Learning Optimal Advantage from Preferences and Mistaking it for Reward

W. Bradley Knox\textsuperscript{1,4}  Stephane Hatgis-Kessell\textsuperscript{1}  Sigurdur Orn Adalgeirsson\textsuperscript{4}  Serena Booth\textsuperscript{2}  Anca Dragan\textsuperscript{5}  Scott Niekum\textsuperscript{6}  Peter Stone\textsuperscript{1,3}

\textsuperscript{1}UT Austin  \textsuperscript{2}MIT CSAIL  \textsuperscript{3}Sony AI  \textsuperscript{4}Google Research  \textsuperscript{5}UC Berkeley  \textsuperscript{6}UMass Amherst
The model of preference

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

(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

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**

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

-1

Indifferent!
**Equal partial return**

**Lower end state value**

**Equal partial return**

**Higher end state value**

**Suboptimal segment**

**Optimal segment**

**Equal partial return**

**Higher start state value**

**Equal partial return**

**Lower start state value**

**Suboptimal segment**

**Optimal segment**
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

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

Showing reward

Regret

\[ P(\sigma_1 > \sigma_2) = \text{logistic} \left( \sum_{(s,a)\in \sigma_1} A^*_r(s,a) - \sum_{(s,a)\in \sigma_2} A^*_r(s,a) \right) \]

Showing optimal advantage

Indifferent

Preferred
The preference model

Proposed model: Regret

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

Showing optimal advantage

[Diagram showing two grids with different outcomes and the preferred one marked]
The preference model

Proposed model: Regret

\[ P(\sigma_1 \succ \sigma_2) = \text{logistic}\left( \sum_{(s,a) \in \sigma_1} A^*_r(s, a) - \sum_{(s,a) \in \sigma_2} A^*_r(s, a) \right) \]
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 $A^*_{\gamma}$ as $\gamma$)

Outline:
- When $A^*_{\gamma}$ is known exactly
- When $A^*_{\gamma}$ is approximated
- Reframing RLHF for LLMs
Assuming the partial return preference model when regret is correct

(Learning $A^*_\gamma$ and using it as $\gamma$)
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 \( \sigma \)

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

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

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

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

\[ = \text{logistic} \left( \sum_{t=0}^{\left| \sigma_1 \right|-1} \tilde{r}(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\left| \sigma_2 \right|-1} \tilde{r}(s_t^\sigma, a_t^\sigma) \right) \quad \text{Partial return} \]

\[ = \text{logistic} \left( \sum_{t=0}^{\left| \sigma_1 \right|-1} A_\tilde{r}^*(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\left| \sigma_2 \right|-1} A_\tilde{r}^*(s_t^\sigma, a_t^\sigma) \right) \quad \text{Regret} \]

\[ = \text{logistic} \left( \sum_{t=0}^{\left| \sigma_1 \right|-1} g(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\left| \sigma_2 \right|-1} g(s_t^\sigma, a_t^\sigma) \right) \quad \text{Unification} \]

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

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

\[ = \text{logistic}\left( \sum_{t=0}^{\lfloor |\sigma_1| - 1 \rfloor} \tilde{r}(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\lfloor |\sigma_2| - 1 \rfloor} \tilde{r}(s_t^\sigma, a_t^\sigma) \right) \text{ Partial return} \]

\[ = \text{logistic}\left( \sum_{t=0}^{\lfloor |\sigma_1| - 1 \rfloor} A^*_\tilde{r}(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\lfloor |\sigma_2| - 1 \rfloor} A^*_\tilde{r}(s_t^\sigma, a_t^\sigma) \right) \text{ Regret} \]

\[ = \text{logistic}\left( \sum_{t=0}^{\lfloor |\sigma_1| - 1 \rfloor} g(s_t^\sigma, a_t^\sigma) - \sum_{t=0}^{\lfloor |\sigma_2| - 1 \rfloor} g(s_t^\sigma, a_t^\sigma) \right) \text{ Unification} \]
## 3 algorithms

<table>
<thead>
<tr>
<th>Dataset created by reward function $r'$ and preference model</th>
<th>Algorithm for learning from preferences</th>
<th>Output of learning from preferences</th>
<th>Additional step to create policy (other than greedy action selection)</th>
</tr>
</thead>
<tbody>
<tr>
<td>partial return preference model</td>
<td>learning $g$</td>
<td>$\hat{r}$</td>
<td>policy improvement</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning by regret algorithm</td>
<td>$\hat{r}$</td>
<td>policy improvement</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning $g$</td>
<td>$\hat{A}_r^*$</td>
<td>nothing</td>
</tr>
</tbody>
</table>
3 algorithms

<table>
<thead>
<tr>
<th>Dataset created by reward function $\gamma'$ and</th>
<th>Algorithm for learning from preferences</th>
<th>Output of learning from preferences</th>
<th>Additional step to create policy (other than greedy action selection)</th>
</tr>
</thead>
<tbody>
<tr>
<td>partial return preference model</td>
<td>learning $g$</td>
<td>$\hat{\gamma}$</td>
<td>policy improvement</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning by regret algorithm</td>
<td>$\hat{\gamma}'$</td>
<td>$\hat{\pi}'$</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning $g$</td>
<td>$\hat{A}'$</td>
<td>nothing</td>
</tr>
</tbody>
</table>

$\hat{\pi}'$
### 4 algorithms

<table>
<thead>
<tr>
<th>Dataset created by reward function $\gamma'$ and regret preference model</th>
<th>Algorithm for learning from preferences</th>
<th>Assumed Output of learning from preferences</th>
<th>Additional step to create policy (other than greedy action selection)</th>
</tr>
</thead>
<tbody>
<tr>
<td>learning $g$</td>
<td>$\hat{\gamma}$</td>
<td>policy improvement</td>
<td>$\hat{\pi}_\gamma$</td>
</tr>
<tr>
<td>learning by regret algorithm</td>
<td>$\hat{\gamma}$</td>
<td>policy improvement</td>
<td>$\hat{\pi}_\gamma$</td>
</tr>
<tr>
<td>learning $g$</td>
<td>$\hat{A}_\gamma$</td>
<td>nothing</td>
<td>$\hat{\pi}_\gamma$</td>
</tr>
</tbody>
</table>
4 algorithms

Dataset created by reward function \( \hat{r} \) and

- **regret**
  - partial return
  - preference model

Algorithm for learning from preferences

- learning \( g \)

Assumed Output of learning from preferences

- \( \hat{r} \)

Additional step to create policy (other than greedy action selection)

- policy improvement

- \( \hat{\pi}_r \)

- learning by regret algorithm

- \( \hat{r} \)

- policy improvement

- \( \hat{\pi}_r \)

- learning \( g \)

- \( \hat{A}_r^* \)

- nothing

- \( \hat{\pi}_r \)
## 4 algorithms

<table>
<thead>
<tr>
<th>Dataset created by reward function ( r' ) and partial return preference model</th>
<th>Algorithm for learning from preferences</th>
<th>Assumed Output of learning from preferences</th>
<th>Additional step to create policy (other than greedy action selection)</th>
</tr>
</thead>
<tbody>
<tr>
<td>regret preference model</td>
<td>learning ( g )</td>
<td>( \hat{r} )</td>
<td>policy improvement</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning by regret algorithm</td>
<td>( \hat{\gamma} )</td>
<td>policy improvement</td>
</tr>
<tr>
<td>regret preference model</td>
<td>learning ( g )</td>
<td>( \hat{A}_r^* )</td>
<td>nothing</td>
</tr>
</tbody>
</table>

**greedy** \( \hat{A}_r^* \)
4 algorithms

Dataset created by reward function $r^*$ and partial return preference model

Algorithm for learning from preferences

Assumed Output of learning from preferences

Additional step to create policy (other than greedy action selection)

\[
greedy \quad Q^*_r A^*_r
\]
Using $A^*_r$ as reward
Optimal policies are preserved.

The set of optimal policies under \( r \) and \( r_A^* \triangleq A_r^* \) is the same, regardless of the discount factor used with \( r A_r^* \).

Intuition:

\[
A_r^*(s, a) = 0 \iff (s, a) \text{ is optimal w.r.t. } r
\]

\[
A_r^*(s, a) < 0 \iff (s, a) \text{ is suboptimal w.r.t. } r
\]

so:

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

trajectory \( \tau \) has return \( < 0 \) under \( r' \) \iff some \((s, a)\) in \( \tau \) 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 \( \Phi \), 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 \( r_{A^*} \triangleq A_{r^*} \), the optimal advantage function with respect to \( r^* \)!
An underspecification issue is resolved.

When segment lengths $|\sigma|$ are 1:

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

| Preferences training set generated via partial return | No |
| Reward function learned via partial return | No |
| The set of optimal policies | Yes |
| The choice of $\gamma$ 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 $\gamma$ 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 $\pi_r^*(s) = \arg\max_a A_r^*(s, a)$?
Using $\hat{A}^*_r$, an approximation of $A^*_r$, as reward
If the max of $\hat{A}_r^*$ in every state is 0, behavior is identical between greedy $\hat{A}_r^*$ and greedy $Q_r^*$. 

Proof is in the paper. Empirical validation:

*Across 90 small gridworld tasks*

\[\text{Mean return} \begin{cases} 1 & \text{if greedy } \hat{A}_r^* \\ 0 & \text{if greedy } Q_r^* \end{cases}
\]

\[\text{i.e., } r_{\hat{A}} \triangleq \hat{A}_r^*, \quad \text{where } \max_a \hat{A}_r^*(\cdot, a) = 0\]

I.e., while $\hat{A}_r^*$ might not be optimal, treating $\hat{A}_r^*$ 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 $g'(s, a) = g(s, a) + \text{constant}$.

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.

<table>
<thead>
<tr>
<th>Condition</th>
<th>$\pi^*_r$ terminates</th>
<th>$\pi^*_r$ does not terminate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max loop partial return &gt; 0</td>
<td>greedy $Q^*_r$</td>
<td>greedy $\tilde{A}^*_r$</td>
</tr>
<tr>
<td>Max loop partial return &lt; 0</td>
<td>greedy $\tilde{A}^*_r$</td>
<td>greedy $Q^*_r$</td>
</tr>
</tbody>
</table>

![Graph showing the relationship between greedy $Q^*_r$ and greedy $\tilde{A}^*_r$ returns vs. maximum loop return.

- Blue dots: MDP in which $\pi^r$ terminates
- Orange dots: MDP in which $\pi^r$ does not terminate]
Reward is also highly shaped with approximation error

For 100 MDPs, each $\hat{A}_r^*$ learned with 100K noiselessly generated preferences.
Is using $\hat{A}_r^*$ as reward advised?

No!

But it's not as bad as we would have expected (if a pitfall is addressed).
Using $A_r^*$ 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.)
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 .
- A ranking of n responses is turned into many preferences (precisely \((n^2-n)/2\) preferences).
- Their "reward model" is our .

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

<table>
<thead>
<tr>
<th>LM framing:</th>
<th>human's prompt</th>
<th>LM's response</th>
<th>human's prompt</th>
<th>LM's response</th>
<th>human's prompt</th>
<th>LM's response</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL framing:</td>
<td>observation</td>
<td>action</td>
<td>observation</td>
<td>action</td>
<td>observation</td>
<td>action</td>
<td></td>
</tr>
<tr>
<td>R(s,a):</td>
<td>( r_0 )</td>
<td></td>
<td>( r_1 )</td>
<td></td>
<td>( r_2 )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RL framing:</td>
<td>observation</td>
<td>action</td>
<td>observation</td>
<td>action</td>
<td>observation</td>
<td>action</td>
</tr>
<tr>
<td>R(s,a):</td>
<td>$r_0$</td>
<td></td>
<td>$r_1$</td>
<td></td>
<td>$r_2$</td>
<td></td>
</tr>
</tbody>
</table>

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

**Partial return** assumes learned function approximates $r$.

$$\pi^*_r(s) = \arg\max_a Q^*_r(s, a)$$

$$= \arg\max_a (r(s, a) + \gamma E_s[V^*_r(s')])$$

$$= \arg\max_a r(s, a) \quad \text{bandit task}$$
Regret
Assumes the learned function approximates $A^*$. No $\gamma$ hyperparameter.

$$\pi^*_r(s) = \arg\max_a A^*_r(s, a)$$

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

![Image of interface]

**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.

Bai et al., 2022
So what?
The algorithm is the same.
When segment length > 1 and γ=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

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

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