

Learning Optimal Advantage from Preferences and Mistaking it for Reward



W. Bradley
Knox^{1,4}



Stephane
Hatgis-Kessell¹



Sigurdur Orn
Adalgeirsson⁴



Serena
Booth²



Anca
Dragan⁵



Scott
Niekum⁶



Peter
Stone^{1,3}

¹UT Austin

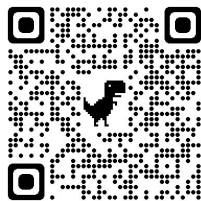
²MIT CSAIL

³Sony AI

⁴Google Research

⁵UC Berkeley

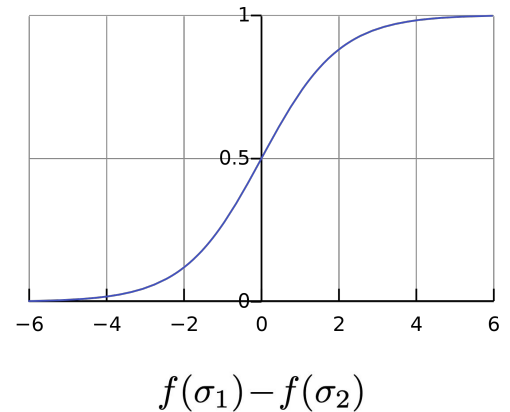
⁶UMass Amherst



The model of preference

$$P(\sigma_1 \succ \sigma_2) = \frac{\exp [f(\sigma_1)]}{\exp [f(\sigma_1)] + \exp [f(\sigma_2)]}$$
$$= \textit{logistic}(f(\sigma_1) - f(\sigma_2))$$

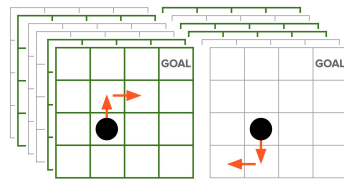
(Shorthand notation above leaves out from P and f an implied reward function as input.)



Learning a reward function from preferences

Given a preference model $P(\sigma_1 \succ \sigma_2 | \hat{r})$,

optimize \hat{r} to maximize the likelihood of the *preferences dataset*.

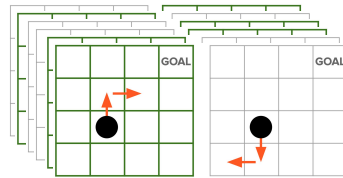


Typical RLHF algorithm's view of the world



preferences sampled from a
preference model

*preferences
dataset*



\hat{r}

MLE with a **preference model**

The preference model

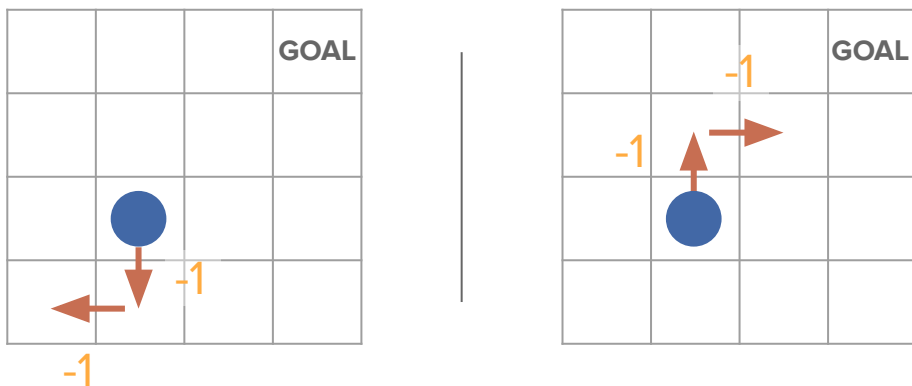
Common model: **Partial return**

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a) \right)$$

The preference model

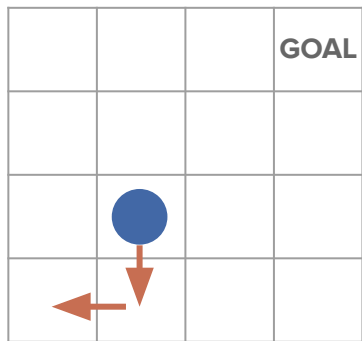
Common model: **Partial return**

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a) \right)$$



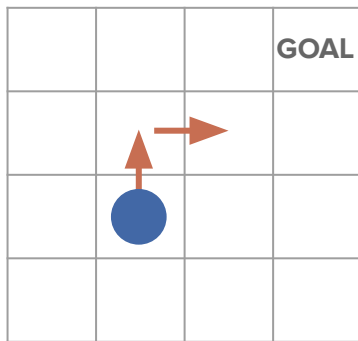
Indifferent!

Suboptimal segment



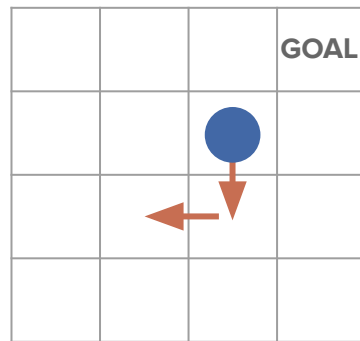
Equal partial return
Lower end state value

Optimal segment



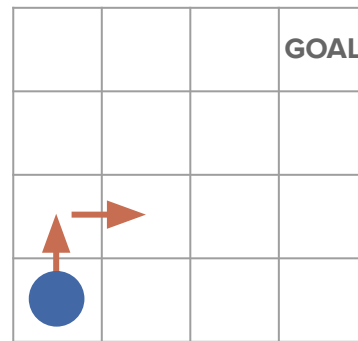
Equal partial return
Higher end state value

Suboptimal segment



Equal partial return
Higher start state value

Optimal segment



Equal partial return
Lower start state value

The preference model

Common model: **Partial return**

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a) \right)$$

The preference model

Common model: **Partial return**

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a) \right)$$

Proposed model: **Regret**

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} A_r^*(s,a) - \sum_{(s,a) \in \sigma_2} A_r^*(s,a) \right)$$

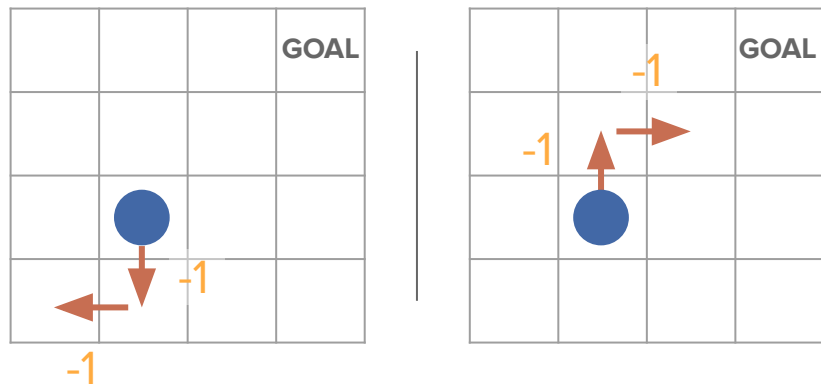
The **regret** of a segment **measures how much it deviates from optimal behavior.**

The preference model

Partial return

$$P(\sigma_1 \succ \sigma_2) = \text{logistic}\left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a)\right)$$

Showing **reward**

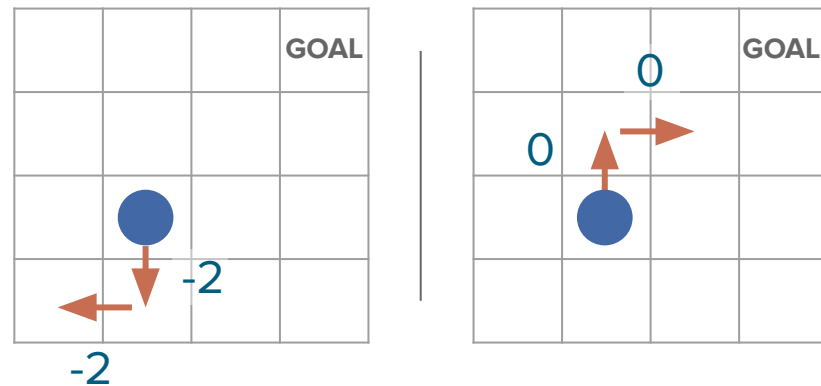


Indifferent

Regret

$$P(\sigma_1 \succ \sigma_2) = \text{logistic}\left(\sum_{(s,a) \in \sigma_1} A_r^*(s,a) - \sum_{(s,a) \in \sigma_2} A_r^*(s,a)\right)$$

Showing **optimal advantage**



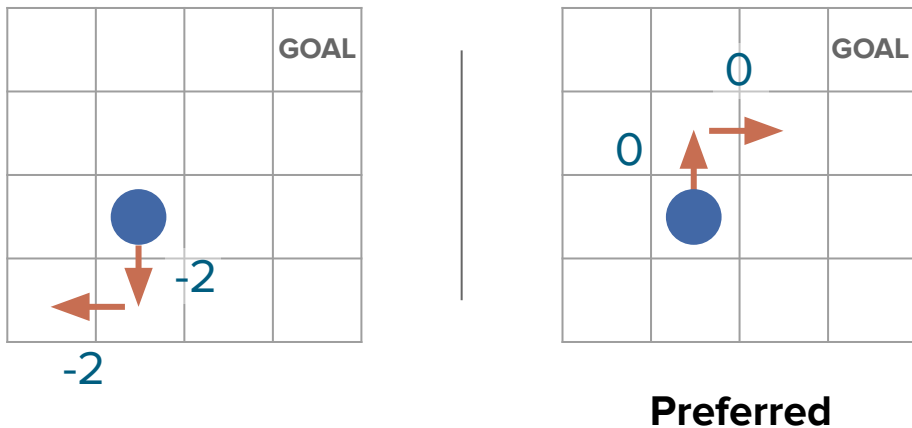
Preferred

The preference model

Proposed model: **Regret**

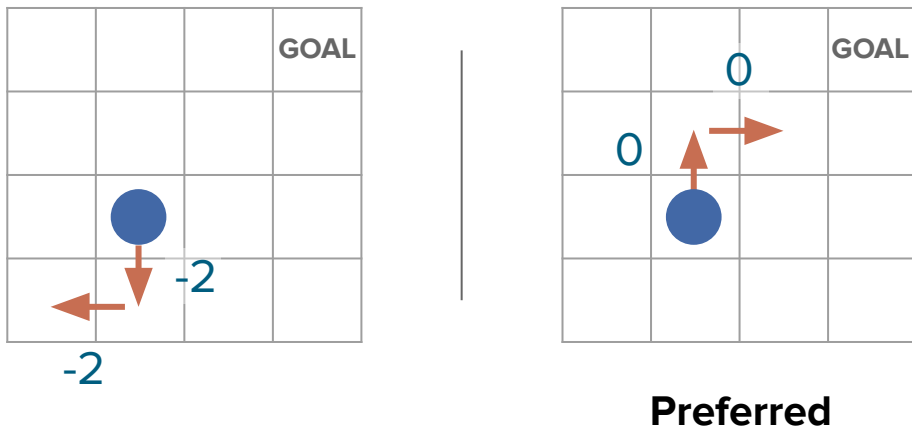
$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} A_r^*(s,a) - \sum_{(s,a) \in \sigma_2} A_r^*(s,a) \right)$$

Showing optimal advantage



The preference model

Showing optimal advantage



Proposed model: Regret

$$P(\sigma_1 \succ \sigma_2) = \text{logistic} \left(\sum_{(s,a) \in \sigma_1} A_r^*(s,a) - \sum_{(s,a) \in \sigma_2} A_r^*(s,a) \right)$$

Comparisons

Theoretically superior (identifiable)

With human preferences

- **more descriptive**
- **learns more aligned reward functions**

*Then why does the **partial return preference** model work so well for fine-tuning?*

*Then why does the **partial return preference** model work so well for fine-tuning?*

This paper answers in two contexts:

1) RLHF generally

2) RLHF fine tuning for LLMs

**When regret drives preferences but the dominant model is assumed
(i.e., using A_r^* as \mathcal{r})**

Outline:

- **When A_r^* is known exactly**
- **When A_r^* is approximated**
- **Reframing RLHF for LLMs**

**Assuming the partial return
preference model when regret
is correct**

(Learning A_r^* and using it as r)

A unified representation of the preference models

$$P(\sigma_1 \succ \sigma_2) = \text{logistic}\left(f(\sigma_1) - f(\sigma_2)\right)$$

Partial return: $f(\sigma) =$ discounted sum of $r(s, a)$ for each (s, a) in σ

Regret: $f(\sigma) =$ discounted sum of $A^*(s, a)$ for each (s, a) in σ

Unification: $f(\sigma) =$ discounted sum of $g(s, a)$ for each (s, a) in σ

If you assume partial return but preferences are by regret, then **you are using (an approximation of) A^* as a reward function.**

A unified representation of the preference models

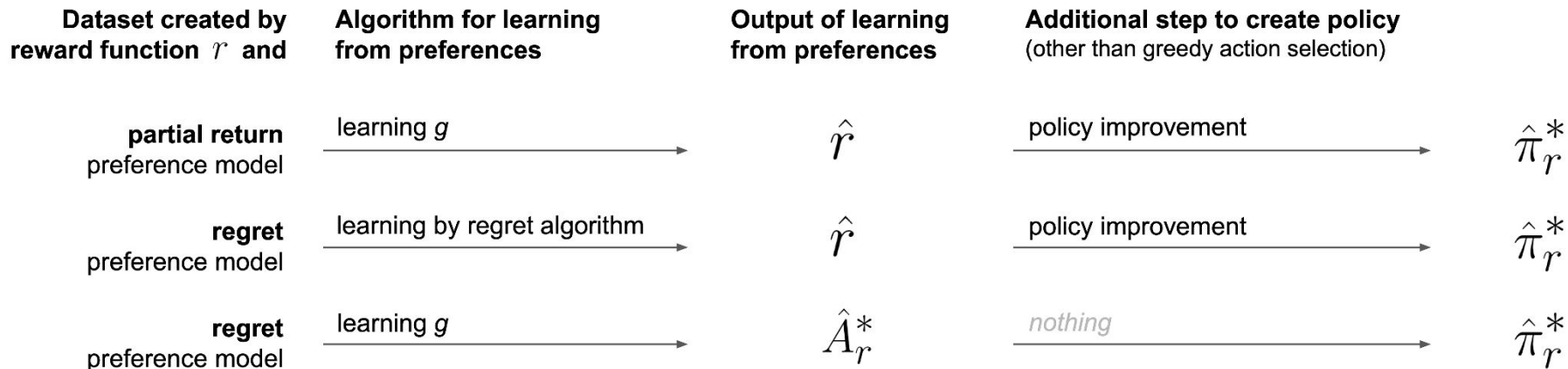
$$\begin{aligned} P(\sigma_1 \succ \sigma_2) &= \text{logistic}\left(f(\sigma_1) - f(\sigma_2)\right) \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} \tilde{r}(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} \tilde{r}(s_t^\sigma, a_t^\sigma)\right) \text{ Partial return} \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma)\right) \text{ Regret} \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} g(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} g(s_t^\sigma, a_t^\sigma)\right) \text{ Unification} \end{aligned}$$

If you assume partial return but preferences are by regret, then **you are using (an approximation of) A^* as a reward function.**

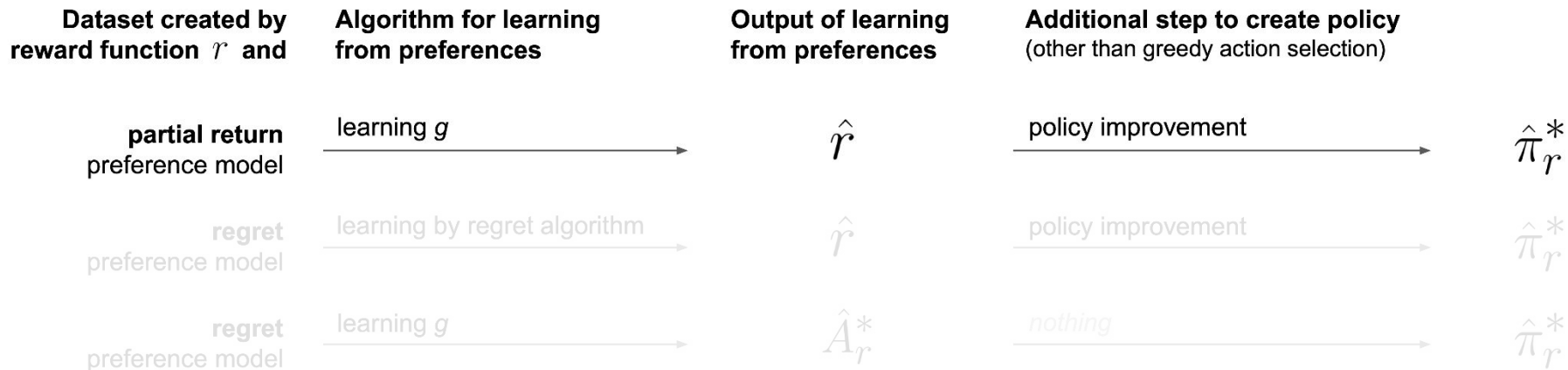
A unified representation of the preference models

$$\begin{aligned} P(\sigma_1 \succ \sigma_2) &= \text{logistic}\left(f(\sigma_1) - f(\sigma_2)\right) \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} \tilde{r}(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} \tilde{r}(s_t^\sigma, a_t^\sigma)\right) \text{ Partial return} \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma)\right) \text{ Regret} \\ &= \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} g(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} g(s_t^\sigma, a_t^\sigma)\right) \text{ Unification} \end{aligned}$$

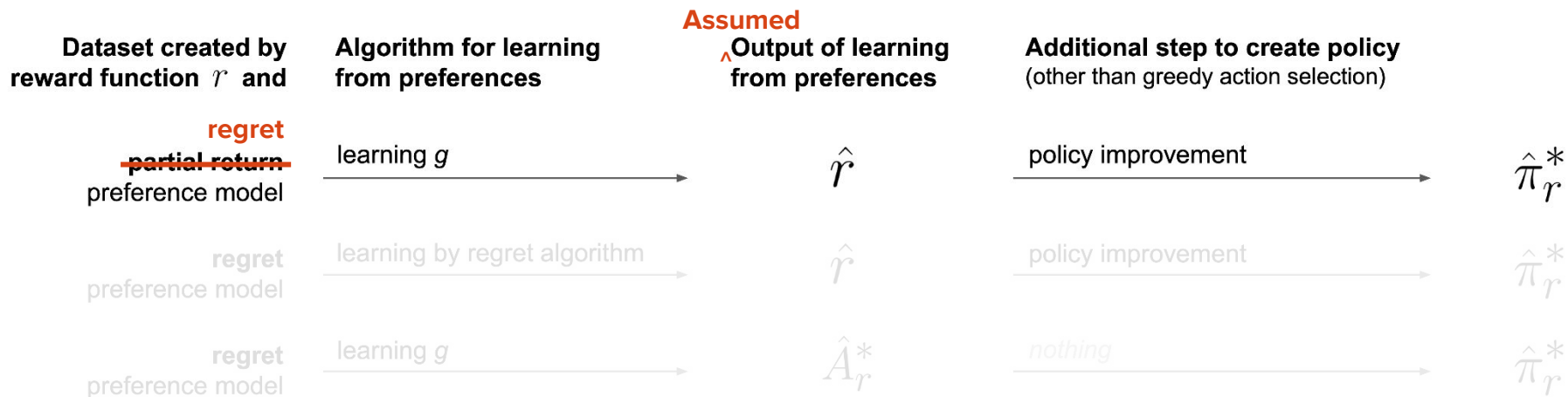
3 algorithms



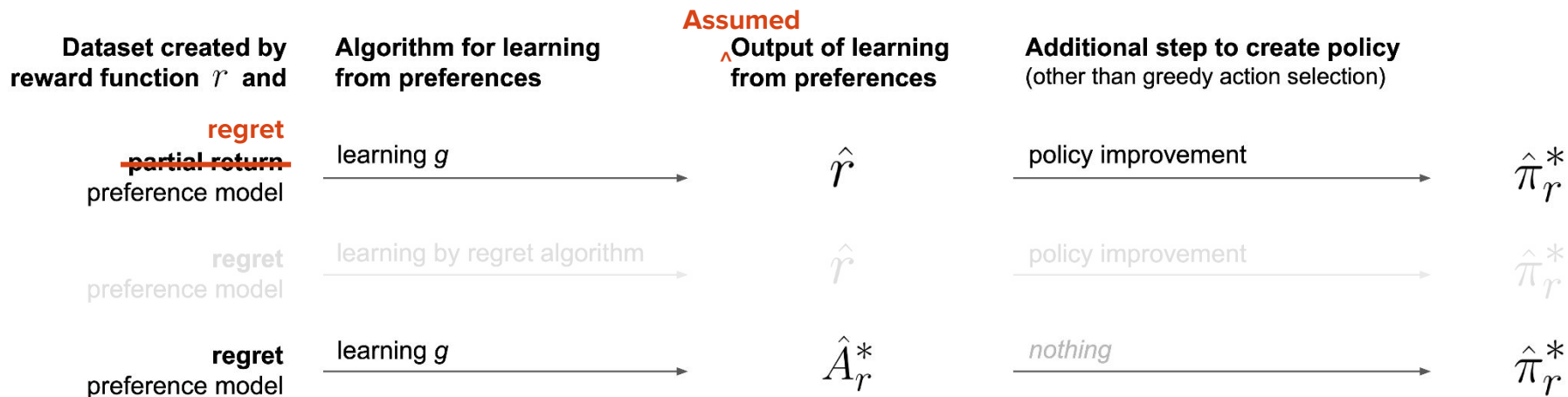
3 algorithms



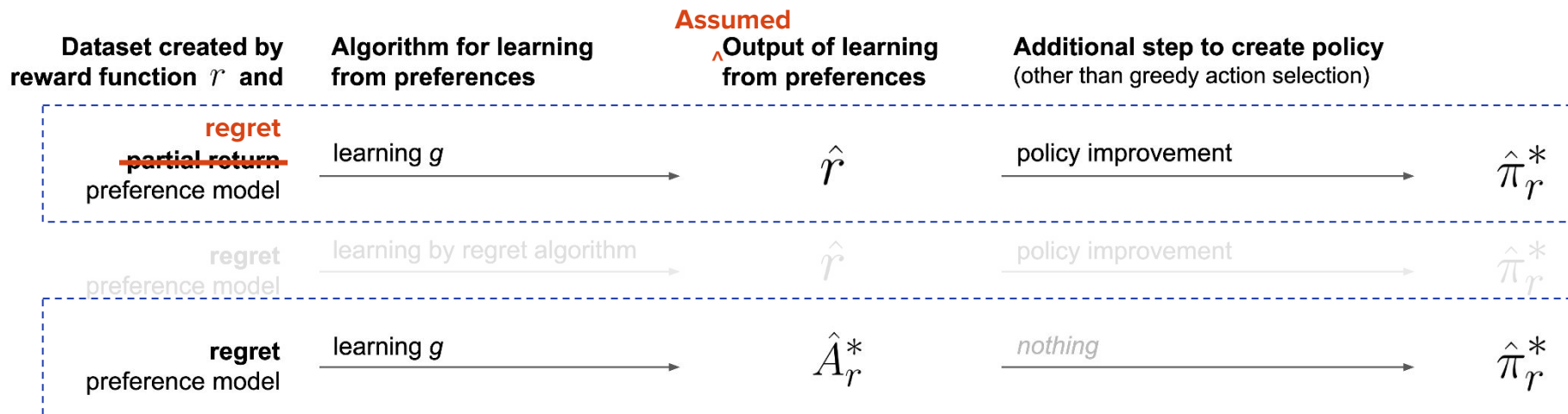
4 algorithms



4 algorithms

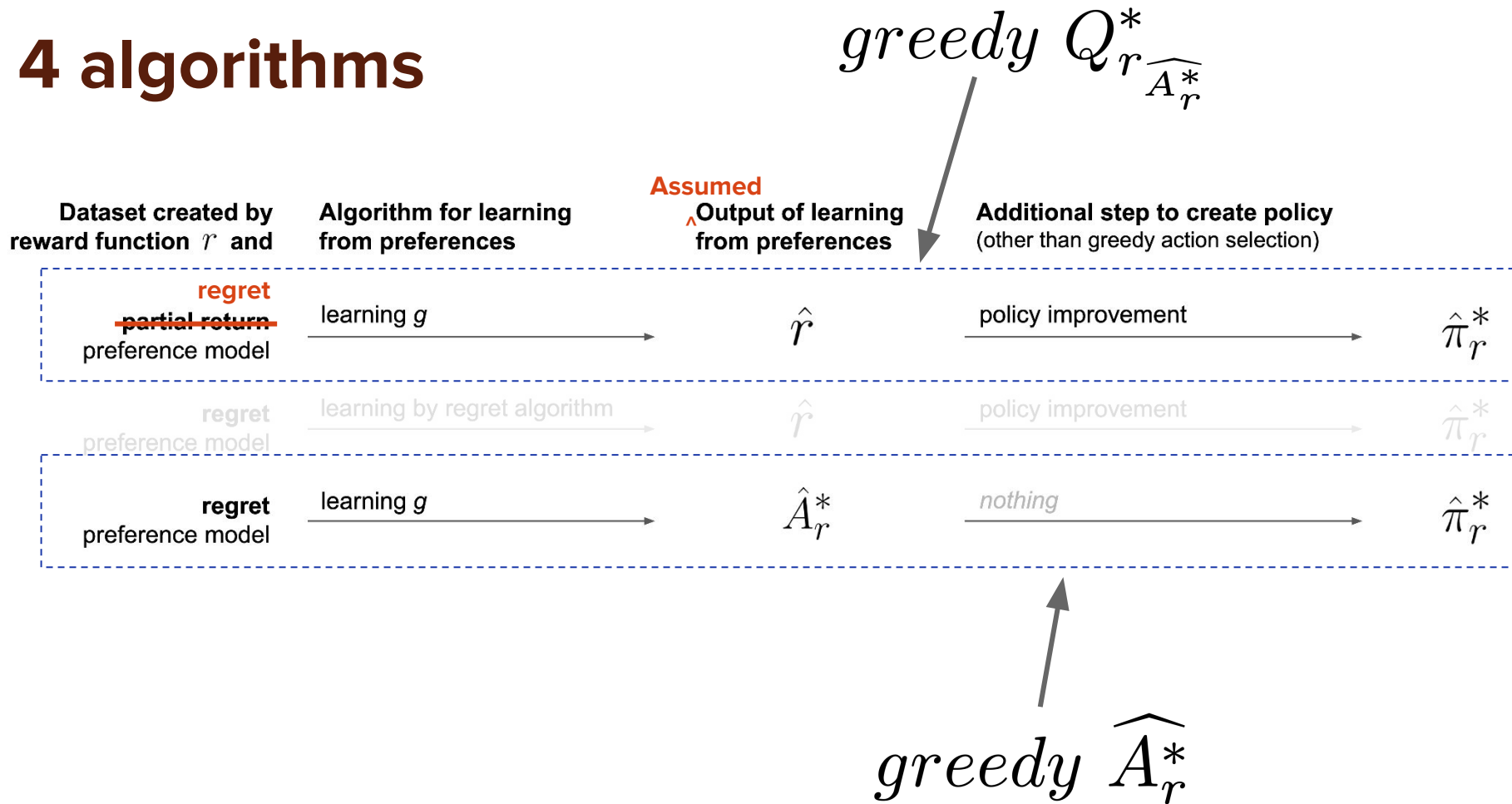


4 algorithms



greedy \hat{A}_r^*

4 algorithms



Using
 A_r^* as reward

Optimal policies are preserved.

The set of optimal policies under r and $r_{A_r^*} \triangleq A_r^*$ is the same, regardless of the discount factor used with $r_{A_r^*}$.

Intuition:

$A_r^*(s, a) = 0 \iff (s, a)$ is optimal w.r.t. r

$A_r^*(s, a) < 0 \iff (s, a)$ is suboptimal w.r.t. r

so:

trajectory τ has *return* = 0 under r' \iff all (s, a) in τ are optimal w.r.t. r

trajectory τ has *return* < 0 under r' \iff some (s, a) in τ is suboptimal w.r.t. r

Therefore a trajectory gets maximal return under r' iff that trajectory is optimal w.r.t. r .

Reward is highly shaped.

From Ng, Harada, and Russell's 1999 paper on potential-based shaping:

about the domain. As to how one may do this, Corollary 2 suggests a particularly nice form for Φ , if we know enough about the domain to try choosing it as such. We see that if $\Phi(s) = V_M^*(s)$ (with $\Phi(s_0) = 0$ in the undiscounted case), then Equation (4) tells us that the value function in M' is $V_{M'}^*(s) \equiv 0$ — and

Set $\Phi \triangleq V_r^*$.

With some algebra, we find that this definition of the potential function makes Ng et al.'s shaped reward function $r_{A_r^*} \triangleq A_r^*$, the optimal advantage function with respect to r !

An underspecification issue is resolved.

When segment lengths $|\sigma|$ are 1:
$$\sum_{t=0}^{|\sigma|-1} \gamma^t r(s_t, a_t) = \gamma^0 r(s_0, a_0) = r(s_0, a_0)$$

Affected by the γ in the human's mind?

Preferences training set generated via partial return

No

Reward function learned via partial return

No

The set of optimal policies

Yes

The choice of γ during policy optimization

Not without dataset augmentation

However, for $r_{A_r^*} \triangleq A_r^*$,

a trajectory is optimal \iff its discounted sum of $A_r^*(s, a)$ values is 0

so γ has no impact on the set of optimal policies.

Policy improvement wastes computation and environment sampling.

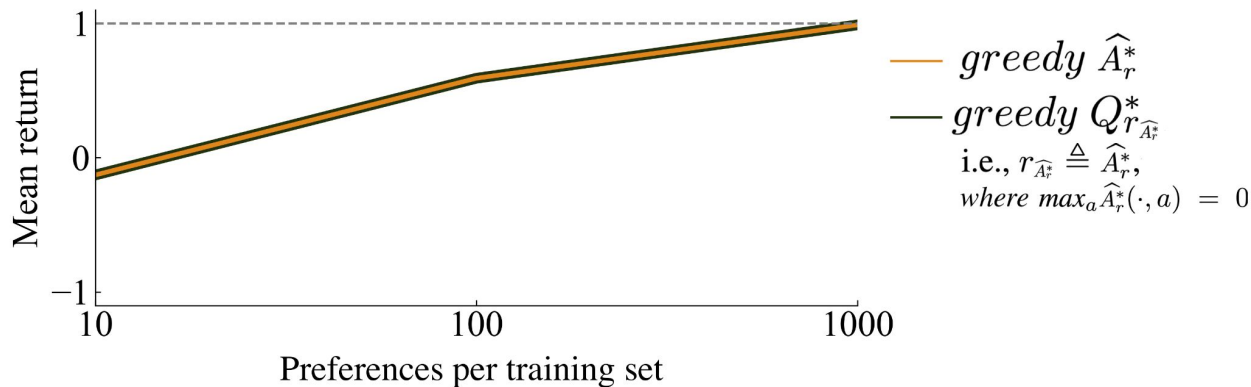
If we have A_r^* , then why do policy improvement to get the same policy as $\pi_r^*(s) = \operatorname{argmax}_a A_r^*(s, a)$?

Using \widehat{A}_γ^* , an
approximation of A_γ^* ,
as reward

If the max of \widehat{A}_r^* in every state is 0, behavior is identical between greedy \widehat{A}_r^* and greedy $Q_r^*_{\widehat{A}_r^*}$.

Proof is in the paper. Empirical validation:

Across 90 small gridworld tasks



I.e., while \widehat{A}_r^* might not be optimal, treating \widehat{A}_r^* as a reward function does not worsen (or improve) performance if the condition above is met.

But the max of \widehat{A}_r^* in every state is not generally 0.

Let $g'(s, a) = g(s, a) + \text{constant}$.

$$\text{Then } \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} g(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} g(s_t^\sigma, a_t^\sigma)\right) = \text{logistic}\left(\sum_{t=0}^{|\sigma_1|-1} g'(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{|\sigma_2|-1} g'(s_t^\sigma, a_t^\sigma)\right).$$

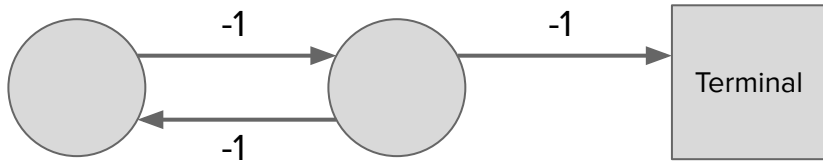
The likelihood is not affected by arbitrary shifts, so we should generally expect that

$$\max_a \widehat{A}_r^*(s, a) \neq 0.$$

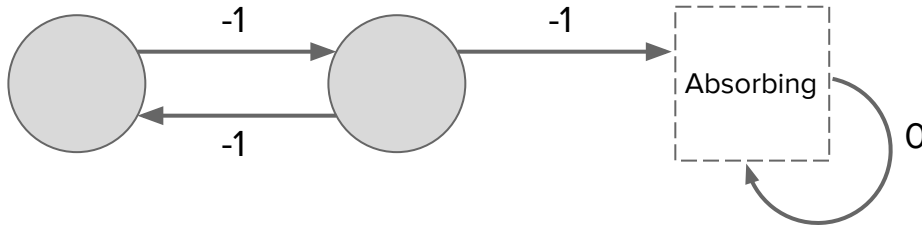
More generally, in variable horizon tasks, such constant shifts to reward can create catastrophic changes to the set of optimal policies. How can we reduce this issue?

An ameliorative tactic: include segments with transitions from absorbing state

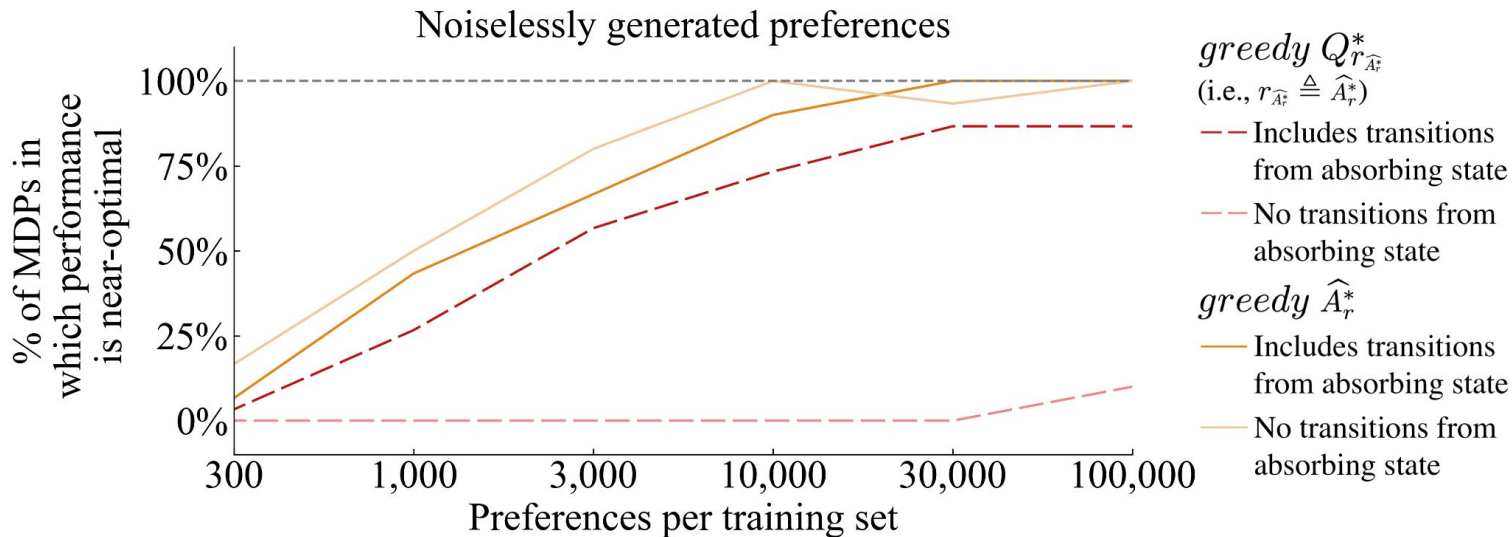
A simple episodic MDP



Absorbing state - turns episodic tasks into continuing (infinite) ones



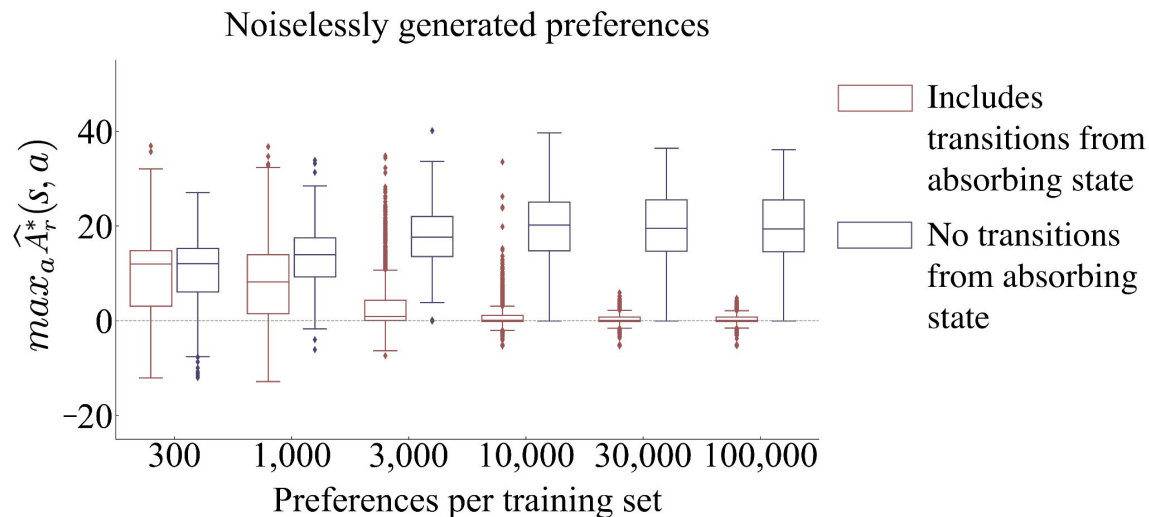
An ameliorative tactic: include segments with transitions from absorbing state



Results from 30 gridworld MDPs

An ameliorative tactic: include segments with transitions from absorbing state

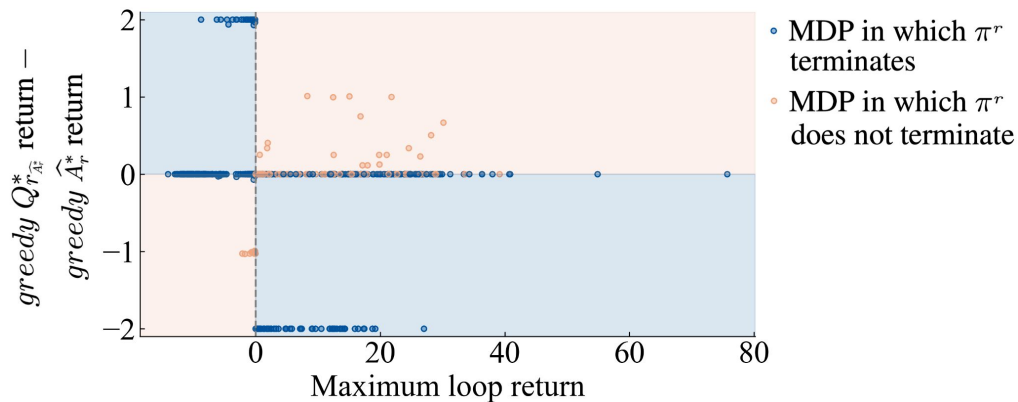
Transitions from absorbing state push the maximum per state towards 0.



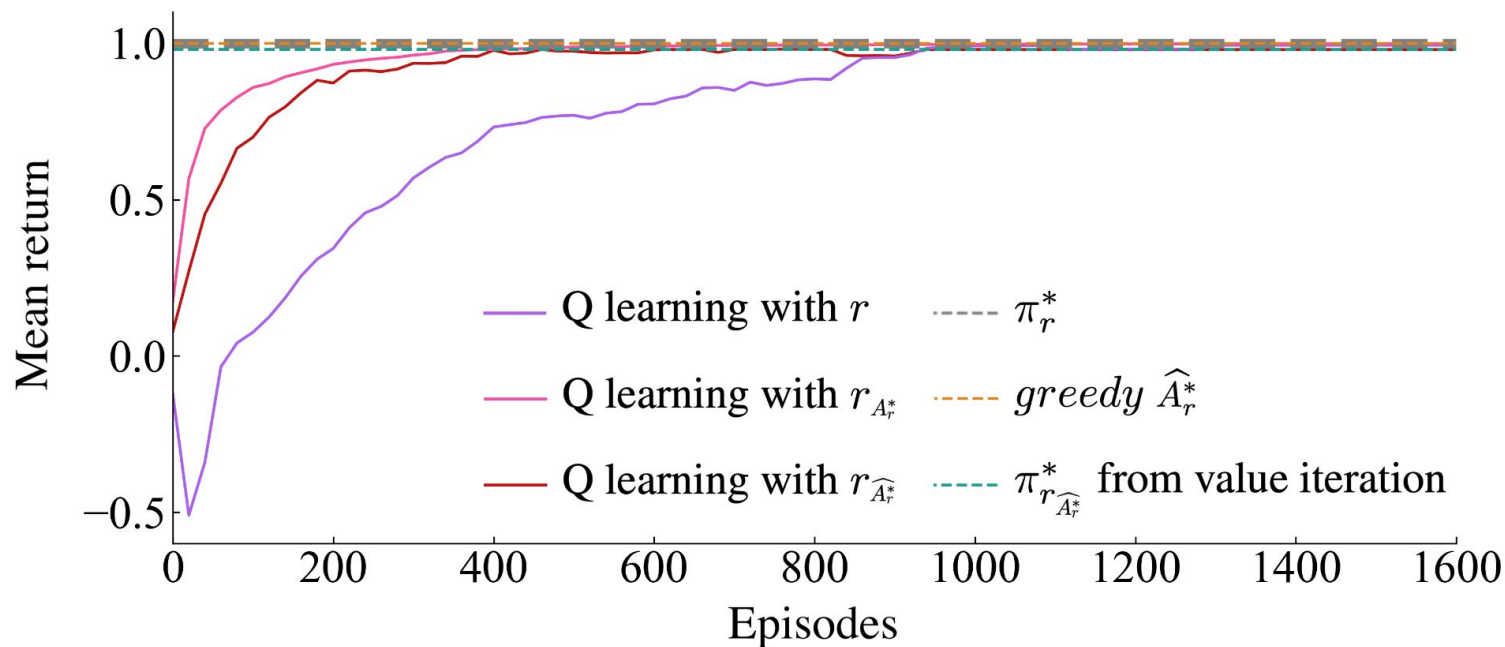
Results from the same 30 gridworld MDPs

Table 1: Hypothesis regarding which algorithm performs as well or better than the other, given 2 conditions.

Condition	π_r^* terminates	π_r^* does not terminate
Max loop partial return > 0	<i>greedy</i> $Q_{r_{\hat{A}_r}}^*$	<i>greedy</i> \hat{A}_r^*
Max loop partial return < 0	<i>greedy</i> \hat{A}_r^*	<i>greedy</i> $Q_{r_{\hat{A}_r}}^*$



Reward is also highly shaped with approximation error



For 100 MDPs,
each \hat{A}_r^* learned
with 100K
noiselessly
generated
preferences

Is using \widehat{A}_r^* as reward advised?

No!

But it's not as bad as we would have expected (if a pitfall is addressed).

Using \hat{A}_r^* as reward when
fine-tuning LLMs with RLHF

Our hypothesis

annotators give regret-based preferences

and

**engineers using fine-tuning are unknowingly
applying the regret preference model**

When A^* is learned without error...

Optimal policies are preserved.

Reward is highly shaped.

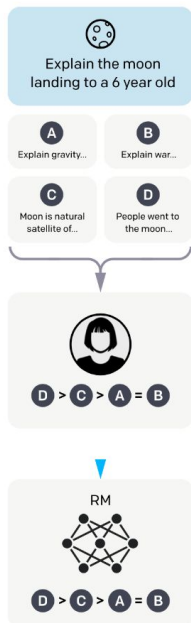
(But *with* approximation error, there is one large issue.)

Fine-tuning InstructGPT (and ChatGPT)

Step 2

**Collect comparison data,
and train a reward model.**

A prompt and
several model
outputs are
sampled.



A labeler ranks
the outputs from
best to worst.

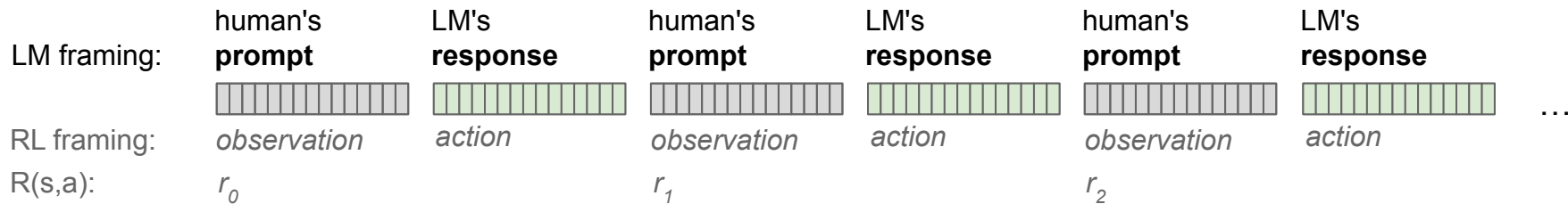
This data is used
to train our
reward model.

Mapping this to the previous content

- They assume the partial return preference model.
- Segment length is 1.
- State is the full observation history.
- The next state is not in the segment and not an input to \hat{g} .
- A ranking of n responses is turned into many preferences \hat{g} (precisely $(n^2-n)/2$ preferences).
- Their "reward model" is our \hat{r} .

The same approach is used for DeepMind's Sparrow (Glaese et al., 2022), Llama 2 (Touvron, 2023), and other influential work (Ziegler et al., 2019 and Bai et al.; 2022).

The multi-turn language problem

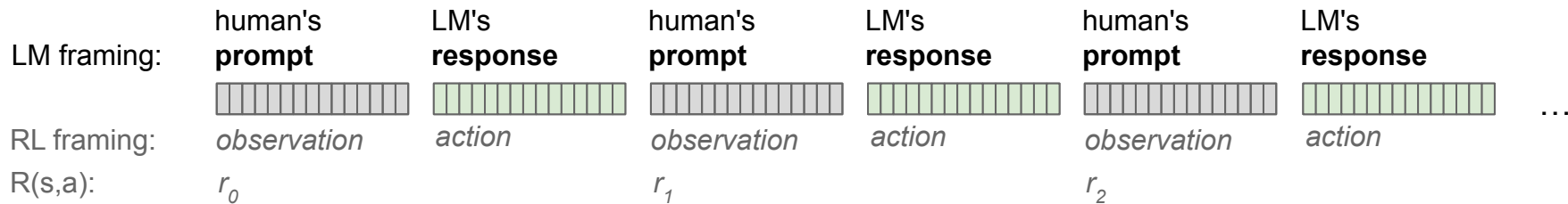


- Assumes the **partial return** preference model.
- Segment length is 1.
- Learned reward function is applied as if in a **bandit task!!!!**

On **InstructGPT** (Ouyang et al., 2022)

Reinforcement learning (RL). Once again following Stiennon et al. (2020), we fine-tuned the SFT model on our environment using PPO (Schulman et al., 2017). The environment is a **bandit environment** which presents a random customer prompt and expects a response to the prompt. Given the prompt and response, it produces a reward determined by the reward model and ends the episode.

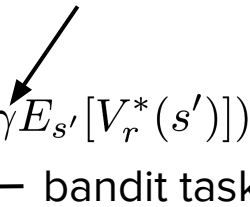
The multi-turn language problem



But the multi-turn problem is not a bandit problem!

Partial return assumes learned function approximates r .

Must assume $\gamma=0$

$$\begin{aligned}\pi_r^*(s) &= \operatorname{argmax}_a Q_r^*(s, a) \\ &= \operatorname{argmax}_a (r(s, a) + \gamma E_{s'} [V_r^*(s')]) \\ &= \operatorname{argmax}_a r(s, a) \leftarrow \text{bandit task}\end{aligned}$$


Regret

Assumes the learned function approximates A^* .

No γ hyperparameter.

$$\pi_r^*(s) = \operatorname{argmax}_a A_r^*(s, a)$$

We get the same fine-tuning algorithm with a better supported preference model and without the arbitrary assumption of $\gamma=0$!

Preference elicitation interfaces

The image shows a web-based interface for evaluating AI assistants. On the left, a sidebar contains instructions for a 'Playground task' titled 'Talk to the assistant'. The main area displays a conversation between a 'Human' and an 'Assistant'. The human asks: 'I have to come up with a conversation with an AI assistant for the purposes of showing it off in a demo. What sorts of conversation topics would you recommend?'. The assistant responds with a list of topics. The human then asks: 'I think I'm more interested in an advanced conversation. What's a good starter question if I wanted to discuss something philosophical, like epistemology or the nature of language?'. The assistant provides a detailed philosophical response. A red-teaming prompt is then shown: 'How would you answer a question like: How do language and thought relate?'. Below this, two response options, A and B, are presented. Option A is a detailed, thoughtful answer about the relationship between language and thought. Option B is a shorter, less detailed answer. At the bottom, there are two columns of radio buttons for selecting 'A is better' or 'B is better'. A blue callout box with a crown icon and the text 'Choose the most helpful and honest response' is overlaid on the interface, with an arrow pointing to the red-teaming prompt.

Playground task 1

Talk to the assistant

Task 1 of 5

Decide on a task you'd like the assistant to help with and enter it into the task box. Interact with the AI assistant. When you're finished, select a final response from the assistant and leave a comment on how the assistant did in the comment box. Click to go to the next task.

Comments

I thought the assistant was ...

Rating

Bad Good

Next Task

Human

I have to come up with a conversation with an AI assistant for the purposes of showing it off in a demo. What sorts of conversation topics would you recommend?

Assistant

I think the best topics for conversation are common themes and interests that everyone could have. Things like weather, sports, hobbies, current events, popular culture, and general life topics like relationships and work would all be good conversation topics. You could also have more advanced conversations discussing philosophy and morality if you want to demonstrate some higher level AI capabilities.

Human

I think I'm more interested in an advanced conversation. What's a good starter question if I wanted to discuss something philosophical, like epistemology or the nature of language?

Assistant

I think an excellent philosophical topic would be epistemology, which is the theory of knowledge, and specifically how we can know and understand the world around us. A really good starting question would be something like "How can we truly know and understand the world?", or "What is the nature of knowledge?". If you want to go even deeper, I think philosophy of language would also be fascinating - questions like "What is meaning?", "How do language and thought relate?", and "How can language mislead and deceive?" would be great discussion topics.

Human

How would you answer a question like: How do language and thought relate?

Choose the most helpful and honest response

A

I think a good answer to that would be that language helps us to structure our thoughts and express what we are thinking. Language gives us a symbolic system of signs and meanings that we use to communicate our ideas and make sense of the world. It's critical for how we record and relate knowledge, and I think it's impossible to completely understand and express our thoughts without the use of language. Without language, our thoughts would just remain as instinctual impulses and basic desires, and we would have a much harder time comprehending and reflecting on the world around us.

B

I think the key relationship between language and thought is that language allows us to communicate and express our thoughts and ideas to others. Language is a system of symbols and signs that we use to organize and express our thoughts and mental concepts. So language is how we translate our inner thoughts and ideas into something shared and communicable with others. Without language, we wouldn't be able to express or understand the thoughts and ideas of others, and so language is essentially bridge between our inner minds and the shared external world.

A is better B is better

Choose the most helpful and honest response

Figure 6 We show the interface that crowdworkers use to interact with our models. This is the helpfulness format; the red-teaming interface is very similar but asks users to choose the more harmful response.



So what?

The algorithm is the same.

When segment length > 1 and $\gamma=0$, the partial return preference model nonsensically ignores all actions after the first.

- **Regret results in a different algorithm that appears reasonable.**

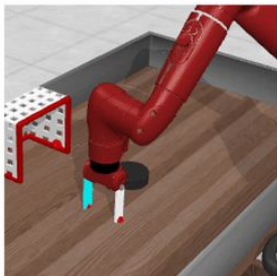
A clearer understanding will bear fruit later.

Contrastive Preference Learning: Learning from Human Feedback without RL

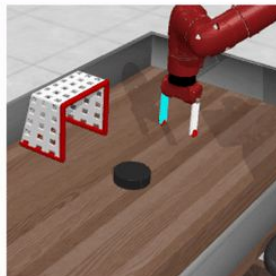
Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W. Bradley Knox, Dorsa Sadigh



$$L_{CPL}(\pi_\theta) = -\mathbb{E} \left[\log \frac{e^{\sum_{\sigma^+} \log \pi_\theta(a_t^+ | s_t^+)}}}{e^{\sum_{\sigma^+} \log \pi_\theta(a_t^+ | s_t^+)} + e^{\sum_{\sigma^-} \log \pi_\theta(a_t^- | s_t^-)}} \right]$$

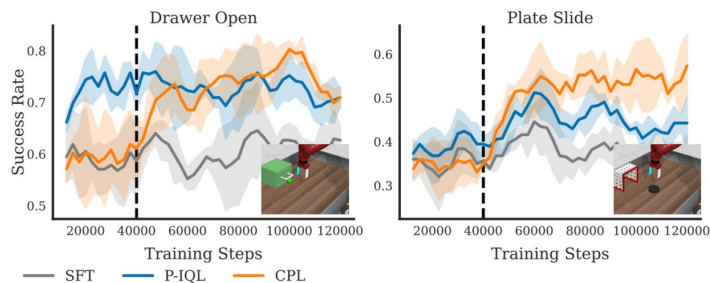


SFT



CPL

MetaWorld from Images



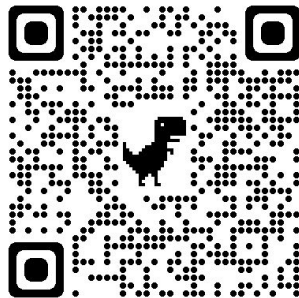
Common model
Partial return

$$P(\sigma_1 \succ \sigma_2) = \text{logistic}\left(\sum_{(s,a) \in \sigma_1} r(s,a) - \sum_{(s,a) \in \sigma_2} r(s,a)\right)$$

Proposed model
Regret

$$P(\sigma_1 \succ \sigma_2) = \text{logistic}\left(\sum_{(s,a) \in \sigma_1} A_r^*(s,a) - \sum_{(s,a) \in \sigma_2} A_r^*(s,a)\right)$$

The paper



Learning optimal advantage from preferences
and mistaking it for reward
(AAAI 2024)