# LEARNING OPTIMAL ADVANTAGE FROM PREFERENCES AND MISTAKING IT FOR REWARD

A PREPRINT

**W. Bradley Knox**[*1,2], **Stephane Hatgis-Kessell**[1], **Sigurdur Orn Adalgeirsson**[2], **Serena Booth**[3], **Anca Dragan**[4], **Peter Stone**[1,5], **and Scott Niekum**[6]
[1]University of Texas at Austin
[2]Google Research
[3]MIT CSAIL
[4]UC Berkeley
[5]Sony AI
[6]University of Massachusetts Amherst

## 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). Most recent work assumes that human preferences are generated based only upon the reward accrued within those segments, or their *partial return*. Recent work casts doubt on the validity of this assumption, proposing an alternative preference model based upon *regret*. We investigate the consequences of assuming preferences are based upon partial return when they actually arise from regret. We argue that the learned function is an approximation of the optimal advantage function, $\widehat{A}^*_r$, *not* a reward function. We find that if a specific pitfall is addressed, this incorrect assumption is not particularly harmful, resulting in a highly shaped reward function. Nonetheless, this incorrect usage of $\widehat{A}^*_r$ is less desirable than the appropriate and simpler approach of greedy maximization of $\widehat{A}^*_r$. From the perspective of the regret preference model, we also provide a clearer interpretation of fine tuning contemporary large language models with RLHF. This paper overall provides insight regarding why learning under the partial return preference model tends to work so well in practice, despite it conforming poorly to how humans give preferences.

[*]Correspondence to: bradknox@cs.utexas.edu

## 1 Introduction

When learning from human preferences (in RLHF), the dominant model assumes that human preferences are determined only by each segment's accumulated reward, or **partial return**. Knox et al. [2022] argued that the partial return preference model has fundamental flaws that are removed or ameliorated by instead assuming that human preferences are determined by the **optimal advantage** of each segment, which is a measure of deviation from optimal decision-making and is equivalent to the negated **regret**. This past work argues for the superiority of the regret preference model (1) by intuition, regarding how humans are likely to give preferences (e.g., see Fig. 2); (2) by theory, showing that regret-based preferences have a desirable identifiability property that preferences from partial return lack; (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) by empirical analysis, showing that with both human and synthetic preferences, the regret model requires fewer preference labels. Section 2 of this paper provides details on the general problem setting and on these two models.

In this paper, we explore the consequences of using algorithms that are designed with the assumption that preferences are determined by partial return when these preferences are instead determined by regret. We show in Section 3 that these algorithms learn an approximation of the optimal *advantage function*, $A^*_r$, not of the reward function, as presumed in many prior works. We then study the implications of this mistaken interpretation. When interpreted as reward, the *exact* optimal advantage is highly shaped and preserves the set of optimal policies, which enables partial-return-based algorithms to perform well. However, the learned *approximation* of the optimal advan-

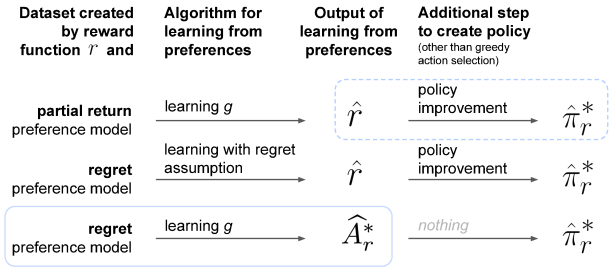


Figure 1: Three algorithms that are justified by their assumed preference model. The top algorithm was popularized by Christiano et al. [2017] and the middle algorithm was proposed by Knox et al. [2022]. The third algorithm is described in Section 3.2. The reward function $\hat{r}$, optimal advantage function $\widehat{A}^*_r$, and optimal policy $\hat{\pi}^*_r$ are approximations of the true versions of these functions. The function $g$ is defined generally in Equation 6 to allow it to represent including $A^*_r$ or $r$. This paper focuses on what occurs when the solid box represents the actual algorithm for learning $g$ but the partial return preference model is assumed, causing $\widehat{A}^*_r$ to be used as if it is the reward in the dashed box.

tage function, $\widehat{A}^*_r$, will have errors. We characterize when and how such errors will affect the set of optimal policies with respect to this mistaken reward, and we uncover a method for reducing a harmful type of error. We conclude that this incorrect usage of $\widehat{A}^*_r$ still permits decent performance under certain conditions, though it is less desirable than the appropriate and simpler approach of greedy maximization of $\widehat{A}^*_r$.

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, Touvron et al. 2023] can be viewed as an instance of learning an optimal advantage function and inadvertently 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 a problematic 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 $\gamma$ to 0.

## 2 Preliminaries: Preference models for learning reward functions

A Markov decision process (MDP) is specified by a tuple $(S, A, T, \gamma, D_0, r)$. $S$ and $A$ are the sets of possible states and actions, respectively. $T : S \times A \to p(\cdot|s, a)$ is a transition function; $\gamma$ is the discount factor; and $D_0$ is the distribution of start states. Unless stated otherwise, we assume tasks are undiscounted ($\gamma = 1$) and 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 $r_t$ is a function of $s_t$, $a_t$, and $s_{t+1}$ at time $t$. 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. $Q^\pi_{\tilde{r}}$ and $V^\pi_{\tilde{r}}$ refer respectively to the state-action value function and state value function for a policy, $\pi$, under $\tilde{r}$, and are defined as follows.

$$V^\pi_{\tilde{r}}(s) \triangleq \mathbb{E}_\pi[\sum_{t=0}^{\infty} \tilde{r}(s_t, a_t, s_{t+1})|s_0 = s]$$
$$Q^\pi_{\tilde{r}}(s, a) \triangleq \mathbb{E}_\pi[\tilde{r}(s, a, s') + V^\pi_{\tilde{r}}(s')]$$

An optimal policy $\pi^*_{\tilde{r}}$ is any policy where $V^{\pi^*_{\tilde{r}}}_{\tilde{r}}(s) \geq V^\pi_{\tilde{r}}(s)$ at every state $s$ for every policy $\pi$. We write shorthand for $Q^{\pi^*_{\tilde{r}}}_{\tilde{r}}$ and $V^{\pi^*_{\tilde{r}}}_{\tilde{r}}$ as $Q^*_{\tilde{r}}$ and $V^*_{\tilde{r}}$, respectively. The optimal advantage function is defined as $A^*_{\tilde{r}}(s, a) \triangleq Q^*_{\tilde{r}}(s, a) - V^*_{\tilde{r}}(s)$; this measures how much an action reduces expected return relative to following an optimal policy.

Throughout this paper, when the preferences are not human-generated, the ground-truth reward function $r$ is used to algorithmically generate preferences. $r$ 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 is commonly learned by minimizing the cross-entropy loss—i.e., maximizing the likelihood—of observed human preference labels [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, Touvron et al. 2023].

**Segments** Let $\sigma$ denote a segment starting at state $s^\sigma_0$. 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 $\sigma_t \triangleq (s^\sigma_t, a^\sigma_t, s^\sigma_{t+1})$. A segment $\sigma$ is **optimal** with respect to $\tilde{r}$ if, for every $i \in \{1, ..., |\sigma|\text{-}1\}$, $A^*_{\tilde{r}}(s^\sigma_i, a^\sigma_i) = 0$. A segment that is not optimal is **suboptimal**. Given some $\tilde{r}$ and a segment $\sigma$, where $\tilde{r}^\sigma_t \triangleq \tilde{r}(s^\sigma_t, a^\sigma_t, s^\sigma_{t+1})$, the undiscounted **partial return** of a segment $\sigma$ is $\sum_{t=0}^{|\sigma|-1} \tilde{r}^\sigma_t$, which we denote in shorthand as $\Sigma_\sigma \tilde{r}$.

**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 $\sigma_1$ is preferred over $\sigma_2$, denoted $\sigma_1 \succ \sigma_2$, $\mu = \langle 1, 0 \rangle$. $\mu$ 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). For a sample $(\sigma_1, \sigma_2, \mu)$, we assume that the two segments have equal lengths (i.e., $|\sigma_1| = |\sigma_2|$).

**Loss function** When learning a reward function from a preference dataset, $D_\succ$, preference labels are typically assumed to be generated by a preference model $P$ based on an unobservable *ground-truth* reward function $r$.We learn $\hat{r}$, an approximation of $r$, by minimizing cross-entropy loss:

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

If $\sigma_1 \succ \sigma_2$, the sample's likelihood is $P(\sigma_1 \succ \sigma_2|\hat{r})$ and its loss is therefore $-logP(\sigma_1 \succ \sigma_2|\hat{r})$. If $\sigma_1 \prec \sigma_2$, its likelihood is $1 - P(\sigma_1 \succ \sigma_2|\hat{r})$. This loss is underspecified until the preference model $P(\sigma_1 \succ \sigma_2|\hat{r})$ is defined. Algorithms in this paper for learning approximations of $r$ or $A^*_r$ from preferences can be summarized simply as "minimize Equation 1".

**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 can be used to model preferences provided by humans or other systems, or to generate synthetic preferences.

### 2.2 Preference models: partial return and regret

**Partial return** The dominant preference model (e.g., Christiano et al. [2017]) 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** Knox et al. [2022] introduced an alternative human preference model. This regret-based model assumes that preferences are based on segments' deviations from optimal decision-making: the regret of each transition in a segment. We first focus on segments with deterministic transitions. For a single transition $(s_t, a_t, s_{t+1})$, $regret_{\mathrm{d}}(\sigma_t|\tilde{r}) \triangleq V^*_{\tilde{r}}(s^\sigma_t) - [\tilde{r}_t + V^*_{\tilde{r}}(s^\sigma_{t+1})]$. For a full 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 $\sigma^*$ always has 0 regret, and a suboptimal segment $\sigma^{\neg *}$ always has positive regret.

Stochastic state 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 the sum of optimal advantages, or the *negated* regret:

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

(Notationally, $A^*_{\tilde{r}}(\sigma_t) = A^*_{\tilde{r}}(s^\sigma_t, a^\sigma_t)$.) Lastly, 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: $P_{regret}(\cdot|\tilde{r}) = P_{\Sigma_r}(\cdot|\tilde{r})$.

Intuitively, the partial return preference model always assumes preferences are based upon outcomes while the regret model is able to account for preferences based upon outcomes (Eq. 3) and preferences over decisions (Eq. 4).

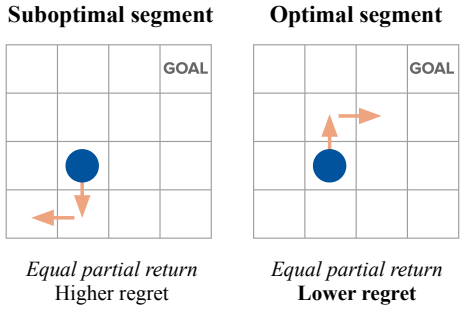


Figure 2: Two segments in an undiscounted task with $-1$ reward each time step. The partial return of both segments with respect to the true reward function is $-2$. The regret of the left segment is 4. The right segment is optimal and therefore has a regret of 0. The regret preference model is more 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.

Knox et al. [2022] showed the regret both has desirable theoretical properties (i.e., it is identifiable where partial

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

return is not) and is a better model of true human preferences. Since regret better models true human preferences, and since many recent works use true human preferences but assume them to be generated according to partial return, we ask: what are the consequences of misinterpreting the optimal advantage function as reward?

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

We ask: what is actually learned when preferences are assumed to arise from partial return but actually come from regret (Equation 2), and what implications does that have?

Our results can be reproduced via our code repository, at github.com/Stephanehk/Learning-OA-From-Prefs.

## 3.1 Learning the optimal advantage function

To start, let us unify the two preference models from Section 2.2 into a single general preference model.

$$P_g(\sigma_1 \succ \sigma_2|\tilde{r}) \triangleq logistic\Big(\sum_{t=0}^{|\sigma_1|-1} g(\sigma_{1,t}) - \sum_{t=0}^{|\sigma_2|-1} g(\sigma_{2,t})\Big) \tag{6}$$

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 preferences are generated according to partial return, $g(\sigma_t) = \tilde{r}(s_t^\sigma, a_t^\sigma, s_{t+1}^\sigma)$, and the reward function $\tilde{r}$ is learned via Equation 1.

When preferences are instead generated according to regret, $g(\sigma_t) = A_{\tilde{r}}^*(\sigma_t) = A_{\tilde{r}}^*(s_t^\sigma, a_t^\sigma)$ and the parameters of this optimal advantage function can be learned directly, also via Equation 1. $\widehat{A}_r^*$ can be learned and then acted upon greedily, via $argmax_a \widehat{A}_r^*(s,a)$, an algorithm we call $\boldsymbol{greedy\ \widehat{A}_r^*}$ (bottom algorithm of Fig. 1). Notably, this algorithm does not require the additional step of policy improvement and instead uses $\widehat{A}_r^*$ directly. No reward function is explicitly represented or learned, though we still assume that preferences were generated by regret under a hidden reward function $r$.

The remainder of this section considers first the consequences of using the error-free $A_r^*$ as a reward function: $\boldsymbol{r_{A_r^*} = A_r^*}$. We call this mistaken approach $\boldsymbol{greedy\ Q^*_{r_{A_r^*}}}$. We then consider the consequences of using the approximation $\widehat{A}_r^*$ as a reward function, $r_{\widehat{A}_r^*} = \widehat{A}_r^*$, which we refer to as $\boldsymbol{greedy\ Q^*_{r_{\widehat{A}_r^*}}}$. The following investigation is an attempt to answer *why learning while assuming the partial return preference model tends to work so well in practice, despite its poor fit as a descriptive model of human preference.*

## 3.2 Using $A_r^*$ as a reward function

Under the assumption of regret-based preferences, learning a reward function with the partial return preference model effectively uses an approximation of $A_r^*$ as a reward function, $\hat{r} = \widehat{A}_r^*$. Let us first assume perfect inference of $A_r^*$ (i.e., that $\widehat{A}_r^* = A_r^*$), and consider the consequences. We will refer to the *non-approximate* versions of $greedy\ \widehat{A}_r^*$ and $r_{\widehat{A}_r^*}$ as $\boldsymbol{greedy\ A_r^*}$ and $\boldsymbol{r_{A_r^*}}$.

**Optimal policies are preserved.** Using $A_r^*$ as a reward function preserves the set of optimal policies. To prove this statement, we first prove a more general theorem.

For $\tilde{r}$, an arbitrary reward function, $max_a A_{\tilde{r}}^*(\cdot, a) = 0$ by definition. Let the set of optimal policies with respect to $\tilde{r}$ be denoted $\Pi_{\tilde{r}}^*$.

**Theorem 3.1** (Greedy action is optimal when the maximum reward in every state is 0.)**.**
$\Pi_{\tilde{r}}^* = \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in argmax_a \tilde{r}(s,a)]\}$ *if* $max_a \tilde{r}(\cdot, a) = 0$.

Theorem 3.1 is proven in Appendix A. The sketch of the proof is that if the maximum reward in every state is 0, then the best possible return from every state is 0. Therefore, $V_{\tilde{r}}^*(\cdot) = 0$, making $\forall (s,a) \in S \times A, Q_{\tilde{r}}^*(s,a) = \tilde{r}(s,a) + \gamma \mathbb{E}_{s'}[V_{\tilde{r}}^*(s)] = \tilde{r}(s,a)$.

We now return to our specific case, proven in Appendix B.

**Corollary 3.1** (Policy invariance of $r_{A_r^*}$)**.**
*Let* $r_{A_r^*} \triangleq A_r^*$. *If* $max_a A_r^*(\cdot, a) = 0$, $\Pi_{r_{A_r^*}}^* = \Pi_r^*$.

**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 $\gamma$, making its choice arbitrary despite it often affecting the set of optimal policies. With $r_{A_r^*}$, however, the lack of $\gamma$ in Corollary 3.1 establishes $\gamma$ does not affect the set of optimal policies. To give intuition, we apply the intermediate result within the proof of Theorem 3.1 that $V_{\tilde{r}}^*(\cdot) = 0$ to the specific case of Corollary 3.1, we see that $V_{r_{A_r^*}}^*(\cdot) = 0$. Therefore, $Q_{r_{A_r^*}}^*(s,a) = r_{A_r^*}(s,a) + \gamma \mathbb{E}_{s'}[0]$, making $\gamma$ have no impact on $Q_{r_{A_r^*}}^*(s,a)$ and therefore on on $\Pi_r^*$.

**Reward is highly shaped.** In Ng et al. [1999]'s seminal research on potential-based reward shaping , they highlight $\phi(s) = V_r^*(s)$ as a particularly desirable potential function. Algebraic manipulation reveals that the MDP that results from this $\phi$ actually uses as a reward function $r_{A_r^*} \triangleq A_r^*$. See Appendix C for the derivation. Ng et al. also note that that it causes $V_{r_{A_r^*}}^*(\cdot) = 0$ and therefore results in "a particularly easy value function to learn; ... all that would remain to be done would be to learn the non-zero Q-values." We characterize this approach as highly shaped because the information required to act optimally is in the agent's immediate reward.

**Policy improvement wastes computation and environment sampling.** When using $A^*_r$ as a reward function, no policy improvement is needed: setting $\pi(s) = argmax_a[A^*_r(s,a)]$ provides an optimal policy.

### 3.3 Using the learned $\widehat{A}^*_r$ as a reward function

A caveat to the preceding analysis is that the algorithm does not necessarily learn $A^*_r$. Rather it learns its approximation, $\widehat{A}^*_r$. We investigate the effects of the approximation error of $\widehat{A}^*_r$. We find that this error only induces a difference in performance from that of *greedy* $\widehat{A}^*_r$ when $max_a \widehat{A}^*_r(s,a) \neq 0$ in at least one state $s$, and the consequence of that error is dependent on the maximum partial return of all *loops*—segments that start and end in the same state—within the MDP.

For the empirical results below, we build upon the experimental setting of Knox et al. [2022], including both for learning and for randomly generating MDPs. Hyperparameters and other experimental settings are identical except where noted. All preferences are synthetically generated by the regret preference model.

**If the maximum value of $\widehat{A}^*_r$ in every state is 0, behavior is identical between *greedy* $Q^*_{r_{\widehat{A}^*_r}}$ and *greedy* $\widehat{A}^*_r$.** From Theorem 3.1, the following trivially holds for a learned approximation $\widehat{A}^*_r$.

**Corollary 3.2.** *Let* $r_{\widehat{A}^*_r} \triangleq \widehat{A}^*_r$. *If* $max_a \widehat{A}^*_r(\cdot,a) = 0$, *then* $\Pi^*_{r_{\widehat{A}^*_r}} = \{\pi : \forall s, \forall a \ [\pi(a|s) > 0 \Leftrightarrow a \in argmax_a \widehat{A}^*_r(s,a)]\}$.

Therefore, if $max_a \widehat{A}^*_r(\cdot,a) = 0$, then a policy from *greedy* $\widehat{A}^*_r$ is identical to an optimal policy for *greedy* $Q^*_{r_{\widehat{A}^*_r}}$, assuming ties are resolved identically. The actual policy from *greedy* $Q^*_{r_{\widehat{A}^*_r}}$ will also be identical unless limitations of the policy improvement algorithm cause it to not find a policy in $\Pi^*_{r_{\widehat{A}^*_r}}$ in this highly shaped setting with the reward function also in hand, not requiring experience to query. However, $max_a \widehat{A}^*_r(\cdot,a) = 0$ is not guaranteed for an approximation of $A^*_r$, which we consider later in this section.

We conduct an empirical test of the assertion above by adjusting $\widehat{A}^*_r$ to have the property $max_a \widehat{A}^*_r(\cdot,a) = 0$ by shifting $\widehat{A}^*_r$ by a state-dependent constant: for all $(s,a)$, $r_{\widehat{A}^*_r\text{-}shifted}(s,a) \triangleq \widehat{A}^*_r(s,a) - max_{a'} \widehat{A}^*_r(s,a')$. Note that $argmax_a r_{\widehat{A}^*_r\text{-}shifted}(s,a) = argmax_a \widehat{A}^*_r(s,a)$. In 90 small gridworld MDPs, we observe no difference between *greedy* $\widehat{A}^*_r$ and *greedy* $Q^*_{r_{\widehat{A}^*_r}}$ with $r_{\widehat{A}^*_r\text{-}shifted}$ (see Figure 9). However, cost is generally incurred from suboptimal behavior and environment sampling *while* a policy improvement algorithm learns this approximately optimal policy, unless the policy improvement algorithm uses the in-hand $r_{\widehat{A}^*_r\text{-}shifted}$ without environment sampling and makes use of knowledge that the state value is 0 in every state, which together allow it to simply define optimal behavior as $argmax_a Q_{r_{\widehat{A}^*_r\text{-}shifted}}(s,a) = argmax_a r_{\widehat{A}^*_r\text{-}shifted}(s,a) = argmax_a \widehat{A}^*_r(s,a)$, which is *greedy* $\widehat{A}^*_r$.

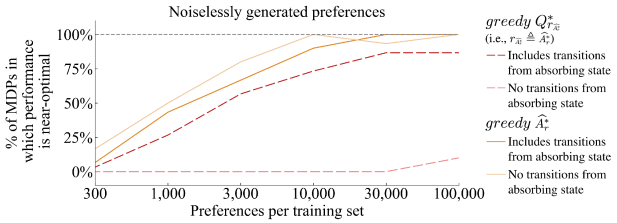


Figure 3: Performance when noiselessly generated preference datasets do and do not include segments with transitions from absorbing state. Results are across 30 randomly generated grid-world MDPs with tabular representations of the $\widehat{A}^*_r$, where segments of length 3 are chosen by uniformly randomly choosing a start state and 3 its actions. When transitions from absorbing states are not included, any segment that terminates before its final transition is rejected and then resampled. For *greedy* $\widehat{A}^*_r$ (in red) Wilcoxon paired signed-rank tests reveal that including transitions from absorbing state results in significantly higher performance for all training set sizes but the smallest, 300, with $p < 0.0007$. No significant difference in performance is detected for *greedy* $Q^*_{r_{\widehat{A}^*_r}}$ with or without terminating transitions except at 30,000 preferences with a more modest $p = 0.04$. Appendix G contains the plot for stochastically generated preferences (Figure 11), which contains similar results.

**Including segments with transitions from absorbing state encourages $max_a \widehat{A}^*_r(\cdot,a) = 0$.** If an algorithm designer is confident that the preferences in their preference dataset were generated via the regret preference model, then the technique above of manually shifting $\widehat{A}^*_r$ may be justified and tenable, depending on upon the size of the action space. Yet with such confidence, acting to greedily maximize $\widehat{A}^*_r$ is more straightforward and efficient. Further, an appeal that will emerge from our analysis is that *algorithmically* assuming preferences arise from partial return can lead to good performance regardless of whether preferences actually reflect partial return or regret. The manual shift technique could change the set of optimal policies when preferences are generated by the partial return preference model. Therefore, we do not recommend applying the shift above in practice. Below we describe another method that, although imperfect, avoids explicitly embracing either preference model.

Adding a constant to $\widehat{A}^*_r$ does not change the likelihood of a preferences dataset, making the *learned* value of $max_{(s,a)} \widehat{A}^*_r(s,a)$ arbitrary. Consequently, it also makes $max_a \widehat{A}^*_r(\cdot,a)$ underspecified. 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 to themselves after termination, where all such transitions receive 0 reward. Note that this issue does not exist when acting directly from $\widehat{A}^*_r$—i.e., $\pi(s) = argmax_a[\widehat{A}^*_r(s,a)]$—for which adding a constant to the output of $\widehat{A}^*_r$ does not change $\pi$. Some past authors have acknowledged this insensitivity to a shift [Christiano et al. 2017, Lee et al. 2021a, Ouyang et al. 2022, Hejna and

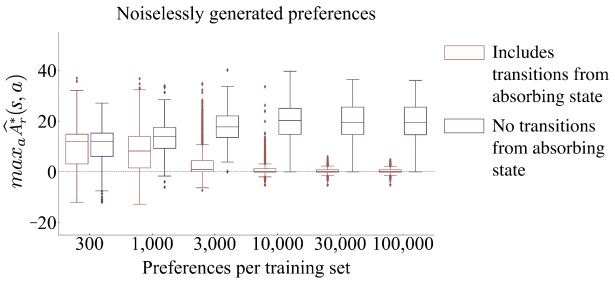


Figure 4: Comparing the effect on $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ of including transitions from absorbing state. For each state within 30 MDPs, the plots above show the $max_a \widehat{A}^*_r(s,a)$ values. The plot shows that including such transitions moves the resultant maximum values closer to 0. The plot for stochastically generated preferences is similar and can be found in Appendix F. After learning with absorbing transitions, $max_a \widehat{A}^*_r(s,a)$ across all states is stochastically closer to 0 than when learning without them. Wilcoxon paired signed-rank tests at every training set size are all extremely significant with $p < 10^{-7}$.

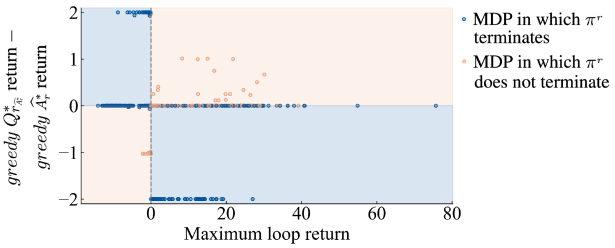


Figure 5: Validation of the hypothesis that maximum partial return by $r_{\widehat{A}^*_r}$ across all loops determines the direction of performance differences between $greedy\ \widehat{A}^*_r$ and $greedy\ Q^*_{r_{\widehat{A}^*_r}}$. 1080 runs are shown, built from the set of 90 MDPs $\times$ $\{10, 100, 1000\}$ preferences in the training set $\times$ $\{1, 2\}$ segment lengths $\times$ $\{$noiselessly, stochastically$\}$ generated preferences. Plot points are colored orange when every $\pi^*_r$ terminates and blue when every $\pi^*_r$ does not terminate. The blue and orange shading of the plot represents where our hypotheses predict circles of each color to be, if $y \neq 0$. Returns are standardized across MDPs within $[-1, 1]$ (detail in Appendix D), and the x axis is the maximum partial return by $r_{\widehat{A}^*_r}$ across all loops in the MDP. Of the 75 runs with a performance difference ($y \neq 0$), 73 conform to our hypothesis. In the remaining 2 runs, both algorithms achieve near-optimal behavior and therefore have a difference of less than 0.1.

Sadigh 2023], and the common practice of forcing all tasks to have a fixed horizon (e.g., as done by Christiano et al. [2017, p. 14] and Gleave et al. [2022]) may be partially attributable to the poor performance that results when using the partial return preference model in variable-horizon tasks without transitions from absorbing states.

Figure 4 shows the large impact of including transitions from absorbing state when $\hat{r} = \widehat{A}^*_r$. As expected, $greedy\ \widehat{A}^*_r$ is not noticeably affected by the inclusions of such transitions. Further, Figure 4 shows that the inclusion of these transitions from absorbing state does indeed push $max_a A^*_{\hat{r}}(\cdot, a)$ towards 0, more so with larger training set sizes (given a fixed number of epochs), though it does not completely accomplish making $max_a A^*_{\hat{r}}(\cdot, a) = 0$.

**Bias towards termination determines performance differences.** When $max_a \widehat{A}^*_r(s,a)$ tends to be near 0, we find the performances of $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ and $greedy\ \widehat{A}^*_r$ to be similar. But their performances sometimes differ. Can we predict which algorithm will perform better? To address this questions understand why, we performed a detailed analysis with 90 small gridworld MDPs, from which the following hypothesis arose. The logic behind the following hypothesis assumes an undiscounted task, though the hypothesized effects should exist in lessened form as discounting is increased. We define a loop to be a segment that begins and ends in the same state and then focus on the maximum partial return by $r_{\widehat{A}^*_r}$ across all loops.

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

| **Condition** | $\pi^*_r$ **terminates** | $\pi^*_r$ **does not terminate** |
|---|---|---|
| Max loop partial return $> 0$ | $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ | $greedy\ \widehat{A}^*_r$ |
| Max loop partial return $< 0$ | $greedy\ \widehat{A}^*_r$ | $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ |

Focusing on tasks with deterministic transitions,[3] the justification for this hypothesis is based on the following biases created by the maximum partial return of all loops:

- When the maximum partial return of all loops is *positive*, any $\pi^*_{r_{\widehat{A}^*_r}}$ will not terminate because it can achieve infinite value.
- When the maximum partial return of all loops is *negative*, any $\pi_{\hat{r}}$ for $r_{\widehat{A}^*_r}$ will terminate, because it can only achieve negative infinity value without terminating.

Results shown in Figure 5 validate this hypothesis. Over 1080 runs of learning $\widehat{A}^*_r$ in various settings, we find that the hypothesis is highly predictive of deviations in performance.

The cause of this predictive measure, the maximum partial return by $r_{\widehat{A}^*_r}$ of all loops, has not yet been characterized. Hence, an algorithm designer should still be wary of mistaking $\widehat{A}^*_r$ for a reward function and relying on this predictive measure to determine whether the resulting policy avoids or seeks termination.

**Reward is also highly shaped with approximation error.** We also test whether the reward shaping that exists when using $A^*_r$ as a reward function is also present when using its approximation, $\widehat{A}^*_r$. Figure 6 finds shows that policy improvement with the Q learning algorithm [Watkins

[3]For stochastic tasks, this concept of loops generalizes to the steady-state distribution with the maximum average reward, across all policies.

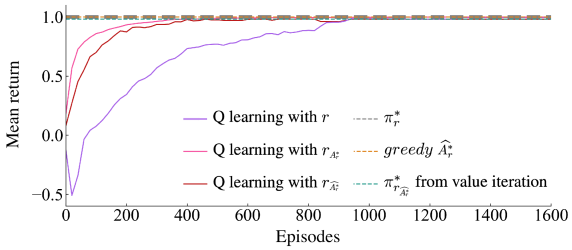


Figure 6: Learning curves for Q learning on the ground truth reward function $r$ and on $r_{\widehat{A}_r^*}$. Each curve represents 100 instances of Q learning, each in a different MDP. $\widehat{A}_r^*$ was learned with noiseless 100,000 regret-based preferences. Even without giving the learning agent access to the known $r_{\widehat{A}_r^*}$, we see that learning is more efficient, indicating that in practice $r_{\widehat{A}_r^*}$ is a helpfully shaped reward function, as is using the true $A_r^*$ as a reward function. We define AAC as the area above a curve and below 1.0. A small AAC indicates better learning performance. Wilcoxon paired signed-rank tests reveal that Q learning with $r$ (purple) has a larger AAC than with $r_{A_r^*}$ (red), which in turn has a larger AAC than with $r_{\widehat{A}_r^*}$ (both $p < 0.00003$).

and Dayan 1992] is more sample efficient with $r_{A_r^*}$ and with $r_{\widehat{A}_r^*}$ than with the ground truth $r$, as was expected.

### 3.4 Summary

When one learns from regret-based preferences using the partial return preference model, the theoretical and empirical consequences are surprisingly less harmful than this apparent misuse suggests it would be. The policy that would have been learned with the correct regret-based preference model is preserved if $\widehat{A}_r^*$ has a maximum of 0 in every state. Further, $\widehat{A}_r^*$ acts as a highly shaped reward. Perhaps this analysis explains why the partial return preference model—shown to not model human preferences well [Knox et al. 2022]—nonetheless has achieved impressive performance on numerous tasks. That said, confusing $\widehat{A}_r^*$ for a reward function has drawbacks compared to $greedy\ \widehat{A}_r^*$, including higher sample complexity and sensitivity to an understudied factor, the maximum partial return by $r_{\widehat{A}_r^*}$ of all loops.

## 4 Reframing related work on fine-tuning generative models

The partial return preference model has been used in several high-profile applications: 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], in work by Bai et al. [2022], and to fine-tune Llama 2 [Touvron et al. 2023]. The use of the partial return model in these works fortuitously allows **an alternative interpretation of their approach: they are applying a regret preference model and are learning an optimal advantage function, not a reward function.** These approaches make several assumptions:

- Preferences are generated by partial return.
- 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 $\gamma$ 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, which is interpreted as a reward function according to the partial return preference model. 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 dialog, such as ChatGPT [OpenAI 2022], Sparrow [Glaese et al. 2022], and work by Bai et al. [2022]. Treating these as bandit tasks (i.e., 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.

Further, the choice of $\gamma$ 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: $\sum_{t=0}^{|\sigma|-1} \gamma^t \tilde{r}(s_t^\sigma, a_t^\sigma) = \tilde{r}(s_0^\sigma, a_0^\sigma)$. Consequently, $\gamma$ 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 $\gamma$ 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 preferences are driven only by partial return and that $\gamma = 0$—that lack justification and appear to have significant consequences, the technique is remarkably effective, producing some of the most capable language models at the time of writing.

**Fine-tuning with regret-based preferences** Let us instead assume preferences come from the regret preference model. As explained in Section 3.2, the $\gamma = 0$ assumption then has no effect. Therefore it 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 $\widehat{A}_r^*$. Consequently, under this regret-based framing, for any state $s$, $\pi^*_{\tilde{r}}(s) = argmax_a A^*_{\tilde{r}}(s,a) = argmax_a g(s,a)$. Therefore, *both the learning algorithm and action selection for a greedy policy in this setting are functionally equivalent to their algorithm, but their interpretations change*.

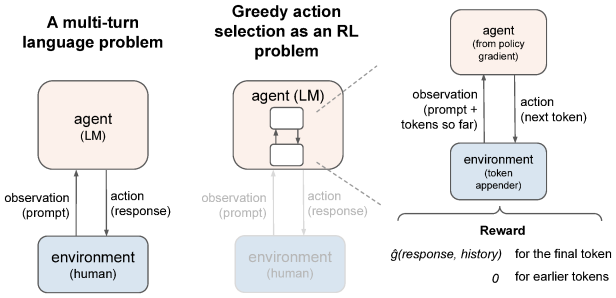


Figure 7: This paper focuses exclusively on the problem to the left, which involves multiple turns of the human providing a prompt and the language-model agent responding. The problem on the right is a common artificial constraint on action selection to make it tractable, using the policy to sample a response one token at a time, sequentially; it does not involve any interaction with the human (i.e., the environment).

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 sequence of tokens. This choice of response is a single action that results in interaction with the environment, the human. As an example, Ouyang et al. [2022] instead 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 7 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 7. 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 $\pi(s) = argmax_a[\widehat{A}^*_r(s,a)]$, 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 fine-tuning 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. In particular, the known method for learning a reward function with the regret preference model requires a differentiable approximation of the optimal advantage function for the reward function arising from parameters that change at each training iteration.

## 5 Conclusion

This paper investigates the consequences of assuming that preferences are generated according to partial return when they instead arise from regret. The regret preference model provides an improved account of the effective method of fine-tuning LLMs from preferences (Section 4). In the general case (Section 3), we find that this mistaken assumption is not ruinous to performance when averaged over many instances of learning, which explains the success of many algorithms which rely on this flawed assumption. Nonetheless, this mistaken interpretation obfuscates learning from preferences, confusing practitioners' intuitions about human preferences and how to use the function learned from preferences. We believe that partial return preference model is rarely accurate for trajectory segments, i.e., it is rare for a human's preferences to be unswayed by any of a segment's end state value, start state value, or luck during transitions. Assuming that humans incorporate *all* of those three segment characteristics, as the regret preference model does, results in a better descriptive model, yet it does not *universally* describe human preferences. To improve the sample efficiency and alignment of agents that learn from preferences, subsequent research should focus further on potential models of human preference and also on methods for influencing people to conform to a desired preference model. Lastly, after reading this paper, one might be tempted to conclude that it's safe to close your eyes, clench your teeth, and put your faith in the partial return preference model. This conclusion is not supported by this paper, since even with the addition of transitions from absorbing states, arbitrary bias to seek or avoid termination is frequently introduced. The implication of this bias is particularly important since RLHF is currently the primary safeguarding mechanism for LLMs [Casper et al. 2023].

## Acknowledgments

This work has taken place in part in the the Interactive Agents and Colloraborative Technologies (InterACT) lab at UC Berkeley, the Learning Agents Research Group (LARG) at UT Austin and the Safe, Correct, and Aligned Learning and Robotics Lab (SCALAR) at The University of Massachusetts Amherst. LARG research is supported in part by NSF (FAIN-2019844, NRT-2125858), ONR (N00014-18-2243), ARO (E2061621), Bosch, Lockheed Martin, and UT Austin's Good Systems grand challenge. Peter Stone is financially compensated as the Executive Director of Sony AI America, the terms of which have been approved by the UT Austin. SCALAR research is supported in part by the NSF (IIS-1749204), AFOSR (FA9550-20-1-0077), and ARO (78372-CS, W911NF-19-2-0333). InterACT research is supported in part by ONR YIP and NSF HCC. Serena Booth is supported by NSF GRFP.

## References

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Erdem Bıyık, Dylan P Losey, Malayandi Palan, Nicholas C Landolfi, Gleb Shevchuk, and Dorsa Sadigh. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, page 02783649211041652, 2021.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4299–4307, 2017.

Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

Adam Gleave, Mohammad Taufeeque, Juan Rocamonde, Erik Jenner, Steven H. Wang, Sam Toyer, Maximilian Ernestus, Nora Belrose, Scott Emmons, and Stuart Russell. imitation: Clean imitation learning implementations. arXiv:2211.11972v1 [cs.LG], 2022. URL https://arxiv.org/abs/2211.11972.

Joey Hejna and Dorsa Sadigh. Inverse preference learning: Preference-based rl without a reward function. *arXiv preprint arXiv:2305.15363*, 2023.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. *arXiv preprint arXiv:1811.06521*, 2018.

W Bradley Knox, Stephane Hatgis-Kessell, Serena Booth, Scott Niekum, Peter Stone, and Alessandro Allievi. Models of human preference for learning reward functions. *arXiv preprint arXiv:2206.02231*, 2022.

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

Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. B-pref: Benchmarking preference-based reinforcement learning. *arXiv preprint arXiv:2111.03026*, 2021b.

A.Y. Ng, D. Harada, and S. Russell. 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.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

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

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Xiaofei Wang, Kimin Lee, Kourosh Hakhamaneshi, Pieter Abbeel, and Michael Laskin. Skill preferences: Learning to extract and execute robotic skills from human feedback. In *Conference on Robot Learning*, pages 1259–1268. PMLR, 2022.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

## A Proof of Theorem 3.1

**Theorem 3.1** (Greedy action is optimal when the maximum reward in every state is 0.)
$\Pi^*_{\tilde{r}} = \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \textit{argmax}_a \tilde{r}(s,a)]\}$ *if* $\textit{max}_a \tilde{r}(\cdot, a) = 0$.

The main idea is that if the maximum reward in every state is 0, then the best possible return from every state is 0. Therefore, $V^*_{\tilde{r}}(\cdot) = 0$, making $\forall (s,a) \in S \times A, Q^*_{\tilde{r}}(s,a) = \tilde{r}(s,a) + \gamma \mathbb{E}_{s'}[V^*_{\tilde{r}}(s)] = \tilde{r}(s,a)$.

The proof follows.

$\forall (s,a) \in S \times A, \tilde{r}(s,a) \leq 0$, so $\forall s \in S, V^*_{\tilde{r}}(s) \leq 0$.

$\forall s \in S, \exists a \in A : \tilde{r}(s,a) = 0$, so $\forall s \in S, V^*_{\tilde{r}}(s) \geq 0$.

$V^*_{\tilde{r}}(s) \leq 0$ and $V^*_{\tilde{r}}(s) \geq 0$ implies $V^*_{\tilde{r}}(s) = 0$, so $\forall s \in S, V^*_{\tilde{r}}(s) = 0$.

$\forall (s,a) \in S \times A$,

$$\begin{aligned} Q^*_{\tilde{r}}(s,a) &= \tilde{r}(s,a) + \gamma \mathbb{E}_{s'}[V^*_{\tilde{r}}(s')] \\ Q^*_{\tilde{r}}(s,a) &= \tilde{r}(s,a) + \gamma \mathbb{E}_{s'}[0] \\ Q^*_{\tilde{r}}(s,a) &= \tilde{r}(s,a) \\ argmax_a Q^*_{\tilde{r}}(s,a) &= argmax_a \tilde{r}(s,a) \end{aligned} \tag{7}$$

By definition, $\Pi^*_{\tilde{r}} = \{\pi : \forall s, \pi(s) = \text{argmax}_a Q^*_{\tilde{r}}(s,a)\}$.

Since $argmax_a Q^*_{\tilde{r}}(s,a) = argmax_a \tilde{r}(s,a)$, $\Pi^*_{\tilde{r}} = \{\pi : \forall s, \pi(s) = \text{argmax}_a \tilde{r}(s,a)\}$. □

## B Proof of Corollary 3.1

**Corollary 3.1** (Policy invariance of $r_{A^*_r}$)
*Let* $r_{A^*_r} \triangleq A^*_r$. *If* $\textit{max}_a A^*_r(\cdot, a) = 0$, $\Pi^*_{r_{A^*_r}} = \Pi^*_r$.

Since $\text{max}_a A^*_r(\cdot, a) = 0$ and $r_{A^*_r} \triangleq A^*_r$, $\text{max}_a r_{A^*_r}(\cdot, a) = 0$.

Therefore, by Theorem 3.1, $\Pi^*_{r_{A^*_r}} = \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \text{argmax}_a r_{A^*_r}(s,a)]\}$.

Also, by definition, $\Pi^*_r = \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \text{argmax}_a A^*_r(s,a)]\}$.

Consequently,

$$\begin{aligned} \Pi^*_{r_{A^*_r}} &= \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \text{argmax}_a r_{A^*_r}(s,a)]\} \\ &= \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \text{argmax}_a A^*_r(s,a)]\} \\ &= \Pi^*_r \end{aligned} \tag{8}$$

## C Used as reward, $A^*_r$ is highly shaped

In Section 3.2, we stated that following the advice below of Ng et al. [1999] is equivalent to using $A^*_r$ as reward. We derive this result after reviewing their advice.

In their paper on potential-based reward shaping, the authors suggest a potent form of setting $\Phi(s)$, which is $\Phi(s) = V^*_M(s)$. Their notation includes MDPs $M$ and $M'$, where $M$ is the original MDP and $M'$ is the potential-shaped MDP. The notation for these two MDPs maps to our notation in that the reward function of $M$ is $r$, and we ultimately derive that the reward function of $M'$ is $r_{A^*_r}$.

Ng et al.'s Corollary 2 includes the statement that, under certain conditions, for any state $s$ and action $a$, $Q^*_{r_{A^*_r}}(s,a) = Q^*_r(s,a) - \Phi(s)$.

$$\begin{aligned} Q^*_{r_{A^*_r}}(s,a) &= Q^*_r(s,a) - \Phi(s) \\ Q^*_{r_{A^*_r}}(s,a) &= Q^*_r(s,a) - V^*_r(s) \\ Q^*_{r_{A^*_r}}(s,a) &= A^*_r(s,a) \\ max_a Q^*_{r_{A^*_r}}(s,a) &= max_a A^*_r(s,a) \\ max_a Q^*_{r_{A^*_r}}(s,a) &= 0 \\ V^*_{r_{A^*_r}}(s,a) &= 0 \end{aligned} \tag{9}$$

Eqn 9 above establishes two things that will be applied within Eqn 10 below, that $Q^*_{r_{A^*_r}}(s,a) = A^*_r(s,a)$ and that $V^*_{r_{A^*_r}}(s,a) = 0$.

$$\begin{aligned} Q^*_{r_{A^*_r}}(s,a) &= r_{A^*_r} + \gamma \mathbb{E}_{s'}[V^*_{\tilde{r}}(s')] \\ Q^*_{r_{A^*_r}}(s,a) &= r_{A^*_r} + \gamma \mathbb{E}_{s'}[0] \\ Q^*_{r_{A^*_r}}(s,a) &= r_{A^*_r} \\ A^*_r(s,a) &= r_{A^*_r} \end{aligned} \tag{10}$$

□

# D Detailed experimental settings

Here we provide details regarding the gridworld tasks and the learning algorithms used in our experiments. The learning algorithms described include both algorithms for learning from preferences and for policy improvement. Because much of the details below are repeated from Knox et al. [2022], some of the description in this section is adapted from that paper with permission from the authors.

## D.1 The gridworld domain and MDP generation

**Gridworld domain** Each instantiation of the gridworld domain consists of a grid of cells. In the following sections, each gridworld domain instantiation is referred to interchangeably as a randomly generated MDP.

A cell can contain up to one of four types of objects: "mildly good" objects, "mildly bad" objects, terminal success objects, and terminal failure objects. Each object has a specific reward component, and a time penalty provides another reward component. The reward received upon entering a cell is the sum of all reward components. The delivery agent's state is its location. The agent's action space consists of a single step in one of the four cardinal directions. The episode can terminate either at a terminal success state for a non-negative reward, or at a terminal failure state for a negative reward. The reward for a non-terminal transition is the sum of any reward components. The procedure for choosing the reward component of each cell type is described later in this subsection.

Actions that would move the agent beyond the grid's perimeter result in no motion and receive reward that includes the current cell's time penalty reward component but not any "mildly good" or "mildly bad" components. In this work, the start state distribution is always uniformly random over non-terminal states. This domain was introduced by [Knox et al. 2022].

**Standardizing return across MDPs and defining *near optimal* performance** To compare performance across different MDPs, the mean return of a policy $\pi$, $V^\pi_r$, is normalized to $(V^\pi_r - V^U_r)/V^*_r$, where $V^*_r$ is the optimal expected return and $V^U_r$ is the expected return of the uniformly random policy (both given the uniformly random start state distribution). Normalized mean return above 0 is better than $V^U_r$. Optimal policies have a normalized mean return of 1, and we consider above 0.9 to be *near optimal*.

Additionally, when plotting the mean of these standardized returns, we floor each such return at -1, which prevent the mean from being dominated by low performing policies that never terminate. Such policies can have, for example,

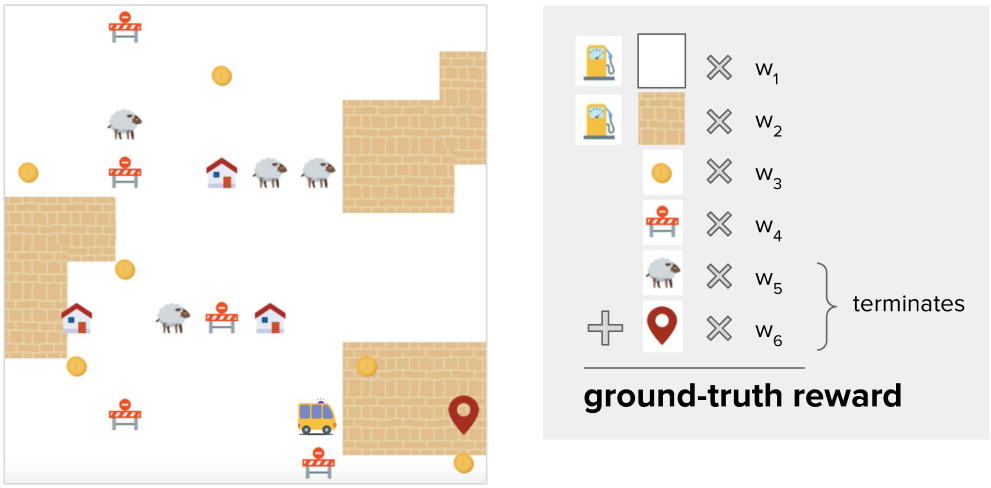


Figure 8: An example task and unspecified reward function from the gridworld delivery domain. Tasks from this domain are used in all experiments. In Appendix D.1, objects described as "mildly good" are shown as coins, objects described as "mildly bad" are shown as orange road blocks, objects described as "terminal failure states" are shown as sheep, and objects described as "terminal success states" are shown as red destination markers. The brick surface and the house object were not used in our experiments. The gridworld image is reprinted with permission, from Knox et al. [2022].

-1000 or -10000 mean standardized return, which we group together as a similar degree of failure, yet without flooring at -1, these two failing policies would have very different effects on the means.

**Generating the random MDPs used to create Figures 3, 4, and 6** Here we describe the procedure for generating the 100 MDPs used in Figure 6, which include the 30 MDPs used in Figures 3 and 4. This procedure was also used by [Knox et al. 2022].

The height for each MDP is sampled from the set $\{5, 6, 10\}$, and the width is sampled from $\{3, 6, 10, 15\}$. The proportion of cells that contain terminal failure objects is sampled from the set $\{0, 0.1, 0.3\}$. There is always exactly one cell with a terminal success object. The proportion of "mildly bad" objects is selected from the set $\{0, 0.1, 0.5, 0.8\}$, and the proportion of "mildly good" objects is selected from $\{0, 0.1, 0.2\}$. Each sampled proportion is translated to a number of objects (rounding down to an integer when needed), and then each of the object types are randomly placed in empty cells until the proportions are satisfied. A cell can have zero or one object in it.

Then the ground-truth reward component for each of the above cell or object types was sampled from the following sets:

- Terminal success objects: $\{0, 1, 5, 10, 50\}$
- Terminal failure objects: $\{-5, -10, -50\}$
- Mildly bad objects: $\{-2, -5, -10\}$

Mildly good objects always have a reward component of 1. An constant time penalty of -1 is also always applied.

**Generating random MDPs as seen in figure 5** For all 90 MDPs, the following parameters were used. The height for each MDP is sampled from the set $\{3, 5\}$, and the width is sampled from $\{1, 2\}$. There is always exactly one positive terminal cell that is randomly placed on one of the four corners of the board. The ground-truth reward component for the positive terminal state is sampled from $\{0, 1.5, 10\}$. These 90 MDPs do not contain any "mildly good" or "mildly bad" cells.

For 30/90 of the MDPs, it is always optimal to eventually terminate at either a terminal failure cell or a terminal success cell:

- For each MDP there is a $50\%$ chance that a terminal failure cell exists. If it does exist it is randomly placed on one of the four corners of the board.
- The ground-truth reward component for the terminal failure cell is sampled from $\{-5, -10\}$.

- The true reward component for blank cells is always $-1$.

For 30/90 of the MDPs, it is always optimal to eventually terminate at a terminal success cell:

- For each MDP there is always a terminal failure cell that exists and is randomly placed on one of the four corners of the board.
- The ground-truth reward component for the terminal failure cell is always $-10$.
- The true reward component for blank cells is always $-1$.

For 30/90 of the MDPs, it is always optimal to loop forever and never terminate:

- For each MDP there is always a terminal failure cell that exists and is randomly placed on one of the four corners of the board.
- The ground-truth reward component for the terminal failure cell is always $-10$.
- The true reward component for blank cells is always $+1$.

All parameters for randomly sampling MDPs that are not explicitly discussed above are the same as for Figures 3, 4, and 6.

### D.2 Learning algorithms

**Doubling the training set by reversing preference samples** To provide more training data and avoid learning segment ordering effects, for all preference datasets we duplicate each preference sample, swap the corresponding segment pairs, and reverse the preference.

**Discounting during value iteration and Q learning** Despite the gridworld domain being episodic, a policy may endlessly avoid terminal states. In some MDPs, such as a subset of those used in Figure 5, this is an optimal behavior. In other MDPs this is the result of a low-performing policy. To avoid an infinite loop of value function updates, we apply a discount factor of $\gamma = 0.999$ during value iteration, Q learning, and when assessing the mean returns of policies with respect to the ground-truth reward function, $r$. We chose this high discount factor to have negligible effect on the returns of high-performing policies (since relatively quick termination is required for high performance) while still allowing for convergence within a reasonable time.

**Hyperparameters for learning $\widehat{A}_r^*$ as seen in Figures 3, 4, 5, and 10** These hyperparameters exactly match those used in [Knox et al. 2022], except that we decreased the number of training epochs. For all experiments, each algorithm was run once with a single randomly selected seed.

- learning rate: 2
- number of seeds used: 1
- number of training epochs: $1,000$
- optimizer: Adam
  - $\beta_1 = 0.9$
  - $\beta_2 = 0.999$
  - eps= $1e-08$

**Hyperparameters for learning $\widehat{A}_r^*$ as seen in Figure 6** These hyperparameters exactly match those used in [Knox et al. 2022]. For all experiments, each algorithm was run once with a single randomly selected seed.

- learning rate: 2
- number of seeds used: 1
- number of training epochs: $30,000$
- optimizer: Adam

- $\beta_1 = 0.9$
- $\beta_2 = 0.999$
- eps= $1e-08$

**Hyperparameters for Q learning as seen in Figure 6** These hyperparameters were tuned on 10 learned $\widehat{A}^*_r$ functions where setting the reward function as $\widehat{A}^*_r$ and using value iteration to derive a policy was known to eventually lead to optimal performance. The hyperparameters were tuned so that, for each $\widehat{A}^*_r$ function in this set, Q-learning also yielded an optimal policy. For all experiments, each algorithm was run once with a single randomly selected seed.

- learning rate: 1
- number of seeds used: 1
- number of training episodes: $1,600$
- maximum episode length: 1000 steps
- initial Q values: 0
- exploration procedure: $\epsilon$-$greedy$
  - $\epsilon = 0.4$
  - decay=0.99

**Computer specifications and software libraries used** The compute used for all experiments had the following specification.

- processor: 1x Core™ i9-9980XE (18 cores, 3.00 GHz) & 1x WS X299 SAGE/10G — ASUS — MOBO;
- GPUs: 4x RTX 2080 Ti;
- memory: 128 GB.

Pytorch 1.7.1 [Paszke et al. 2019] was used to implement all reward learning models, and statistical analyses were performed using Scikit-learn 0.23.2 [Pedregosa et al. 2011].

## E Shifting such that the maximum value of $\widehat{A^*_r}$ in every state is 0

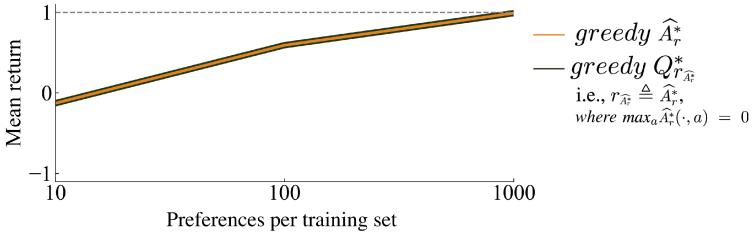


Figure 9: In 90 small gridworld MDPs, when we explicitly shift $\widehat{A}^*_r$ by a state-dependent constant such that $max_a \widehat{A}^*_r(\cdot, a) = 0$, we empirically observe no difference between $greedy\ \widehat{A}^*_r$ and $greedy\ Q^*_{r_{\widehat{A}^*_r}}$.

Corollary 3.2 claims that if $\max_a \widehat{A}^*_r(\cdot, a) = 0$, then $\Pi^*_{r_{\widehat{A}^*_r}} = \{\pi : \forall s, \forall a\ [\pi(a|s) > 0 \Leftrightarrow a \in \text{argmax}_a \widehat{A}^*_r(s, a)]\}$. Figure 9 shows the results of our empirical validation of this claim. Specifically, this figure shows that when $\max_a \widehat{A}^*_r(\cdot, a) = 0$, we observe no difference in the mean standardized return between $greedy\ \widehat{A}^*_r$ and $greedy\ Q^*_{r_{\widehat{A}^*_r}}$.

## F Encouraging $max_a \widehat{A}^*_r(\cdot, a) = 0$ without shifting learned values manually

Figure 4 in Section 3.3 uses noiselessly generated preferences. Figure 10 presents an analogous analysis for stochastically generated preferences. The pattern is similar results in the noiseless setting, with even less variance for large training sets. Specifically, Figure 10 shows that including transitions from absorbing states moves the resultant maximum values of the approximated optimal advantage function closer to 0.

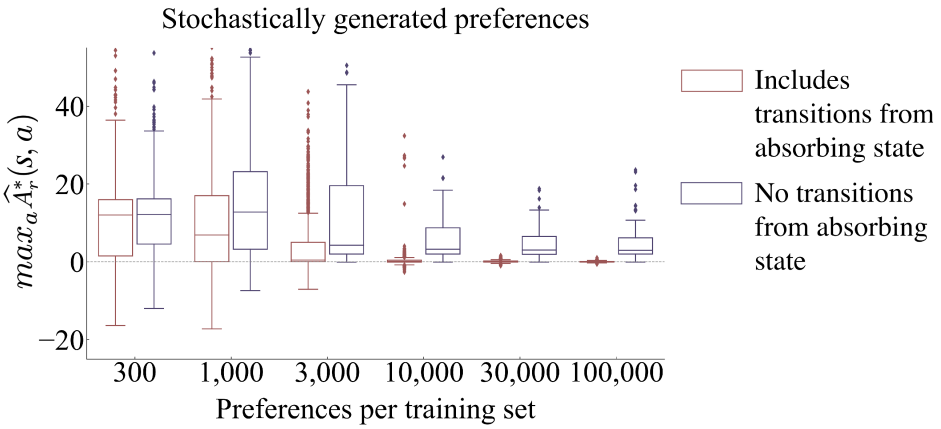


Figure 10: Comparing the effect on $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ of including transitions from absorbing state. For each state within 30 MDPs, the plots above show the $max_a \widehat{A}^*_r(s,a)$ values. The plot shows that including such transitions moves the resultant maximum values closer to 0. This plot complements Figure 4, which contains a similar plot for noiseless preferences. After learning with absorbing transitions, $max_a \widehat{A}^*_r(s,a)$ across all states is stochastically closer to 0 than when learning without them. Wilcoxon paired signed-rank tests at every training set size above 300 are extremely significant with $p < 10^{-57}$. For 300 preferences, $p = 0.0002$.

## G Investigation of performance differences

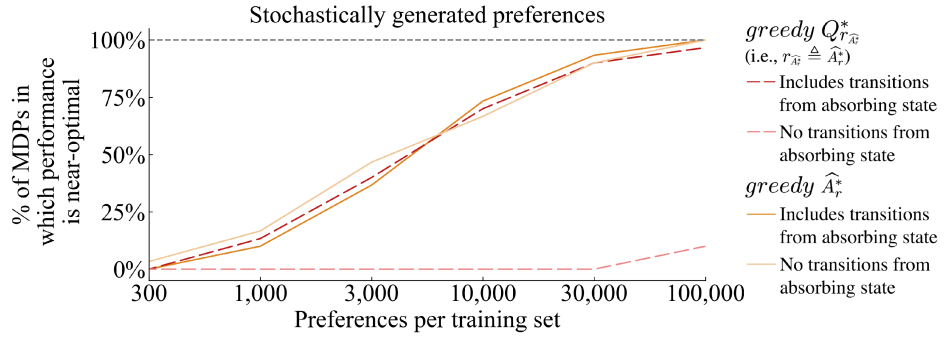


Figure 11: Performance when stochastically generated preference datasets do and do not include segments with transitions from absorbing state, complementing Figure 3. Results are across 30 randomly generated gridworld MDPs with tabular representations of the $\widehat{A}^*_r$, where segments of length 3 are chosen by uniformly randomly choosing a start state and 3 its actions. When transitions from absorbing states are not included, any segment that terminates before its final transition is rejected and then resampled. This plot complements Figure 4, which shows a similar plot for noiseless preferences. For $greedy\ \widehat{A}^*_r$ (in red) Wilcoxon paired signed-rank tests reveal that including transitions from absorbing state results in significantly higher performance for all training set sizes but the smallest, 300, with $p < 0.02$ for 1000, and $p < 0.0002$ for others. No significant difference in performance is detected for $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ with or without terminating transitions.

Figure 3 in Section 3.3 uses noiselessly generated preferences. As in the section above, Figure 11 presents an analogous analysis for stochastically generated preferences. This plot likewise shows that $greedy\ Q^*_{r_{\widehat{A}^*_r}}$ learned without transitions from absorbing state performs poorly. We also note that the performance of the other three conditions is more similar in comparison to Figure 3.