Human Feedback Guided Reinforcement Learning for Unknown Temporal Tasks via Weighted Finite Automata

Nathaniel Smith [1] Nicholas Hirsch [1] Yu Wang [1]

Abstract

Robots in real-world settings are commonly expected to complete complex temporal tasks—goals that have explicit order dependencies. Because such goals are difficult to specify mathematically, we explore how humans with intuitive task understanding can provide meaningful feedback on observed robot behavior. This work focuses on inferring the structure of a temporal task from pairwise human preferences and using it to guide reinforcement learning toward behavior aligned with human intent. Our method leverages limited feedback to construct a weighted finite automaton (WFA) that tracks task progress and shapes the learning process. We validate the approach in a Minigrid case study using real human feedback, showing that it enables agents to learn temporally structured tasks that would otherwise remain unlearned.

1. Introduction

Robots deployed in real-world settings are increasingly expected to accomplish complex temporal tasks—goals that unfold over time rather than being specified as static, one-shot objective (Choudhury et al., 2022; Nunes & Gini, 2015). Examples include navigating through a series of waypoints while avoiding restricted zones, monitoring equipment at regular intervals, or responding to events in a structured sequence (e.g., "after picking up object A, deliver it before returning to base") (Kivrak et al., 2022). Encoding such temporal behavior is crucial for enabling robust, high-level autonomy (Muscettola et al., 1998).

A human with an intuitive understanding of the task at hand (no math) can provide feedback on collected samples that significantly help train the robot to accomplish difficult goals (Meriçli et al., 2011; Wang et al., 2020; Knox et al., 2013). Crucially, this feedback need not come from an expert in logic or programming. A domain expert or any user with an intuitive understanding of the task can express preferences between different robot behaviors (St. Clair & Mataric, 2015; Otero et al., 2008). This kind of feedback encodes rich information about the task, even if it is not expressed in formal symbols. Traditionally, one might attempt to formalize these temporal tasks using Linear Temporal Logic (LTL) (Kloetzer & Belta, 2008; Lindemann et al., 2021). However, specifying correct and complete LTL formulas is often challenging for human operators, especially when they are not formally trained in logic (Barringer et al., 2013). Furthermore, in many practical scenarios, the desired behavior is not easily expressible as a strict logical formula: it may involve preferences, exceptions, or soft constraints that LTL does not easily capture. For these reasons, we propose to use human preference feedback as a means of teaching robots temporal behaviors in a more intuitive and flexible way.

We investigate how to learn the underlying structure of a temporal task from human preferences and use it to guide reinforcement learning toward behaviors that align with the human intent. In our framework, a human observes pairs of robot trajectories and provides ordinal feedback indicating which trajectory better aligns with the intended temporal task. These preferences are then used to train an agent to capture the underlying temporal pattern implicit in the preferred trajectories. Preferences feedback is often simple and robust—humans are generally better at saying "this is better than that" than at assigning absolute values or writing down formal rules (Tajwar et al., 2024; Gao et al., 2024; Chaudhari et al., 2024; Li et al., 2024). Furthermore, our preference-based feedback differentiates this work from many others which require feedback throughout the entire trajectory (Kauffman, 2023).

The novelty of this work lies in a method for leveraging human preference feedback to guide reinforcement learning toward behaviors that align with temporal human intent, along with a case study demonstrating its effectiveness using real human feedback. Given the assumption of limited data, the method employs statistical inference to identify the trajectory most likely to reflect human intent, rather than attempting to learn a full model of human preferences. To capture the temporal structure of the task, the method augments the agent's state space to track significant events without requiring storage of the entire trajectory. We validate this approach through a case study in a Minigrid environment, where we show that—under identical hyperparameters—the agent fails to learn the temporal task without our feedback-guided framework (Chevalier-Boisvert et al., 2023).

The rest of the manuscript is organized as follows. Section 2 introduces preliminary notations needed to describe transition systems. We introduce the problem formulation in Section 3. Section 4 describes how we can train an agent to follow human intent culminating with Algorithm 1 . A case study with real human feedback is presented in Section 5 to demonstrate the effectiveness of our novel Algorithm. Lastly, we conclude the work in Section 5

Related Works

RLHF has emerged as an effective tool in AI, with notable success in fine-tuning large language models (LLMs) (Yao et al., 2024; Vaswani et al., 2017). Beyond LLMs, RLHF has also demonstrated promise in areas such as robotics, where human feedback guides agents to perform complex tasks more efficiently than reward engineering in isolation (Yang et al., 2024; Meriçli et al., 2011; Knox & Stone, 2010). In recommendation systems and content moderation, RLHF techniques are being used to incorporate subjective human judgments, improving personalization and fairness (Stiennon et al., 2020). These success stories highlight RLHF's role in bringing raw model capabilities closer to real-world human expectations.

There have been few developments to the best of our knowledge that apply RLHF for temporal tasks. The important developments made in (Loftin et al., 2016; Knox & Stone, 2011; Wirth et al., 2017) do provide effective ways of learning a reward function from human feedback; however, they differ from our work in a few key ways. They presume human feedback is provided over an entire path. This approach requires a human to carefully observe the entire trajectory and provide feedback at each transition—such as by pressing a ”clicker” to indicate reward—which demands significantly more attention and effort than our preference-based feedback framework. From this style of feedback much effort has gone into deriving a state-dependent reward function. This assumes that the agent should receive the same reward each time it enters a specific state. We focus on temporal tasks where this does not hold true. The work in (Wang & Atanasov, 2022) shares similarities with ours in using a weighted finite automaton (WFA) to train agents on temporal tasks. However, it differs in two key ways. First, their approach relies on trajectories labeled with scalar scores, whereas we use only pairwise preferences. Second, they assume access to large amounts of labeled data to learn an explicit model of the task, while our method is designed to operate under limited human feedback. This distinction is significant: our approach demonstrates that effective learning is possible with fewer trajectories and without requiring numerical scores. Pairwise preferences not only reduce the burden on human annotators but are also less prone to error than scalar evaluations.

2. Preliminaries

In this section, we describe how we model the agent's interaction with its environment with a deterministic Markov decision process (DMDP). The results of this work are easily extendable to probabilistic models, and this is demonstrated in Sections 4 and 5 . Let $\mathbb{N}$ denote the natural numbers and $\mathbb{R}^n$ be the real-valued n-dimensional vector space. Let I_n denote the n-dimensional identity matrix and e_i be the ith standard basis vector.

Definition 2.1. A deterministic Markov decision process (DMDP) is a tuple

$$\mathcal{M} = (S, A, \Delta, \Sigma, L, s_0, R)$$

where:

- S is a finite set of *states*,
- A is a finite set of *actions*,
- $\Delta : S \times A \to S$ is a deterministic *transition function* that maps a state and action to a successor state,
- Σ is a finite set of events with the empty word $\epsilon \in \Sigma$,
- $L : S \to \Sigma$ is a *labeling function* that assigns to each state and event that is true in that state,
- $R : S \times A \to \mathbb{R}$ is a *reward function*,
- $s_0 \in S$ is the *initial state*.

Let $\mathcal{M} = (S, A, \Delta, \Sigma, L, s_0, R)$ be a deterministic DMDP. A **finite trajectory of length** n is a sequence of state-action pairs $\tau = ((s_0, a_0), (s_1, a_1), \dots, (s_n, a_n)) \in (S \times A)^{n+1}$, where for each $i = 0, 1, \dots, n$, $s_i \in S$, $a_i \in A$, and for $i < n$, the transition satisfies $s_{i+1} = \Delta(s_i, a_i)$. A **policy** is a function $\pi : S \to A$ that maps each state $s \in S$ to an action $a \in A$. The trajectory of length n induced by a policy π is $\tau_\pi = ((s_0, a_0), (s_1, a_1), \dots, (s_n, a_n)) \in (S \times A)^{n+1}$, such that for each $i = 0, 1, \dots, n-1$, the next state is given by $s_{i+1} = \Delta(s_i, \pi(s_i))$. With a slight abuse of notation, we extend the labeling function and reward function from individual states to full trajectories. We extend the labeling function using the Kleene Star * with $L(\tau) = L(s_0)L(s_1)...L(s_n) \in \Sigma^*$ and refer to such a sequence of events as σ (Cassandras & Lafortune, 2008). Similarly, for any, trajectory $\tau = ((s_0, a_0), (s_1, a_1), \dots, (s_n, a_n))$ we extend the reward function as

$$R(\tau) = \sum_{i=0}^{n} R(s_i, a_i). \tag{1}$$

The empty event ϵ has no effect on a sequence of events. The empty event is very important in practice because most states are not associated with a significant event that the human should be expected to monitor. To illustrate the role of ϵ we will provide a trajectory example.

Example 1. *Let $\mathcal{M}$ be a DMDP and $\tau = ((s_0, a_0), (s_1, a_1))$ is a trajectory in $\mathcal{M}$ with $L(s_0) = \sigma_0$ and $L(s_0) = \epsilon$. Then $L(\tau) = L(s_0)L(s_1)$ which after applying the labeling function becomes $L(\tau) = \sigma_0\epsilon = \sigma_0$ where the last equality holds because ϵ has no effect on the event sequence. The empty event is very important in practice because most states are not associated with a significant event that the human should be expected to monitor.*

3. Problem Formulation

Our goal is to learn a policy that completes a temporal task as inferred from human preference feedback. A human trainer, with intuitive knowledge of the task, observes pairs of trajectories and identifies which trajectory in each pair better fulfills the temporal objective. The likelihood of incorrect labeling is assumed to be inversely proportional to the difference in scores between the two trajectories. A higher score implies the path does a better job of accomplishing the unknown temporal task. Given a collection of such ordered preference pairs, our aim is to synthesize a policy that best reflects the preferences demonstrated by the human

This problem presents a significant challenge because human preferences are both **unknown** and **trajectory-dependent**, which prevents the use of a memoryless optimal policy. If a quantitative model of these preferences were known in advance, the optimal policy $\pi^{\max}$ could be computed efficiently using techniques such as shortest-path algorithms (e.g., Dijkstra's algorithm) (Shu-Xi, 2012; Gallo & Pallottino, 1988). However, human intent is often complex and difficult to model explicitly. Unlike traditional reward functions, which are typically defined over individual state-action pairs, human preferences depend on the entire trajectory τ_π, making them unsuitable for local evaluation or optimization. As a result, purely state-dependent policies are insufficient, as they cannot track which task-relevant events have occurred. Although memory-based policies could theoretically resolve this by storing the full trajectory, they quickly become computationally intractable, especially in probabilistic environments (Sutton & Barto, 2018). Furthermore, because human feedback is expensive to obtain, we do **not** assume that the dataset $\mathcal{D}$ is large enough to support full preference model learning.

Consider the human observes a series of trajectories through $\mathcal{M}$ and provides a dataset:

$$\mathcal{D} = \left\{(\sigma_i^+, \sigma_i^-)\right\}_{i=1}^N$$

where $\sigma = L(\tau)$ denotes the sequence of events or word associated with the trajectory τ through a DMDP $\mathcal{M}$. Each pair indicates that the σ_i^+ is **preferred** over σ_i^- by the human, meaning it better satisfies the underlying (and unknown) temporal task. We assume the human observer has an intuitive understanding of the temporal task, enabling the provision of useful preference based feedback. We also assume the human **only** focuses on the sequence of events associated with τ rather than the entire trajectory itself. This reduces the workload on the human. Furthermore, subtle differences in trajectory are difficult for a human to discern and may distract the human from focusing on the more important event sequences.

We use the Bradley-Terry (BT) probabilistic model to account for human labeling errors. It is reasonable to assume that a human is more likely to "mislabel" a pair if the two paths receive similar levels of approval from the human with regards to the temporal task. This intuitive concept is precisely provided with the BT probabilistic model. The BT model defines the probability that σ^+ is preferred to σ^- as

$$P(\sigma^+ \succ \sigma^-) = \frac{\exp(f(\sigma^+))}{\exp(f(\sigma^+)) + \exp(f(\sigma^-))} \tag{2}$$

where $f : \Sigma^* \to \mathbb{R}$ is an unknown "scoring" function and $\succ$ denotes preference. The BT model has shown to be very effective in modeling randomness with ordered pairs obtained from interactions (Matthews & Morris, 1995; Cattelan et al., 2013).

Our formal objective is to recover a policy that aligns with the temporal preferences encoded in $\mathcal{D}$. This leads to the optimization problem:

$$\pi^{\max} = \arg\max_{\pi} f(L(\tau_\pi)) + R(\tau_\pi) \tag{3}$$

where R is the reward function for the DMDP $\mathcal{M}$ associated with the environment.

In Equation (3), the term $f(L(\tau_\pi))$ represents the reward associated with aligning the agent's behavior with human-specified temporal intent, while $R(\tau_\pi)$ captures the reward for efficient interaction with the environment. These two components are treated separately for both practical and conceptual reasons. First, human feedback is often sparse and coarse, making it unrealistic to expect detailed feedback on low-level motion or step-by-step decisions. Instead, humans typically express preferences over high-level temporal patterns—such as completing subtasks in the correct order—which is modeled by $f(L(\tau_\pi))$, where $L(\tau_\pi)$ is the sequence of labels induced by the policy π. Second, we assume that R is a known, environment-defined reward function that encourages efficient behavior, such as minimizing time or energy. For example, $f(L(\tau_\pi))$ might reward executing tasks in the correct order (e.g., picking up a key before opening a door), while $R(\tau_\pi)$ rewards completing this task in the fewest possible steps.

4. Learning a Policy from Human Preferences

To compute a policy that aligns with human feedback, we augment the environment's state space to track task-relevant events observed by the agent. This begins by identifying the event sequence that best reflects the temporal objective encoded in the preference dataset $\mathcal{D}$ (see Section 4.1) since no model is available for the human preferences. We use this sequence to construct a weighted finite automaton (WFA) that assigns real-valued scores to trajectories based on how closely their induced event sequences match the inferred human intent in Section 4.2. Augmenting the WFA with the transition space allows an agent to accomplish goals with a memoryless policy that otherwise would have required a memory-dependent policy. This construction enables efficient event tracking and supports direct computation of the optimal policy via shortest-path algorithms in the deterministic case in Section 4.3. These results are extended to the probabilistic case in Section 4.4 where the WFA not only augments the state space, but also provides a state-dependent reward function leading to a memoryless policy.

4.1. Statistical Inference of Temporal Human Intent

We use statistical inference to identify the sequence of events $\sigma^{\max}$ that best aligns with human preferences by maximizing the objective in Equation (3). Since human feedback is costly and limited, we do not attempt to learn the full scoring function f. Furthermore, we do not assume f is of any specific form. Thus, we instead focus on extracting the most preferred event sequence as a surrogate for human intent. The agent is then rewarded based on how closely its behavior matches this inferred sequence $\sigma^{\max}$, thereby guiding learning without requiring full recovery of f.

To identify the highest-scoring event sequence from human preference data, we perform maximum likelihood estimation (MLE) using the Bradley-Terry (BT) probabilistic model. Given a dataset of pairwise preferences $\mathcal{D} = \{(\sigma_i^+, \sigma_i^-)\}_{i=1}^N$, where each pair indicates that event sequence σ_i^+ is preferred over σ_i^-, the likelihood of the dataset under the BT model is given by:

$$\mathcal{L}(\mathcal{D}; f) = \prod_{i=1}^{N} \frac{\exp(f(\sigma_i^+))}{\exp(f(\sigma_i^+)) + \exp(f(\sigma_i^-))}. \tag{4}$$

As shown in (Glickman, 2013) the following iterative update can be used to compute the event sequence scores that maximize the log-likelihood in (4):

$$f_i = \log\left(\frac{\sum_j w_{ij}}{\sum_j (w_{ij} + w_{ji})/(e^{f_i} + e^{f_j})}\right). \tag{5}$$

where f_i is the score assigned to event sequence σ_i, and w_{ij} denotes the number of times σ_i was preferred over σ_j in the dataset. In practice, we solve for the scores f_i and the highest scoring event sequence using the implementation provided in the Choix Python package (Maystre, 2022). This result is formalized with the following Lemma.

Lemma 4.1. *Let $\mathcal{F} = \{\sigma_1, \dots, \sigma_M\}$ be a finite set of event sequences, and let $\mathcal{D} = \{(\sigma_i^+, \sigma_i^-)\}_{i=1}^N \subseteq \mathcal{F} \times \mathcal{F}$ be a dataset of pairwise preferences, where each ordered pair (σ_i^+, σ_i^-) indicates that sequence σ_i^+ is preferred over σ_i^-. We assume the following:*

- *For all $i \in \{1, \dots, N\}$, it holds that $\sigma_i^+, \sigma_i^- \in \mathcal{F}$,*
- *For all $\sigma \in \mathcal{F}$, there exists at least one $i \in \{1, \dots, N\}$ such that $\sigma = \sigma_i^+$ or $\sigma = \sigma_i^-$.*

That is, the dataset $\mathcal{D}$ is defined over elements of $\mathcal{F}$, and every element of $\mathcal{F}$ appears in at least one preference pair. Suppose the preferences are modeled according to the Bradley-Terry (BT) probabilistic model with (2).

The maximum likelihood estimates $\{\hat{f}_i\}_{i=1}^M$, where $\hat{f} : \mathcal{F} \to \mathbb{R}$ is the inferred scoring function and $\hat{f}_i = \hat{f}(\sigma_i)$, can be computed by maximizing the log-likelihood defined in Equation (4). These estimates are obtained using the following iterative fixed-point update (Glickman, 2013):

$$\hat{f}_i^{t+1} = \log\left(\frac{\sum_j w_{ij}}{\sum_j (w_{ij} + w_{ji})/(e^{\hat{f}_i^t} + e^{\hat{f}_j^t})}\right) \tag{6}$$

where w_{ij} denotes the number of times event sequence σ_i was preferred over σ_j in $\mathcal{D}$. The update converges to a local maximum of the log-likelihood under mild regularity conditions (Bong & Rinaldo, 2022; Hunter, 2004). We define the highest-scoring event sequence in the maximum likelihood sense as the element of $\mathcal{F}$ with the greatest score under the inferred function $\hat{f}$. That is,

$$\sigma^{max} = \arg\max_{\sigma \in \mathcal{F}} \hat{f}(\sigma), \tag{7}$$

where $\hat{f} : \mathcal{F} \to \mathbb{R}$ is the scoring function estimated from the dataset $\mathcal{D}$ using the Bradley-Terry model.

We use Lemma 4.1 to obtain a ranking system for the event sequences recorded in $\mathcal{D}$ that aligns with the human intent while simultaneously accounting for mislabels. The estimate function $\hat{f}$ is only used to obtain the ranking system. Unlike learning the full scoring function f, obtaining a preference-consistent ranking requires significantly less data (Braga-Neto, 2024), making it more practical in settings where human feedback is limited. Because the model maximizes the likelihood of observed preferences, it naturally handles mislabels. For instance, if the pair (σ_a, σ_b) is presented ten times and the human selects σ_a nine out of ten times, the model will still assign a higher score to σ_a, reflecting the overall trend in the feedback.

4.2. Temporal Task Tracking via Weighted Finite Automata

We derive a mechanism from the event sequence $\sigma^{\max}$ in the form of a Weighted Finite Automaton (WFA) to guide the reinforcement learning process. This WFA enables the agent to accomplish temporal tasks that are dependent on the entire trajectory rather than each state-action pair in isolation. The WFA guides training by acting as a "flow chart". More precisely, it augments the state space with temporal progress, enabling the agent to track task completion independently of the DMDP state. It also supports structured reward shaping by assigning intermediate rewards or penalties based on how well the agent follows the target temporal sequence $\sigma^{\max}$ for probabilistic systems. A formal definition of a WFA is given below.

Definition 4.2. We define a weighted finite automaton (WFA) as

$$\mathcal{A} = (\boldsymbol{t_0}, \{\mathbf{T}_\sigma\}_{\sigma\in\Sigma}, \boldsymbol{t_f})$$

where:

- $\boldsymbol{t_0} \in \mathbb{R}^n$ is the initial weight vector,
- $\boldsymbol{t_f} \in \mathbb{R}^n$ is the final weight vector,
- For each event $\sigma \in \Sigma$, $\mathbf{T}_\sigma \in \mathbb{R}^{n\times n}$ is a transition weight matrix.

Given a finite word or event sequence $\sigma = \sigma_1\sigma_2\ldots\sigma_k \in \Sigma^*$, the WFA defines a real-valued function:

$$f_{\mathcal{A}}(\sigma) = \boldsymbol{t_0}^\top \mathbf{T}_{\sigma_1}\mathbf{T}_{\sigma_2}\cdots\mathbf{T}_{\sigma_k}\boldsymbol{t_f}.$$

This score $f_{\mathcal{A}}(\sigma) \in \mathbb{R}$ represents how well the sequence σ aligns with a temporal pattern and we colloquially refer to it as the score of a word. This scoring function is used later used to shape both the reward function and state augmentation in the RL framework.

We will use $\sigma^{\max}$ to create a specific WFA $\mathcal{A}^{\max}$ such that its associated scoring function $f_{\mathcal{A}^{\max}}$ will increase as the trajectory follows $\sigma^{\max}$ and decrease if the agent's trajectory deviates from $\sigma^{\max}$. The scoring function $f_{\mathcal{A}^{\max}}$ achieves this by applying a linear transformation to the current score at each step, where the transformation depends on the most recently observed event in Σ. If the event aligns with the next expected event in $\sigma^{\max}$, the event sequence's score increases. Otherwise, the score decreases, thereby discouraging divergence from the target sequence. Finally, this form of WFA provides a more expressive representation than classical automata, which only indicate binary acceptance or rejection. In contrast, the WFA assigns a real-valued score that quantifies how closely a given sequence aligns with the preferred sequence $\sigma^{\max}$ inferred from human behavior (Khoussainov & Nerode, 2012).

Definition 4.3. Let $\sigma^{\max} = \sigma_0^{\max}\sigma_1^{\max}\ldots\sigma_{n-1}^{\max} \in \Sigma^{\max}$ be a finite word of length n, and define a function $N : \Sigma \to 2^{\mathbb{N}}$ such that:

$$N(\sigma) = \{i \mid i \in \{0, 1, \ldots, n-1\}, \sigma_i^{\max} = \sigma\}.$$

This function records the indices at which each event $\sigma \in \Sigma$ appears in the sequence $\sigma^{\max}$. We say that a Weighted Finite Automaton $\mathcal{A} = (\boldsymbol{t_0}, \{\mathbf{T}_\sigma\}_{\sigma\in\Sigma}, \boldsymbol{t_f})$ *accepts* $\sigma^{\max}$ if the following conditions are satisfied:

- $\boldsymbol{t_0} = e_1 \in \mathbb{R}^n$, the first standard basis vector,
- $\boldsymbol{t_f} = [1, 1, \ldots, 1]^\top \in \mathbb{R}^n$,
- $T_\epsilon = I_n$ where ϵ is the empty word,
- For each $\sigma \in \Sigma$, the transition matrix is defined as:

$$\mathbf{T}_\sigma = cI_n + \sum_{i\in N(\sigma)} \left(pe_ie_{i+1}^\top - ce_ie_i^\top\right),$$

- where $p > 1$ and $0 < c < 1$ are constants.

Here, n is the length of $\sigma^{\max}$, and $e_i \in \mathbb{R}^n$ denotes the i-th standard basis vector. We refer to p as the progress factor and c as the stationary factor.

The scoring capabilities of $\mathcal{A}^{\max}$ that accepts $\sigma^{\max}$ will now be demonstrated with an example. Observe that $f_{\mathcal{A}^{\max}}$ is not meant to model the unknown scoring function f entirely, but rather just its inferred maximum $\sigma^{\max}$

Example 2. *Consider the the scenario where σ^{max} = (pickup box = σ_0, open door = σ_1). The WFA, $\mathcal{A}^{max}$, that accepts σ^{max} is as follows.*

$$\boldsymbol{t}_0 = \begin{bmatrix}1 & 0 & 0\end{bmatrix}, \boldsymbol{t}_f = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

$$T_{\sigma_0} = \begin{bmatrix}0 & 1.25 & 0\\0 & 0.75 & 0\\0 & 0 & 0.75\end{bmatrix}, T_{\sigma_1} = \begin{bmatrix}0.75 & 0 & 0\\0 & 0 & 1.25\\0 & 0 & 0.75\end{bmatrix}$$

where $p = 1.25$ is a hyperparemeter used to reward forward progress through the temporal objective and $c = 0.75$ is a hyperparemeter that penalizes events that are not aligned with the temporal objective. Observe that the score of the highest scoring word is σ^{max} and $f_{\mathcal{A}}(\sigma_0\sigma_1) = 1.5625$ which is greater than zero. On the contrary for a less desired word $\sigma_0\sigma_0\sigma_1$ we have that $f_{\mathcal{A}}(\sigma_0\sigma_0\sigma_1)$ =1.17 which is less than $f_{\mathcal{A}}(\sigma^{max})$.

Lastly, we conclude with a Lemma summarizing how a WFA that accepts a word $\sigma^{\max}$ is maximized at $\sigma^{\max}$. This Lemma is significant because it proves that if the agent maximizes $f_{\mathcal{A}^{\max}}$ then it will also follow $\sigma^{\max}$.

Lemma 4.4. *Let Σ be an alphabet, $\sigma^{max} \in \Sigma^*$ and $\mathcal{A}^{max}$ be a WFA that accepts σ^{max}. Then $f_{\mathcal{A}^{max}}(\sigma^{max}) = p^n$ where n is the length of σ^{max} and p is the progress factor. Furthermore, for all $\sigma \in \Sigma^*$ for which $\sigma^{max} \neq \sigma$ we have that $f_{\mathcal{A}^{max}}(\sigma^{max}) > f_{\mathcal{A}^{max}}(\sigma)$.*

Proof. Let Σ, $\sigma^{\max}$, and $\mathcal{A}^{\max}$ be as defined above. Proving $f_{\mathcal{A}^{\max}}(\sigma^{\max}) = p^n$ falls immediately from induction. Showing $f_{\mathcal{A}^{\max}}(\sigma^{\max}) > f_{\mathcal{A}^{\max}}(\sigma)$ is also trivial. We can demonstrate that if an event occurs that does not follow $\sigma^{\max}$ then the score will be multiplied by the stationary factor c, which is less than one. $\square$

4.3. WFA Guided Policy for Deterministic Systems

We will now present how the WFA that accepts $\sigma^{\max}$ can provide us with a closed form solution for an optimal policy by replacing the unknown scoring function with the WFA in (3) from Section 3. Specifically, we reformulate the original unconstrained optimization problem into a constrained one, where the policy must generate a trajectory whose induced event sequence follows $\sigma^{\max}$. By constructing the product of the WFA and the DMDP, we can apply a shortest-path algorithm to compute a trajectory that both satisfies the temporal structure inferred from human preferences and maximizes the known state-dependent reward function.

The product between a WFA and DMDP that tracks the agent's progress through the temporal task described with $\sigma^{\max}$ will now be defined

Definition 4.5. Let $\mathcal{M} = (S, A, \Delta, \Sigma, L, s_0)$ be a DMDP, and let

$$\mathcal{A} = (\boldsymbol{t_0}, \{\mathbf{T}_\sigma\}_{\sigma\in\Sigma}, \boldsymbol{t_f})$$

be a Weighted Finite Automaton (WFA) with $\boldsymbol{t_0} \in \mathbb{R}^n$, $\boldsymbol{t_f} \in \mathbb{R}^n$, and transition matrices $\mathbf{T}_\sigma \in \mathbb{R}^{n\times n}$ for each $\sigma \in \Sigma$.

The **product DMDP** $\mathcal{M} \times \mathcal{A}$ is defined as:

$$\tilde{\mathcal{M}} = (\tilde{S}, A, \tilde{\Delta}, \Sigma, \tilde{L}, (s_0, \boldsymbol{t_0}))$$

where:

- $\tilde{S} = S \times \mathbb{R}^n$ is the product state space,
- $\tilde{\Delta} : \tilde{S} \times A \to \tilde{S}$ is the transition function defined by:

  $$\tilde{T}((s, \boldsymbol{t}), a) = (s', \boldsymbol{t}'),$$

  where $s' = \Delta(s, a)$ and $\boldsymbol{t}' = \mathbf{T}_{L(s')}\boldsymbol{t}$,
- $\tilde{L}((s, \boldsymbol{t})) = L(s)$ extends the labeling function to the product state.

This augmented space tracks both the environment state and the automaton's internal state, allowing the agent to reason about its progress through $\sigma^{\max}$ The resulting product MDP enables reward shaping and policy learning to account for temporal structure. The WFA state has no direct way of affecting the original state dynamics. It only affects the policy. Example 3 provides a graphical representation of how the product space tracks the agent's progress relative to $\sigma^{\max}$.

We will now provide an example that is an extension of Example 2 to demonstrate how the reward function guides the policy to follow $\sigma^{\max}$.

Example 3. *Continuing from Example 2 with σ^{max} = (pickup box = σ_0, open door = σ_1). We provide the graphical representation of the WFA in Figure 1 to illustrate how the WFA state can easily account for which events have been recorded in the trajectory. The "up" arrows at a transition indicate a positive reward while the "down" arrows indicate a negative reward. If no events have occurred then the WFA portion of the product transition state $\boldsymbol{t}1$will be e_1. As the agent follows σ^{max} the WFA state transitions from $\boldsymbol{t}_1 = [0, 1.25, 0]$ and $\boldsymbol{t}_2 = [0, 0, 1.5625]$. Thus, as t evolves along the trajectory the agent knows where along σ^{max} it is.*

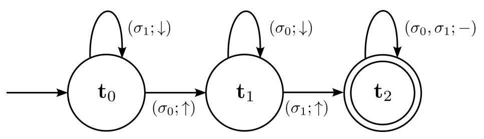

Figure 1. Graphical representation of WFA that accepts $\sigma^{\max} = \sigma_0\sigma_1$. The "up" arrows at a transition indicate a positive reward while the "down" arrows indicate a negative reward. Thus rewarding the agent if it follows $\sigma^{\max}$.

A lemma will now be introduced to show a closed-form solution for obtaining a policy that follows $\sigma^{\max}$ while maximizing the known reward function discussed in Section 3.

Lemma 4.6. *Let $\mathcal{M}$ be a DMDP, $\mathcal{D} = \left\{(\sigma_i^+, \sigma_i^-)\right\}_{i=1}^N$ be a set of preferences, where each σ is a sequence of events, and σ^{max} is the preferred sequence derived from (5). Let $\mathcal{A}^{max}$ be the WFA that accepts σ^{max}. Then the policy that maximizes the reward function defined with $\mathcal{M}$ while following σ^{max} is*

$$\pi^{max} = \arg\max_{\pi} R(\tau_\pi) \text{ such that } f_{\mathcal{A}^{max}}(L(\tau_\pi)) = p^n \quad (8)$$

where p is the progress factor in $\mathcal{A}^{max}$ and n is the number of events in σ^{max}. This equation is solved through any shortest path algorithm.

Although Lemma 4.6 applies only to deterministic DMDPs—which may not fully capture the uncertainty present in real-world environments—the same underlying principles can be extended to the probabilistic case. In particular, we continue to use a weighted finite automaton (WFA) to track the agent's progress in following the preferred event sequence, enabling temporal structure to inform policy learning even when transitions are uncertain.

4.4. WFA augmented RL Training

A WFA that accepts the highest scoring recording word can help guide the reinforcement learning process towards a policy that aligns with the human preferences. The accepting WFA influences training in two key ways. First, it extends the state space to include the automaton state, which tracks the agent's progress through the temporal task described with $\sigma^{\max}$. This addition makes the policy more robust to reward hacking, as the WFA state changes in response to task-relevant events—even when the underlying DMDP state does not. In this sense, the WFA acts as a "temporal counter," recording which parts of the task have been completed. Second, the WFA enables the design of structured reward signals by assigning intermediate rewards or penalties based on the agent's advancement through the temporal pattern encoded by $\sigma^{\max}$.

The scoring function $f_{\mathcal{A}^{\max}}$, derived from the WFA, provides a reward mechanism that promotes forward progress through the target sequence $\sigma^{\max}$, while penalizing deviations. This reward function serves as a heuristic that operates independently of the DMDP state. Importantly, this independence makes the reward robust to reward hacking, as the agent cannot exploit repeated DMDP states to consistently receive the same reward—rewards are instead tied to progress through the temporal task (Sutton & Barto, 2018). Specifically, the agent is rewarded or penalized based on whether the score $f_{\mathcal{A}^{\max}}$ increases or decreases after incorporating a new event. The reward function only needs to track the current automaton state $\boldsymbol{t}_i$ and the previous one $\boldsymbol{t}_{i-1}$, making it independent of the entire sequence of events. We will define it below

Definition 4.7. Let $\boldsymbol{t}_{i-1}, \boldsymbol{t}_i \in \mathbb{R}^n$ be the previous and current WFA state vectors, and let $\boldsymbol{t}_f \in \mathbb{R}^n$ be the final weight vector of the WFA. Let R be a reward function from a probabilistic MDP. The reward function at time step i is defined as:

$$R(s_i, \boldsymbol{t}_i) = \begin{cases} r_p + R(s_i) & \text{if } (\boldsymbol{t}_i - \boldsymbol{t}_{i-1})^\top \boldsymbol{t}_f > 0 \\ -r_c + R(s_i) & \text{if } (\boldsymbol{t}_i - \boldsymbol{t}_{i-1})^\top \boldsymbol{t}_f < 0, \end{cases}$$

where:

- $r_p > 0$: reward for making forward progress through the WFA,
- $r_c > 0$: penalty for regressing from the temporal objective.

We conclude this section by presenting an algorithm that summarizes our method for both DMDP and MDP. Once the reward function is obtained with Definition (1) we can employ any reinforcement learning algorithm of our choice since the reward is purely state dependent (Sutton & Barto, 2018).

Algorithm 1 Temporal Preferences to Policy (TP2P)

```
1: Input: MDP $\mathcal{M}$, and preference set $\mathcal{D} = \{(\sigma_i^+, \sigma_i^-)\}_{i=1}^N$, where each $\sigma$ is a sequence of events
2: Output: A memoryless optimal policy $\pi^{\max}$
3: Obtain $\sigma^{\max}$ through statistical inference using iterative updates of Equation (5)
4: Construct a weighted finite automaton that accepts $\sigma^{\max}$, denoted $\mathcal{A}^{\max}$, and define the corresponding scoring function $f_{\mathcal{A}^{\max}}$
5: if $\mathcal{M}$ is deterministic then
6:     Build product DMDP between $\mathcal{M}$ and $\mathcal{A}^{\max}$: $\tilde{\mathcal{M}} = (\tilde{S}, A, \tilde{\Delta}, \Sigma, \tilde{L}, (s_0, \boldsymbol{t_0}))$
7:     Compute $\pi^{\max} = \arg\max_\pi [f_{\mathcal{A}^{\max}}(\tau_\pi) + R(\tau_\pi)]$ using a shortest-path solver
8: else
9:     Build product system between the probabilistic MDP $\mathcal{M}$ and $f_{\mathcal{A}^{\max}}$
10:    Use $\mathcal{A}^{\max}$ to define the reward function $R_{\mathcal{A}^{\max}}$ from Definition 4.7
11:    Procure $\pi^{\max}$ from any reinforcement learning algorithm on the product space with reward $R_{\text{prod}} = R_{\mathcal{A}^{\max}} + R$
12: end if
```

5. Minigrid Case Study

5.1. Simulation Setup

Our TP2P Algorithm was implemented on a temporal task using the `Minigrid` library in Python to demonstrate the effectiveness of our algorithm with real human feedback. An agent was trained on a POMDP gridworld designed to simulate a warehouse environment in which the following sequence of events was to be followed: locate and pickup a key, use said key to unlock and open a door, pass through the door and close it once on the other side, drop the key, and finally locate and pickup a box. The agent can move forward, turn left or right, pickup and drop objects, and interact with certain objects (in this case being able to unlock/open and close doors). First, a human observer with knowledge of the temporal goal reviewed a series of trajectory pairs and indicated a preference for one trajectory in each pair. Next, the accepting WFA of the temporal task, $\mathcal{A}^{\max}$ is obtained via TP2P (Algorithm 1). A graphical representation of $\mathcal{A}^{\max}$ is provided in 3. Then, this WFA is augmented with the environment state space to provide intermediary rewards for moving through the WFA. Lastly, a policy is procured

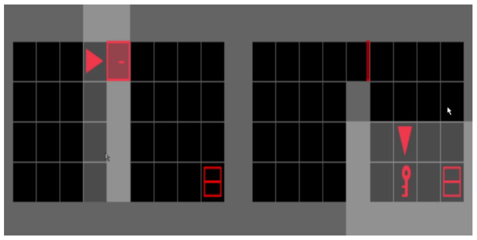

Figure 2. The gridworld on which the agent was trained. Shown are example rollout trajectories presented to the user to obtain ordinal feedback.

through proximal policy optimization (PPO) (Schulman et al., 2017).

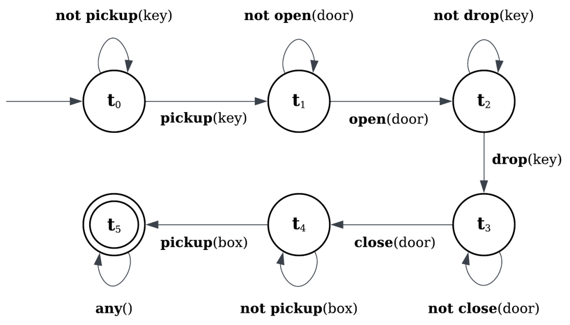

Figure 3. The WFA learned from human feedback on the rollout trajectories.

5.2. Results

It was found that when using PPO the WFA augmented POMDP converged to a steady-state average return of slightly over 1.00 per episode while the baseline using only the diminishing temporal reward failed to converge in the same time frame. Human feedback was provided on 15 different trajectory pairs and required a total of 5 minutes and 10 seconds of human feedback to watch the 15 pairs. The training scheme used $1,000,000$ frames on `Minigrid` with four parallel environments. The entire end-to-end algorithm took 7 minutes to run on a 12th Gen Intel(R) Core(TM) i5-12500 processor with 32 GB of RAM. It was found that without a pseudo-memory policy as allowed by the WFA the agent was unable to converge to a successful policy in the time allotted with all other hyperparameters being equal. However, with the WFA convergence was rapid – occurring in less than $800,000$ frames on average.

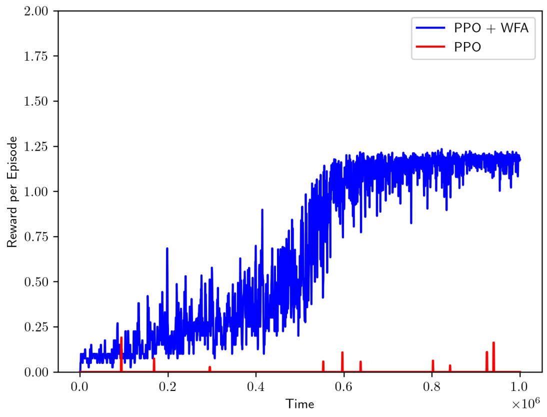

Figure 4. Average reward per episode over time for both the PPO and PPO + WFA schemes.

6. Conclusion

This work presents a method for leveraging pairwise human preferences to guide reinforcement learning in temporally structured tasks. By inferring a preferred event sequence and encoding it as a weighted finite automaton, we enable agents to track task progress and align their behavior with human intent using limited feedback. Our case study in a Minigrid environment demonstrates the effectiveness of this approach, showing that it enables successful learning of temporal tasks that would otherwise remain unlearned.

References

Barringer, H., Fisher, M., Gabbay, D. M., and Gough, G. *Advances in Temporal Logic*. Springer Science & Business Media, November 2013. ISBN 978-94-015-9586-5. Google-Books-ID: vob1CAAAQBAJ.

Bong, H. and Rinaldo, A. Generalized Results for the Existence and Consistency of the MLE in the Bradley-Terry-Luce Model. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 2160–2177. PMLR, June 2022. URL `https://proceedings.mlr.press/v162/bong22a.html`. ISSN: 2640-3498.

Braga-Neto, U. *Fundamentals of Pattern Recognition and Machine Learning*. Springer International Publishing, Cham, 2024. ISBN 978-3-031-60949-7 978-3-031-60950-3. doi: 10.1007/978-3-031-60950-3. URL `https://link.springer.com/10.1007/978-3-031-60950-3`.

Cassandras, C. G. and Lafortune, S. *Introduction to discrete event systems*. Springer, New York, NY, 2. ed edition,

2008. ISBN 978-0-387-33332-8 978-1-4419-4119-0 978-0-387-68612-7.

Cattelan, M., Varin, C., and Firth, D. Dynamic Bradley–Terry Modelling of Sports Tournaments. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 62(1):135–150, January 2013. ISSN 0035-9254. doi: 10.1111/j.1467-9876.2012.01046.x. URL https://doi.org/10.1111/j.1467-9876.2012.01046.x.

Chaudhari, S., Aggarwal, P., Murahari, V., Rajpurohit, T., Kalyan, A., Narasimhan, K., Deshpande, A., and Silva, B. C. d. RLHF Deciphered: A Critical Analysis of Reinforcement Learning from Human Feedback for LLMs, April 2024. URL http://arxiv.org/abs/2404.08555. arXiv:2404.08555 [cs].

Chevalier-Boisvert, M., Dai, B., Towers, M., Perez-Vicente, R., Willems, L., Lahlou, S., Pal, S., Castro, P. S., and Terry, J. Minigrid & Miniworld: Modular & Customizable Reinforcement Learning Environments for Goal-Oriented Tasks. *Advances in Neural Information Processing Systems*, 36:73383–73394, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e8916198466e8ef218a2185a491b49fa-Abstract-Datasets_and_Benchmarks.html.

Choudhury, S., Gupta, J. K., Kochenderfer, M. J., Sadigh, D., and Bohg, J. Dynamic multi-robot task allocation under uncertainty and temporal constraints. *Autonomous Robots*, 46(1):231–247, January 2022. ISSN 1573-7527. doi: 10.1007/s10514-021-10022-9. URL https://doi.org/10.1007/s10514-021-10022-9.

Gallo, G. and Pallottino, S. Shortest path algorithms. *Annals of Operations Research*, 13(1):1–79, December 1988. ISSN 1572-9338. doi: 10.1007/BF02288320. URL https://doi.org/10.1007/BF02288320.

Gao, G., Taymanov, A., Salinas, E., Mineiro, P., and Misra, D. Aligning LLM Agents by Learning Latent Preference from User Edits, November 2024. URL http://arxiv.org/abs/2404.15269. arXiv:2404.15269 [cs].

Glickman, M. E. Introductory note to 1928 (= 1929). In Zermelo, E., Ebbinghaus, H.-D., and Kanamori, A. (eds.), *Ernst Zermelo - Collected Works/Gesammelte Werke II: Volume II/Band II - Calculus of Variations, Applied Mathematics, and Physics/Variationsrechnung, Angewandte Mathematik und Physik*, pp. 616–671. Springer, Berlin, Heidelberg, 2013. ISBN 978-3-540-70856-8. doi: 10.1007/978-3-540-70856-8_13. URL https://doi.org/10.1007/978-3-540-70856-8_13.

Hunter, D. R. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics*, 32(1):384–406, February 2004. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1079120141. Publisher: Institute of Mathematical Statistics.

Kauffman, T. A SURVEY OF REINFORCEMENT LEARNING FROM HUMAN FEEDBACK. *arXiv preprint arXiv:2312.14925 10 (2023).*, October 2023.

Khoussainov, B. and Nerode, A. *Automata Theory and its Applications*. Springer Science & Business Media, December 2012. ISBN 978-1-4612-0171-7. Google-Books-ID: wR_oBwAAQBAJ.

Kivrak, H., Cakmak, F., Kose, H., and Yavuz, S. Waypoint based path planner for socially aware robot navigation. *Cluster Computing*, 25(3):1665–1675, June 2022. ISSN 1573-7543. doi: 10.1007/s10586-021-03479-x. URL https://doi.org/10.1007/s10586-021-03479-x.

Kloetzer, M. and Belta, C. A Fully Automated Framework for Control of Linear Systems from Temporal Logic Specifications. *IEEE Transactions on Automatic Control*, 53(1):287–297, February 2008. ISSN 1558-2523. doi: 10.1109/TAC.2007.914952. URL https://ieeexplore.ieee.org/abstract/document/4459804.

Knox, W. B. and Stone, P. Combining Manual Feedback with Subsequent MDP Reward Signals for Reinforcement Learning. 2010.

Knox, W. B. and Stone, P. Augmenting Reinforcement Learning with Human Feedback. 2011.

Knox, W. B., Stone, P., and Breazeal, C. Training a Robot via Human Feedback: A Case Study. In Herrmann, G., Pearson, M. J., Lenz, A., Bremner, P., Spiers, A., and Leonards, U. (eds.), *Social Robotics*, pp. 460–470, Cham, 2013. Springer International Publishing. ISBN 978-3-319-02675-6. doi: 10.1007/978-3-319-02675-6_46.

Li, X., Zhou, R., Lipton, Z. C., and Leqi, L. Personalized Language Modeling from Personalized Human Feedback, December 2024. URL http://arxiv.org/abs/2402.05133. arXiv:2402.05133 [cs].

Lindemann, L., Nowak, J., Schönbächler, L., Guo, M., Tumova, J., and Dimarogonas, D. V. Coupled Multi-Robot Systems Under Linear Temporal Logic and Signal Temporal Logic Tasks. *IEEE Transactions on Control Systems Technology*, 29(2):858–865, March 2021. ISSN 1558-0865. doi: 10.1109/TCST.2019.2955628. URL https://ieeexplore.ieee.org/abstract/document/8943945.

Loftin, R., Peng, B., MacGlashan, J., Littman, M. L., Taylor, M. E., Huang, J., and Roberts, D. L. Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning. *Autonomous Agents and Multi-Agent Systems*, 30(1):30–59, January 2016. ISSN 1573-7454. doi: 10.1007/s10458-015-9283-7. URL https://doi.org/10.1007/s10458-015-9283-7.

Matthews, J. N. S. and Morris, K. P. An Application of Bradley-Terry-Type Models to the Measurement of Pain. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 44(2):243–255, 1995. ISSN 1467-9876. doi: 10.2307/2986348. URL https://onlinelibrary.wiley.com/doi/abs/10.2307/2986348. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.2307/2986348.

Maystre, L. choix Documentation, 2022.

Meriçli, c., Veloso, M., and Akı n, H. L. Task Refinement for Autonomous Robots Using Complementary Corrective Human Feedback. *International Journal of Advanced Robotic Systems*, 8(2):16, June 2011. ISSN 1729-8806. doi: 10.5772/10575. URL https://doi.org/10.5772/10575. Publisher: SAGE Publications.

Muscettola, N., Morris, P., Pell, B., and Smith, B. Issues in temporal reasoning for autonomous control systems. In *Proceedings of the second international conference on Autonomous agents*, AGENTS '98, pp. 362–368, New York, NY, USA, May 1998. Association for Computing Machinery. ISBN 978-0-89791-983-8. doi: 10.1145/280765.280862. URL https://dl.acm.org/doi/10.1145/280765.280862.

Nunes, E. and Gini, M. Multi-Robot Auctions for Allocation of Tasks with Temporal Constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), February 2015. ISSN 2374-3468. doi: 10.1609/aaai.v29i1.9440. URL https://ojs.aaai.org/index.php/AAAI/article/view/9440. Number: 1.

Otero, N., Alissandrakis, A., Dautenhahn, K., Nehaniv, C., Syrdal, D. S., and Koay, K. L. Human to robot demonstrations of routine home tasks: exploring the role of the robot's feedback. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, HRI '08, pp. 177–184, New York, NY, USA, March 2008. Association for Computing Machinery. ISBN 978-1-60558-017-3. doi: 10.1145/1349822.1349846. URL https://dl.acm.org/doi/10.1145/1349822.1349846.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal Policy Optimization Algorithms, August 2017. URL http://arxiv.org/abs/1707.06347. arXiv:1707.06347 [cs].

Shu-Xi, W. The Improved Dijkstra's Shortest Path Algorithm and Its Application. *Procedia Engineering*, 29:1186–1190, January 2012. ISSN 1877-7058. doi: 10.1016/j.proeng.2012.01.110. URL https://www.sciencedirect.com/science/article/pii/S1877705812001208.

St. Clair, A. and Mataric, M. How Robot Verbal Feedback Can Improve Team Performance in Human-Robot Task Collaborations. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '15, pp. 213–220, New York, NY, USA, March 2015. Association for Computing Machinery. ISBN 978-1-4503-2883-8. doi: 10.1145/2696454.2696491. URL https://dl.acm.org/doi/10.1145/2696454.2696491.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning, second edition: An Introduction*. MIT Press, November 2018. ISBN 978-0-262-35270-3. Google-Books-ID: uWV0DwAAQBAJ.

Tajwar, F., Singh, A., Sharma, A., Rafailov, R., Schneider, J., Xie, T., Ermon, S., Finn, C., and Kumar, A. Preference Fine-Tuning of LLMs Should Leverage Suboptimal, On-Policy Data, June 2024. URL http://arxiv.org/abs/2404.14367. arXiv:2404.14367 [cs].

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Wang, G., Trimbach, C., Lee, J. K., Ho, M. K., and Littman, M. L. Teaching a Robot Tasks of Arbitrary Complexity via Human Feedback. In *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '20, pp. 649–657, New York, NY, USA, March 2020. Association for Computing Machinery. ISBN 978-1-4503-6746-2. doi: 10.1145/3319502.

3374824. URL https://dl.acm.org/doi/10.1145/3319502.3374824.

Wang, T. and Atanasov, N. WFA-IRL: Inverse Reinforcement Learning of Autonomous Behaviors Encoded as Weighted Finite Automata. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7429–7435, October 2022. doi: 10.1109/IROS47612.2022.9981874. URL https://ieeexplore.ieee.org/abstract/document/9981874. ISSN: 2153-0866.

Wirth, C., Akrour, R., Neumann, G., and Fürnkranz, J. A Survey of Preference-Based Reinforcement Learning Methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017. ISSN 1533-7928. URL http://jmlr.org/papers/v18/16-634.html.

Yang, Y., Bhatt, N. P., Ingebrand, T., Ward, W., Carr, S., Wang, Z., and Topcu, U. Fine-Tuning Language Models Using Formal Methods Feedback: A Use Case in Autonomous Systems. *Proceedings of Machine Learning and Systems*, 6:339–350, May 2024.

Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., and Zhang, Y. A survey on large language model (LLM) security and privacy: The Good, The Bad, and The Ugly. *High-Confidence Computing*, 4(2):100211, June 2024. ISSN 2667-2952. doi: 10.1016/j.hcc.2024.100211. URL https://www.sciencedirect.com/science/article/pii/S266729522400014X.