Learning Optimal Advantage from Preferences and Mistaking it for Reward

W. Bradley Knox [1 2] **Stephane Hatgis-Kessell** [1] **Sigurdur Orn Adalgeirsson** [2] **Serena Booth** [3] **Anca Dragan** [4] **Peter Stone** [1 5] **Scott Niekum** [6]

Abstract

We consider algorithms for learning reward functions from human preferences over pairs of trajectory segments—as used in reinforcement learning from human feedback (RLHF)—including those used to fine tune ChatGPT and other contemporary language models.[1] Most recent work on such algorithms assumes that human preferences are generated based only upon the reward accrued within those segments, which we call their partial return function. But if this assumption is false because people base their preferences on information other than partial return, then what type of function *is* their algorithm learning from preferences? We argue that this function is better thought of as an approximation of the optimal advantage function, *not* as a partial return function as previously believed.

1. Introduction

The dominant model of human preference is the **partial return** preference model, wherein preferences are determined only by accumulated reward during each segment. Knox et al. (2022) recently argued that the partial return preference model has fundamental flaws that are removed or ameliorated by a model of human preference based upon the **optimal advantage** of each segment, therein cast as (negated) **regret**. Optimal advantage is a measure of deviation from optimal decision-making.

This past work provides arguments for the superiority of the regret preference model (1) by intuition, regarding how humans are likely to give preferences (e.g., see Figure 2); (2) by theory, regarding how we would normatively want humans to give preference labels to improve the alignment of the resulting learned reward function with their true preferences; (3) by descriptive analysis, showing that the likelihood of a human preferences dataset is higher under the regret preference model than under the partial return preference model; and (4) and by empirical results, showing that with both human and synthetic preference labels, the regret preference model is consistently more sample efficient, requiring fewer preference labels to reach a certain level of performance. Section 2 of this paper provides details on the general problem setting and on these two models.

In Section 3, we assume that preference labels are based upon regret and then explore the consequences for these common algorithms that assume the partial return preference model. The discussion is built from the insight that these algorithms are using optimal advantage as reward.

We then show in Section 4 that recent algorithms used to fine-tune state-of-the-art language models ChatGPT (OpenAI, 2022), Sparrow (Glaese et al., 2022), and others (Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022) can be viewed as an instance of learning an optimal advantage function and, surprisingly (because their authors describe them as learning a reward function), treating it as one. In multi-turn (i.e., sequential) settings such as that of ChatGPT, Sparrow, and research by Bai et al. (2022), this alternative framing allows the removal of an arbitrary and perplexing assumption of these algorithms: that a reward function learned for a sequential task is instead used in a bandit setting, effectively setting the discount factor γ to 0.

2. Preliminaries: Preference models for learning reward functions

The background from here through Section 2.2 is lightly modified text from Knox et al. (2022) (used with permission).

We assume that the task environment is a Markov decision process (MDP) specified by the tuple $(S, A, T, \gamma, D_0, r)$. S and A are the sets of possible states and actions, respectively. T is a transition function, $T : S \times A \to S$. γ is the discount factor and D_0 is the distribution of start states. Unless otherwise stated, we assume undiscounted tasks (i.e., $\gamma = 1$)

[1]University of Texas at Austin [2]Google Research [3]MIT CSAIL [4]UC Berkeley [5]Sony AI [6]University of Massachusetts Amherst. Correspondence to: W. Bradley Knox <bradknox@cs.utexas.edu>.

The Many Facets of Preference Learning Workshop at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023.

[1]They are a special case where segments include only one state-action pair.

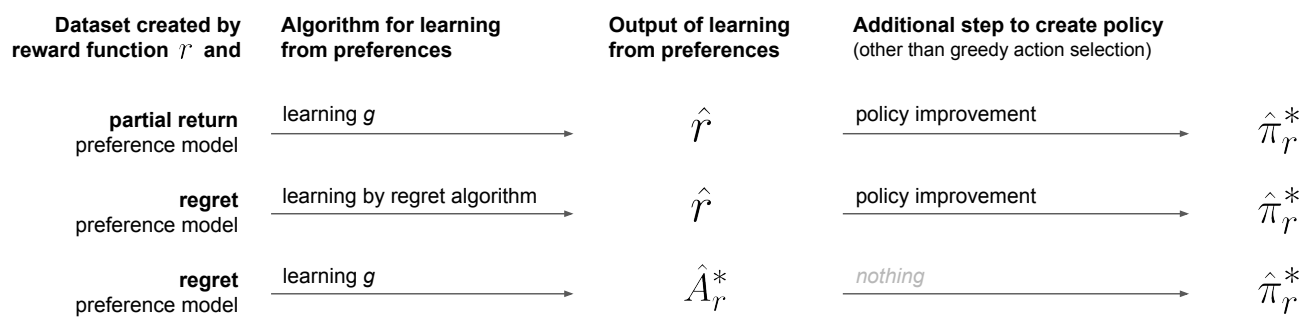

Figure 1. Three algorithms that are each justified by their assumed preference model. The top algorithm was popularized by Christiano et al. (2017) and the middle algorithm was proposed by Knox et al. (2022). Here we briefly discuss the bottom algorithm. The reward function $\hat{r}$, optimal advantage function $\hat{A}^*_r$, and optimal policy $\hat{\pi}^*_r$ are merely approximations of the true versions of these functions. The function g is defined in Equation 6 and allows for different interpretations of what the learned g function represents, including $\hat{A}^*_r$ or $\hat{\pi}^*_r$. This paper focuses on what occurs when one believes they are running the top algorithm but have actually learned $\hat{A}^*_r$ and are using it as the reward function.

that have terminal states, after which only 0 reward can be received. r is a reward function, $r : S \times A \times S \to \mathbb{R}$, where the reward r_t at time t is a function of s_t, a_t, and s_{t+1}. An MDP$\backslash r$ is an MDP without a reward function.

Throughout this paper, r refers to the ground-truth reward function for some MDP; $\hat{r}$ refers to a learned approximation of r; and $\tilde{r}$ refers to any reward function (including r or $\hat{r}$). A policy ($\pi : S \times A \to [0,1]$) specifies the probability of an action given a state. $Q^*_{\tilde{r}}$ and $V^*_{\tilde{r}}$ refer respectively to the state-action value function and state value function for an optimal policy, π^*, under $\tilde{r}$. The **optimal advantage function** is defined as $A^*_{\tilde{r}}(s,a) \triangleq Q^*_{\tilde{r}}(s,a) - V^*_{\tilde{r}}(s)$. Like our notation for reward, $\hat{Q}^*_{\tilde{r}}$, $\hat{V}^*_{\tilde{r}}$, and $\hat{A}^*_{\tilde{r}}$ express learned approximations of $Q^*_{\tilde{r}}$, $V^*_{\tilde{r}}$, and $A^*_{\tilde{r}}$ for some reward function $\tilde{r}$, and $\tilde{Q}^*_{\tilde{r}}$, $\tilde{V}^*_{\tilde{r}}$, and $\tilde{A}^*_{\tilde{r}}$ represent both approximations and ground-truth functions. Throughout this paper, the ground-truth reward function r is used to algorithmically generate preferences when they are not human-generated, is hidden during reward learning, and is used to evaluate the performance of optimal policies under a learned $\hat{r}$.

2.1. Reward learning from pairwise preferences

A reward function can be learned by minimizing the cross-entropy loss—i.e., maximizing the likelihood—of observed human preference labels, a common approach in recent literature (Christiano et al., 2017; Ibarz et al., 2018; Wang et al., 2022; Bıyık et al., 2021; Sadigh et al., 2017; Lee et al., 2021a;b; Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022; Glaese et al., 2022; OpenAI, 2022).

Segments Let σ denote a segment starting at state s^σ_0. Its length $|\sigma|$ is the number of transitions within the segment. A segment includes $|\sigma|+1$ states and $|\sigma|$ actions: $(s^\sigma_0, a^\sigma_0, s^\sigma_1, a^\sigma_1, ..., s^\sigma_{|\sigma|})$. In this problem setting, segments lack any reward information. As shorthand, we define $\sigma_t \triangleq (s^\sigma_t, a^\sigma_t, s^\sigma_{t+1})$ and allow functions to ignore the s^σ_{t+1} component of σ_t when their input does not include the next state. A segment σ is **optimal** with respect to $\tilde{r}$ if, for every $i \in \{1,...,|\sigma|\text{-}1\}$, $Q^*_{\tilde{r}}(s^\sigma_i, a^\sigma_i) = V^*_{\tilde{r}}(s^\sigma_i)$. A segment that is not optimal is **suboptimal**. Given some $\tilde{r}$ and a segment σ, $\tilde{r}_t \triangleq \tilde{r}(s^\sigma_t, a^\sigma_t, s^\sigma_{t+1})$, and the **partial return** of a segment σ is $\sum_{t=0}^{|\sigma|-1} \gamma^t \tilde{r}_t$, denoted in shorthand as $\Sigma_\sigma\, r$.

Preference datasets Each preference over a pair of segments creates a sample $(\sigma_1, \sigma_2, \mu)$ in a preference dataset $D_\succ$. Vector $\mu = \langle \mu_1, \mu_2 \rangle$ represents the preference; specifically, if σ_1 is preferred over σ_2, denoted $\sigma_1 \succ \sigma_2$, $\mu = \langle 1,0 \rangle$. μ is $\langle 0,1 \rangle$ if $\sigma_1 \prec \sigma_2$ and is $\langle 0.5,0.5 \rangle$ for $\sigma_1 \sim \sigma_2$ (no preference). We assume that the two segments in a sample are equally sized.

Loss function To learn a reward function from a preference dataset, $D_\succ$, a common assumption is that the preference labels were generated by a preference model P that arises from an unobservable *ground-truth* reward function r. We approximate r by minimizing cross-entropy loss to learn $\hat{r}$:

$$loss(\hat{r}, D_\succ) = \sum_{(\sigma_1,\sigma_2,\mu) \in D_\succ} -\mu_1 \log P(\sigma_1 \succ \sigma_2 | \hat{r}) + \mu_2 \log P(\sigma_1 \prec \sigma_2 | \hat{r}) \quad (1)$$

This loss is under-specified until $P(\sigma_1 \succ \sigma_2 | \hat{r})$ is defined, which is the focus of Knox et al. (2022) and this paper.

Preference models A preference model determines the probability of one trajectory segment being preferred over another, $P(\sigma_1 \succ \sigma_2 | \tilde{r})$. Preference models could be applied to model preferences provided by humans or other systems. Preference models can also directly generate preferences, and in such cases we refer to them as **preference generators**.

2.2. Two preference models: partial return and regret

Partial return The aforementioned recent work assumes human preferences are generated by a Boltzmann distribution over the two segments' partial returns, expressed here

as a logistic function:[2]

$$P_{\Sigma_r}(\sigma_1 \succ \sigma_2|\tilde{r}) = logistic\Big(\Sigma_{\sigma_1}\tilde{r} - \Sigma_{\sigma_2}\tilde{r}\Big). \quad (2)$$

Regret We introduce an alternative preference model that Knox et al. (2022) designed to reflect segments' deviations from optimal decision-making, based on the regret of each transition in a segment. We first focus on segments with deterministic transitions. For a transition (s_t, a_t, s_{t+1}) in a deterministic segment, $regret_{\text{d}}(\sigma_t|\tilde{r}) \triangleq V^*_{\tilde{r}}(s^\sigma_t) - [\tilde{r}_t + V^*_{\tilde{r}}(s^\sigma_{t+1})]$. For a full deterministic segment,

$$\begin{aligned} regret_d(\sigma|\tilde{r}) &\triangleq \sum_{t=0}^{|\sigma|-1} regret_d(\sigma_t|\tilde{r}) \\ &= V^*_{\tilde{r}}(s^\sigma_0) - (\Sigma_\sigma \tilde{r} + V^*_{\tilde{r}}(s^\sigma_{|\sigma|})), \end{aligned} \quad (3)$$

with the right-hand expression arising from cancelling out intermediate state values. Therefore, deterministic regret measures how much the segment reduces expected return from $V^*_{\tilde{r}}(s^\sigma_0)$. An optimal segment, σ^*, always has 0 regret, and a suboptimal segment, $\sigma^{\neg *}$, will always have positive regret.

Stochastic transitions, however, can result in $regret_d(\sigma^*|\hat{r}) > regret_d(\sigma^{\neg *}|\tilde{r})$, losing the property above. To retain it, we note that the effect on expected return of transition stochasticity from a transition (s_t, a_t, s_{t+1}) is $[\tilde{r}_t + V^*_{\tilde{r}}(s_{t+1})] - Q^*_{\tilde{r}}(s_t, a_t)$ and add this expression once per transition to get $regret(\sigma)$, removing the subscript d that refers to determinism. The regret for a single transition becomes $regret(\sigma_t|\tilde{r}) = [V^*_{\tilde{r}}(s^\sigma_t) - [\tilde{r}_t + V^*_{\tilde{r}}(s^\sigma_{t+1})]] + [[\tilde{r}_t + V^*_{\tilde{r}}(s^\sigma_{t+1})] - Q^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t)] = V^*_{\tilde{r}}(s^\sigma_t) - Q^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t) = -A^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t)$. Regret for a full segment is

$$\begin{aligned} regret(\sigma|\tilde{r}) &= \sum_{t=0}^{|\sigma|-1} regret(\sigma_t|\tilde{r}) \\ &= \sum_{t=0}^{|\sigma|-1} \Big[V^*_{\tilde{r}}(s^\sigma_t) - Q^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t)\Big] \\ &= \sum_{t=0}^{|\sigma|-1} -A^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t). \end{aligned} \quad (4)$$

The regret preference model is the Boltzmann distribution over negated regret:

$$\begin{aligned} &P_{regret}(\sigma_1 \succ \sigma_2|\tilde{r}) \\ &\quad \triangleq logistic\Big(regret(\sigma_2|\tilde{r}) - regret(\sigma_1|\tilde{r})\Big) \\ &\quad = logistic\Big(\sum_{t=0}^{|\sigma_1|-1} A^*_{\tilde{r}}(\sigma^1_t) - \sum_{t=0}^{|\sigma_2|-1} A^*_{\tilde{r}}(\sigma^2_t)\Big). \end{aligned} \quad (5)$$

(Recall $A^*_{\tilde{r}}(\sigma_t) = A^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t)$.) Lastly, we note that if two segments have deterministic transitions, end in terminal states, and have the same starting state, this regret model reduces to the partial return model: $P_{regret}(\cdot|\tilde{r}) = P_{\Sigma_r}(\cdot|\tilde{r})$.

[2]Unless otherwise stated, we ignore the temperature because scaling reward has the same effect.

Algorithms in this paper All algorithms for learning from preferences can be summarized simply as "minimize Equation 1". The two algorithms for learning reward functions differ only in how the preference probabilities are calculated.

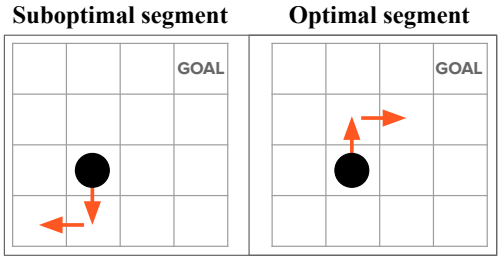

Figure 2. Two segments in an undiscounted task with -1 reward each time step. The partial return of both segments with respect to the true reward function is -2. The regret of the left segment is 4. The right segment is optimal and therefore has a regret of 0. The regret preference model is more likely to prefer the right segment—as we suspect our human readers are—whereas the partial return preference model is equally likely to prefer each segment.

2.3. Past results comparing these preference models

In addition to intuitively considering how each of these preference models fit human preferences (see Figure 2), Knox et al. (2022)'s main analyses of these two preference models focus on two questions.

Normative analysis First, they ask which preference model is a more desirable generator of preferences, ignoring how well a human can follow the preference model.

Part of their analysis focuses on the property of *reward identifiability*. Reward identifiability holds if, for any MDP, an exhaustively infinite dataset labeled by a preference model will always contain the information needed to recover a reward function with the same set of optimal policies as the ground-truth reward function underlying the preferences. Knox et al. prove that the regret preference model has reward identifiability with either noiseless or stochastic preferences. And they prove that the partial return preference model lacks reward identifiability for noiseless preferences.

Additionally, they consider how performant reward functions are that are learned by each preference model under the seemingly ideal conditions of having each model generate its own preferences dataset, making each preference model exactly correct for its own training set. On 100 randomly generated grid world MDPs, near-optimal performance is defined as having mean return that is no more than 10% of the distance from an optimal policy's mean return to

a uniformly random policy's mean return. They find that learning with the regret preference model results in a near-optimal policy more often in these MDPs than with the partial return preference model, consistently across training set sizes from 3 to 3000 preferences.

Analysis with human-generated preferences Second, Knox et al. ask which preference model is more compatible with human-generated preferences. On a rich grid world MDP, they gathered a dataset of 1812 preferences from 50 subjects. This dataset has higher likelihood under the regret preference model than under the partial return preference model. Although learning a reward function with either model reaches near-optimal performance on the full human preferences dataset, when presented with smaller subsets of the dataset, the regret preference model more frequently produces a near-optimal policy.

Based on both types of analysis, they thus conclude that the regret preference model is superior to the (heretofore standard in the literature) partial return model. However, they do not examine the implications of assuming that people are providing preference labels based on the partial return model when they are actually doing so based on the regret model. That is the question examined in this paper.

3. Learning optimal advantage from preferences and using it as reward

Here we examine what is actually learned when preferences are assumed to arise from partial return but actually come from regret (Equation 2). To start, let us unify the two preference models from Section 2.2 into a single general preference model.

$$P(\sigma_1 \succ \sigma_2|\tilde{r}) = logistic\Big(\sum_{t=0}^{|\sigma_1|-1} g(\sigma_t^1) - \sum_{t=0}^{|\sigma_2|-1} g(\sigma_t^2)\Big) \quad (6)$$

In the above unification, the segment statistic in the preference model is expressed as a sum of some function g over each transition in the segment: $\sum_{t=0}^{|\sigma|-1} g(\sigma_t) = \sum_{t=0}^{|\sigma|-1} g(s_t^\sigma, a_t^\sigma, s_{t+1}^\sigma)$. When the partial return preference model generates preferences, $g(\sigma_t) = \tilde{r}(s_t^\sigma, a_t^\sigma, s_{t+1}^\sigma)$, and the parameters of the reward function $\tilde{r}$ are learned via Equation 1.

However, evidence suggests that *human* preferences are more likely to be generated by the regret preference model (Knox et al., 2022). When the regret preference model generates preferences, we instead define $g(\sigma_t) = A_{\tilde{r}}^*(\sigma_t) = A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma)$ and the parameters of this optimal advantage function can be learned directly, also via Equation 1. This creates the bottom algorithm of Figure 1, deviating from the middle algorithm there by Knox et al., for learning a reward function. No reward function is represented or learned, though we still assume that preferences in the training set were generated by regret under the hidden reward function r. Therefore, the A_r^* that is estimated as $\hat{A}_r^*$ is the advantage function for an optimal policy under this hidden, preference-generating reward function, r.

3.1. Using A_r^* as a reward function

Under our assumption of regret-based preferences, learning a reward function with the partial return preference model—the dominant method of RLHF—effectively uses an approximation of A_r^* as a reward function, $\hat{r} = \hat{A}_r^*$. Let us first assume perfect inference of A_r^* (i.e., that $\hat{A}_r^* = A_r^*$), and consider the consequences.

Optimal policies are preserved. Contrary to our initial intuition, using A_r^* as a reward function preserves the set of optimal policies. The proof sketch follows.

Let M be the ground-truth MDP, with r, and M' be M modified such that its reward function is $r' = A_r^*$. In M, $A_r^*(s,a) = 0$ if and only if action a is optimal in state s under the ground-truth reward function r. Otherwise, $A_r^*(s,a) < 0$. Therefore, only trajectories that are optimal with respect to r will consist of transitions whose optimal advantages are all 0. Consequently a trajectory will have a return of 0 in M' if and only if it is optimal in M.

In M, since $A_r^*(s,a) \leq 0$ for any state and action and $A_r^*(s,a) = 0$ for at least one action per state, the maximum from any state in M' is 0. Therefore, in M', a trajectory is optimal if and only if its return is 0.

Putting together the final assertions of each of the previous two paragraphs, a trajectory is optimal in M if and only if it is optimal in M'.

An underspecification issue is resolved. As we discuss in Section 4, when segment lengths are 1, the partial return preference model ignores the discount factor, making the choice of γ arbitrary, despite that its choice often affects the set of optimal policies. However, in M' defined above, the discount factor does not affect the set of optimal policies, as we explain below. Optimal trajectories have returns of 0 and suboptimal trajectories have negative returns. This line of separation remains under any $\gamma > 0$, since 0 reward discounted with $\gamma > 0$ is still 0 and negative reward discounted with $\gamma > 0$ is still negative. For $\gamma = 0$, $Q_{r'}^* = r' = A_r^*$, and so $\pi_{M'}^*(s) = argmax_a[Q_{r'}^*(s,a)] = argmax_a[A_r^*(s,a)] = \pi_M^*$, also preserving the set of optimal policies from M, which we established above has the same optimal policies as M' with $\gamma > 0$.

Reward is highly shaped. In the research by Ng et al. (1999) on potential-based reward shaping, they suggest that a particularly potent form of their approach is to define the potential function, $\phi(s) = V_M*(s)$. Some algebraic manipulation reveals that the resulting MDP uses A_r^* as

reward and is the same as M' defined above. We suspect further analysis will show that the shaped reward—despite preserving the set of optimal policies—will generalize poorly to other MDPs with the same reward function but different transition dynamics.

Policy improvement wastes computation and environment sampling. Although conducting policy improvement in this highly shaped M' may at first seem appealing, there is actually nothing that needs to be learned from experience in the environment: setting $\pi(s) = argmax_a[A^*_r(s,a)]$ provides an optimal policy.

3.2. Using the learned $\hat{A}^*_r$ as a reward function

An important caveat to the preceding analysis is that the algorithm does not necessarily learn A^*_r. Rather it learns its approximation, $\hat{A}^*_r$.

One potential issue is that adding a constant to $\hat{A}^*_r$ does not change the likelihood of a preferences dataset, making the *learned* value of $max_a \hat{A}^*_r(\cdot,a)$ arbitrary. (For A^*_r, it is 0.) If tasks have varying horizons, then different choices for this maximum value can determine different sets of optimal policies (e.g., by changing whether termination is desirable). One solution is to convert varying horizon tasks to continuing tasks by including infinite transitions from absorbing states after termination, where all such transitions receive 0 reward. Note that this issue does not exist when acting directly from $\hat{A}^*_r$—i.e., $\pi(s) = argmax_a[\hat{A}^*_r(s,a)]$—for which adding a constant to the output of $\hat{A}^*_r$ does not change π.

Another issue is that $\hat{A}^*_r$ will almost surely have some error, and adding policy improvement could compound that error with its own.

3.3. Summary

When regret-based preferences are learned from using the partial return preference model, the theoretical result is surprisingly not as harmful as this apparent misuse suggests it would be. Perhaps this analysis explains why the partial return preference model—shown by Knox et al. (2022) to not model human preferences well—nonetheless has achieved impressive performance on numerous tasks. Yet it has several potential weaknesses, including potentially being a waste of computation and environment sampling.

4. Reframing related work on fine-tuning generative models

We now argue that certain high-profile applications of the partial return preference model—to fine-tune large language models for text summarization (Ziegler et al., 2019), to create InstructGPT and ChatGPT (Ouyang et al., 2022; OpenAI, 2022), to create Sparrow (Glaese et al., 2022), and in work by Bai et al. (2022)—include additional assumptions—one of which appears unjustified in sequential language tasks—that fortuitously allow **an alternative interpretation of their approach: they are applying a regret preference model and are learning an optimal advantage function, not a reward function.**

In these approaches, several assumptions are made:

- Preferences are generated by the partial return preference model.
- During policy improvement, the sequential task is treated as a bandit task at each time step. That treatment is equivalent to setting the discount factor γ to 0 during policy improvement.
- The reward function is $R \to S \times A$, not taking the next state as input.

These approaches learn g as in Equation 6. Since these algorithms assume that preferences are generated by the partial return preference model, the learned g function is a reward function. They also assume $\gamma = 0$ during what would be the policy improvement stage. Therefore, $\tilde{r}(s,a) = Q^*_{\tilde{r}}(s,a)$, and for any state s, $\pi^*_{\tilde{r}}(s) = argmax_a Q^*_{\tilde{r}}(s,a) = argmax_a \tilde{r}(s,a) = argmax_a g(s,a)$.

Problems with the above assumptions Many of the language models considered here are applied in the sequential setting of multi-turn, interactive chat, such as ChatGPT (OpenAI, 2022), Sparrow (Glaese et al., 2022), and work by Bai et al. (2022). Treating these as bandit tasks (or equivalently setting $\gamma = 0$) is an unexplained decision that contradicts how reward functions are used in sequential tasks, to accumulate throughout the task to score a trajectory as return.

Worse, the choice of γ is arbitrary in the original framing of their algorithms. Because they also assume $|\sigma| = 1$, then the partial return of a segment reduces to the immediate reward without discounting: $\sum_{t=0}^{|\sigma|-1} \gamma^t \tilde{r}(s^\sigma_t, a^\sigma_t) = \tilde{r}(s^\sigma_0, a^\sigma_0)$. Consequently, γ curiously has no impact on what reward function is learned from the partial return preference model (assuming the standard definition in this setting that $0^0 = 1$). This lack of impact is a generally problematic aspect of learning reward functions with partial return preference models, since changing γ for a fixed reward function is known to often change the set of optimal polices. (Otherwise MDPs could be solved much more easily by setting $\gamma = 0$ and myopically maximizing immediate reward.)

Despite two assumptions that are unjustified and appear to have significant consequences—that preferences are driven only by partial return and that $\gamma = 0$—the technique is remarkably effective, producing some of the most capable language models at the time of writing.

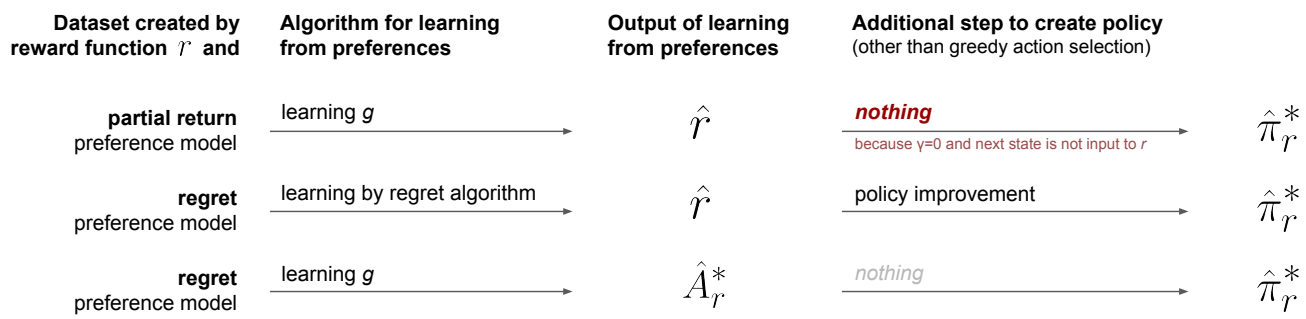

Figure 3. The three algorithms from Figure 1, with the top algorithm modified to reflect the assumptions by related work on fine-tuning language models, discussed in Section 4. With these assumptions, the top algorithm—learning a reward function using the partial return preference model—is procedurally equivalent to the bottom algorithm for learning an optimal advantage function.

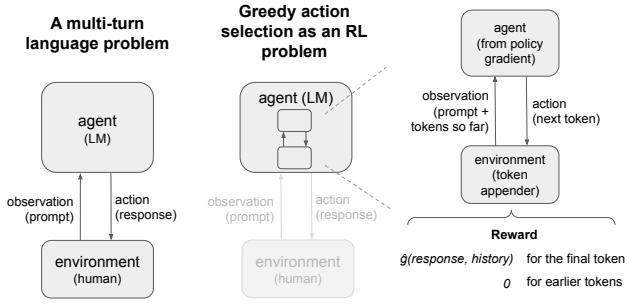

Figure 4. This paper focuses exclusively on the problem to the left, which involves multiple turns of the human providing a prompt and the language-model agent responding. The problem on the right is a common artificial constraint on action selection to make it tractable, forcing the policy to choose one token at a time, sequentially; it does not involve any interaction with the human (i.e., the environment).

Fine-tuning with regret-based preferences Let us instead assume preferences come from the regret preference model, which Knox et al. (2022) and we have argued better describes human preferences. As we subsequently explain, the $\gamma = 0$ assumption then has no effect and therefore can be removed, avoiding both of the troubling assumptions. Specifically, if preferences come from the regret preference model, then the same algorithm's output g is $\hat{A}^*$. Therefore, under this regret-based framing, for any state s, $\pi^*_{\hat{r}}(s) = argmax_a A^*_{\hat{r}}(s,a) = argmax_a g(s,a)$. Therefore, *both the learning algorithm and action selection for a greedy policy are functionally equivalent to their algorithm, but their interpretations change*. Since the discount factor, γ, is already included in the calculation of advantage, its value has no effect on action selection (or the learning algorithm) and the assumption that $\gamma = 0$ can be removed.

In summary, assuming that learning from preferences produces an optimal advantage function—the consequence of adopting the more empirically supported regret preference model—provides a more consistent framing for these algorithms.

A common source of confusion Greedy action selection can itself be challenging for large action spaces. These language models have large action spaces, since choosing a response to the latest human prompt involves selecting a large number of tokens. This choice of response is a single action that results in interaction with the environment, the human. Instead Ouyang et al. (2022) artificially restrict the selection of an action to itself be a sequential decision-making problem, forcing the tokens to be selected one at a time, in order from the start to the end of the text, as Figure 4 illustrates. They use a policy gradient algorithm, PPO (Schulman et al., 2017), to learn a policy for this sub-problem, where the RL agent receives 0 reward until the final token is chosen. At that point, under their interpretation, it receives the learned bandit reward from the left problem in Figure 4. This paper does not focus on *how to do greedy action selection*, and we do not take a stance on whether to treat it as a token-by-token RL problem. However, if one desires to take such an approach to greedy action selection while seeking $\pi(s) = argmax_a[\hat{A}^*_r(s,a)]$, then the bandit reward is simply replaced by the optimal advantage, again executable by the same code, since both are simply the outputs of g.

Implications for future work on fine-tuning language models and other generative models Extensions of the discussed fine-tuning work may seek to learn a reward function to use beyond a bandit setting. Motivations for doing so include reward functions generalizing better when transition dynamics change and allowing the language model to improve its behavior based on experienced long-term outcomes. To learn a reward function to use in such a sequential problem setting, framing the preferences dataset as having been generated by the regret preference model would provide a different algorithm for doing so (in Section 2). It would also avoid the arbitrariness of setting $\gamma > 0$ and learning with the partial return preference model, which outputs the same reward function under these papers' assumptions regardless of the discount factor. The regret-based algorithm for learning a reward function is more internally consistent

and appears to be more aligned with human stakeholder's preferences. However, it does present research challenges for learning reward functions in complex tasks such as those for which these language models are fine-tuned.

References

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Bıyık, E., Losey, D. P., Palan, M., Landolfi, N. C., Shevchuk, G., and Sadigh, D. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, pp. 02783649211041652, 2021.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 4299–4307, 2017.

Glaese, A., McAleese, N., Trębacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. *arXiv preprint arXiv:1811.06521*, 2018.

Knox, W. B., Hatgis-Kessell, S., Booth, S., Niekum, S., Stone, P., and Allievi, A. Models of human preference for learning reward functions. *arXiv preprint arXiv:2206.02231*, 2022.

Lee, K., Smith, L., and Abbeel, P. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021a.

Lee, K., Smith, L., Dragan, A., and Abbeel, P. B-pref: Benchmarking preference-based reinforcement learning. *arXiv preprint arXiv:2111.03026*, 2021b.

Ng, A., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. *Sixteenth International Conference on Machine Learning (ICML)*, 1999.

OpenAI. Chatgpt: Optimizing language models for dialogue. OpenAI Blog https://openai.com/blog/chatgpt/, 2022. Accessed: 2022-12-20.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Sadigh, D., Dragan, A. D., Sastry, S., and Seshia, S. A. Active preference-based learning of reward functions. *Robotics: Science and Systems*, 2017.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Wang, X., Lee, K., Hakhamaneshi, K., Abbeel, P., and Laskin, M. Skill preferences: Learning to extract and execute robotic skills from human feedback. In *Conference on Robot Learning*, pp. 1259–1268. PMLR, 2022.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.