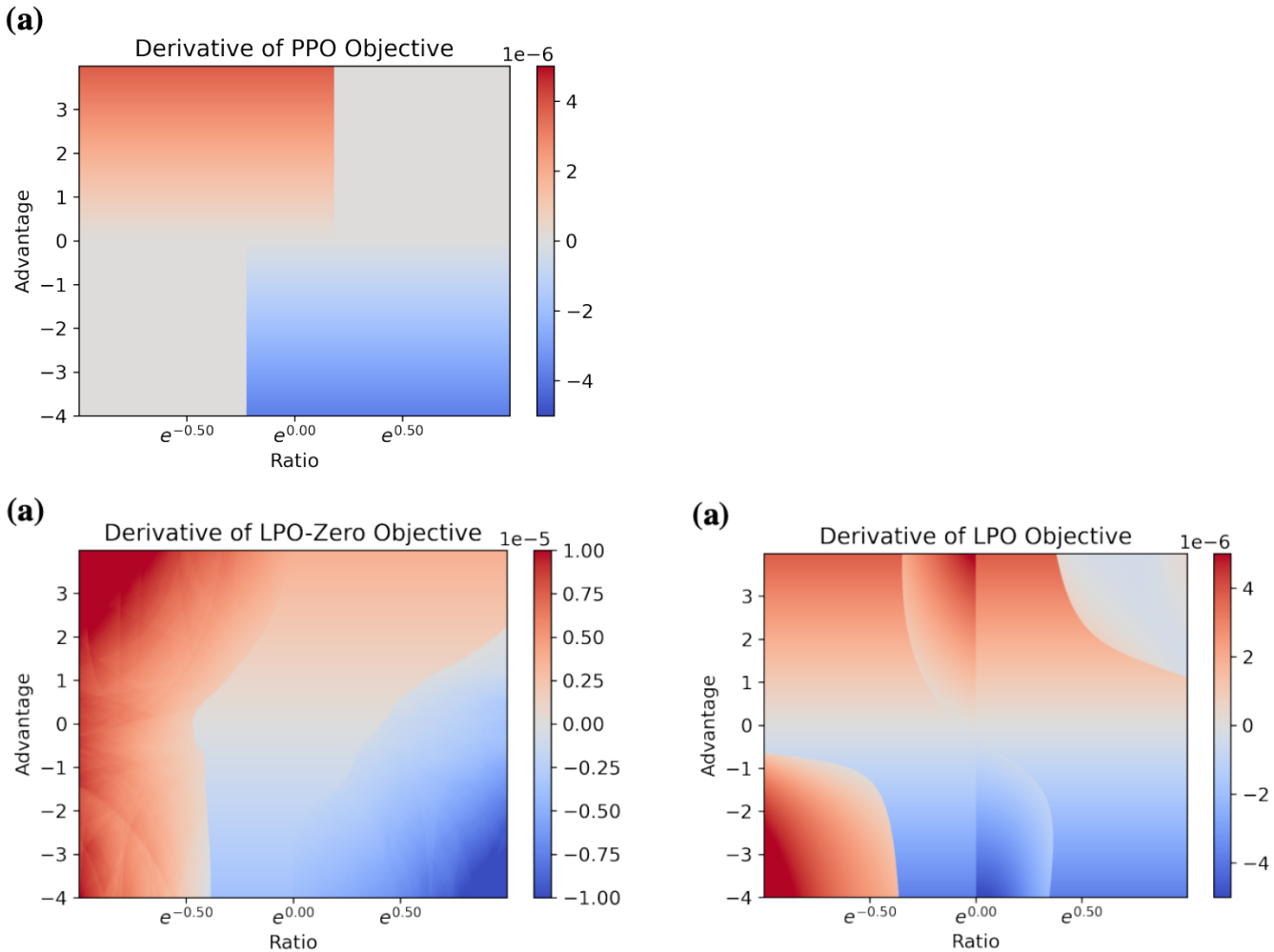



Figure: Derivative of the Objective for PPO, LPO-Zero, and LPO [6].

# From LPO to DPO

- A closed-form formula would be much better than a neural network!

$$f_{\text{DPO}}(r, A) = \begin{cases} \text{ReLU}\left((r-1)A - \alpha \tanh\left(\frac{(r-1)A}{\alpha}\right)\right) & \text{if } A \geq 0, \\ \text{ReLU}\left(\log(r)A - \beta \tanh\left(\frac{\log(r)A}{\beta}\right)\right) & \text{if } A < 0. \end{cases} \tag{4}$$

- For $\alpha = 2$ and $\beta = 0.6$, the formula reproduces key features of LPO.

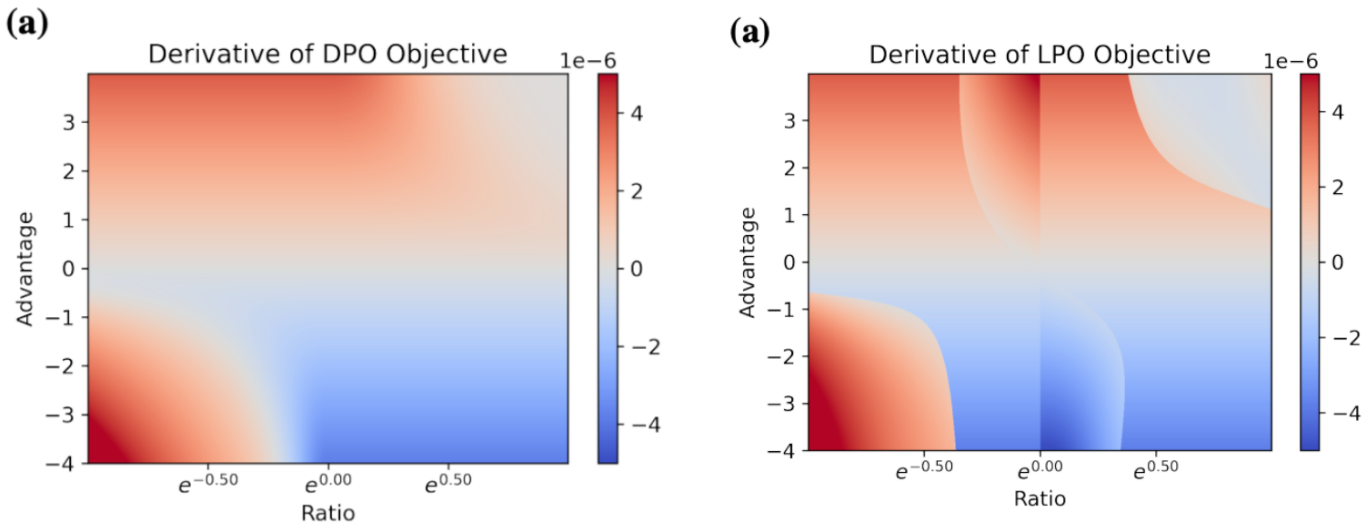



Figure: Derivative of the Objective for DPO and LPO [6].

# From LPO to DPO

- A closed-form formula would be much better than a neural network!

$$f_{\text{DPO}}(r, A) = \begin{cases} \text{ReLU}\left((r-1)A - \alpha \tanh\left(\frac{(r-1)A}{\alpha}\right)\right) & \text{if } A \geq 0, \\ \text{ReLU}\left(\log(r)A - \beta \tanh\left(\frac{\log(r)A}{\beta}\right)\right) & \text{if } A < 0. \end{cases} \tag{4}$$

- For $\alpha = 2$ and $\beta = 0.6$, the formula reproduces key features of LPO.

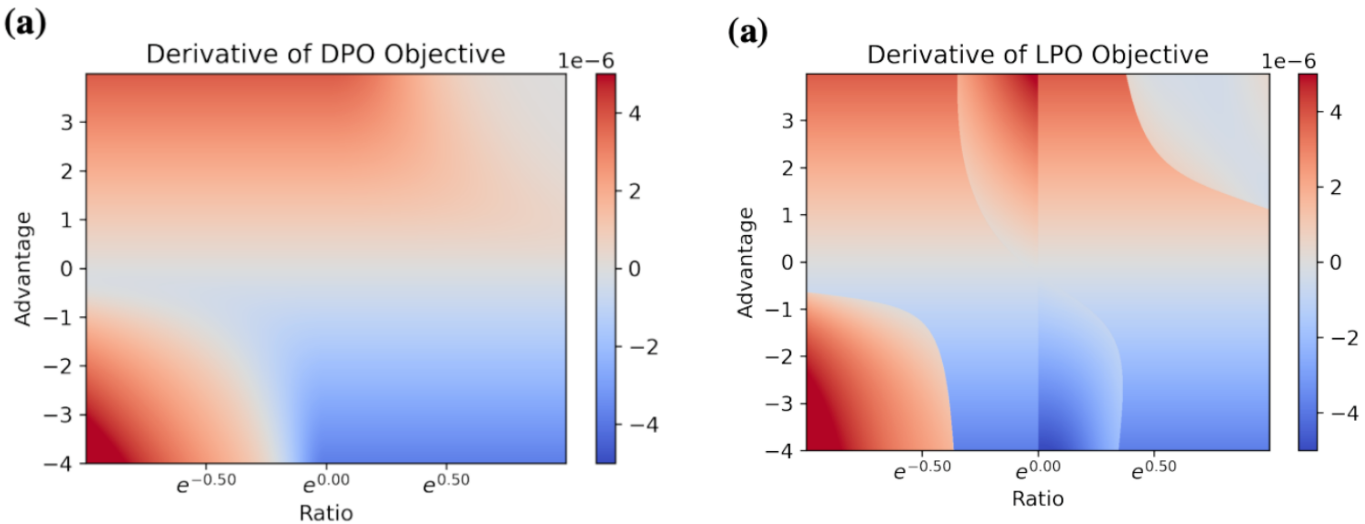



Figure: Derivative of the Objective for DPO and LPO [6].

- Moreover, the formula satisfies the drift function properties [6], as a bonus, it is two lines of code update of PPO to integrate into RL libraries!

# Evaluation

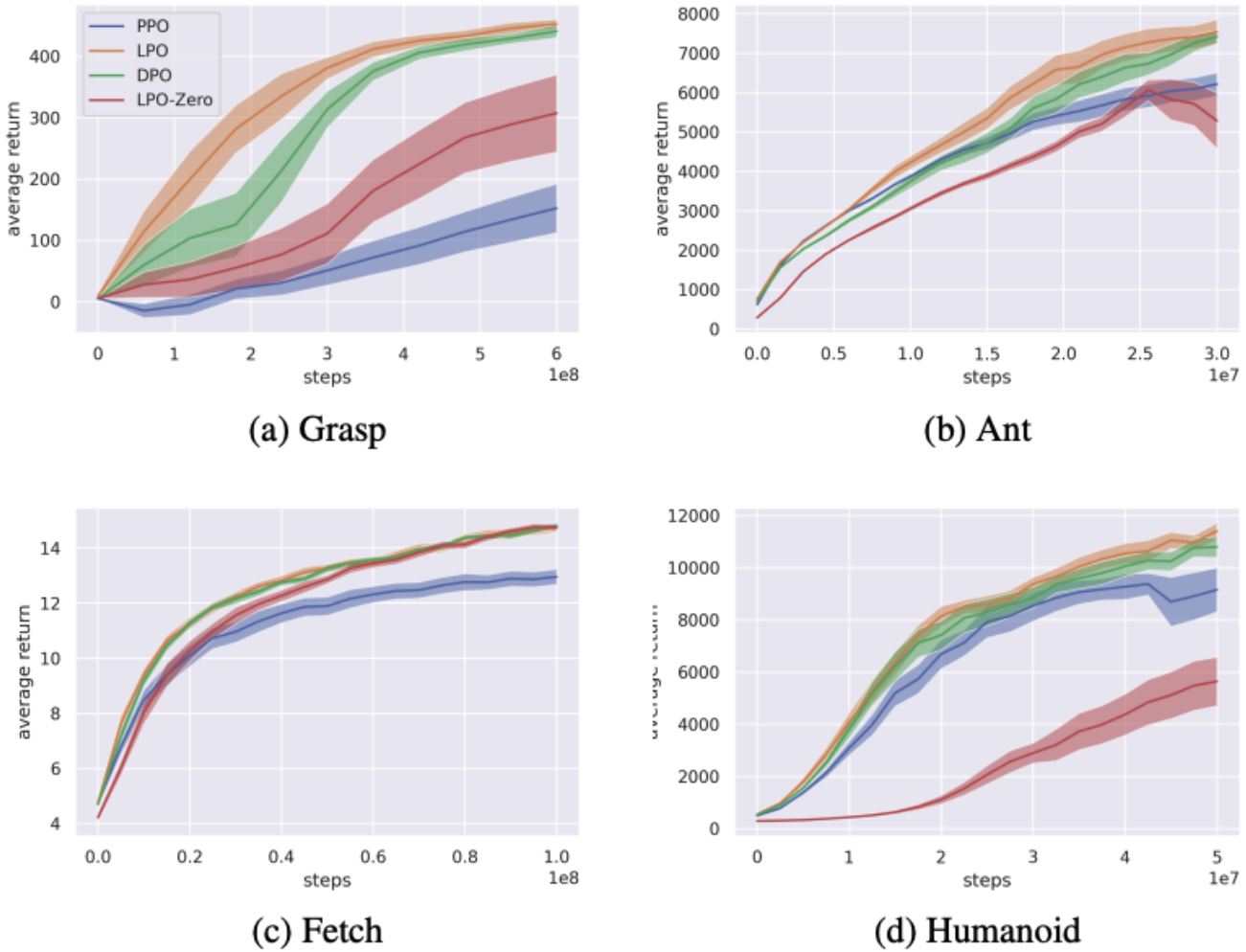



Figure: LPO, DPO, PPO performance comparison [6] on Brax [8] environments (10 seeds).

# Evaluation

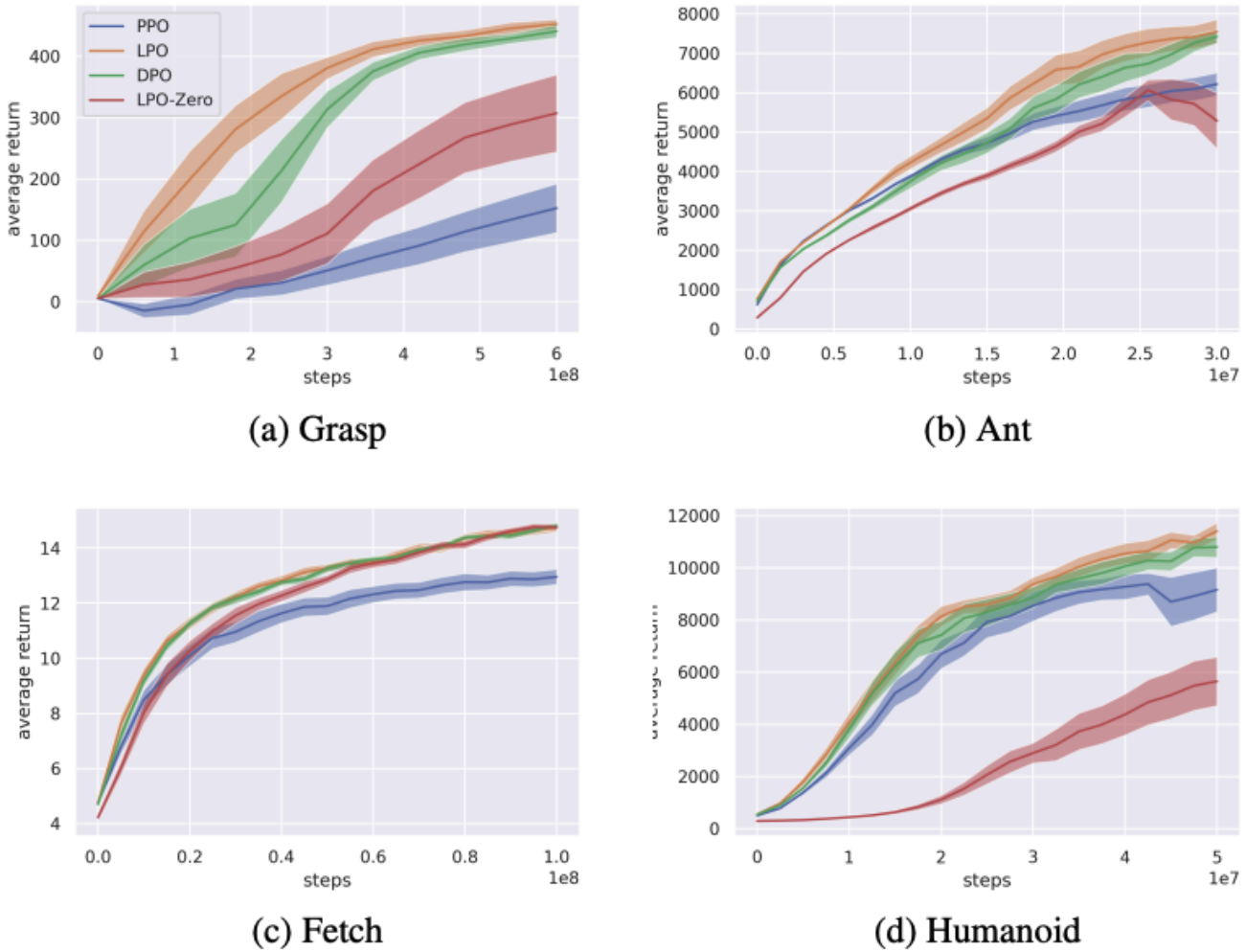
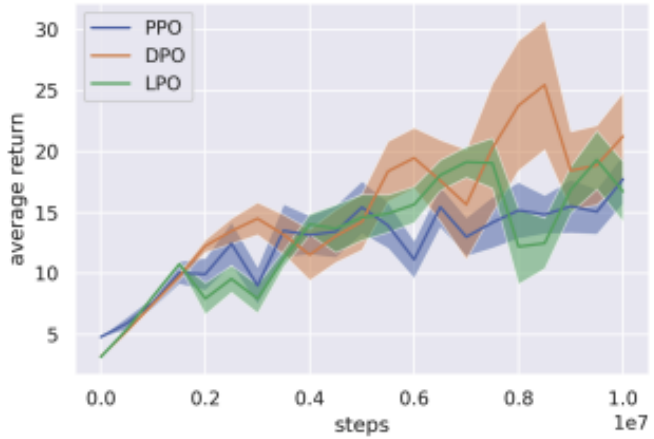


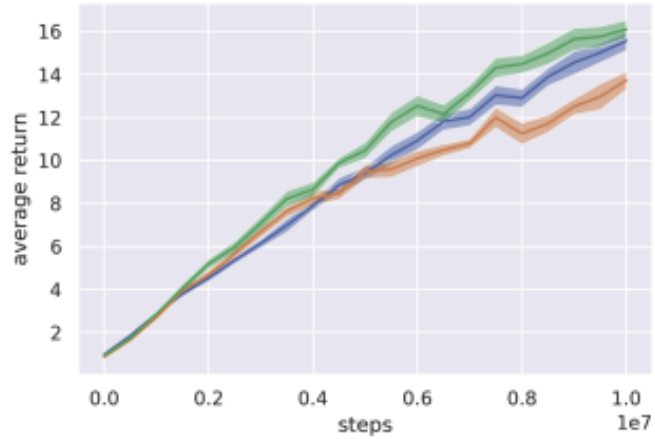

Figure: LPO, DPO, PPO performance comparison [6] on Brax [8] environments (10 seeds).

- Note: LPO was meta-trained on a single environment (ant) only!

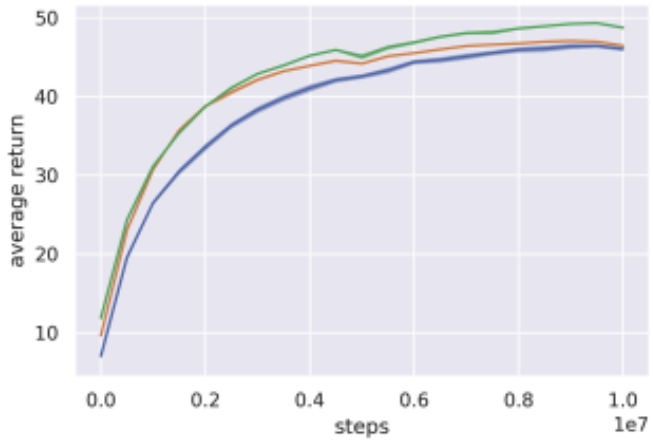
# Evaluation (cont.)



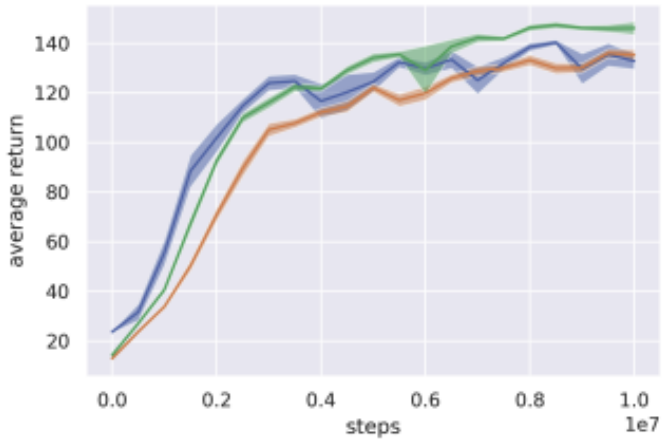

(a) Breakout-Minatar




(b) Asterix-Minatar




(c) Freeway-Minatar




(d) SpaceInvaders-Minatar

Figure: LPO, DPO, PPO performance comparison [6] on Minatar [9] environments (10 seeds).

# Evaluation (cont.)

- Both LPO and DPO outperform PPO on tasks they were not meta-learned for.
- Moreover, they use default PPO's finetuned hyperparameters for the given task.

# Evaluation (cont.)

- Both LPO and DPO outperform PPO on tasks they were not meta-learned for.
- Moreover, they use default PPO's finetuned hyperparameters for the given task.
- Both LPO and DPO show both greater stability, possibly due to better exploration [6].

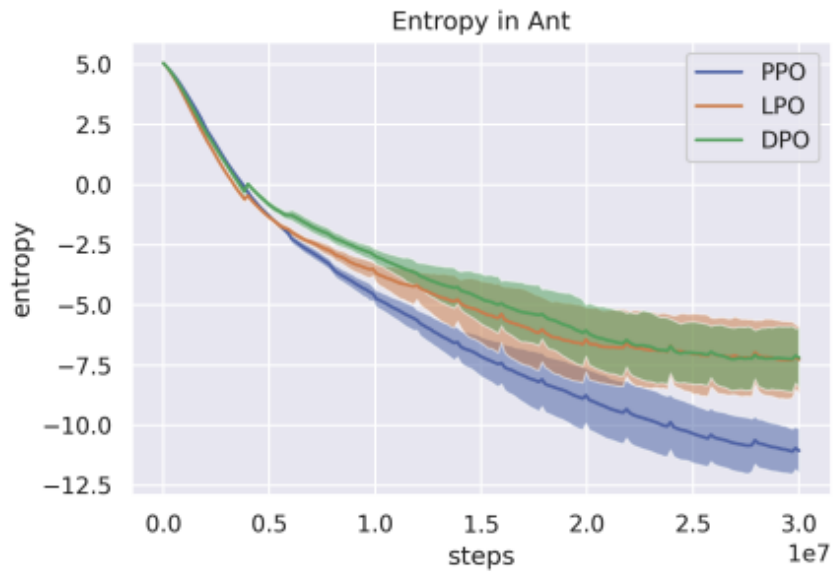



Figure: Comparison of entropy between PPO, LPO, and DPO on ant environment [6].

# Takeaways & Future Work

- **Key takeaways**:
  - Mirror learning shows there is an infinite space of theoretically sound RL algorithms

# Takeaways & Future Work

- **Key takeaways**:
  - Mirror learning shows there is an infinite space of theoretically sound RL algorithms
  - This space can be searched and SOTA algorithms discovered not by handcrafting but through meta-learning.

# Takeaways & Future Work

- **Key takeaways**:
  - Mirror learning shows there is an infinite space of theoretically sound RL algorithms
  - This space can be searched and SOTA algorithms discovered not by handcrafting but through meta-learning.
- **Future directions**:
  - Find the optimal drift function for each time-step.

# Takeaways & Future Work

- **Key takeaways**:
  - Mirror learning shows there is an infinite space of theoretically sound RL algorithms
  - This space can be searched and SOTA algorithms discovered not by handcrafting but through meta-learning.
- **Future directions**:
  - Find the optimal drift function for each time-step.
  - LLM-guided Discovery?

# LLM-guided Discovery! (2024)

## Discovering Preference Optimization Algorithms with and for Large Language Models


**Chris Lu**[*]
Sakana AI and FLAIR
chrislu@sakana.ai

**Samuel Holt**[*]
University of Cambridge
sih31@cam.ac.uk

**Claudio Fanconi**[*]
University of Cambridge
caf83@cam.ac.uk

**Alex J. Chan**[†]
University of Cambridge
ajc340@cam.ac.uk

**Jakob Foerster**[‡]
FLAIR, University of Oxford
jakob.foerster@eng.ox.ac.uk

**Mihaela van der Schaar**[‡]
University of Cambridge
mv472@cam.ac.uk

**Robert Tjarko Lange**[‡]
Sakana AI
robert@sakana.ai



### Abstract

Offline preference optimization is a key method for enhancing and controlling the quality of Large Language Model (LLM) outputs. Typically, preference optimization is approached as an offline supervised learning task using manually crafted convex loss functions. While these methods are based on theoretical insights, they are inherently constrained by human creativity, so the large search space of possible loss functions remains under-explored. We address this by performing LLM-driven *objective discovery* to automatically discover new state-of-the-art preference optimization algorithms without (expert) human intervention. Specifically, we iteratively prompt an LLM to propose and implement new preference optimization loss functions based on previously evaluated performance metrics. This process leads to the discovery of previously unknown and performant preference optimization algorithms. The best performing of these we call *Discovered Preference Optimization* (DiscoPOP)[1], a novel algorithm that adaptively blends logistic and exponential losses. Experiments demonstrate the state-of-the-art performance of DiscoPOP and its successful transfer to held-out tasks.


Figure: DPO follow-up paper [10].

# References I


[1] John Schulman et al. “Trust Region Policy Optimization”. In: *CoRR* abs/1502.05477 (2015). arXiv: 1502.05477. URL: http://arxiv.org/abs/1502.05477.

[2] Christopher Berner et al. “Dota 2 with large scale deep reinforcement learning”. In: *arXiv preprint arXiv:1912.06680* (2019).

[3] Yuhui Wang, Hao He, and Xiaoyang Tan. “Truly proximal policy optimization”. In: *Uncertainty in artificial intelligence*. PMLR. 2020, pp. 113–122.

[4] Jakub Grudzien, Christian A Schroeder De Witt, and Jakob Foerster. “Mirror Learning: A Unifying Framework of Policy Optimisation”. In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 7825–7844. URL: https://proceedings.mlr.press/v162/grudzien22a.html.

[5] Lilian Weng. *Meta Reinforcement Learning*. Accessed: 2025-03-07. 2019. URL: https://lilianweng.github.io/posts/2019-06-23-meta-rl/.

# References II


[6] Chris Lu et al. “Discovered Policy Optimisation”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 16455–16468. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/688c7a82e31653e7c256c6c29fd3b438-Paper-Conference.pdf.

[7] Niru Maheswaranathan et al. “Guided evolutionary strategies: Augmenting random search with surrogate gradients”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 4264–4273.

[8] C. Daniel Freeman et al. *Brax - A Differentiable Physics Engine for Large Scale Rigid Body Simulation*. Version 0.12.1. 2021. URL: http://github.com/google/brax.

[9] Kenny Young and Tian Tian. “MinAtar: An Atari-Inspired Testbed for Thorough and Reproducible Reinforcement Learning Experiments”. In: *arXiv preprint arXiv:1903.03176* (2019).

[10] Chris Lu et al. “Discovering preference optimization algorithms with and for large language models”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 86528–86573.

MASARYK
UNIVERSITY