Aligning Agent Policies with Preferences: Human-Centered Interpretable Reinforcement Learning

Stephanie Milani[1], **Zhicheng Zhang**[1], **Nicholay Topin**[2], **Lirong Xia**[3], **Fei Fang**[1]
[1]Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213 USA
[2]Matchmatch, 251 Little Falls Drive, Wilmington, DE 19808 USA
[3]DIMACS and Rutgers University, 57 US Highway 1, New Brunswick, NJ 08901 USA
smilani@cs.cmu.edu, zczhang@cmu.edu, nicholay.topin@gmail.com, lirong.xia@rutgers.edu, feifang@cmu.edu

Abstract

An unaddressed challenge in interpretable reinforcement learning (RL) is to enable AI agents to integrate preference feedback into the policy generation process. Existing methods collect feedback only after training is complete, neglecting opportunities to *inform* the learning process. To address this gap, we propose a novel framework to align interpretable policies with human feedback during training. Our framework interleaves preference learning with an evolutionary algorithm, using updated preference estimates to guide the generation of better-aligned policies, and using newly-generated policies to query users to refine the preference model. Evolutionary algorithms enable the exploration of the full space of policies; however, it is intractable to maintain separate preference estimates—like win rates or utility values—for each individual policy in this infinite space. To handle this challenge, we propose to represent policies as feature vectors consisting of a finite set of meaningful attributes. For example, among a set of policies with similar performance, some may be more intuitive or more amenable to human intervention. To maximize the value of each user query, we employ a novel filtering technique to avoid presenting policies that are dominated in all dimensions, as repeated selections of clearly superior policies provides little information. We validate our method with experiments on synthetic preference data on two RL environments. We show that it produces RL policies that are not only better-aligned with user preferences but also more efficient in the number of user queries.

1 Introduction

AI agents are increasingly developed for high-stakes decision-making, like finance (Xiao et al. 2024) and education (Zhang et al. 2024). In these settings, AI agents will make or recommend a sequence of actions toward some overarching goal. These decisions are captured by a policy, which defines the agent's behavior across situations and contexts. A natural choice for training these policies is reinforcement learning (RL) (Sutton and Barto 2018), as it provides a principled framework to enable agents to optimize long-term objectives through trial and error. Achieving strong performance in complex settings typically requires representing policies with expressive function approximators (Mnih et al. 2013; Carroll et al. 2022).

While effective, these representations are often opaque, hindering our ability to understand and collaborate with these agents (Kenny, Tucker, and Shah 2023). Moreover, more widespread AI deployment means actions can impact human well-being at scale. In these contexts, regulatory and ethical frameworks (AI 2023) increasingly demand that AI agents be not only performant but also understandable and subject to meaningful human oversight. Interpretable models—such as decision trees (DTs) (McCallum 1996; Pyeatt, Howe et al. 2001; Gupta, Talvitie, and Bowling 2015) or rule lists—are frequently proposed as a way to meet these expectations by exposing the internal structure of the agent's policy. The key advantage of these structures is that they expose the reasoning process itself, in a way that is faithful to the true decision-making process of the agent.

However, interpretability is itself nebulous and context-specific. In practice, many desirable properties emerge from the overall structure of a policy, not from its individual components. Among a set of interpretable policies with similar performance, some may better support fairness, be easier to audit by being more concise, or offer clearer hooks for human intervention. At the same time, some of these properties are emergent characteristics of a whole policy, and policy subcomponents cannot be evaluated for these properties in isolation. For example, a limit on the number of nodes used by a policy cannot be factored into equivalent local limits. Likewise, fairness is generally measured in expectation, across all paths.

Standard RL approaches excel at incrementally improving local portions of a policy, but these approaches cannot account for preferences about an entire policy. To address this limitation, we introduce a framework that learns user preferences across high-level attributes and selects a policy that maximally fits these preferences. These attributes reflect key desiderata for these policies, which can be established from both regulatory guidance (e.g., the EU AI Act (Act 2024)) and human-centered design (e.g., the ease of tracing decisions for oversight and the clarity of policy outputs). This representation allows us to define user or institutional preferences as utility functions over the policy space, enabling explicit reasoning about tradeoffs and alignment. This compact representation also enables the direct usage of human feedback during *training* or test-time adaptation to develop better-aligned policies.

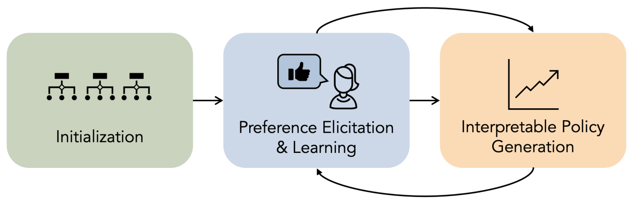

Initialization
Preference Elicitation & Learning
Interpretable Policy Generation

Figure 1: Overview of PASTEL, a novel algorithm for interpretable RL consisting of two main parts: i) preference elicitation and learning and ii) interpretable policy generation. In i), users provide feedback on interpretable models. During ii), the current preference estimate guides generation of models that better align with user preferences.

In this work, we propose a novel learning approach to support developing preference-aligned interpretable agents. Our approach, shown at a high level in Figure 1, explicitly learns underlying user preferences, and our policy generation step selects for policies that better fit user preferences. This in-the-loop policy training enables simultaneous refinement of preference estimates and the generation of new policies to align with those estimates. In both cases, our method centers on a shared insight: representing policies as feature vectors over deployment-relevant attributes enables the modeling of preferences over the space of policies, not just their outputs. This idea also enables us to construct an *interpretable* latent representation of the user preference, such that an end user or decision maker could audit, inspect, and reason about it.

Contributions. We make the following contributions. We propose *Preference Aligned Selection of Trees via Evolutionary Learning* (PASTEL), the first algorithm to leverage preference feedback for interpretable RL training. This algorithm consists of three main parts. First, we introduce a simple but effective evolutionary algorithm (EA) to guide training toward DT policies that are better tailored to user preferences. Second, we maintain a population of preference estimates to more informatively select the policies for eliciting user feedback. Third, we improve efficiency by pruning the set of candidate DT policies for querying by maintaining the Pareto frontier. Through experiments on two RL environments using synthetic but empirically justified preference data, we show that PASTEL yields policies that are better aligned with user preferences and is more sample efficient in the number of user queries, decreasing the burden on human users. By bridging the gap between training RL agents and evaluating their explanations, we believe our work opens new avenues for developing more interpretable, user-centered RL systems. In summary, we contribute:

- The first formal framework for preference-based interpretable RL,
- PASTEL, the first algorithm to leverage preference feedback to train interpretable RL policies, and
- An empirical evaluation that demonstrates PASTEL produces more preference-aligned policies compared with standard techniques.

2 Background

Reinforcement Learning. RL is a framework in which an agent learns to make decisions through interacting with an environment (Sutton and Barto 2018). An RL problem is typically modeled as a Markov decision process (Puterman 2014), which is a tuple $(\mathcal{S}, \mathcal{A}, T, r, \gamma)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, T is the transition probability function, r is the reward function, and γ is the discount factor. The agent aims to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected cumulative discounted reward. To maximize performance, policies often take the form of neural networks (Schulman et al. 2017).

Interpretable Policies. There exist various interpretable alternatives to neural networks, such as domain-specific programming languages (Verma et al. 2018), human-friendly prototypes (Kenny, Tucker, and Shah 2022), and simple symbolic rules (Ma et al. 2021). We focus on DT policies (Lipton 2018; Rudin 2019), which are widely used to model or explain policies in RL (Pyeatt, Howe et al. 2001; Milani et al. 2023; Chen et al. 2024; Tambwekar and Gombolay 2024).

Preference Elicitation. Pairwise comparisons are a common supervision signal in preference elicitation (Chen et al. 2013; Bengs et al. 2021). Rather than assigning absolute scores to items (Lai and Robbins 1985), users compare two alternatives i and j, expressing a preference $i \succ j$ if item i is preferred. The Bradley–Terry model (Bradley and Terry 1952) assigns each item i a latent utility score $u_i \in \mathbb{R}$, and models the probability of a pairwise preference as:

$$\mathbb{P}[i \succ j] = \frac{\exp(u_i)}{\exp(u_i) + \exp(u_j)}. \quad (1)$$

This formulation captures stochasticity in human preferences while remaining simple to estimate via maximum likelihood, as we will later see. Pairwise comparisons are also widely used in practice, particularly for applications where absolute utility is hard to elicit but relative judgments are easier and/or more reliable (Carterette 2011; Pelánek 2016; Ouyang et al. 2022).

3 Problem Statement

We first formulate the problem of incorporating user feedback into interpretable RL to identify the most preferred policy. We propose to model the problem as an online, iterative process in which a learner interacts with a user by eliciting pairwise comparisons between candidate policies. To capture practical constraints on the amount of feedback, the interaction proceeds over a fixed query budget, T.

Preference Elicitation Framework. At each iteration $t \in [T]$, the learner queries the user to obtain additional preference information. Standard bandit settings assume precise numerical reward specification; however, users may struggle to articulate this feedback, as it requires communicating exact values and tradeoffs. To address this issue, pairwise comparisons have emerged as a popular supervision signal (Bengs et al. 2021; Carterette 2011; Chen et al. 2013;

Christiano et al. 2017; Ouyang et al. 2022). Rather than assigning absolute scores (Lai and Robbins 1985), users compare two alternatives i and j, expressing a preference $i \succ j$ if item i is preferred. We adopt this form of feedback in our work. Specifically, the learner presents the user with a pair of policies $Q_t = (\hat{\pi}_t^i, \hat{\pi}_t^j)$ from a set $\hat{\Pi}_t$. The user gives stochastic feedback $O_t \in \{0, 1\}$, realized as o_t, with the probability of selection determined by their utility function v.

Preference Feedback Model. Human preferences are often context-dependent and noisy (Kahneman, Sibony, and Sunstein 2021; Otto et al. 2022). Rather than assuming users always choose the utility-maximizing option, we adopt a stochastic response model. We follow the standard Bradley-Terry model (Bradley and Terry 1952; Luce 2012) (Equation (1)) with a rationality coefficient (Laidlaw and Dragan 2022; Shah et al. 2016). Specifically, a user's stochastic response o to queries $Q = (\hat{\pi}^i, \hat{\pi}^j)$ is based on their underlying linear utility function v (Chu et al. 2011; Li et al. 2010; Saha 2021). Given two policies $\hat{\pi}_i, \hat{\pi}_j$ the probability that $\hat{\pi}^i$ is preferred to $\hat{\pi}^j$ is:

$$\mathbb{P}(\hat{\pi}^i \succ \hat{\pi}^j | \boldsymbol{\theta}) = \frac{1}{1 + \exp\left(\beta\Big(v\left(\hat{\pi}^j; \boldsymbol{\theta}\right) - v\left(\hat{\pi}^i; \boldsymbol{\theta}\right)\Big)\right)}, \tag{2}$$

where β is an inverse temperature parameter governing the stochasticity of user responses and $\boldsymbol{\theta} \in \mathbb{R}^d$ is a latent parameter vector representing user preferences.

Learning Objective. The learner aims to identify a high-utility policy using a bounded number of queries T. After T queries, the learner outputs a recommended policy $\hat{\pi}_T$. We define the objective as selecting a policy to maximize the user's true utility:

$$\max_{\hat{\pi} \in \hat{\Pi}} v(\hat{\pi}; \boldsymbol{\theta}), \tag{3}$$

where $\hat{\Pi}$ is the set of candidate policies. Since $v(\cdot; \boldsymbol{\theta})$ is not directly observable, the learner must infer which policy to recommend based only on the observed comparisons.

4 PASTEL

Equipped with our problem formulation, we now ask: how can we find interpretable policies that align with what people actually want? In nearly all existing preference learning settings, generating entirely new items (e.g., new movies or products) is infeasible, so the learner focuses on ranking existing options. In contrast, our setting allows for the generation of new policies during interaction.

Leveraging this insight, we propose PASTEL, an algorithm that interleaves preference learning and policy generation. In each iteration $t \in [T]$, the algorithm updates its estimate of the user's preferences based on pairwise comparisons, then uses this estimate to guide the generation of new interpretable policies. Because the flexibility of policy generation poses the challenge of efficiently searching what is effectively an infinite set, PASTEL estimates user preferences over a compact space rather than tracking performance for each policy individually. We detail PASTEL in Algorithm 1.

Algorithm 1: PASTEL

```
 1: $\{\hat{\boldsymbol{\theta}}_0^i\}_{i=1}^N \leftarrow$ INITIALIZEESTIMATES()
 2: $\pi^* \leftarrow$ TRAINRLEXPERT()
 3: $\Pi_0 \leftarrow$ GENERATEINITIALPOPULATION($\pi^*$)
 4: $\mathbf{f} \leftarrow \{\phi(\hat{\pi}, \mathcal{E}) \mid \hat{\pi} \in \Pi_0\}$
 5: for $t = 1, \ldots, T$ do
 6:     $\Pi_{t-1} \leftarrow$ PARETOFILTER($\Pi_{t-1}$)
 7:     $Q_{t-1} \leftarrow$ SELECTQUERY($\Pi_{t-1}, \{\hat{\boldsymbol{\theta}}_{t-1}^i\}_{i=1}^N$)
 8:     $o_{t-1} \leftarrow$ QUERYUSER($Q_{t-1}$)
 9:     $\{\hat{\boldsymbol{\theta}}_t^i\}_{i=1}^N \leftarrow$ UPDATE($\{\hat{\boldsymbol{\theta}}_{t-1}^i\}_{i=1}^N, Q_{t-1}, o_{t-1}$)
10:     for each $\hat{\boldsymbol{\theta}}_t^i$ do
11:         $\Pi_t^i \leftarrow$ RUNEVOLUTIONARYALGORITHM($\hat{\boldsymbol{\theta}}_t^i, \Pi_0$)
12:     end for
13:     $\Pi_t \leftarrow$ AGGREGATEPOPULATIONS($\{\Pi_t^i\}_{i=1}^N$)
14: end for
15: Calculate utility $\hat{v}_T$ under each $\hat{\boldsymbol{\theta}}_T$
16: return $\hat{\pi}^* \leftarrow \arg\max_{\pi \in \Pi_T} \hat{v}_T(\pi)$
```

Preference Modeling Over Interpretable Policies

The space of candidate policies is often large but reducible to a set of important factors. For example, two DTs might achieve similar performance but have different depths. Evaluating and estimating utility for each policy independently ignores these shared attributes. As a result, we propose to share information *across* policies by representing each policy with a feature vector that encodes key attributes. Now, we learn a preference model over *features* rather than discrete policies. This approach supports transparent reasoning about preferences. For example, if a loan approval policy has features for approval rate and fairness with learned weights $\hat{\boldsymbol{\theta}} = [0.3, 0.8]$, respectively, then the company prioritizes fairness over maximizing approvals. We now explain how we transform the policies and utility model to accommodate this new featurization.

Featurize Policy. Let $\hat{\Pi}$ denote the set of all possible policies of a specific form and $\mathcal{E}$ represent the set of all possible environmental or contextual information. Let $\phi : \hat{\Pi} \times \mathcal{E} \to \mathbb{R}^d$ be a function that encodes a policy $\hat{\pi}$ and the environment information into a d-dimensional feature vector $\mathbf{f}_{\hat{\pi}} \in \mathbb{R}^d$. Because users evaluate policies by examining observable properties, like the structure, we assume that $\phi(\cdot)$ captures all aspects of the policy that contribute to the user preference evaluation. In other words, we assume that the mapping is known and sufficient to describe preferences, such that,

$$\mathbf{f}_{\hat{\pi}} = \phi(\hat{\pi}, \mathcal{E}). \tag{4}$$

These features must be well-defined computations and may include structural attributes of the policy, performance metrics, and explanation characteristics.

Reparameterize Utility Model. We now handle the user's utility function. Observe that, given $\mathbf{f}_{\hat{\pi}}$ and $\hat{\boldsymbol{\theta}}$, we can preserve the original linearity assumption by simply rewriting $v(\hat{\pi}; \hat{\boldsymbol{\theta}})$ as a linear combination of the policy features:

$$v(\hat{\pi}; \boldsymbol{\theta}) = v(\phi(\hat{\pi}, \mathcal{E}); \boldsymbol{\theta}) = \mathbf{f}_{\hat{\pi}}^\top \hat{\boldsymbol{\theta}}. \tag{5}$$

An appropriate choice of $\phi(\cdot)$ can capture potentially non-linear and non-differentiable preferences over policies. Then, we derive the recommendation as:

$$\hat{\pi}_T = \text{argmax}_{\hat{\pi} \in \hat{\Pi}} \hat{v}_t(\hat{\pi}; \hat{\boldsymbol{\theta}}_T), \tag{6}$$

where $\hat{v}$ is the current utility estimate based on the estimated preference vector $\hat{\boldsymbol{\theta}}_T$ at iteration T and $\hat{\Pi}_T$ is the set of policies considered up to iteration T.

Preference Elicitation and Learning

Equipped with the ability to handle potentially infinite policies with our choice of representation, we now introduce our preference elicitation and learning process. Our approach combines a novel selection strategy to navigate the space of preferences with an update mechanism to move the preference estimates closer to the ground-truth preference. With our approach, we aim to converge on accurate preference estimates while handling stochasticity.

Maintain Ensemble of Preference Estimates. Inspired by the Query by Committee paradigm in active learning (Seung, Opper, and Sompolinsky 1992), we initialize a diverse ensemble of guesses $\{\hat{\boldsymbol{\theta}}^i_{t-1}\}_{i=1}^N$ about the true user preference to promote exploration (Algorithm 1, line 1). Each $\hat{\boldsymbol{\theta}}$ can be thought of as a hypothesis about the user preference. Accompanying each estimate is an EA, such that the output of the EA maximizes the current guess. During the policy selection step, each estimate $\hat{\boldsymbol{\theta}}^i$ contributes equally to the decision through a voting mechanism, breaking ties uniformly at random (Algorithm 1, line 16). Because these estimates are trained on the same data, they should eventually converge to similar policies. The idea is to use these estimates to not only evaluate the quality of the current policies but also guide the generation of new policies, as we will later discuss.

Filter Data by the Pareto Frontier. Given that users have limited time, we want each query to be informative. Consequently, we avoid presenting policies that are dominated in all dimensions, as repeated selections of clearly superior policies provide little information about how users trade off different feature dimensions. Instead, we propose filtering the policy set based on the Pareto frontier (Algorithm 1, line 6). Given a dataset D in a multi-dimensional space and a corresponding preference direction for each dimension, we identify the points that comprise the Pareto frontier:

$$\mathcal{P} = \{x \in D \mid \nexists y \in D, \text{ such that } y \text{ dominates } x\}, \tag{7}$$

where a point y dominates x if it is at least as good in all dimensions and strictly better in at least one dimension. PASTEL then chooses from this reduced set for preference elicitation. In many practical cases, the size of the Pareto frontier grows sublinearly with respect to n. For instance, in two-dimensional problems, it follows a logarithmic relationship (Bentley et al. 1978). Filtering reduces the complexity from $O(Tn^2)$ to $O(T \log^2 n)$, so we can explore the use of more computationally intensive techniques for identifying the most informative pairs for comparison.

Select Queries for Feedback Using Uncertainty. We introduce a novel approach to identify the most informative query, a pair of items $(\mathbf{f}_{\hat{\pi}_j}, \mathbf{f}_{\hat{\pi}_k})$ from a set F, given $\{\hat{\boldsymbol{\theta}}^i\}_{i=1}^N$. We seek to identify the most uncertain examples over the entire ensemble. Specifically, we compute hypothetical gradient updates for both possible outcomes for each parameter. As a result, for each potential pair and each $\hat{\boldsymbol{\theta}}_i$, the algorithm computes two hypothetical updates:

$$\hat{\boldsymbol{\theta}}_i^{(j)} = \hat{\boldsymbol{\theta}}_i - \eta \nabla \mathcal{L}(\mathbf{f}_{\hat{\pi}_j} \succ \mathbf{f}_{\hat{\pi}_k} | \hat{\boldsymbol{\theta}}_i) \text{ and}$$
$$\hat{\boldsymbol{\theta}}_i^{(k)} = \hat{\boldsymbol{\theta}}_i - \eta \nabla \mathcal{L}(\mathbf{f}_{\hat{\pi}_k} \succ \mathbf{f}_{\hat{\pi}_j} | \hat{\boldsymbol{\theta}}_i),$$

where η is the learning rate and $\nabla \mathcal{L}$ is the gradient of the logistic loss. We then compute the cosine similarity $\cos(\hat{\boldsymbol{\theta}}_j^{(i)}, \hat{\boldsymbol{\theta}}_j^{(k)})$ for each $\hat{\boldsymbol{\theta}}_j$ to measure the degree of alignment in the parameter space. Low cosine similarity means the two potential updates point in opposite directions, indicating more uncertainty about the outcome. Consequently, the algorithm selects the pair $(\mathbf{f}_{\hat{\pi}_j}, \mathbf{f}_{\hat{\pi}_k})$ that minimizes the average cosine similarity across all $\hat{\boldsymbol{\theta}}_i$:

$$(\mathbf{f}_{\hat{\pi}_j}, \mathbf{f}_{\hat{\pi}_k}) = \underset{(\mathbf{f}_{\hat{\pi}_j}, \mathbf{f}_{\hat{\pi}_k})}{\arg\min} \frac{1}{m} \sum_{i=1}^{m} \cos(\hat{\boldsymbol{\theta}}_i^{(j)}, \hat{\boldsymbol{\theta}}_i^{(k)}). \tag{8}$$

Preference Update. To learn preferences, PASTEL must query users, then leverage that feedback to update preference estimates. To update the estimates, we adopt a logistic regression model with stochastic gradient updates. After receiving (noisy) feedback o_t about Q_t from the user based on Equation (2), PASTEL updates each $\hat{\boldsymbol{\theta}}$ as:

$$\hat{\boldsymbol{\theta}}_{t+1} = \hat{\boldsymbol{\theta}}_t + \alpha \cdot \left(o_t - \sigma\left(\hat{\boldsymbol{\theta}}_t^\top (\mathbf{f}_{\hat{\pi}} - \mathbf{f}'_{\hat{\pi}})\right)\right) \cdot (\mathbf{f}_{\hat{\pi}} - \mathbf{f}'_{\hat{\pi}}), \tag{9}$$

where α is the learning rate, $\sigma(\cdot)$ is the logistic function, $o_t \in \{0, 1\}$ is the binary label derived from the user's feedback, and $\mathbf{f}_{\hat{\pi}} - \mathbf{f}'_{\hat{\pi}}$ is the difference between the feature vectors (Algorithm 1, line 9).

Interpretable Policy Generation

We now focus on generating new structures to align with those preferences. The goal is to create these structures such that they can be checked against (and possibly added to) the current Pareto frontier for further preference elicitation. Because many popular structures used in interpretable RL, such as DTs, are not differentiable, we require an algorithm that does not rely on gradients.[1] As a result, we introduce a novel module that leverages the power of evolutionary algorithms (EAs) to generate new DT policies. EAs (Bäck and Schwefel 1993) do not require the objective function to be differentiable. They work by evolving a population of solutions over generations to optimize a given objective function, called the fitness function. It is called in Algorithm 1, line 11, and additional details are in Algorithm 2.

[1]When the interpretable models or explanations can be updated using gradients, one can directly leverage work that uses preference feedback for RL (Christiano et al. 2017; Bai et al. 2022).

Algorithm 2: RUNEVOLUTIONARYALGORITHM

```
 1: Input: Preference vector $\hat{\boldsymbol{\theta}}_t^i$, initial population $\Pi_0$,
    crossover rate $\rho$
 2: Output: New population $\Pi_t^i$
 3: for $g = 0, \ldots, G-1$ do
 4:     Evaluate fitness of each policy in $\Pi_g$ using $\mathbf{f}_{\hat{\pi}}^\top \hat{\boldsymbol{\theta}}_t^i$
 5:     Initialize next population $\Pi_{g+1}^i \leftarrow \emptyset$
 6:     while $|\Pi_{g+1}^i| < |\Pi_g^i|$ do
 7:         $(p_1, \mathbf{f}_{p_1}), (p_2, \mathbf{f}_{p_2}) \leftarrow$ SelectParents($\Pi_g^i$)
 8:         Generate a random number $a \in [0, 1]$
 9:         if $a < \rho$ then
10:             $p_1, p_2 \leftarrow$ Crossover($p_1, p_2$)
11:         end if
12:         Mutate($p_1$), Mutate($p_2$)
13:         $\mathbf{f}_{p_1}, \mathbf{f}_{p_2} \leftarrow$ EXTRACTFEATUREVECTORS($p_1, p_2$)
14:         Add $(p_1, \mathbf{f}_{p_1}), (\mathbf{f}_{p_2}, p_2)$ to $\Pi_{g+1}^i$
15:     end while
16: end for
17: return $\Pi_t^i = \Pi_G$
```

Generate Policies with EA. In PASTEL, we call the EA for each $\{\hat{\boldsymbol{\theta}}_t^i\}_{i=1}^N$, such that the fitness function for the associated EA evaluates the quality of each individual in the population with $\hat{\boldsymbol{\theta}}^i$. Intuitively, this means that each EA aims to find solutions that maximize the current preference estimate. Each EA runs for G generations to iteratively modify the policy population. Within a generation g, each policy $\hat{\pi} \in \Pi_g$ is scored by the utility $\mathbf{f}_{\hat{\pi}}^\top \hat{\boldsymbol{\theta}}_t^i$ (Algorithm 2, line 2). The next population Π_{g+1} is first formed by tournament *selection*, retaining parents with probability proportional to fitness. Specifically, P random individuals from the population compete based on their fitness, and the best among them is chosen as a parent. Then, new policies are formed using *crossover* (recombining selected pairs to exploit complementary structure) and *mutation* (randomly perturbing offspring to sustain diversity). These mechanisms can be thought of as balancing exploitation of known high-quality policies with exploration of promising regions of the policy space. The procedure then outputs $\Pi_t^i = \Pi_G$, from which the highest-fitness policy can be extracted. The new policy is compared with the set of policies in the Pareto frontier and added to the set if it is non-dominated. The idea is to iteratively add more candidate policies to this set as our estimates of the user preferences improve given more user feedback.

Create New Policies with Mutation and Crossover. We now specify the two main processes for creating new policies: crossover and mutation. For crossover, we introduce a procedure called subtree swapping. Once two DTs have been selected for crossover, subtree swapping first randomly selects a node within the tree structure of the two parents, denoted as $n_{\text{cross}} \in N$. Then, the subtrees rooted at this node are exchanged, resulting in two new individuals. Mutation occurs over randomly-selected nodes in one of three ways, determined by tunable weights: random, subtree rebuilding, and subtree removal. Random mutation ($\mathcal{M}_{\text{rand}}$) selects a random feature and a corresponding split value (or selects a new random action) to create a new node n'. Subtree rebuilding ($\mathcal{M}_{\text{subtree}}$) first retrieves the subset of data $\mathcal{D}_{\text{path}} \in \mathcal{D}$ corresponding to a chosen node. The subtree is then reconstructed using imitation learning with $\mathcal{D}_{\text{path}}$. Subtree removal replaces the subtree rooted at the selected node with an action.

Expedite Run-Time. Because we envision this system running in real-time with real users providing preference feedback, we introduce a method to speed up one of the more computationally-expensive parts of our method: estimating the reward of a policy during feature extraction (Algorithm 2, line 13). With a large number of candidate policies to precisely evaluate at each round of the EA, this estimation step quickly becomes burdensome. As a result, we use a UCB-inspired bound (Slivkins et al. 2019) to adaptively pick policies for roll-out.

5 Experimental Setup

We evaluate the performance of PASTEL on two different RL environments. Because DTs are generally considered interpretable, we conduct a *functionally-grounded* evaluation (Doshi-Velez and Kim 2017), in which we define a specific set of features that would inform user preferences for interpretable models and simulate these preferences. A crucial advantage of simulated environments is we can evaluate how well our model is able to recover the preferences of the "ground truth" users, which provides insights about how our methods could perform with real users. We seek to answer the following questions:

RQ 1. Preference Alignment: Does PASTEL produce more preference-aligned trees? Does it do so in a query-efficient manner?

RQ 2. Human-AI Interaction: What benefits does PASTEL have in terms of user interactions?

RQ 3. Noise Robustness: How robust is PASTEL to preference noise?

RQ 4. Ablation Study: Which components of PASTEL contribute most to its performance?

Environments. We select CartPole-v1 and PotholeWorld-v0 as our test environments because a reward-optimal policy can be represented as a DT. CartPole-v1, a classic control problem, offers a domain with a low-dimensional state space (cart position, velocity, pole angle, and angular velocity) and binary actions (push left or right). In contrast, PotholeWorld-v0 (Topin et al. 2021) presents a more complex, driving-inspired scenario where an agent navigates lanes to avoid obstacles. We augment PotholeWorld-v0 with a controllable parameter governing lane reward trade-offs. The state is the current position, with three possible actions corresponding to lane choices. As a result, the agent must learn through experience to navigate around the potholes.

Features. For our feature vectors (computed as in Equation (4)), we test with $d = 4$ features for each environment. Table 1 shows the features used.

CartPole-v1			PotholeWorld-v0		
Feature	Type	Values	Feature	Type	Values
Reward	Performance	$[0, 500]$	Reward	Performance	$[-170, 49.95]$
Depth	Structural	$\{0, \ldots, 10\}$	Depth	Structural	$\{0, \ldots, 10\}$
Num. leaves	Structural	$\{1, \ldots, 1024\}$	Num. leaves	Structural	$\{1, \ldots, 1024\}$
Feature used	Explanation	$\{0, 1\}$	Action taken	Explanation	$\{0, 1\}$

Table 1: Features used in the preference vectors for the two environments. Each environment incudes performance, structural, and explanation-based attributes to highlight the diversity of attributes that can be accommodated by our method. Here, "Feature used" corresponds to the inclusion of a specific feature in the state space as a feature that is tested in the internal node of the DT.

Preference Generation. To generate the values for the user preference $\boldsymbol{\theta}$, we use a vector scaling technique (Ehrgott 2005). Specifically, we linearly combine the basis vectors, with each vector maximizing only one preference. This setup enables the exploration of explicit trade-offs among these preferences by adjusting the scaling factor for each vector entry α, such that $\sum_{i=1}^{d} |\boldsymbol{\theta}_i| = 1$. The scaling factor is 0.25, resulting in 35 different $\boldsymbol{\theta}$ values per environment. We run each $\boldsymbol{\theta}$ with 3 random seeds for each experiment.

Baselines. We compare with two baselines: VIPER (Bastani, Pu, and Solar-Lezama 2018) and Randomized Dueling Preference Selection (RDPS). VIPER is an interactive imitation learning algorithm in which an expert policy π^* provides corrections to a DT student. It does not incorporate preferences in its learning process, so the best DT policy is chosen based on reward only. The RDPS baseline is inspired by randomized dueling bandit algorithms used in preference-based learning. The algorithm maintains the current best policy. At each iteration t, the algorithm compares the current best with a randomly selected challenger. After eliciting pairwise preference feedback, the winner of the duel becomes the updated best explanation. This process is repeated for T queries. Note that this algorithm does not attempt to estimate the ground-truth preference model, which could make it more susceptible to preference noise.

Policy Set Initialization. To create the initial policy population, we use a modified version of VIPER (Bastani, Pu, and Solar-Lezama 2018). Specifically, to obtain a broader variety of tree policies, we introduce a depth randomization procedure (in which we choose a set of M maximum depths and call the VIPER algorithm under the maximum depth constraint). Rather than initializing the population only with the single best DT, we use all DTs generated over all iterations.

Performance Metrics. Our goal is to generate agent policies that align with preferences. Our main performance metric is the *alignment* score, which we compute using $\hat{v}t(\hat{\pi}; \boldsymbol{\theta})$. We normalize the alignment score using the practical maximum and minimum possible alignment scores of a policy of that structure. We obtain these values by running the EA for each $\boldsymbol{\theta}$ and its inverse, $-\boldsymbol{\theta}$, respectively. The normalized values are in $[0, 1]$.

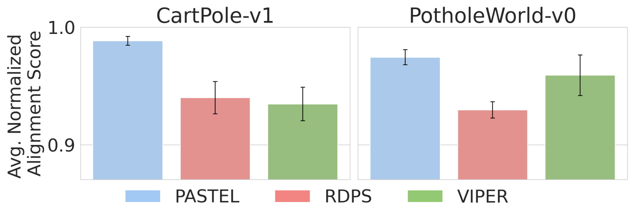


Figure 2: Comparison of normalized alignment scores. Error bars correspond to standard error. By explicitly incorporating preferences over policies and using those preferences to generate new policies, PASTEL creates more aligned interpretable policies than the baselines in both environments.

6 Results

RQ 1: Preference Alignment

To evaluate whether PASTEL produces better-aligned DT policies, we first examine performance under an *easy* noisy regime by setting $\beta = 10$ in Equation (2). We assess policy alignment quality at the maximum training horizon T ($T = 40$ for PotholeWorld-v0 and $T = 20$ for Cartpole-v1) then examine sample efficiency across various values of T. We then analyze the ability of PASTEL to recover the underlying user preference.

PASTEL produces better aligned policies. After a predetermined number of T queries, PASTEL produces substantially more preference-aligned trees than both VIPER and RDPS in both environments. As shown in Figure 2, this result indicates that personalizing policies to support end users in their goals and to align with their preferences is possible.

PASTEL more efficiently produces better-aligned policies. Because our goal is to not overburden a user, we aim to *efficiently* output highly-aligned policies. As a result, we evaluate the quality of the recommended policies with increasing T. Specifically, in Figure 3, we plot the quality over number of queries, starting from $T = 1$. PASTEL more rapidly outputs higher-quality policies. Interestingly, for PotholeWorld-v0, RDPS chooses strong initial guesses but struggles to improve with more queries. This result indicates that PASTEL can more effectively make use of the human feedback for policy selection and generation.

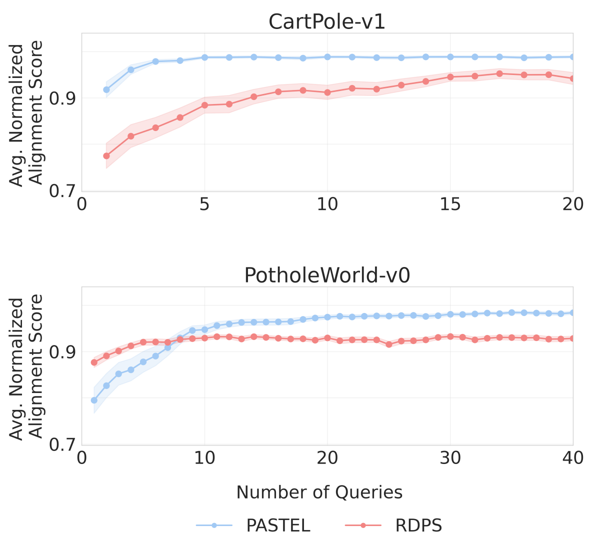


Figure 3: Normalized alignment scores with varying T. Error bars correspond to standard error. Compared with the baseline, PASTEL more effectively utilizes a small number of human preference queries to produce more aligned interpretable policies. Interestingly, for PotholeWorld-v0, RDPS happens to choose strong initial guesses, but it struggles to improve the alignment of the policy with more queries.

PASTEL generates more aligned policies because it iteratively refines its preference estimate. We now seek to understand how PASTEL generates more aligned policies. We calculate the cosine similarity between the ground-truth preference vector θ and the ensemble members. Specifically, over successive preference queries, we first sort all members by the cosine similarity, then average along that axis across all runs and preference vectors. Figure 4 shows that every curve climbs monotonically: a handful of early comparisons eliminate blatantly wrong weightings. After this rapid phase, the curves asymptote. This plateau is expected: once the algorithm learns a *sufficient* direction to rank the set, additional queries may offer diminishing discriminative power.

RQ 2: Human-AI Interaction

We now illustrate the benefits of a few of the modeling choices introduced by PASTEL. We first show the envisioned interactions that users could have with the entire system. We then show how a few design choices contribute to the comprehensibility of the system.

Modeling the factors that contribute to preference alignment enables users to better understand preference tradeoffs. One distinguishing factor of PASTEL is that we represent each policy as a compact subset of features in contrast to uninterpretable latent modeling approaches and approaches that bypass these factors to simply output a point estimate. For example, RDPS only outputs the current best policy, which does not enable the user to intuitively understand the contribution of each feature to the final policy. We visualize how a user can interact with the learned preference

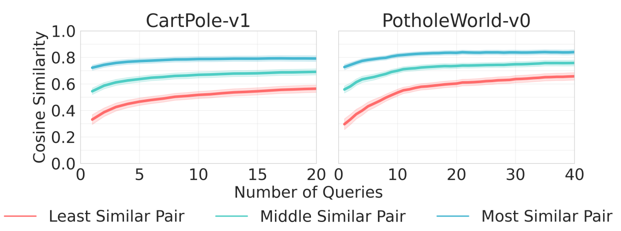


Figure 4: Similarity of each $\{\hat{\boldsymbol{\theta}}_t^i\}_{i=1}^N$ to the $\boldsymbol{\theta}$. We calculate the cosine similarity of the $N = 3$ learned preference estimates in the ensemble for each environment. We do so by first ordering them in terms of least to most similar according to cosine similarity, then averaging over runs within each similarity bin (least, middle, and most similar). This plot shows that, in both environments, PASTEL is indeed refining and improving its estimate of the true user preference.

models and policy features in Figure 5. The tables show that the user can investigate which $\hat{\boldsymbol{\theta}}$ voted for each $\hat{\pi}$, in addition to the overall winner. The line plots highlight that the user can decompose the assessment into pairwise comparisons over features to see the direction that the $\hat{\boldsymbol{\theta}}$s point in for each combination. The Pareto frontier filtering means that users would only have to inspect a small subset of alternatives, rather than hundreds or thousands.

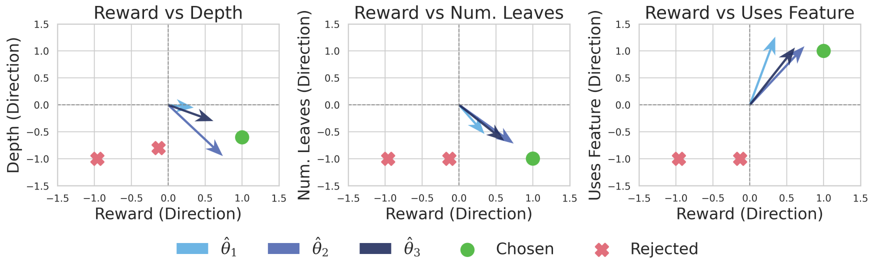
Implementation details enable the system to potentially be used in real time. Because we envision that this system could be used to elicit from downstream users, we prioritize efficiency in our approach. As a result, each EA call only takes around 1 minute of wall clock time due to algorithmic details and parallelization. Consequently, with well-spaced parallelized runs of the EA, each individual run could finish in less than 10 minutes. Although RDPS completes 1 run of 20 queries in around 1 minute, PASTEL requires around 5 minutes but produces much more aligned policies.

RQ 3: Robustness to Preference Noise

We investigate whether PASTEL is robust to preference noise in CartPole-v0 for three different noise regimes: easy, medium, and hard. We achieve these regimes by setting β in Equation (2) to values that result in progressively *more* probabilistic disparity between the best and the worst policies in the original set. We set $\beta = 10$, $\beta = 30$, and $\beta = 50$ for the easy, medium, and hard regimes, respectively.

PASTEL is more robust to preference noise. Figure 6 shows that PASTEL finds better preference-aligned trees in all three noise regimes. Note that VIPER does not learn preferences and therefore is resistant to noise (it will deterministically pick the reward-maximizing option). Unsurprisingly, this resistance is beneficial in cases where more preference weight is placed on reward. Compared to RDPS, PASTEL enjoys both higher preference alignment and lower variance.

$\hat{\pi}$	Reward	Depth	Num. Leaves	Uses Feature	Score $(\hat{\theta}_1)$	Score $(\hat{\theta}_2)$	Score $(\hat{\theta}_3)$
1	500	2	3	1	2.18	3.11	2.52
2	217	1	2	0	-0.73	0.29	-0.24
3	9	0	1	0	-1.01	-0.13	-0.69



$\hat{\pi}$	Reward	Depth	Num. Leaves	Takes Action	Score $(\hat{\theta}_1)$	Score $(\hat{\theta}_2)$	Score $(\hat{\theta}_3)$
1	24	4	15	1	0.55	0.57	0.55
2	23	4	11	1	0.56	0.58	0.56
3	19	2	4	1	0.99	0.87	0.99
4	8	2	3	1	0.99	0.83	0.99
5	5	0	1	0	2.48	2.57	2.48
6	24	5	7	1	0.35	0.44	0.35
7	5	1	2	1	1.2	0.97	1.20

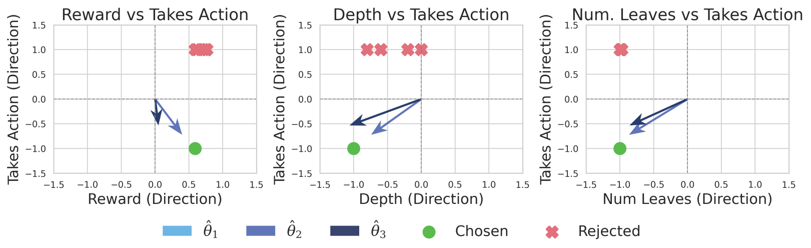


Figure 5: Example of how one can leverage and inspect the learned preferences and attributes from PASTEL in CartPole-v1 (top) and PotholeWorld-v0 (bottom). For each environment and policy $\hat{\pi}$ in the Pareto frontier, the top table shows the different values of the features (Reward, Depth, etc.) and the overall score according to each $\hat{\boldsymbol{\theta}}$. The overall $\hat{\pi}$ selected is in green and those not selected are in pink. The three plots at the bottom show the directions of the $\hat{\boldsymbol{\theta}}$ for different feature comparisons.

RQ 4: Ablation Study

Table 2 presents the results of our ablation study of the PASTEL algorithm. Overall, we find that different components of PASTEL have different, environment-dependent contributions to the overall performance.

Generating new policies most impacts the overall performance. We first discuss the effect of removing the EA component (PASTEL-DTEA). PASTEL-DTEA learns the underlying user preference by only querying from and recommending the policies in the initial set. We find that, compared with PASTEL, PASTEL-DTEA results in a notable drop in performance—especially in CartPole-v1, where the score decreases from 0.99 ± 0.00 to 0.89 ± 0.02. This result indicates that the additional ability to *generate* new policies based on preference estimates most substantially impacts the ability to produce better-aligned policies.

Interleaving preference learning and policy generation helps in some cases. The non-iterative variant (PASTEL-ITER) first exhausts the T preference queries, then runs the EA. In contrast, PASTEL interleaves preference learning and policy generation. As a result, this ablation tests whether generating new policies throughout the algorithm is more useful than first learning the preferences and then using the preference estimate to guide the EA. PASTEL-ITER exhibits a significant performance degradation in PotholeWorld-v0; in contrast, PASTEL-ITER maintains comparable performance in CartPole-v1. In CartPole-v1, the initial VIPER–extracted tree set is already sufficient for learning a reasonable preference estimate, so expanding the set of candidate policies throughout the learning process is not necessary for extracting additional preference information. However, in PotholeWorld-v0, the initial VIPER-extracted trees provide poor coverage of high-quality re-

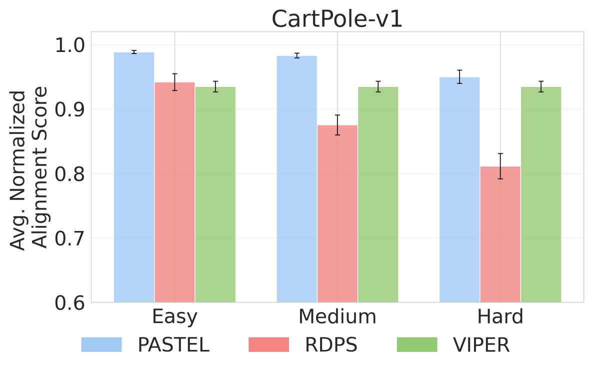


Figure 6: PASTEL creates and finds better aligned policies than the baselines in all three noise regimes. Error bars correspond to standard error. PASTEL is more robust to preference noise than the baselines.

Algorithm	**PotholeWorld-v0**	**CartPole-v1**
PASTEL	0.977 ± 0.004	0.987 ± 0.003
PASTEL-DTEA	0.935 ± 0.004	0.887 ± 0.020
PASTEL-ITER	0.716 ± 0.047	0.986 ± 0.003
PASTEL-PFF	0.967 ± 0.005	0.964 ± 0.008
PASTEL-CSS	0.971 ± 0.005	0.988 ± 0.003

Table 2: Ablation study: average normalized alignment scores for all PASTEL ablations. Error is standard error. For PotholeWorld-v0, all components of PASTEL contribute to its performance. However, CartPole-v1 is simpler, so the major benefit stems from the EA to generate more aligned policies (PASTEL vs. PASTEL-DTEA). Overall, the algorithmic components of PASTEL contribute to its performance.

gions; without interleaving, the system spends its entire feedback budget comparing sub-optimal policies, so the eventual EA search is guided by a potentially lower-quality preference model. By contrast, the alternating scheme continuously injects improved trees, steering both the preference model and subsequent evolutionary steps toward promising areas and yielding higher final utility.

Pareto frontier filtering provides modest benefit. We now discuss the impact of not filtering by the Pareto frontier (PASTEL-PFF). Compared with PASTEL, PASTEL-PFF has a slight alignment reduction in both environments, indicating that filtering contributes modestly but consistently to overall performance. We note that some of the benefit of this filtering is not only performance but also in reducing the number of policies that the human would inspect (as discussed in Section 6).

PASTEL is relatively robust to the choice of elicitation strategy. We now ablate the choice of elicitation strategy. Specifically, we compare the designed preference elicitation strategy used by PASTEL with random sampling (PASTEL-CSS). Interestingly, PASTEL-CSS leads to only a minor performance drop in both environments compared with PASTEL. This robustness is particularly encouraging given that many preference elicitation methods employ a "warm start" approach, beginning with random queries to broadly explore the preference space before transitioning to more targeted querying (Saha 2021). PASTEL could benefit from such hybrid approaches in future work.

7 Discussion

Although we study our approach with DT policies, our framework is general enough to accommodate policies of various forms and architectures (Kenny, Tucker, and Shah 2022; Verma et al. 2018; Ma et al. 2021; Hein, Udluft, and Runkler 2018). The key insight that interleaving preference elicitation with policy evaluation can improve alignment extends beyond any specific policy representation. Through our design choices—including focusing on the pairwise comparison setting, query efficiency, and algorithm speed—we hope our approach could be used for real-world tasks where users must quickly converge on a policy.

We also highlight how PASTEL could be incorporated into real-world domains. In medical imaging–based diagnosis, for example, a DT policy might branch on clinically relevant features (e.g., lesion size, texture), making its reasoning transparent to clinicians. Preferences could then encode properties valued by domain experts, such as shallow depth for ease of review, or the inclusion of high-value diagnostic features. In self-driving cars, a DT policy might branch on scene attributes (e.g., distance to nearest vehicle, pedestrian proximity) to determine driving actions. Human preferences could specify trade-offs over structural attributes, favoring trees that, for example, prioritize safety-related features early. In both cases, the ability to generate new candidate DTs during learning enables the system to explore novel combinations of features guided by these preferences.

8 Related Work

Interpretable Machine Learning. Interpretable machine learning (ML) has become increasingly important as AI agents are deployed for more complex and sensitive use cases (Prashanth et al. 2016; Chebotar et al. 2021). The need for interpretability arises from the desire to understand, appropriately trust, and effectively manage these systems. The most applicable line of work is interpretable RL. Although many structures have been proposed to use in lieu of neural networks for RL policies (Verma et al. 2018; Kenny, Tucker, and Shah 2022), we focus on DT policies due to the general consensus that they are human-understandable (Lipton 2018; Rudin 2019). However, their discrete structure is not immediately amenable to gradient-based training. Previous work applied gradient-based training by introducing nonlinearities in the splits (Silva et al. 2020), but the conversion of the so-called soft tree (Irsoy, Yıldız, and Alpaydın 2012) to a standard DT results in severe performance loss, making it inapplicable in our setting. Moreover, prior work trains DT policies only from the perspective of reward maximization (Bastani, Pu, and Solar-Lezama 2018; Topin et al. 2021). Instead, we view reward-based metrics as one class of properties, so our method aligns policies with both reward and policy-wide interpretability metrics.

Learning from Human Feedback. There is growing interest in the area of learning from human feedback (Griffith et al. 2013; Hadfield-Menell et al. 2016; Stiennon et al. 2020) due to the difficulty of specifying concrete objectives that capture the full extent of desired behavior. Early work collects preferences over agent trajectories to train a reward model r_θ, and the resulting cumulative reward $R(\tau;\theta) = \sum_t r_\theta(s_t, a_t)$ is used to model preference probabilities via the Bradley-Terry model (Christiano et al. 2017). Similar techniques are used for language models, where preferences over two *completions* y_i, y_j of queries x are used to train a reward model $R(x, y)$. The language model is then fine-tuned using RL (Ouyang et al. 2022) with the reward model. In contrast, we estimate a utility function over interpretable policies using policy-level features, and use it to actively guide the generation of new policies that align with user preferences. This utility function need not be decomposable into a sum over transition utilities.

Preference Elicitation. Preference elicitation and learning have been extensively studied across various disciplines (Chajewska, Koller, and Parr 2000; Conen and Sandholm 2001; Loepp, Hussein, and Ziegler 2014; Weernink et al. 2014; Li et al. 2023). The objective of most work in this area is to determine a full or partial ranking over alternatives by employing methods such as pairwise comparisons (Eric, Freitas, and Ghosh 2007; Lu and Boutilier 2011a; Branke et al. 2017) or asking individuals to rank their top choices among all or a subset of alternatives (Lu and Boutilier 2011b; Soufiani, Parkes, and Xia 2013; Drummond and Boutilier 2014; Zhao et al. 2018). Our work builds on these works, but it shifts the focus from eliciting rankings to learning the underlying preference model parameters to aid in *generating* preference-aligned interpretable policies. Our approach shares similarities with Bayesian Experimental Design (Chaloner and Verdinelli 1995; Rainforth et al. 2024), where the objective is to learn information about the underlying user preference. However, we integrate preference elicitation and learning with an emphasis on aligning models with the preferences of downstream users.

9 Limitations and Future Work

One limitation of our work is the assumption of static and linear user preferences. Although these are common assumptions, future work could explore techniques for modeling and adapting to evolving user preferences over time (Kahneman and Tversky 2013). Furthermore, interpretability is inherently subjective; it can vary significantly depending on the user's background, experiences, and domain knowledge (Zhou et al. 2021). Although we aim to capture *task-specific* factors that influence a user's judgment of a policy, future work could incorporate *user-specific* factors.

Another limitation is the use of synthetic data, which may not capture the diversity of real-world user perspectives. In future studies, incorporating a wider range of user input and adopting an *application-grounded* evaluation approach (Doshi-Velez and Kim 2017) could provide additional insights by leveraging real-world data to model the latent factors that influence user preferences. Involving end users would also help determine the best user interface and presentation of such policies before deployment.

10 Conclusion

We proposed to incorporate user feedback directly into the interpretable policy generation process. Through experiments in two environments, we demonstrated that our method not only produces DT policies that are better aligned with the underlying preferences but also does so efficiently, in 40 queries or fewer. Our work lays the groundwork for future research on developing more user-centered, interpretable RL agents that prioritize human alignment.

Acknowledgments

This work was supported in part by NSF grant IIS-2046640 (CAREER).

Ethical Statement

Though interpretable ML tools can provide insight into system behavior, they can also give a user the false impression of understanding. Likewise, a user may place more trust in a system simply because it is labeled "interpretable," even without interacting with the available oversight functionality. We ensure the user participates in our system by explicitly posing queries to them, and we emphasize the need to integrate our approach into a broader system. Relatedly, when asking a user for preferences, the system may fail: the user may misunderstand their role, the preference model may poorly represent the user's preferences, or the preferences may not be satisfied by the returned result. Toward mitigating these concerns, we take the following actions. First, we use a simple query form: "Which of these two options do you prefer?". Here, we aim to make the system as easy as possible for end users to interact with, while also enabling them maximal flexibility in providing pairwise feedback. Second, we directly expose learned preferences in a concise format (one scalar weight per property) so a user can evaluate whether their preferences were captured. Third, our system evaluates the final result using these preferences, showing which properties are present and to which degree.

This work did not involve any human subjects or data about users. All of the preference data is synthetic. We rely on synthetic user feedback both for ease of experimentation and due to ethical considerations. In particular, there is growing concern regarding researchers conducting user studies for their own sake, leading to end users being overburdened without clear benefit to the research goal. Because our work seeks to understand how one could integrate preferences into the generation of interpretable policies, relying on well-studied assumptions in the literature for the preference elicitation component, we opt not to engage with real users for the purpose of this work. We believe Responsible AI Development (RAI) includes responsibly *choosing* when it is appropriate to involve end users.

References

Act, E. A. I. 2024. The EU Artificial Intelligence Act.

AI, N. 2023. Artificial intelligence risk management framework (AI RMF 1.0). *URL: https://nvlpubs. nist. gov/nistpubs/ai/nist. ai*, 100–1.

Bäck, T.; and Schwefel, H.-P. 1993. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1): 1–23.

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

Bastani, O.; Pu, Y.; and Solar-Lezama, A. 2018. Verifiable reinforcement learning via policy extraction. *Advances in Neural Information Processing Systems*, 31.

Bengs, V.; Busa-Fekete, R.; El Mesaoudi-Paul, A.; and Hüllermeier, E. 2021. Preference-based online learning with dueling bandits: A survey. *Journal of Machine Learning Research*, 22(7): 1–108.

Bentley, J. L.; Kung, H.-T.; Schkolnick, M.; and Thompson, C. D. 1978. On the average number of maxima in a set of vectors and applications. *Journal of the ACM (JACM)*, 25(4): 536–543.

Bradley, R. A.; and Terry, M. E. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4): 324–345.

Branke, J.; Corrente, S.; Greco, S.; and Gutjahr, W. 2017. Efficient pairwise preference elicitation allowing for indifference. *Computers & Operations Research*, 88: 175–186.

Carroll, M.; Paradise, O.; Lin, J.; Georgescu, R.; Sun, M.; Bignell, D.; Milani, S.; Hofmann, K.; Hausknecht, M.; Dragan, A.; et al. 2022. Uni [mask]: Unified inference in sequential decision problems. *Advances in neural information processing systems*, 35: 35365–35378.

Carterette, B. 2011. System effectiveness, user models, and user utility: a conceptual framework for investigation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in information retrieval*, 903–912.

Chajewska, U.; Koller, D.; and Parr, R. 2000. Making rational decisions using adaptive utility elicitation. In *Aaai/Iaai*, 363–369.

Chaloner, K.; and Verdinelli, I. 1995. Bayesian experimental design: A review. *Statistical science*, 273–304.

Chebotar, Y.; Hausman, K.; Lu, Y.; Xiao, T.; Kalashnikov, D.; Varley, J.; Irpan, A.; Eysenbach, B.; Julian, R.; Finn, C.; et al. 2021. Actionable models: Unsupervised offline reinforcement learning of robotic skills. In *International Conference on Machine Learning*.

Chen, J.; Zhou, H.; Mei, Y.; Joe-Wong, C.; Adam, G. C.; Bastian, N.; and Lan, T. 2024. RGMDT: Return-Gap-Minimizing Decision Tree Extraction in Non-Euclidean Metric Space. *Advances in Neural Information Processing Systems*, 37: 18806–18847.

Chen, X.; Bennett, P. N.; Collins-Thompson, K.; and Horvitz, E. 2013. Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*, 193–202.

Christiano, P. F.; Leike, J.; Brown, T.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Chu, W.; Li, L.; Reyzin, L.; and Schapire, R. 2011. Contextual bandits with linear payoff functions. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 208–214.

Conen, W.; and Sandholm, T. 2001. Preference elicitation in combinatorial auctions. In *Proceedings of the 3rd ACM Conference on Electronic Commerce*, 256–259.

Doshi-Velez, F.; and Kim, B. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Drummond, J.; and Boutilier, C. 2014. Preference elicitation and interview minimization in stable matchings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Ehrgott, M. 2005. *Multicriteria optimization*, volume 491. Springer Science & Business Media.

Eric, B.; Freitas, N.; and Ghosh, A. 2007. Active preference learning with discrete choice data. *Advances in neural information processing systems*, 20.

Griffith, S.; Subramanian, K.; Scholz, J.; Isbell, C. L.; and Thomaz, A. L. 2013. Policy shaping: Integrating human feedback with reinforcement learning. *Advances in Neural Information Processing Systems*, 26.

Gupta, U. D.; Talvitie, E.; and Bowling, M. 2015. Policy tree: Adaptive representation for policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Hadfield-Menell, D.; Russell, S. J.; Abbeel, P.; and Dragan, A. 2016. Cooperative inverse reinforcement learning. *Advances in Neural Information Processing Systems*, 29.

Hein, D.; Udluft, S.; and Runkler, T. A. 2018. Interpretable policies for reinforcement learning by genetic programming. *Engineering Applications of Artificial Intelligence*, 76: 158–169.

Irsoy, O.; Yıldız, O. T.; and Alpaydın, E. 2012. Soft decision trees. In *Proceedings of the 21st International Conference on Pattern Recognition*, 1819–1822. IEEE.

Kahneman, D.; Sibony, O.; and Sunstein, C. R. 2021. *Noise: A flaw in human judgment*. Hachette UK.

Kahneman, D.; and Tversky, A. 2013. Prospect theory: An analysis of decision under risk. In *Handbook of the fundamentals of financial decision making: Part I*, 99–127. World Scientific.

Kenny, E. M.; Tucker, M.; and Shah, J. 2022. Towards interpretable deep reinforcement learning with human-friendly prototypes. In *The 11th International Conference on Learning Representations*.

Kenny, E. M.; Tucker, M.; and Shah, J. 2023. Towards interpretable deep reinforcement learning with human-friendly prototypes. In *The Eleventh International Conference on Learning Representations*.

Lai, T. L.; and Robbins, H. 1985. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1): 4–22.

Laidlaw, C.; and Dragan, A. 2022. The boltzmann policy distribution: Accounting for systematic suboptimality in human models. In *Proceedings of the International Conference on Learning Representations*.

Li, B. Z.; Tamkin, A.; Goodman, N.; and Andreas, J. 2023. Eliciting human preferences with language models. *arXiv preprint arXiv:2310.11589*.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 661–670.

Lipton, Z. C. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3): 31–57.

Loepp, B.; Hussein, T.; and Ziegler, J. 2014. Choice-based preference elicitation for collaborative filtering recommender systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 3085–3094.

Lu, T.; and Boutilier, C. 2011a. Robust approximation and incremental elicitation in voting protocols. In *IJCAI*, volume 1, 287–293.

Lu, T.; and Boutilier, C. 2011b. Vote elicitation with probabilistic preference models: Empirical estimation and cost tradeoffs. In *Algorithmic Decision Theory: Second International Conference, ADT 2011, Piscataway, NJ, USA, October 26-28, 2011. Proceedings 2*, 135–149. Springer.

Luce, R. D. 2012. *Individual choice behavior: A theoretical analysis*. Courier Corporation.

Ma, Z.; Zhuang, Y.; Weng, P.; Zhuo, H. H.; Li, D.; Liu, W.; and Hao, J. 2021. Learning symbolic rules for interpretable deep reinforcement learning. *arXiv preprint arXiv:2103.08228*.

McCallum, A. K. 1996. *Reinforcement learning with selective perception and hidden state*. University of Rochester.

Milani, S.; Zhang, Z.; Topin, N.; Shi, Z. R.; Kamhoua, C.; Papalexakis, E. E.; and Fang, F. 2023. MAVIPER: Learning Decision Tree Policies for Interpretable Multi-Agent Reinforcement Learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part IV*, 251–266. Springer.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Otto, A. R.; Devine, S.; Schulz, E.; Bornstein, A. M.; and Louie, K. 2022. Context-dependent choice and evaluation in real-world consumer behavior. *Scientific Reports*, 12(1): 17744.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Pelánek, R. 2016. Applications of the Elo rating system in adaptive educational systems. *Computers & Education*, 98: 169–179.

Prashanth, L.; Jie, C.; Fu, M.; Marcus, S.; and Szepesvári, C. 2016. Cumulative prospect theory meets reinforcement learning: Prediction and control. In *International Conference on Machine Learning*, 1406–1415. PMLR.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Pyeatt, L. D.; Howe, A. E.; et al. 2001. Decision tree function approximation in reinforcement learning. In *Proceedings of the 3rd International Symposium on Adaptive Systems: Evolutionary Computation and Probabilistic Graphical Models*, volume 2, 70–77.

Rainforth, T.; Foster, A.; Ivanova, D. R.; and Bickford Smith, F. 2024. Modern Bayesian experimental design. *Statistical Science*, 39(1): 100–114.

Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5): 206–215.

Saha, A. 2021. Optimal algorithms for stochastic contextual preference bandits. *Advances in Neural Information Processing Systems*, 34: 30050–30062.

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

Seung, H. S.; Opper, M.; and Sompolinsky, H. 1992. Query by committee. In *Proceedings of the Annual Workshop on Computational Learning Theory*, 287–294.

Shah, N. B.; Balakrishnan, S.; Bradley, J.; Parekh, A.; Ramch, K.; and Wainwright, M. J. 2016. Estimation from pairwise comparisons: Sharp minimax bounds with topology dependence. *Journal of Machine Learning Research*, 17(58): 1–47.

Silva, A.; Gombolay, M.; Killian, T.; Jimenez, I.; and Son, S.-H. 2020. Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, 1855–1865. PMLR.

Slivkins, A.; et al. 2019. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2): 1–286.

Soufiani, H. A.; Parkes, D. C.; and Xia, L. 2013. Preference elicitation for general random utility models. *arXiv preprint arXiv:1309.6864*.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33: 3008–3021.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tambwekar, P.; and Gombolay, M. 2024. Towards reconciling usability and usefulness of policy explanations for sequential decision-making systems. *Frontiers in Robotics and AI*, 11: 1375490.

Topin, N.; Milani, S.; Fang, F.; and Veloso, M. 2021. Iterative bounding MDPs: Learning interpretable policies via non-interpretable methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9923–9931.

Verma, A.; Murali, V.; Singh, R.; Kohli, P.; and Chaudhuri, S. 2018. Programmatically interpretable reinforcement learning. In *International Conference on Machine Learning*, 5045–5054. PMLR.

Weernink, M. G.; Janus, S. I.; Van Til, J. A.; Raisch, D. W.; Van Manen, J. G.; and IJzerman, M. J. 2014. A systematic review to identify the use of preference elicitation methods in healthcare decision making. *Pharmaceutical medicine*, 28: 175–185.

Xiao, Y.; Sun, E.; Luo, D.; and Wang, W. 2024. TradingAgents: Multi-Agents LLM Financial Trading Framework. *arXiv preprint arXiv:2412.20138*.

Zhang, Z.; Zhang-Li, D.; Yu, J.; Gong, L.; Zhou, J.; Hao, Z.; Jiang, J.; Cao, J.; Liu, H.; Liu, Z.; et al. 2024. Simulating classroom education with llm-empowered agents. *arXiv preprint arXiv:2406.19226*.

Zhao, Z.; Li, H.; Wang, J.; Kephart, J.; Mattei, N.; Su, H.; and Xia, L. 2018. A cost-effective framework for preference elicitation and aggregation. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.

Zhou, J.; Gandomi, A. H.; Chen, F.; and Holzinger, A. 2021. Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, 10(5): 593.