RULE-BOTTLENECK RL: LEARNING TO DECIDE AND EXPLAIN FOR SEQUENTIAL RESOURCE ALLOCATION VIA LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep Reinforcement Learning (RL) has demonstrated remarkable success in solving sequential resource allocation problems, but often suffers from limited explainability and adaptability—barriers to integration with human decision-makers. In contrast, LLM agents, powered by large language models (LLMs), provide human-understandable reasoning but may struggle with effective sequential decision making. To bridge this gap, we introduce Rule-Bottleneck RL (`RBRL`), **the first LLM agent framework for resource allocation problems that jointly optimizes language-based decision policy and explainability.** At each step within `RBRL`, an LLM first generates candidate rules—language statements capturing decision priorities tailored to the current state. RL then optimizes rule selection to maximize environmental rewards and explainability, with the LLM acting as a judge. Finally, the LLM chooses the action (optimal allocation) based on the rule. We provide conditions for `RBRL` performance guarantees as well as the finite-horizon evaluation gap of the learned `RBRL` policy. Furthermore, we provide evaluations in real-world scenarios, particularly in public health, showing that `RBRL` not only improves the performance of baseline LLM agents, but also approximates the performance of Deep RL while producing more desirable human-readable explanations. We conduct a human survey validating the improvement in the quality of the explanations.

1 INTRODUCTION

Sequential resource allocation is a fundamental problem in many domains, including healthcare, finance, public policy, and operations research (Considine et al., 2025; Boehmer et al., 2024; Yu et al., 2024; Balaji et al., 2019). This task involves allocating limited resources over time while accounting for dynamic changes and competing demands. Deep reinforcement learning (RL) is an effective method to optimize decision-making in resource allocation offering scalable high-reward policies (Yu et al., 2021; Talaat, 2022; Xiong et al., 2023), albeit generally providing action recommendations without human-readable reasoning and explanations. Such lack of interpretability poses a major challenge in critical high-stake domains where decisions must be transparent, justifiable, and in line with human decision-makers to ensure trust and compliance with ethical and regulatory standards.

For example, in healthcare settings, doctors may need to decide whether to prioritize intervention for Patient A or Patient B based on their current vital signs (Boehmer et al., 2024). An RL algorithm might suggest: *"Intervene with Patient A "* with the implicit goal of maximizing the value function. However, the underlying reasoning may not be clear to the doctors, leaving them uncertain about the factors influencing the decision (Milani et al., 2024). For doctors, a more effective suggestion could be risk-based with specific information, e.g., *"Patient A's vital signs are likely to deteriorate leading to higher potential risk compared to Patient B, so intervention with Patient A is prioritized"* (Gebrael et al., 2023; Boatin et al., 2021).

LLM agents (Sumers et al., 2024), on the other hand, leverage large language models (LLMs) for multi-step decision-making using reasoning techniques like chain of thought (CoT) (Wei et al., 2022). They enable natural language goal specification (Du et al., 2023) and enhance human understanding (Hu & Sadigh, 2023; Srivastava et al., 2024). However, agents based solely on LLM reasoning often

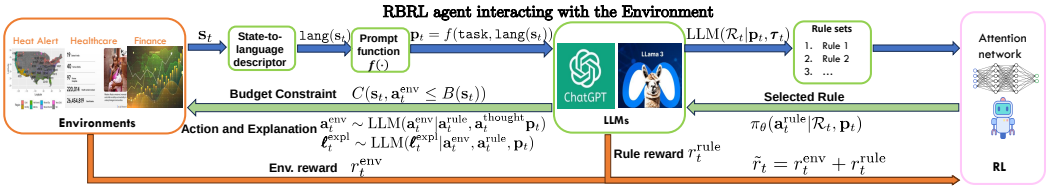


Figure 1: Overview of the framework of RBRL for joint sequential decision-making and explanation generation at time instance t. Starting with current state $\mathbf{s}_t$, a state-to-language descriptor generates $\texttt{lang}(\mathbf{s}_t)$, which is used to create the input prompt $\mathbf{p}_t$. The LLM processes $\mathbf{p}_t$ to produce a thought $\boldsymbol{\tau}_t$ and a set of candidate rules $\mathcal{R}_t$. An attention-based policy network selects a rule $\mathbf{a}_t^{\text{rule}}$ obeying the budget constraint $B(\mathbf{s}_t)$, which is used by LLM to derive an executable action $\mathbf{a}_t^{\text{env}}$ for the environment and a human-readable explanation $\boldsymbol{\ell}_t^{\text{expl}}$, while also providing a rule reward r_t^{rule} . The environment transitions to the next state $\mathbf{s}_{t+1}$, returning an environment reward r_t^{env} . This process is repeated iteratively at subsequent time steps.

struggle with complex sequential decision-making out of the box (Furuta et al., 2024), making RL a crucial tool for grounding to specific tasks (Carta et al., 2023; Tan et al., 2024; Wen et al., 2024; Zhai et al., 2024).

Consequently, aiming to combine the strengths of both deep RL and LLM agents, we pose the following question:

> *Can we design an LLM agent framework that can simultaneously perform sequential resource allocation and provide human-readable explanations?*

Similar to the celebrated index policy for prioritizing arms in resource allocation problems (Whittle, 1988), we explore the potential of using rules-based prioritization in resource allocation tasks. In the context of LLM agents, rules are defined as "structured statements" that capture prioritization among choices in a given state, aligning with the agent's goals (Srivastava et al., 2024). Building on this, we propose a novel LLM agent framework called Rule-Bottleneck Reinforcement Learning (RBRL), which integrates the strengths of LLM and RL to bridge the gap between decision-making and interpretability. RBRL provides an agent framework (as shown in Figure 1) that *simultaneously* makes sequential resource allocation decisions and provides human-readable explanations, in contrast to prior work that generates post-hoc explanations for a learned policy (Peng et al., 2022; Milani et al., 2024). RBRL leverages LLMs to generate candidate rules and employs RL to optimize policy, allowing the creation of effective decision policies while simultaneously providing human-understandable explanations. RBRL aims to increase efficiency and avoid the computational cost of directly fine-tuning LLM agents, which can be highly challenging in interactive environments due to the heavy computational costs and the complexity of token-level optimization (Rashid et al., 2024).

Our contributions are summarized as follows. *First*, LLMs are leveraged to generate a diverse set of rules according to the environment state, where each rule serves as a prioritization strategy for individuals in resource allocation, enhancing interpretability in decision-making. *Second*, we extend the conventional environmental state-action space by integrating the rules into states generated by LLMs, creating a novel framework that enables RL to operate on a richer, more interpretable decision structure. *Third*, we introduce an attention-based training framework that maps states and rules to queries and keys of a cross-attention network. The rule selection process is optimized by a policy network trained using the Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018), ensuring robust and efficient decision-making. In particular, the LLM also acts as a feedback mechanism, providing guidance during RL exploration to improve policy optimization and promote more effective learning. To the best of our knowledge, this is the first work to jointly optimize decision-making and explanation generation in constrained RL tasks.

We evaluate our method in environments from three real-world domains: HeatAlerts, where resources are allocated to mitigate extreme heat events; WearableDeviceAssignment, for distributing monitoring devices to patients; and BinPacking, which models allocating limited space in containers under constraints to optimize space utilization and minimize overflows. Using cost-effective LLMs such as gpt-4o-mini (OpenAI, 2024) and Llama 3.1 8B (Meta AI, 2024), we first assess decision performance by comparing RBRL with pure RL methods and language agent baselines.

Step 1: Generate Thoughts

Two example thoughts:
- There are only four warnings remaining in the budget.
- The current heat index is high, and issuing alert could raise public awareness.

Step 2: Generate Rules Based on Thoughts and the Current State

An example rule:
- Background : Maintaining a balance in warning issuance is crucial for future effectiveness
- Rule: If there are 3 or more warnings remaining, issue a warning when the heat index is above 105 F.
- State Relevance: There are 4 warnings remaining, allowing for proactive issuance given the current heat index of 107 F.

(a) Examples of generated rules for the Heat Alert Issuance task.

Example Language Wrapper for Heat Alert Issuance

Task: Assist policymakers in deciding when to issue public warnings to protect against heatwaves. Your goal is to minimize the long-term impact on health and mortality. Your decision should be based on the remaining budget, weather conditions, day of the week, past warning history, and remaining warnings for the season. The goal is to issue warnings when they are most effective, minimizing warning fatigue and optimizing for limited resources.

Action: A single integer value representing the decision: 1 = issue a warning, 0 = do not issue a warning. Warning can only be issued if the 'Remaining number of warnings/budget' is positive. Response in JSON format. For example: {'action': 1}.

State: Remaining warning budget: 4, - Current date and day of summer: 2008-07-10, - Current heat index: 107 F.

(b) Examples of language wrapper, containing task description, available actions and current state.

Figure 2: Examples of task prompts and generated rules for `HeatAlerts` domain.

We then evaluate explanation quality through a human survey conducted under IRB approval. The results demonstrate `RBRL`'s effectiveness in both decision quality and interpretability.

2 Related Work

Our work is positioned at the intersection of RL for resource allocation, LLM agents, and Explainable RL (XRL). While traditional RL methods effectively optimize rewards for resource allocation (Boehmer et al., 2024), they often lack the interpretability required for high-stakes domains. Conversely, LLM agents that provide reasoning (Wei et al., 2022) can struggle with sequential optimization. Our framework is novel compared to hierarchical approaches that use LLMs for high-level planning (Carta et al., 2023; Szot et al., 2023), as RBRL is the first to treat the natural-language rule as a primary output, jointly optimizing for both decision-making performance and the rule's quality as an explanation. Furthermore, unlike post-hoc or attribution-based XRL methods that analyze decisions after the fact (Guo et al., 2021; Chen et al., 2024), RBRL provides intrinsic explanations, as the rule is a functional component within the decision-making loop. A detailed discussion of related literature is provided in Appendix B.

3 Preliminary, Key Concepts, and Problem Formulation

3.1 Preliminary: Resource-Constrained Allocation

Resource-constrained allocation tasks are usually formulated as a special type of constrained Markov Decision Process, which is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, P, R, C, h, \gamma \rangle$, where $\mathcal{S}$ denotes a state space and $\mathcal{A}$ denotes a finite action space. The transition probability function, specifying the probability of transitioning to state $\mathbf{s}' \in \mathbb{R}^{d_1}$ after taking action $\mathbf{a} \in \mathbb{R}^{d_2}$ in state $\mathbf{s}$, is $P(\mathbf{s}'|\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \Delta(\mathcal{S})$, $R(\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ represents the reward function, defining the immediate reward received after taking action $\mathbf{a}$ in state $\mathbf{s}$, and we let $C(\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^{d_3}$ be the immediate cost incurred after taking action $\mathbf{a}$ in state $\mathbf{s}$. Often, each dimension $i \in [d_2]$ in $\mathbf{a}$ is either 0 or 1 in resource-constrained allocation tasks. In addition, h is the time horizon and $\gamma \in [0, 1]$ denotes the discount factor, which determines the present value of future rewards.

The goal is to find a policy $\pi \colon \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the expected cumulative discounted reward while satisfying the cost constraints with a budget function $B \colon \mathcal{S} \to \mathbb{R}^{d_3}$:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} J(\pi) := \left[\sum_{t=1}^{h} \gamma^{t-1} R(\mathbf{s}_t, \mathbf{a}_t) \right], \; s.t. \quad \forall t \in [h] \colon C(\mathbf{s}_t, \mathbf{a}_t) \leq B(\mathbf{s}_t). \tag{1}$$

3.2 KEY CONCEPTS FOR RULE-BASED LLM AGENTS

Our challenge is to design a rule-based LLM agent that jointly optimizes a language policy to both solve the optimization problem and improve explanation quality—a direction rarely explored. We next introduce the key concepts and terminologies underlying our main contribution.

LLM Agent For our LLM agent, the action space includes internal language actions $\tilde{\mathcal{A}} = \mathcal{A} \cup \mathcal{L}$ (Yao et al., 2023). The LLM agent has two types of internal language actions: First, *thoughts* $\mathbf{a}^{\text{thought}} \in \mathcal{L}$, are reasoning traces from the current problem state used to inform environment action selection $\mathbf{a}^{\text{env}} \in \mathcal{A}$. Second, *explanations* $\boldsymbol{\ell}^{\text{expl}}$, are generated from actions and thoughts to enhance human trust and interpretability (Zhang et al., 2023), a focus of this work.

Rules Thoughts are useful to highlight relevant aspects of a problem. However, they often lack detailed information to identify the next optimal action. In this work, we will consider "rules" $\mathbf{a}^{\text{rule}} \in \mathcal{L}$, which are structured language statements derived from thoughts that generally take the form "[if/when][do/prioritize] ". More formally, each rule $\mathbf{a}^{\text{rule}}$ consists of a triple $(\texttt{background},\ \texttt{rule\_statement},\ \texttt{state\_relevance})$. Figure 2a shows examples of generated rules from one of the domains used in the experiments.

Task and Constraints Description Language agents require: (1) a language description of the environment and the agent's goal, denoted $\texttt{task}$, containing the available actions for the task; (2) a function describing the state of the environment in natural language, denoted $\texttt{lang} : \mathcal{S} \to \mathcal{L}$. At each state $\mathbf{s}_t$, these descriptors are used to construct a natural language prompt $\mathbf{p}_t = f(\texttt{task}, \texttt{lang}(\mathbf{s}_t))$. Figure 2b exemplifies language wrapper generated for one of the environments in our experiments.

Rule-based Language Policy The objective is to jointly optimize the reward and explainability of the environment. Hence, we have an LLM agent-driven policy π_{LLM} for online interaction with the environment:

$$\begin{aligned} \mathbf{a}_t^{\text{thought}} &\sim \pi_{\text{LLM}}(\mathbf{a}_t^{\text{thought}} \mid \mathbf{p}_t),\ \ \mathbf{a}_t^{\text{rule}} \sim \pi_{\text{LLM}}(\mathbf{a}_t^{\text{rule}} \mid \mathbf{a}_t^{\text{thought}}, \mathbf{p}_t), \\ \mathbf{a}_t^{\text{env}} &\sim \pi_{\text{LLM}}(\mathbf{a}_t^{\text{env}} \mid \mathbf{a}_t^{\text{rule}}, \mathbf{a}_t^{\text{thought}}, \mathbf{p}_t),\ \ \boldsymbol{\ell}_t^{\text{expl}} \sim \pi_{\text{LLM}}(\boldsymbol{\ell}_t^{\text{expl}} \mid \mathbf{a}^{\text{env}}, \mathbf{a}_t^{\text{rule}}, \mathbf{p}_t). \end{aligned} \tag{2}$$

The rule acts as a "bottleneck" to the action and explanation. In the next section, we will introduce RBRL, which allows an RL-based learnable selection policy π_θ choosing among a set of dynamically generated candidate rules.

3.3 PROBLEM STATEMENT

We aim to increase the quality of $\boldsymbol{\ell}^{\text{expl}}$ while also optimizing decision-making by selecting rules that encourage both good quality explanations and high reward. To achieve this goal, we construct a surrogate explainabilty "rule reward" $R_{\text{LLM}}^{\text{rule}}(\mathbf{a}^{\text{rule}})$ using an LLM as judge (Shen et al., 2024; Bhattacharjee et al., 2024; Gu et al., 2024), which will be detailed in Section 4. Then, we propose the following augmented optimization objective under the joint environment/rule reward as $\tilde{R}(\mathbf{s}_t, \mathbf{a}_t^{\text{env}}) = R(\mathbf{s}_t, \mathbf{a}_t^{\text{env}}) + R_{\text{LLM}}^{\text{rule}}(\mathbf{a}_t^{\text{rule}})$:

$$\max_{\pi} \mathbb{E}_\pi \tilde{J}(\pi) := \left[\sum_{t=1}^{h} \gamma^{t-1} \tilde{r}_t \right],\ s.t.\ \text{constraint in (1)}, \tag{3}$$

where $\tilde{r}_t = \tilde{R}(\mathbf{s}_t, \mathbf{a}_t^{\text{env}})$. We emphasize that LLMs cannot fully replace the ultimate human assessment, but they they provide a scalable alternative during the optimization process.

4 RULE-BOTTLENECK REINFORCEMENT LEARNING (RBRL)

In this section, we propose RBRL, a novel LLM agent based on the key concepts in Section 3.2, which leverages the strengths of LLMs and RL to achieve both interpretability and robust sequential decision-making for (3), thereby achieving our goal of jointly optimizing policies and explanations for resource-constrained allocation in (1).

Algorithm Overview The framework of RBRL shown in Algorithm 1 involves four steps: (1) RULE SET GENERATION (line 3), where the LLM processes the state-task $\mathbf{p}_t$ to create candidate rules $\mathcal{R}_t$

Algorithm 1 RBRL

Require: Rule-selection policy π_θ; and Replay buffer $\mathcal{B}$.

1: **Initialization:** Initial state $\mathbf{s}_0$ and task-state prompt $\mathbf{p}_0$.
2: **for** $t = 0, \ldots,$ `max_iters` $- 1$ **do**
3: Generate rule candidates $\mathcal{R}_t$ using CoT from $\mathbf{p}_t$ and $\mathbf{a}_t^{\text{thought}}$. // Section 4.1
4: Select rule $\mathbf{a}_t^{\text{rule}}$ using the RL policy π_θ from $\mathcal{R}_t$ and $\mathbf{s}_t$. // Section 4.2
5: Generate the environment action $\mathbf{a}_t^{\text{env}}$ with the LLM from $\mathbf{a}_t^{\text{rule}}$, $\mathbf{p}_t$, and previous thoughts.
6: Apply the action in the environment and obtain new state $\mathbf{s}_{t+1}$ and environment reward r_t^{env}.
7: Generate explanation with the LLM from $\mathbf{a}_t^{\text{env}}$, $\mathbf{r}_t^{\text{rule}}$, $\mathbf{p}_t$, and previous thoughts.
8: Generate rule reward r_t^{rule} with the LLM as judge. // Section 4.3
9: Update the prompt $\mathbf{p}_{t+1}$ from $\mathbf{s}_{t+1}$, and the constraints C and budget B.
10: Append transition to the replay buffer $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}, \tilde{r}_t, \tilde{\mathbf{s}}_{t+1})\}$.
11: Sample from the replay buffer and update the policy network $\pi_\theta(\mathbf{a}_t^{\text{rule}}|\tilde{\mathbf{s}}_t)$. // Section 4.4
12: **end for**

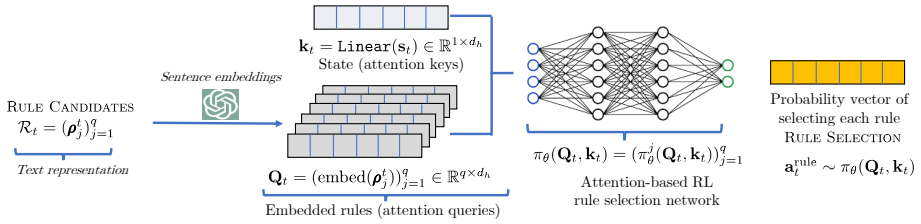


Figure 3: Overview of the Rule Selection step. The current state is encoded as a key vector, while candidate rules are encoded as Queries using a text embedding API (e.b., BERT sentence embedding). An attention-based policy network π_θ computes a probability distribution over the candidate rules, enabling the selection of the most suitable rule for decision-making and explanation.

for action selection; (2) Rule Selection (line 4), where an attention-based RL policy π_θ selects the best rule $\mathbf{a}_t^{\text{rule}} \in \mathcal{R}$; (3) Decision, Rule Reward and Explanation (lines 5-8), where the LLM generates an environment action $\mathbf{a}_t^{\text{env}}$ and based on the chosen rule $\mathbf{a}_t^{\text{rule}}$ gives a human-readable explanation $\boldsymbol{\ell}_t^{\text{expl}}$; (4) Reinforcement Learning (line 11), where it updates the policy π_θ based on collected data with standard RL algorithm Haarnoja et al. (2018) and the combined environment and rule reward $\tilde{r}_t$. Algorithm 1 details the entire process. Further sections elaborate on these steps.

4.1 Rule Set Generation

The rule generation process seeks to create interpretable and actionable guidelines for decision-making. Under this framework, a set of candidate rules $\mathcal{R}_t$ is generated according to $\mathcal{R}_t \sim \pi_{\text{LLM}}(\mathcal{R}_t|\mathbf{p}_t, \mathbf{a}_t^{\text{thought}})$. To enhance interpretability, each rule is accompanied by a rationale explaining the reasoning behind the decision. The LLM is instructed to generate rules as a JSON format, which is common for integration of LLMs with downstream applications (Shen et al., 2024). An example generated rule is given in Figure 2a. See Figure 12 in the Appendix for the prompt templates used rules generation.

4.2 Rule Selection

In this step, rules are converted from text to vector form, and a trainable attention-based policy network π_θ provides the probability distribution for sampling a rule. Figure 3 illustrates the process, with a detailed procedure in Algorithm 2 of the Appendix. Below are the major components of the architecture of π_θ. We propose to base the architecture on cross-attention layers (Bahdanau et al., 2015; Vaswani et al., 2017), with the state acting as the keys and values, and the rules as the queries. This allows to learn from the embedding representations of rules, and efficiently handle dynamically changing number of rules if needed.

State Representation The numeric state is projected by a linear layer: $\mathbf{k}_t = \texttt{Linear}(\mathbf{s}_t) \in \mathbb{R}^{1\times d_h}$, with d_h being to denote the architecture hidden dimension.

Rule Candidate Embedding Each rule in the list of rule candidates $\mathcal{R}_t = \{\boldsymbol{\rho}_1^t, \boldsymbol{\rho}_2^t, \ldots, \boldsymbol{\rho}_q^t\}$ is embedded into a numeric representation using a Sentence Embedding language model (e.g., SentenceBERT (Reimers & Gurevych, 2019)) and further processed by a projection layer similar to the state representation. This results in a *query* matrix $\mathbf{Q}_t \in \mathbb{R}^{q \times d_h}$.

Attention-based Policy Network $\boldsymbol{\pi_\theta}$ The vector $\mathbf{k}_t$, serving as keys, engages with the rule embeddings $\mathbf{Q}_t$, acting as queries, via a cross-attention mechanism to derive a hidden state representation $\mathbf{h}_t^{(1)} = \texttt{Attention}(\mathbf{Q}_t, \mathbf{k}_t^\top, \mathbf{k}_t^\top) \in \mathbb{R}^{q \times d_h}$, computed as $\texttt{Attention}(\mathbf{Q}_t, \mathbf{k}_t^\top, \mathbf{k}_t^\top) = \text{softmax}\left(\frac{\mathbf{Q}_t \mathbf{k}_t^\top}{\sqrt{d_h}}\right)\mathbf{k}_t^\top$, which facilitates the rule candidate vector embeddings in attending to the environment state. Subsequently, we sequentially apply self-attention layers to the hidden representation $\mathbf{h}^{(k+1)} = \texttt{Attention}(\mathbf{h}_t^{(k)}, \mathbf{h}_t^{(k)}, \mathbf{h}_t^{(k)})$, enabling the rule embeddings to attend to one another to rank an optimal candidate. Ultimately, following $K-1$ self-attention layers, a final linear layer converts the rule representations into logit vectors $\boldsymbol{\alpha}_\theta^t = \texttt{Linear}(\mathbf{h}_t^{(k)}) \in \mathbb{R}^q$ used for the computation of the probability of selecting each rule.

Rule Selection The policy distribution over the rules is calculated as: $\pi_{\theta,i}(\mathbf{Q}_t, \mathbf{k}_t) = \frac{\exp(\alpha_{\theta,i}^t(\mathbf{Q}_t, \mathbf{k}_t))}{\sum_{j=1}^q \exp(\alpha_{\theta,j}^t(\mathbf{Q}_t, \mathbf{k}_t))}, \quad i = 1, \ldots, q$. Therefore, a rule is selected at random from the distribution: $\mathbf{a}_t^{\text{rule}} \sim \texttt{Categorical}(\mathcal{R}; (\pi_{\theta,i}(\mathbf{Q}_t, \mathbf{k}_t))_{i=1}^q)$.

4.3 Decision, Rule Reward, and Explanation

Upon selection of rule $\mathbf{a}_t^{\text{rule}}$, the LLM determines the action to be applied within the environment $\mathbf{a}_t^{\text{env}} \sim \pi_{\text{LLM}}(\mathbf{a}_t^{\text{env}} | \mathbf{a}_t^{\text{rule}}, \mathbf{a}_t^{\text{thought}}, \mathbf{p}_t)$, ensuring concordance with the chosen strategy. Subsequently, the LLM formulates an explanation $\boldsymbol{\ell}_t^{\text{expl}} \sim \pi_{\text{LLM}}(\boldsymbol{\ell}_t^{\text{expl}} | \mathbf{a}^{\text{env}}, \mathbf{a}^{\text{rule}}, \mathbf{a}^{\text{thought}}, \mathbf{p}_t)$ contingent upon the rule. Figure 12 in the Appendix illustrates the prompt template employed to generate both the action and explanation.

This procedure concurrently produces the rule reward $R_{\text{LLM}}^{\text{rule}}(r_t^{\text{rule}})$, used for RL in the next step. This rewards is derived from using the LLM as a judge to answer the following three questions: $\texttt{ER}_1$. Without providing $\mathbf{a}_t^{\text{env}}$, is $\mathbf{a}_t^{\text{rule}}$ sufficient to predict the optimal action? $\texttt{ER}_2$. Does $\mathbf{a}_t^{\text{rule}}$ contain enough details about the applicability of the rule to current state? $\texttt{ER}_3$. Given $\mathbf{a}_t^{\text{env}}$, is $\mathbf{a}_t^{\text{rule}}$ compatible with this selection? Each question scores as 0 if negative or 1 if positive. The rule reward is calculated as $r_t^{\text{rule}} = R_{\text{LLM}}^{\text{rule}}(\mathbf{a}_t^{\text{rule}}) \propto (1/3)\sum_i \texttt{ER}_i$. Refer to Figure 12 in the Appendix for the full prompt.

4.4 Policy Update through RL

Augmented state space Traditional RL frameworks fail to directly return a policy based on current environment state due to intermediate steps: generating the rule set $\mathcal{R}_t$, mapping rules $\mathbf{a}_t^{\text{rule}}$ to actions $\mathbf{a}_t^{\text{env}}$ in an LLM-driven environment. RBRL addresses this issue by creating an augmented state $\tilde{\mathbf{s}}_t := (\mathbf{s}_t, \mathcal{R}_t)$ with transition dynamics $P(\tilde{\mathbf{s}}_{t+1} | \tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}})$, integrating rules into the state space for reasoning over both the environment's dynamics and decision rules $\mathbf{a}_t^{\text{rule}}$. The following theorem explains the transition computation.

Theorem 4.1. *The state transition of the RBRL MDP can be calculated as*

$$P(\tilde{\mathbf{s}}_{t+1} | \tilde{\mathbf{s}}_t, \mathbf{a}_t^{rule}) = P(\mathcal{R}_{t+1} | \mathbf{s}_{t+1}) \times \int_{\mathbf{a}} P(\mathbf{s}_{t+1} | \mathbf{a}^{env}, \mathbf{s}_t) \cdot P(\mathbf{a}^{env} | \mathbf{a}_t^{rule}, \mathbf{s}_t) d\mathbf{a}^{env}, \tag{4}$$

where $P(\mathcal{R}_{t+1} | \mathbf{s}_{t+1}) = \pi_{LLM}(\mathcal{R}_{t+1} | \mathbf{p}_t, \boldsymbol{\tau}_t)$ is the probability of the LLM generating rule set $\mathcal{R}_{t+1}$ provided the state $\mathbf{s}_{t+1}$, $P(\mathbf{s}_{t+1} | \mathbf{a}^{env}, \mathbf{s}_t)$ is the original environment dynamics, and $P(\mathbf{a}^{env} | \mathbf{a}_t^{rule}, \mathbf{s}_t) = \pi_{LLM}(\mathbf{a}^{env} | \mathbf{p}_t, \mathbf{a}_t^{rule})$ is the probability of the LLM selecting the environment action $\mathbf{a}^{env}$.

Policy update step The attention-based policy network in Section 4.2 is optimized using the standard SAC algorithm, which balances reward maximization with exploration. The policy network in SAC is updated by minimizing the KL divergence between the policy and the Boltzmann distribution induced by Q networks $Q_{\phi_i}, \forall i = 1, 2$, which is expressed as

$$L_\pi(\theta) = \mathbb{E}_{\mathcal{D}}\Big[\beta \log \pi_\theta(\mathbf{a}_t^{\text{rule}} | \tilde{\mathbf{s}}_t) - \min_{i=1,2} Q_{\phi_i}(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}})\Big], \tag{5}$$

where β is a temperature parameter. The detailed implementation for SAC update procedure is detailed in Algorithm 3 in Appendix D.

5 PERFORMANCE GUARANTEE

In this section, we derive and prove conditions under which RBRL can learn the optimal task policy, as well as characterize the potential trade-off between explainability and task performance when rewarding rules for higher explainability.

Proposition 5.1 (Rule Set Coverage)**.** *Let $\mathcal{A}$ be a finite action space and $Q^*(\mathbf{s}, \mathbf{a}^{env})$ the optimal state-action value function, with $\mathbf{a}^{env,*}(\mathbf{s}) := \arg\max_{\mathbf{a}^{env} \in \mathcal{A}} Q^*(\mathbf{s}, \mathbf{a}^{env})$ denoting the optimal action at state $\mathbf{s}$. Given state $\mathbf{s}$, an LLM samples N rules independently from a conditional distribution $\pi_{LLM}(\cdot \mid \mathbf{s})$, and each rule $\boldsymbol{\rho}_i$ maps $\mathbf{s}$ to an action $\mathbf{a}_i^{env} \sim \pi_{LLM}(\mathbf{a}_i^{env} | \boldsymbol{\rho}_i, \mathbf{s})$. Assume there exists $\delta > 0$ and $\eta_s \in (0, 1]$ such that: $\mathbb{P}_{\boldsymbol{\rho}_i \sim \pi_{LLM}(\cdot|\mathbf{s})} \left[Q^*(\mathbf{s}, \mathbf{a}_i^{env}) \geq Q^*(\mathbf{s}, \mathbf{a}^{env,*}(\mathbf{s})) - \delta \right] \geq \eta_s$. Define the δ-optimal rule set as: $\mathcal{R}^\delta(\mathbf{s}) := \{\boldsymbol{\rho}_i : Q^*(\mathbf{s}, \mathbf{a}_i^{env}) \geq Q^*(\mathbf{s}, \mathbf{a}^{env,*}(\mathbf{s})) - \delta\}$. Then with high probability over the sampled rules, there at least has a rule $\boldsymbol{\rho}_i$ and the induced action $\mathbf{a}_i^{env} \sim \pi_{LLM}(\mathbf{a}_i^{env} | \boldsymbol{\rho}_i, \mathbf{s})$ that satisfies:*

$$\mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{env,*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{env})\right] \leq \delta + \epsilon_{worst} \cdot (1 - \eta_s)^N, \tag{6}$$

where $\epsilon_{worst} := \max_{\boldsymbol{\rho} \notin \mathcal{R}^\delta(\mathbf{s})} \left(Q^*(\mathbf{s}, \mathbf{a}^{env,*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{env})\right)$ is the worst-case loss outside the δ-optimal set.

Remark 5.2. Proposition 5.1 states the *rule diversity* property in the rule candidate set such that the best possible action (when $\delta \to 0$) is included is guaranteed with high probability when number of rules N goes large. This is crucial in guaranteeing that RBRL can learn a near-optimal policy with high probability (with optimality when $\delta = 0$ and $\eta_s = 1$). See Section E in Appendix for more detail. We also numerically evaluate rule-coverage in Figure 8 of Appendix G.5.

Define the T-step value function $V_{\mathcal{M}'}^{\pi,T}(\mathbf{s}_0) = [\sum_{t=0}^{T-1} \gamma^t R_t^{\mathcal{M}'}(\mathbf{s}_t, \pi(\mathbf{s}_t))|\mathbf{s}_0]$, where $R^{\mathcal{M}'}$ is the reward function in $\mathcal{M}'$. We will denote the original MDP as $\mathcal{M}$ and use $\tilde{\mathcal{M}}$ to denote the MDP for the RBRL agent with transition function as in Theorem 4.1 and reward $\tilde{R}$. We have the following theorem.

Theorem 5.3. *The evaluation gap $Gap(T, s_0) := V_{\mathcal{M}}^{\pi^*,T}(\mathbf{s}_0) - V_{\mathcal{M}}^{\pi_{RBRL},T}(\mathbf{s}_0)$ of RBRL is bounded as*

$$Gap(T, s_0) = V_{\mathcal{M}}^{\pi^*,T}(\mathbf{s}_0) - V_{\tilde{\mathcal{M}}}^{\pi_{RBRL},T}(\mathbf{s}_0) + V_{\tilde{\mathcal{M}}}^{\pi_{RBRL},T}(\mathbf{s}_0) - V_{\mathcal{M}}^{\pi_{RBRL},T}(\mathbf{s}_0) \leq \lambda \cdot \frac{1 - \gamma^T}{1 - \gamma}, \tag{7}$$

where λ is a constant depending on the magnitude of the rule reward, and, with a slight notational abuse, $V_{\mathcal{M}}^{\pi_{RBRL},T}$ is the value of the RBRL policy when seen as a policy in the original MDP mapping states to actions (i.e., by integrating out the rule generation and action selection via LLMs.)

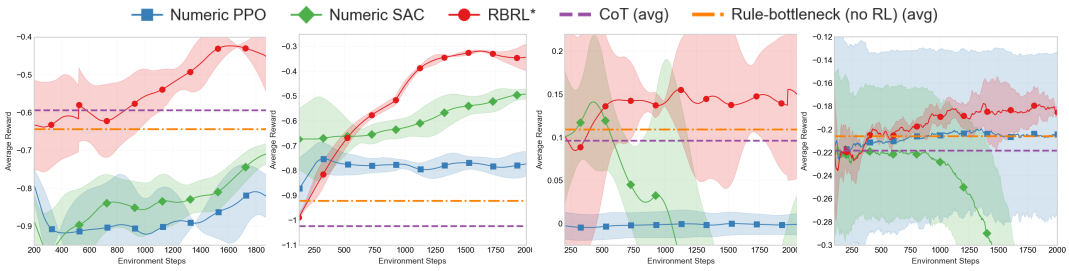
Remark 5.4. This analysis focuses on the evaluation gap between the optimal policy π^* under the original MDP $\mathcal{M}$ and the policy $\pi_{\texttt{RBRL}}$, captures the suboptimality of using $\pi_{\texttt{RBRL}}$ instead of the true optimal policy π^*, assuming RBRL is optimized under the extended MDP $\tilde{\mathcal{M}}$ (with same transitions as $\mathcal{M}$ but additional rule-based reward). It can be decomposed into two interpretable terms. The first part captures the optimism of using $\pi_{\texttt{RBRL}}$ under the extended MDP $\hat{\mathcal{M}}$ rather than the original MDP, which is non-positive. The second part quantifies the accumulated reward difference induced by the additional explanation rewards when using the same RBRL policy in both MDPs.

6 EXPERIMENTS & HUMAN SURVEY

In this section, we evaluate RBRL and empirically show that it can achieve a joint improvement in both reward and explainability over comparable baselines. We briefly summarize these environments here, with additional details in Appendix F.1.

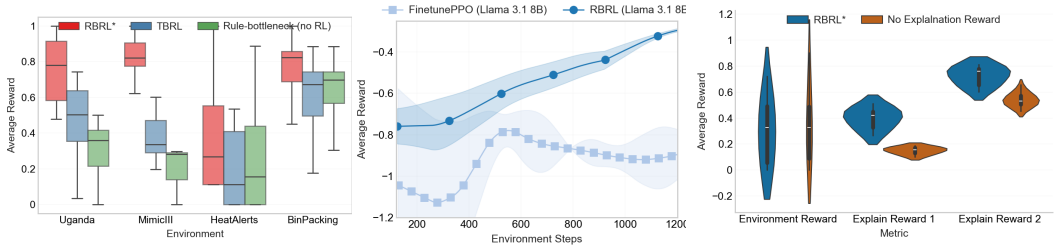
Domains We evaluate RBRL in three main distinct resource-constrained allocation domains:

▷ `WearableDeviceAssignment`: We use two environments, `Uganda` and `MimicIII`, from the vital sign monitoring domain introduced by Boehmer et al. (2024), modeling the allocation of limited wireless devices among postpartum mothers as an MDP setting. ▷ `HeatAlerts`: We use the



(a) `Uganda` (b) `MimicIII` (c) `HeatAlerts` (d) `BinPacking`

Figure 4: Results from Q1 using ChatGPT 4o-mini. The plots show the mean and standard error across three seeds, using exponentially weighted moving averages ($\lambda_{\text{ema}} = 200$).



(a) Q2: ablations (b) Q3: LLM finetuning (c) Q4: no rule rewards

Figure 5: Additional experiments and ablations. (a) Comparison of `RBRL` with thoughts-based RL (TBRL) and the baseline rule-based LLM without RL training; (b) comparison against LLM finetuning with PPO at the token level from the environment reward with CoT generation for the `Mimic`; (c) shows the effect of removing the rule reward in the `HeatAlerts` environments. For (a) and (c), we show distribution of rewards in the last 20% training steps.

`weather2alert` environment from Considine et al. (2025) , which formulates issuing heat alerts as a constrained MDP to reduce hospitalization risk from the alert. ▷ `BinPacking` We adopt the online stochastic `BinPacking`: environment introduced by Balaji et al. (2019), which Sequentially places arriving items into bins with fixed capacity to minimize total waste, following the online stochastic formulation. Detailed domain description can be found in Appendix F.1.

6.1 Environment Reward Optimization

We discuss the main results and refer to Appendix F for the detailed experiment setup and Appendix G for additional experiments. Unless otherwise specified, we use `gpt-4o-mini` as LLM due to its reasonable cost and high performance.

Q1. Did `RBRL` optimize the reward function? `RBRL` is compared to CoT (Wei et al., 2022) for language reasoning and PPO (Schulman et al., 2017) for numeric states. Figure 4 indicates `RBRL` outperforms CoT, showing RL-optimized rule selection improves decision-making. `RBRL` also exceeds PPO in all environments with equal environment steps, suggesting a better online learning performance. Notice that `RBRL` is compatible with a baseline LLM trained for advanced reasoning techniques (e..g, GRPO Shao et al. (2024)). However, GRPO or similar cannot be used directly in MDPs. Nevertheless, our experiments with the comparable GPT o3 (see Appendix G) prove that RBRL can also help improve reasoning models in our tasks.

Q2. Did structured rules help optimization? We conduct two ablation studies on structured rules. First, we benchmark the use of structured rules without RL, called baseline `Rule-bottleneck(no RL)`, which is shown in Equation (2)-(5). Next, we compare `RBRL` with a variant optimizing unstructured thoughts, termed thoughts-based RL (`TBRL`). The implementation mimics `RBRL`, utilizing a candidate pool $\mathcal{P}$ with the CoT prompt. Results in Figure 5a show that comparing `RBRL` with `RulesLLMOnly` highlights RL training gains, suggesting rule generation alone does not explain `RBRL`'s performance. Additionally, significant improvements over `TBRL` suggest optimizing structured rules is more effective than optimizing free reasoning traces.

Q3. How does RBRL compare to token-level LLM finetuning with RL? We implement LLM finetuning on a Llama 3.1 8B model, termed FinetunePPO. A value head is trained on final hidden states, with KL divergence from a reference model as regularization (Ziegler et al., 2019). CoT is generated, followed by an action query, optimizing both. Training runs for 18 hours on 3 seeds using an A100 40G GPU (1200 steps/seed). For fair comparison, RBRL is also run on Llama 3.1 8B. Figure 5b shows RBRL outperforms the flatter trend of finetuning, indicating better online learning. Moreover, RBRL runs on a regular laptop, whereas FinetunePPO requires specialized hardware and takes 4× longer per step. Due to compute limits, results are shown only for the less noisy MimicII domain.

Additional comparison with XRL benchmarks We further compare RBRL against a representative XRL method that also targets joint optimization and intrinsic interpretability: Decision Diffusion Trees (DDTs) (Silva et al., 2020). As shown in Table 1, RBRL is consistently competitive and often outperforms the tree-based baseline across most domains, particularly in the early stages of training, underscoring its sample efficiency. Although DDT achieves a higher average reward than RBRL in HeatAlerts, it exhibits substantially higher variance, highlighting the greater stability of RBRL.

6.2 Human Survey and Explainability

Q4. Did RBRL increase the explainability of explanations? A survey with 40 participants was conducted to assess explanation quality, detailed in Appendix J. Each prompt included the task, state, and action space as originally given to the LLM, followed by actions and explanations from the CoT agent and the RBRL agent, without disclosing agent types. Participants were asked to choose preference for explanation A, B, or none. Prompts were split between WearableDeviceAssignment and HeatAlerts domains. Figure 6 shows results, favoring RBRL's explanations in both domains, with a detailed breakdown in J. An additional experiment with an LLM judge using a large gpt-4o model showed strong agreement with humans, preferring RBRL's explanations in all cases.

Table 1: XRL Baselines Results Table

Dataset (@steps)	RBRL	SAC	PPO	DDT	DDT w/rules
Uganda (@500)	$\mathbf{-0.56 \pm 0.18}$	-0.83 ± 0.14	-0.91 ± 0.14	-1.01 ± 0.20	-1.20 ± 0.31
Uganda (@2500)	$\mathbf{-0.60 \pm 0.20}$	-0.75 ± 0.14	-0.74 ± 0.05	-1.28 ± 0.35	-1.20 ± 0.30
MimicIII (@500)	$\mathbf{-0.36 \pm 0.05}$	-0.61 ± 0.11	-0.78 ± 0.05	-0.92 ± 0.10	-1.02 ± 0.10
MimicIII (@2500)	$\mathbf{-0.39 \pm 0.07}$	-0.43 ± 0.10	-0.64 ± 0.10	-0.97 ± 0.11	-0.99 ± 0.13
HeatAlerts (@500)	0.14 ± 0.11	-0.04 ± 0.33	0.00 ± 0.01	$\mathbf{0.22 \pm 0.25}$	0.15 ± 0.29
HeatAlerts (@2500)	0.13 ± 0.14	0.05 ± 0.04	0.00 ± 0.01	$\mathbf{0.38 \pm 0.57}$	0.38 ± 0.56
BinPacking (@500)	$\mathbf{-0.03 \pm 0.00}$	-0.03 ± 0.00	-0.03 ± 0.00	-0.19 ± 0.03	-0.19 ± 0.04
BinPacking (@2500)	$\mathbf{-0.03 \pm 0.00}$	-0.06 ± 0.00	-0.03 ± 0.00	-0.21 ± 0.02	-0.21 ± 0.02

Discussion on Explainability The trustworthiness of explanations is a core challenge in XAI. Following recent work (Kunz & Kuhlmann, 2024; Parcalabescu & Frank, 2023; Jacovi & Goldberg, 2020), we highlight three concepts: *Plausibility:* whether an explanation is convincing to humans (validated via our survey, Figure 6). *Consistency:* whether the stated reason logically entails the action (Appendix G.3). *Faithfulness:* whether the explanation reflects the true decision mechanism.

Our work is motivated by the gap between plausibility and faithfulness in post-hoc methods. By design, RBRL ensures consistency: explanations factually follow the State → Rule → Action pipeline, where the rule is the verifiable cause of the action. As shown in Table 8 in Appendix G.3, including reasoning traces does not affect the decision, confirming that consistency stems from the state and rule rather than LLM self-explanation. While our experiments validate consistency, establishing faithfulness—verifying the LLM's internal reasoning for rule generation—remains an open challenge.

Q5. What was the effect of the rule reward? During training of RBRL, rules received rewards from two prompts. We examine an ablation without this reward. Figure 5c illustrates results for the HeatAlerts environment, noted for high variance and a challenging reward function. We extended training to 5k steps to understand these dynamics. Without rule reward, environment reward remains steady (slightly increasing), but explainability scores drop significantly. Refer to Section 4.3 for the definition of the rule reward metrics. A decline in metric 1 indicates that rules are less predictive of the optimal actions. A decline in metric 2 suggests rules lack detailed applicability to the current problem state, indicating more generic rather than specialized rule selection. Metric 3 (not shown) was always 1 in all steps, indicating the limitations of directly evaluating post hoc explanations. Although judged by the LLM, these results are encouraging, as our previous experiment showed alignment between the LLM and human assessments.

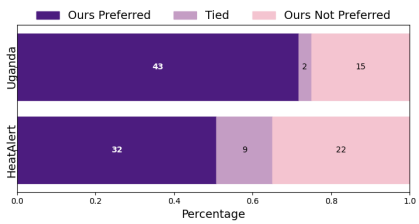


Figure 6: Results from the human survey.

Ethics and Reproducibility Statements

The authors of this work adhere to the ICLR Code of Ethics. Our research involves domains with significant ethical considerations, particularly in healthcare and public policy, which we have carefully addressed. Our work includes a human survey to evaluate the quality of explanations, which was conducted under Institutional Review Board (IRB) approval.

We have taken extensive measures to ensure the reproducibility of our research. The complete source code for our framework, including environment implementations and experiment scripts, is provided as supplementary material, with an anonymous link included at the beginning of the Appendix. All algorithmic details and hyperparameters for our proposed method (RBRL) and all baselines are comprehensively documented in Appendix D and Appendix F.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Bharathan Balaji, Jordan Bell-Masterson, Enes Bilgin, Andreas Damianou, Pablo Moreno Garcia, Arpit Jain, Runfei Luo, Alvaro Maggiar, Balakrishnan Narayanaswamy, and Chun Ye. Orl: Reinforcement learning benchmarks for online stochastic optimization problems. *arXiv preprint arXiv:1911.10641*, 2019.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. Towards llm-guided causal explainability for black-box text classifiers. In *AAAI 2024 Workshop on Responsible Language Models*, 2024.

Adeline A Boatin, Joseph Ngonzi, Blair J Wylie, Henry M Lugobe, Lisa M Bebell, Godfrey Mugyenyi, Sudi Mohamed, Kenia Martinez, Nicholas Musinguzi, Christina Psaros, et al. Wireless versus routine physiologic monitoring after cesarean delivery to reduce maternal morbidity and mortality in a resource-limited setting: protocol of type 2 hybrid effectiveness-implementation study. *BMC Pregnancy and Childbirth*, 21:1–12, 2021.

Niclas Boehmer, Yunfan Zhao, Guojun Xiong, Paula Rodriguez-Diaz, Paola Del Cueto Cibrian, Joseph Ngonzi, Adeline Boatin, and Milind Tambe. Optimizing vital sign monitoring in resource-constrained maternal care: An rl-based restless bandit approach. *arXiv preprint arXiv:2410.08377*, 2024.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *ICML*, 2023.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. Selfie: Self-interpretation of large language model embeddings. *arXiv preprint arXiv:2403.10949*, 2024.

Zelei Cheng, Xian Wu, Jiahao Yu, Sabrina Yang, Gang Wang, and Xinyu Xing. Rice: Breaking through the training bottlenecks of reinforcement learning with explanation. *arXiv preprint arXiv:2405.03064*, 2024.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. Babyai: A platform to study the sample efficiency of grounded language learning. *arXiv preprint arXiv:1810.08272*, 2018.

Cédric Colas, Tristan Karch, Clément Moulin-Frier, and Pierre-Yves Oudeyer. Language and culture internalization for human-like autotelic ai. *Nature Machine Intelligence*, 4(12):1068–1076, 2022.

Ellen M Considine, Rachel C Nethery, Gregory A Wellenius, Francesca Dominici, and Mauricio Tec. Optimizing heat alert issuance with reinforcement learning. In *AAAI*, 2025.

CSL. Senate bill no. 896: Generative artificial intelligence accountability act. https://legiscan.com/CA/text/SB896/id/3023382, 2024. Accessed: 2025-02-06.

Devleena Das, Sonia Chernova, and Been Kim. State2explanation: Concept-based explanations to benefit agent learning and user understanding. *NeurIPS*, 2023.

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

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. In *ICML*, pp. 8657–8677, 2023.

Kelly N DuBois. Deep medicine: How artificial intelligence can make healthcare human again. *Perspectives on Science and Christian Faith*, 71(3):199–201, 2019.

EPC. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence. http://data.europa.eu/eli/reg/2024/1689/oj, 2024. Accessed: 2025-02-06.

Hiroki Furuta, Yutaka Matsuo, Aleksandra Faust, and Izzeddin Gur. Exposing limitations of language model agents in sequential-task compositions on the web. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

Georges Gebrael, Kamal Kant Sahu, Beverly Chigarira, Nishita Tripathi, Vinay Mathew Thomas, Nicolas Sayegh, Benjamin L Maughan, Neeraj Agarwal, Umang Swami, and Haoran Li. Enhancing triage efficiency and accuracy in emergency rooms for patients with metastatic prostate cancer: a retrospective analysis of artificial intelligence-assisted triage using chatgpt 4.0. *Cancers*, 15(14): 3717, 2023.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

Wenbo Guo, Xian Wu, Usmann Khan, and Xinyu Xing. Edge: Explaining deep reinforcement learning policies. *Advances in Neural Information Processing Systems*, 34:12222–12236, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1856–1865. PMLR, 2018.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Hengyuan Hu and Dorsa Sadigh. Language instructed reinforcement learning for human-ai coordination. In *ICML*, 2023.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and Jo ao G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL `http://jmlr.org/papers/v23/21-1342.html`.

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? *arXiv preprint arXiv:2004.03685*, 2020.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

Jenny Kunz and Marco Kuhlmann. Properties and challenges of llm-generated explanations. *arXiv preprint arXiv:2402.10532*, 2024.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. Towards faithful model explanation in nlp: A survey. *Computational Linguistics*, 50(2):657–723, 2024.

Meta AI. Introducing llama 3.1: Our most capable models to date. *AI at Meta*, 2024. URL https://ai.meta.com/blog/meta-llama-3-1/.

Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. Explainable reinforcement learning: A survey and comparative review. *ACM Computing Surveys*, 56(7):1–36, 2024.

OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. *OpenAI Blog*, 2024.

Letitia Parcalabescu and Anette Frank. On measuring faithfulness or self-consistency of natural language explanations. *arXiv preprint arXiv:2311.07466*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Xiangyu Peng, Mark Riedl, and Prithviraj Ammanabrolu. Inherently explainable reinforcement learning in natural language. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *NeurIPS*, 2022.

Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based explainable artificial intelligence: A survey. *arXiv preprint arXiv:2312.12936*, 2023.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*, 2024.

Ahmad Rashid, Ruotian Wu, Julia Grosse, Agustinus Kristiadi, and Pascal Poupart. A critical look at tokenwise reward-guided text generation. *arXiv preprint arXiv:2406.07780*, 2024.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on EMNLP-IJCNLP*, pp. 3980–3990. Association for Computational Linguistics, 2019.

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

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yi Shen, Benjamin McClosky, and Michael Zavlanos. Multi-agent reinforcement learning for resource allocation in large-scale robotic warehouse sortation centers. 2023.

Yiran Shen, Aditya Emmanuel Arokiaraj John, and Brandon Fain. Explainable rewards in RLHF using LLM-as-a-judge, 2024. URL https://openreview.net/forum?id=FaOeBrlPst.

Daria Shevtsova, Anam Ahmed, Iris WA Boot, Carmen Sanges, Michael Hudecek, John JL Jacobs, Simon Hort, Hubertus JM Vrijhoef, et al. Trust in and acceptance of artificial intelligence applications in medicine: Mixed methods study. *JMIR Human Factors*, 11(1):e47031, 2024.

Andrew Silva, Matthew Gombolay, Taylor Killian, Ivan Jimenez, and Sung-Hyun Son. Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In *International conference on artificial intelligence and statistics*, pp. 1855–1865. PMLR, 2020.

Sean R. Sinclair, Felipe Vieira Frujeri, Ching-An Cheng, Luke Marshall, Hugo Barbalho, Jingling Li, Jennifer Neville, Ishai Menache, and Adith Swaminathan. Hindsight learning for mdps with exogenous inputs. In *ICML*, 2023.

Chandan Singh, Jeevana Priya Inala, Michel Galley, Rich Caruana, and Jianfeng Gao. Rethinking interpretability in the era of large language models. *arXiv preprint arXiv:2402.01761*, 2024.

Megha Srivastava, Cédric Colas, Dorsa Sadigh, and Jacob Andreas. Policy learning with a language bottleneck. In *RLC Workshop on Training Agents with Foundation Models*, 2024.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas Griffiths. Cognitive architectures for language agents. *TMLR*, 2024. URL https://openreview.net/forum?id=1i6ZCvflQJ.

Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *ICLR*, 2023.

Fatma M Talaat. Effective deep q-networks (edqn) strategy for resource allocation based on optimized reinforcement learning algorithm. *Multimedia Tools and Applications*, 81(28):39945–39961, 2022.

Weihao Tan, Wentao Zhang, Shanqi Liu, Longtao Zheng, Xinrun Wang, and Bo An. True knowledge comes from practice: Aligning large language models with embodied environments via reinforcement learning. In *ICLR*, 2024.

Mark Towers, Ariel Kwiatkowski, Jordan K Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024. URL https://arxiv.org/abs/2407.17032.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Kukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Alexander Sasha Vezhnevets, John P Agapiou, Avia Aharon, Ron Ziv, Jayd Matyas, Edgar A Duéñez-Guzmán, William A Cunningham, Simon Osindero, Danny Karmon, and Joel Z Leibo. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*, 2023.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, Yitao Liang, and Team CraftJarvis. Describe, explain, plan and select: interactive planning with large language models enables open-world multi-task agents. In *NeurIPS*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Muning Wen, Ziyu Wan, Jun Wang, Weinan Zhang, and Ying Wen. Reinforcing LLM agents via policy optimization with action decomposition. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, 25(A):287–298, 1988.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, et al. Agentgym: Evolving large language model-based agents across diverse environments. *arXiv preprint arXiv:2406.04151*, 2024.

Guojun Xiong, Xudong Qin, Bin Li, Rahul Singh, and Jian Li. Index-aware reinforcement learning for adaptive video streaming at the wireless edge. In *Proceedings of the Twenty-Third International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, pp. 81–90, 2022.

Guojun Xiong, Shufan Wang, Gang Yan, and Jian Li. Reinforcement learning for dynamic dimensioning of cloud caches: A restless bandit approach. *IEEE/ACM Transactions on Networking*, 31 (5):2147–2161, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

Yangyang Yu, Zhiyuan Yao, Haohang Li, Zhiyang Deng, Yupeng Cao, Zhi Chen, Jordan W Suchow, Rong Liu, Zhenyu Cui, Zhaozhuo Xu, et al. Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making. *arXiv preprint arXiv:2407.06567*, 2024.

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, et al. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *arXiv preprint arXiv:2405.10292*, 2024.

Xijia Zhang, Yue Guo, Simon Stepputtis, Katia Sycara, and Joseph Campbell. Understanding your agent: Leveraging large language models for behavior explanation. *arXiv preprint arXiv:2311.18062*, 2023.

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

Supplementary Materials

In this supplementary material, we provide additional details omitted from the main paper for brevity. We also present extended experimental results to further support our findings. The complete source code for this work is available at: `https://anonymous.4open.science/r/rule-bottleneck-reinforcement-learning-6F0D`.

A LLM USAGE DISCLOSURE

We used LLMs solely as algorithmic components within our proposed method. No LLMs were employed for writing, editing, or generating the content of this manuscript.

B RELATED WORK

Our work intersects with three distinct areas within the RL literature. We discuss related work in each of these domains.

RL for Sequential Resource Allocation RL has been widely studied for constrained resource allocation across domains. In maternal health, Boehmer et al. (2024) apply RL to a restless multi-armed bandit (RMAB) problem (Whittle, 1988) to compute stochastic intervention probabilities. Also in an RMAB setting, Xiong et al. (2022) propose a model-based RL approach that prioritizes users via an index and allocates resources under budget constraints. In public health, Considine et al. (2025) propose RL to optimize extreme heat warnings under a budget on the number of possible alerts. Other works include multi-agent RL for robotic warehouse allocation (Shen et al., 2023) and exogenous MDPs for cloud resource management (Sinclair et al., 2023). While these methods optimize rewards effectively, they often lack interpretability—critical for deployment in sensitive domains requiring trust, transparency, and accountability.

RL and LLM Agents One stream of research in LLM agents (Sumers et al., 2024) has developed somewhat independently of RL, with works like ReAct prompting (Yao et al., 2023) extending chain-of-thought (CoT) (Wei et al., 2022) to action settings. These works have focused on tasks such as open-ended web navigation (Putta et al., 2024), social simulations (Park et al., 2023), and virtual assistants (Vezhnevets et al., 2023). Meanwhile, LLM agents have also been proposed for dealing with complex Markov decision processes such as GLAM (Carta et al., 2023), TWOSOME Tan et al. (2024), BAD (Wen et al., 2024), and AgentGym (Xi et al., 2024), which use LLM finetuning techniques in RL environments with a reward function. While our work is related to hierarchical methods that leverage LLMs for high-level planning (Wang et al., 2023; Szot et al., 2023), our framework is novel in its objective. Unlike prior work that uses language solely to guide a policy toward high task rewards, `RBRL` is the first to treat the language-based "rule" as a primary output, jointly optimizing for both decision-making performance and the rule's quality as a human-readable explanation via a dedicated reward signal.

Explainable RL (XRL) Early XRL relied on methods like decision trees and concept-based explanations (Das et al., 2023), but these struggled with scalability in dynamic environments (Poeta et al., 2023). Recent advances introduced LLMS for post-hoc explanations, such as explaining decision paths from policy trees (Zhang et al., 2023) or adding language descriptions to RL policies (Colas et al., 2022). However, these approaches focus on interpreting pre-existing policies rather than enabling LLMs to generate inherently explainable decisions, with challenges in aligning explanations to human reasoning (Singh et al., 2024). By contrast, inherently (also known as intrinsically) interpretable policies are those that have internal representation that allow explanations (Peng et al., 2022; Milani et al., 2024). Our work sits within this literature by using LLM reasoning traces as the basis for environment action selection. Other methods like EDGE (Guo et al., 2021) and RICE (Cheng et al., 2024) are primarily attribution-based; they identify which inputs (e.g., pixels or state features) were most critical to a decision. Similarly, SelfIE (Chen et al., 2024) provides a post-hoc, mechanistic explanation of an LLM's internal mechanics (hidden states). In contrast, `RBRL` generates high-level, user-facing policy rules that are functional components within the RL loop, providing an explanation of the agent's intent.

C Impact Statement and Limitations

This work advances the development of transparent AI systems for high-stakes decision-making in domains like healthcare, public policy, industry, and many other applications. By enabling LLM agents to generate human-readable rules and explanations while attaining reward maximization via RL, `RBRL` improves trust and accountability, critical for ethical deployment in settings where lives and resources are at stake. While the framework prioritizes alignment with human reasoning, potential risks include over-reliance on imperfect LLM-generated rules or explanations that may inadvertently obscure biases in training data. Mitigation requires rigorous validation of rules by domain experts and ongoing monitoring of LLM outputs. Additionally, `RBRL` 's reliance on LLMs raises computational and accessibility challenges in resource-constrained environments. By addressing these considerations, this research contributes to safer, more equitable AI systems that empower—rather than replace—human decision-makers. To further validate the interpretability of our method, we obtained IRB approval and conducted a human subject study to evaluate the quality of the generated explanations.

Notice that the Uganda dataset used in this study is derived from a simulator that models vital sign trajectories of patients, as provided by Boehmer et al. (2024). Importantly, this simulator only replicates vital sign transitions and does not include any feature information or identifying details of real patients. Thus, the data generated by the simulator cannot be traced to or represent actual individuals, ensuring privacy and ethical compliance. We emphasize that this is purely a simulated patient study; and recognize that for any next steps towards real world use, there is a need to conduct rigorous simulation studies on a large scale with real patient data, with detailed assessments of potential biases, verification of policy convergence and its robustness to distribution shifts in patient populations, and making necessary adjustments. Beyond that, there will be a need to obtain ethics and regulatory approval to test the policy in a real-world setting for future comprehensive field testing, addressing issues of participant consent and privacy; and ultimately there would need to be sufficient human oversight for any future deployment.

Interpretability vs. Performance Tradeoff Various works acknowledge the trade-off between interpretability and performance (Rudin, 2019). In practice, prioritizing interpretability is crucial in practice in many high-stake applications: an approach that we subscribe to in this work. For example, in the clinical AI domain, high-performing black-box systems often face rejection in clinical workflows due to distrust (Shevtsova et al., 2024; DuBois, 2019) as physicians require transparency to validate recommendations and uphold ethical accountability, as mandated by regulatory frameworks (e.g., (EPC, 2024; CSL, 2024)). Interpretable models enable clinicians to audit biases, adapt decision logic to local contexts, and iteratively refine recommendations-fostering collaborative decision-making over reliance on inflexible oracles-whereas opaque policies are prone to failure under real-world distribution shifts(Rudin, 2019; Doshi-Velez & Kim, 2017; Shevtsova et al., 2024; DuBois, 2019). Empirical surveys show clinicians favor models that enable shared decision-making, error accountability, and ethical oversight despite modest performance penalties (Shevtsova et al., 2024)-a critical stance in high-stakes healthcare environments where trust and adaptability outweigh narrow efficiency gains.

D Algorithmic Details

In this section, we present the detailed pseudocodes for `Rule_Search` in Algorithm 2 and the SAC for attention-based policy network in Algorithm 3.

Algorithm 2 outlines the process of rule selection using attention-based policies. First, each rule candidate $\boldsymbol{\rho}_i^t$ is embedded into a numeric vector $\mathbf{q}_i^t$ using a sentence embedding technique (e.g., Sentence-BERT), forming a query matrix $\mathbf{Q}_t$. The state $\mathbf{s}_t$ is also converted into a numeric vector $\mathbf{k}_t$. Cross-attention is applied between $\mathbf{Q}_t$ and $\mathbf{k}_t$ to generate an attention representation $\mathbf{h}$, which may optionally be refined using a self-attention mechanism. A linear layer processes to $\mathbf{h}$ produce score vector $\boldsymbol{\alpha}_\theta^t$. These scores define the policy distribution π_θ, from which a rule $\mathbf{a}_t^{\text{rule}}$ is sampled. This attention-based approach ensures efficient selection of rules by leveraging contextual relationships between the state and rule candidates. For the implementation, the attention layer is realized using the multi-headed attention module from Vaswani et al. (2017). We incorporate a dropout layer, fixed

at 0.05 for the experiments, along with SiLU activation and layer normalization, which are excluded from the notation for brevity.

Algorithm 2 `Rule_Search`: Rule Selection via Attention-Based Policies

Require: Numeric state representation $\mathbf{s}_t \in \mathbb{R}^{d_s}$ and rule set $\mathcal{R}_t = \{\boldsymbol{\rho}_i^t\}_{i=1}^q$. Hidden dimension d_h.

1: Embed each rule $\boldsymbol{\rho}_i^t \in \mathcal{R}_t$ into a numeric vector using sentence embeddings $\mathbf{q}_i^t = \texttt{embed}(\boldsymbol{\rho}_i^t) \in \mathbb{R}^{d_h}$ (e.g., Sentence-BERT) and stack to form a query matrix $\mathbf{Q}_t \in \mathbb{R}^{d_h \times q}$. *// Rule Candidate Embedding*

2: The state $\mathbf{s}_t$ is projected by a linear layer with SiLU activation: $\mathbf{k}_t = \texttt{SiLU}(\texttt{Linear}(\mathbf{s}_t)) \in \mathbb{R}^{1\times d_h}$, with d_h being to denote the architecture hidden dimension. *// State Representation*

3: Use cross-attention to obtain $\mathbf{h} = \mathrm{CrossAttention}(\mathbf{Q}_t, \mathbf{k}_t, \mathbf{k}_t) \in \mathbb{R}^{q\times d_h}$.

4: (Optional) Further apply a self-attention network $\mathbf{h} \leftarrow \mathrm{SelfAttention}(\mathbf{h})$.

5: Apply linear layer to obtain logits vector $\boldsymbol{\alpha}_\theta^t = \mathrm{Linear}(\mathbf{h}) \in \mathbb{R}^{q\times 1}$.

6: Calculate the policy distribution: $\pi_{\theta,i}(\mathbf{Q}_t, \mathbf{k}_t) = \frac{\exp(\alpha_{\theta,i}^t(\mathbf{Q}_t,\mathbf{k}_t))}{\sum_{j=1}^q \exp(\alpha_{\theta,j}^t(\mathbf{Q}_t,\mathbf{k}_t))}$, where $\alpha_{\theta,i}^t$ is i-th element in $\boldsymbol{\alpha}_\theta^t$.

7: Sample rule $\mathbf{a}_t^{\text{rule}} \sim \texttt{Categorical}(\mathcal{R}; (\pi_{\theta,i}(\mathbf{Q}_t, \mathbf{k}_t))_{i=1}^q)$. *// Rule Selection with Attention*

8: **Return** $\mathbf{a}_t^{\text{rule}}$.

Algorithm 3 SAC for Attention-based Policy Network

Initialize Q networks Q_{ϕ_1}, Q_{ϕ_2} and policy network π_θ with random parameters ϕ_1, ϕ_2, θ.
Initialize target Q networks $\bar{Q}_{\bar{\phi}_1}$, $\bar{Q}_{\bar{\phi}_2}$ with weights $\bar{\phi}_1 \leftarrow \phi_1$, $\bar{\phi}_2 \leftarrow \phi_2$.
Initialize temperature parameter β and target entropy $\mathcal{H}_{\text{target}}$; Initialize replay buffer $\mathcal{D}$.
for episode $= 1, \ldots, M$ **do**
  Initialize environment and observe initial state $\tilde{\mathbf{s}}_1$.
  for step $t = 1, \ldots, T$ **do**
    Sample action $\mathbf{a}_t^{\text{rule}} \sim \pi_\theta(\cdot|\tilde{\mathbf{s}}_t)$.
    Execute action $\mathbf{a}_t^{\text{rule}}$, observe reward $\tilde{r}_t$ and next state $\tilde{\mathbf{s}}_{t+1}$.
    Store transition $(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}, \tilde{r}_t, \tilde{\mathbf{s}}_{t+1})$ in replay buffer $\mathcal{D}$.
    if enough samples in $\mathcal{D}$ **then**
      Sample a mini-batch of transitions $(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}, \tilde{r}_t, \tilde{\mathbf{s}}_{t+1})$ from $\mathcal{D}$.
      Compute target Q values: $y_t = \tilde{r}_t + \gamma \mathbb{E}_{\mathbf{a}_{t+1}^{\text{rule}} \sim \pi_\theta(\cdot|\tilde{\mathbf{s}}_{t+1})} \left[\min_{j=1,2} \bar{Q}_{\bar{\phi}_j}(\tilde{\mathbf{s}}_{t+1}, \mathbf{a}_{t+1}^{\text{rule}}) - \alpha \log \pi_\theta(\mathbf{a}_{t+1}^{\text{rule}}|\tilde{\mathbf{s}}_{t+1}) \right]$.
      Update Q networks by minimizing:
      $L_Q(\phi_i) = \mathbb{E}_{(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}, \tilde{r}_t, \tilde{\mathbf{s}}_{t+1})} \left[\left(Q_{\phi_i}(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}) - y_t \right)^2 \right]$ for $i = 1, 2$.
      Update policy network by minimizing:
      $L_\pi(\theta) = \mathbb{E}_{\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}} \sim \pi_\theta} \left[\beta \log \pi_\theta(\mathbf{a}_t^{\text{rule}}|\tilde{\mathbf{s}}_t) - \min_{j=1,2} Q_{\phi_j}(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}) \right]$.
      Update temperature parameter by minimizing:
      $L_\beta(\beta) = \mathbb{E}_{\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}} \sim \pi_\theta} \left[-\beta \left(\log \pi_\theta(\mathbf{a}_t^{\text{rule}}|\tilde{\mathbf{s}}_t) + \mathcal{H}_{\text{target}} \right) \right]$.
      Update target Q networks:
      $\bar{\phi}_i \leftarrow \tau \phi_i + (1-\tau)\bar{\phi}_i$ for $i = 1, 2$.
    end if
  end for
end for

Algorithm 3 presents the SAC algorithm tailored for training an attention-based policy network in selecting the desired rule. This method combines entropy-regularized policy optimization with a structured approach to handle rule-selection effectively. The algorithm begins with the initialization of key components: Q networks Q_{ϕ_1}, Q_{ϕ_2}, target Q networks $\bar{Q}_{\phi_1}, \bar{Q}_{\phi_2}$, and a policy network π_θ. Random parameters are assigned to these networks, and the target Q networks are synchronized with the initial Q networks. A temperature parameter α is initialized to regulate the entropy $\mathcal{H}_{target}$ in the policy objective, ensuring a balance between exploration and exploitation. A replay buffer $\mathcal{D}$ is set

up to store transition data. Notice the entropy is defined as

$$\mathcal{H}_{target} = -\sum_{i=1}^{q} \pi_\theta(\mathbf{a}_t^{\text{rule}}|\mathbf{k}_t, \mathbf{Q}_t) \log \pi_\theta(\mathbf{a}_t^{\text{rule}}|\mathbf{k}_t, \mathbf{Q}_t). \tag{8}$$

During training, each episode starts with the initialization of the environment, and the agent observes the initial state $\tilde{\mathbf{s}}_1$. At every time step, the policy network generates an action $\mathbf{a}_t^{\text{rule}}$ based on the current state. This action is executed in the environment, resulting in a reward $\tilde{r}_t$ and a state transition to $\tilde{\mathbf{s}}_{t+1}$. These transitions are stored in the replay buffer for optimization. When sufficient transitions are available in the buffer, the algorithm samples a mini-batch of transitions and computes the target Q values. The target Q values incorporate entropy regularization and are computed using the minimum of the target Q networks to ensure stability. The Q networks are updated by minimizing the mean squared error between the predicted Q values and the computed targets. The policy network is optimized by minimizing a loss function that combines the policy entropy with the expected Q value, ensuring a stochastic and exploratory policy. The temperature parameter β is updated to maintain the desired balance between exploration and exploitation. Finally, the target Q networks are softly updated to stabilize training. This iterative process continues across episodes and time steps, progressively refining the policy network to achieve optimal rule selection.

E Mathematical Details

E.1 Proof of Theorem 4.1

In this section, we provide the detailed proofs for Theorem 4.1. We start with the following equation

$$\begin{aligned}
&P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}) \cdot \int_a P(\mathbf{s}_{t+1}|\mathbf{a}^{\text{env}}, \mathbf{s}_t) \cdot P(\mathbf{a}^{\text{env}}|\mathbf{a}_t^{\text{rule}}, \mathbf{s}_t) d\mathbf{a}^{\text{env}} \\
&\quad = \underbrace{P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}) \cdot \int_a P(\mathbf{s}_{t+1}|\mathbf{a}^{\text{env}}, \mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) \cdot P(\mathbf{a}^{\text{env}}|\mathbf{a}_t^{\text{rule}}, \mathbf{s}_t) d\mathbf{a}^{\text{env}}}_{(a)\ P(\mathbf{s}_{t+1}|\mathbf{a}^{\text{env}}, \mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) = P(\mathbf{s}_{t+1}|\mathbf{a}^{\text{env}}, \mathbf{s}_t)} \\
&\quad = \underbrace{P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}) \cdot \int_a P(\mathbf{s}_{t+1}|\mathbf{a}^{\text{env}}, \mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) \cdot P(\mathbf{a}^{\text{env}}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) d\mathbf{a}^{\text{env}}}_{(b)\ P(\mathbf{a}^{\text{env}}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) = P(\mathbf{a}^{\text{env}}|\mathbf{s}_t, \mathbf{a}_t^{\text{rule}})} \\
&\quad = \underbrace{P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}) \cdot \int_a P(\mathbf{s}_{t+1}, \mathbf{a}^{\text{env}}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) d\mathbf{a}^{\text{env}}}_{P(A|B,C) \cdot P(B|C) = P(A,B|C)} \\
&\quad = \underbrace{P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}) \cdot P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}})}_{P(A) = \int P(A,B) dB} \\
&\quad = \underbrace{P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}, \mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) \cdot P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}})}_{(c)\ P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1}, \mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}}) = P(\mathcal{R}_{t+1}|\mathbf{s}_{t+1})} \\
&\quad = \underbrace{P(\mathbf{s}_{t+1}, \mathcal{R}_{t+1}|\mathbf{s}_t, \mathcal{R}_t, \mathbf{a}_t^{\text{rule}})}_{P(A|B,C) \cdot P(B|C) = P(A,B|C)} \\
&\quad = \underbrace{P(\tilde{\mathbf{s}}_{t+1}|\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}})}_{\tilde{s}_t := (\mathbf{s}_t, \mathcal{R}_t)},
\end{aligned} \tag{9}$$

where (a) follows from the fact that the transition to $\mathbf{s}_{t+1}$ is fully determined by current state $\mathbf{s}_t$ and current action to the environment $\mathbf{a}_t^{\text{env}}$, i.e., independent on rule set $\mathcal{R}_t$ and selected rule $\mathbf{a}_t^{\text{rule}}$; (b) holds since $\mathbf{a}_t^{\text{env}}$ is determined only by the selected rule $\mathbf{a}_t^{\text{rule}}$ and the state $\mathbf{s}_t$; (c) is due to our designed rule generation procedure where $\mathcal{R}_{t+1}$ is generated by the LLM from the the latest state $\mathbf{s}_{t+1}$. This completes the proof.

E.2 Proof of Proposition 5.1

We decompose the expected suboptimality into two cases:

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{\rho}_i \sim \pi_{\text{LLM}}(\cdot|\mathbf{s})} \left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \right] \\
&\quad = \mathbb{P}\left[\boldsymbol{\rho}_i \in \mathcal{R}^\delta(\mathbf{s})\right] \cdot \mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \mid \boldsymbol{\rho}_i \in \mathcal{R}^\delta(\mathbf{s})\right] \\
&\qquad + \mathbb{P}\left[\boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})\right] \cdot \mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \mid \boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})\right] \\
&\quad \leq \mathbb{P}\left[\exists i \in [N], \boldsymbol{\rho}_i \in \mathcal{R}^\delta(\mathbf{s})\right] \cdot \mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \mid \boldsymbol{\rho}_i \in \mathcal{R}^\delta(\mathbf{s})\right] \\
&\qquad + \mathbb{P}\left[\forall i, \boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})\right] \cdot \mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \mid \boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})\right]. \qquad (10)
\end{aligned}
$$

When $\boldsymbol{\rho}_i \in \mathcal{R}^\delta(\mathbf{s})$, we have:

$$Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \leq \delta.$$

When $\boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})$, we use:

$$Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}}) \leq \epsilon_{\text{worst}}.$$

The probability that none of the sampled rules are in $\mathcal{R}^\delta(\mathbf{s})$ is:

$$\mathbb{P}\left[\forall i, \boldsymbol{\rho}_i \notin \mathcal{R}^\delta(\mathbf{s})\right] \leq (1 - \eta_s)^N.$$

Therefore, the overall expected suboptimality is bounded by:

$$\mathbb{E}\left[Q^*(\mathbf{s}, \mathbf{a}^{\text{env},*}(\mathbf{s})) - Q^*(\mathbf{s}, \mathbf{a}_i^{\text{env}})\right] \leq \delta + \epsilon_{\text{worst}} \cdot (1 - \eta_s)^N. \qquad (11)$$

This completes the proof.

E.3 Proof of Theorem 5.3

Notation. We will denote as $\mathcal{M}$ the original MDP as $\tilde{\mathcal{M}}$ the MDP for the rule-selection agent with transition function as in Theorem 4.1 and reward function $\tilde{R}(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}}) = R(\mathbf{s}_t, \mathbf{a}_t^{\text{env}}) + \lambda R^{\text{rule}}(\mathbf{a}_t^{\text{rule}})$, where $\lambda \geq 0$ is a coefficient weight to balance the rule reward (typically very small) and $\text{image}(R^{\text{rule}}) = [0, 1]$. Throughout, we consistently use the tilde-notation $\sim$ for objects in $\tilde{\mathcal{M}}$ and non-tilde notation for objects in $\mathcal{M}$. Denote the optimal policy of $\mathcal{M}$ as π^*. By standard MDP arguments, we may assume that π^* is deterministic. Recall that the states for $\tilde{\mathcal{M}}$ consist of states-rules pairs $\tilde{\mathbf{s}}_t := (\mathbf{s}_t, \mathcal{R}_t)$ and the LLM agent's actions are the selected rules $\mathbf{a}_t^{\text{rule}}$. The environment action is determined as $\mathbf{a}_t^{\text{env}} = \text{LLM}(\mathbf{a}_t^{\text{rule}}, \mathbf{p}_t)$ where $\mathbf{p}_t$ is the task-state prompt. The `RBRL` policy is trained to maximize the reward $\tilde{R}(\tilde{\mathbf{s}}_t, \mathbf{a}_t^{\text{rule}})$.

Basically, we need to characterize

$$
\begin{aligned}
\text{Gap}(T, s_0) &:= V_{\mathcal{M}}^{\pi^*,T}(\mathbf{s}_0) - V_{\mathcal{M}}^{\pi_{\text{RBRL}},T}(\mathbf{s}_0) \\
&= \underbrace{V_{\mathcal{M}}^{\pi^*,T}(\mathbf{s}_0) - V_{\tilde{\mathcal{M}}}^{\pi_{\text{RBRL}},T}(\mathbf{s}_0)}_{Term_1} + \underbrace{V_{\tilde{\mathcal{M}}}^{\pi_{\text{RBRL}},T}(\mathbf{s}_0) - V_{\mathcal{M}}^{\pi_{\text{RBRL}},T}(\mathbf{s}_0)}_{Term_2},
\end{aligned}
$$

where π^* is the optimal polciy for original MDP $\mathcal{M}$ and π_{RBRL} is the optimal policy for the extended MDP $\tilde{\mathcal{M}}$.

First, $Term_1 \leq 0$. The reason is that $\mathcal{M}$ and $\tilde{\mathcal{M}}$ share the same transition dynamics and $\tilde{\mathcal{M}}$ has a higher reward function than $\mathcal{M}$, we always have the following inequality

$$V_{\mathcal{M}}^{\pi^*,T}(\mathbf{s}_0) \leq V_{\tilde{\mathcal{M}}}^{\pi^*,T}(\mathbf{s}_0) \leq V_{\tilde{\mathcal{M}}}^{\pi_{\text{RBRL}},T}(\mathbf{s}_0), \qquad (12)$$

where the second inequality holds due to the fact that π_{RBRL} is optimal for $\tilde{\mathcal{M}}$.

Next, we bound $Term_2$. It is also very straightforward as the single policy π_{RBRL} on the two MDPs $\mathcal{M}$ and $\tilde{\mathcal{M}}$ will results in the same trajectories $(\mathbf{s}_0, \mathbf{a}_0; \mathbf{s}_1; \mathbf{a}_1; \dots)$. Hence it can be bounded as

$$
\begin{aligned}
Term_2 &= \sum_{t=0}^{T-1} \gamma^t [\tilde{r}(\mathbf{s}_t, \mathbf{a}_t) - r(\mathbf{s}_t, \mathbf{a}_t)] \\
&= \lambda \cdot \sum_{t=0}^{T-1} \gamma^t R^{\text{rule}}(\mathbf{a}_t^{\text{rule}}) \leq \lambda \sum_{t=0}^{T-1} \gamma^t \\
&= \lambda \cdot \frac{1 - \gamma^{T+1}}{1 - \gamma}, \qquad (13)
\end{aligned}
$$

where the inequality comes from the fact that rule reward $R^{\text{rule}}(\mathbf{a}_t^{\text{rule}}) \leq 1, \forall t$. Therefore, we have

$$\text{Gap}(T, \mathbf{s}_0) \leq \lambda \cdot \frac{1-\gamma^T}{1-\gamma}.$$

This completes the proof.

F Experiment Setup

All baselines were trained on 3 seeds for 2000 environment steps each.

F.1 Environments Details

F.1.1 Wearable Device Assignment Domain

The simulator for the Uganda domain is adapted from (Boehmer et al., 2024) with minor modifications to simplify the problem. This section provides an overview of the environment, with additional details available in the original paper. In this environment, they want to allocate vital sign monitoring devices to mothers arriving in a maternal unit in order to better monitor mothers' health condition. Each mother's state is modeled by her vital signs (heart rate, respiratory rate, and blood oxygen saturation) along with each vital sign's variability. The mother's vital sign transition is governed by a multivariate Gaussian distribution defined over her vital signs at the current timestep and next timestep, learned from de-identified vital sign data collected from patients at Mbarara Regional Referral Hospital. MIMIC-III Johnson et al. (2016) is another de-identified clinical vital sign dataset that includes the same set of vital signs as the Uganda domain. The key difference is they have different data sources, as MIMIC-III's data comes from Beth Israel Deaconess Medical Center in Boston.

Wearing a monitoring device does not directly alter a mother's vital sign trajectory but has an indirect positive effect by triggering alerts when vital signs deviate from the normal range. These alerts often lead to medical interventions that improve the mother's condition. If no monitoring device is assigned (passive action), the mother's next state is sampled from the multivariate Gaussian distribution conditioned on the current state. If a monitoring device is assigned and the vital signs remain within the normal range, the vital signs evolve as under the passive action. However, if any vital sign deviates from the normal range, there is a 30% chance the vital signs evolve as under the passive action, based on empirical evidence suggesting that clinicians fail to respond in such cases 30% of the time (Boatin et al., 2021). Otherwise, vital signs are probabilistically adjusted towards the normal range before sampling the next state, modeling the positive impact of medical intervention.

The algorithm's goal is to optimize monitoring device allocation to maximize the aggregate reward across all mothers. We simplify the problem by requiring exactly one new mother to join the maternal unit at each timestep, starting with a single mother in the unit. The system has a budget of five monitoring devices. A device must be allocated to the new mother, and if all devices are already in use, one must be removed from a current user. Once removed, a device cannot be reassigned to the same mother. Each mother remains in the maternal unit for 10 timesteps, after which her vital sign trajectory no longer contributes to the reward. Once a device is taken from a mother, we directly sample her entire vital sign trajectory under passive action for the remaining timesteps she stays in the maternal unit and compute all her future rewards. We can directly compute future rewards because the mother will not receive the device again, so she will only undergo passive action in the remaining time. This observation enables us to maintain a smaller observation space, as we only need to keep track of the states of the mothers who own the device.

In this domain, the constraints can be written as $\|\mathbf{a}_t \in \mathcal{R}^{d_2}\|_1 \leq B, \forall t$, which d_2 represents the number of patients in the system at each time slot, and the 1-norm of the action vector must remain within the budget B.

F.1.2 Heat Alerts Domain

The heat alert issuance problem can be modeled as an MDP in the context of RL Considine et al. (2025). The state at any given time, denoted as $\mathbf{s}_t$, encompasses both exogenous and endogenous factors. Exogenous factors include current weather conditions, such as the heat index, temperature,

and humidity, which directly influence the risk of heat-related health issues. Endogenous factors include the history of issued alerts, such as the number and timing of past alerts, their effectiveness, and the remaining budget for the season. Additionally, the day of the week is considered, as public responsiveness to alerts may vary between weekdays and weekends. The action space is binary, with $\mathbf{a}_t \in \mathbb{Z}_2$. The decision to issue a heatwave alert $\mathbf{a}_t = 1$ or not $\mathbf{a}_t = 0$ is constrained by the remaining alert budget. If the budget is exhausted, no further alerts can be issued. The reward function is designed to reflect the reduction in heat-related hospitalizations, which depends on the effectiveness of the alert under current weather conditions. A Bayesian hierarchical framework could be employed to model the health impact of alerts, capturing the uncertainty in their effectiveness. Importantly, consecutive alerts tend to lose effectiveness, introducing a diminishing returns effect that must be accounted for in the decision-making process.

The transition dynamics, $P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, describe how the system evolves over time. The next state is influenced by weather trajectories, the action taken, and public responsiveness to alerts. For instance, issuing an alert reduces the remaining budget and updates the history of issued alerts, while the weather conditions may change independently. Public responsiveness may also vary based on the frequency and timing of past alerts. A key constraint in this problem is the limited alert budget, which necessitates strategic allocation of alerts throughout the season. The goal is to learn a policy $\pi(\mathbf{s}_t|\mathbf{a}_t)$ that maximizes cumulative rewards by effectively issuing alerts during severe heat conditions to minimize hospitalizations, while conserving the budget for future use. This involves balancing immediate health benefits against the potential need for alerts later in the season, addressing the trade-offs between short-term and long-term outcomes.

For this domain, the budget constraints can be expressed as $\sum_{t=1}^{h} \mathbf{a}_t \in \mathbb{R} \leq B$, where the total sum of all actions over time horizon h must not exceed a budget threshold B.

F.1.3 Bin Packing Domain

We adopt the online stochastic bin packing environment from Balaji et al. (2019), which models the sequential allocation of items with random sizes into fixed-capacity bins. At each time step t, an item $x_t \in (0, 1)$ is sampled from a known distribution and must be immediately placed into one of the currently open bins or into a new bin. Each bin has a fixed capacity B, and assigning an item to a bin must satisfy the constraint that the total usage does not exceed B. The environment is formulated as MDP where state $\mathbf{s}_t$ is a tuple containing the size of the arriving item x_t and the current fill levels of all open bins, action $\mathbf{a}_t$ selects a valid bin from the set of open bins or opening a new bin; invalid actions (bins that would overflow) are masked. The transition probablity $P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ determines update of the selected bin's fill level, followed by stochastic sampling of the next item x_{t+1}. The reward function r_t is defined as the negative incremental waste after placing item x_t, i.e., the change in unused space across all open bins. The objective is to learn a policy $\pi(\mathbf{a}_t|\mathbf{s}_t)$ that minimizes the total accumulated waste over time. This domain reflects a broad class of real-world resource allocation settings-such as server packing, memory management, or warehouse logistics, where agents must make immediate, irrevocable decisions under hard capacity constraints.

For this domain, the budget constraints can be expressed as $x_t + C_{\mathbf{a}_t} \leq B$, where at every time no bin can exceed the capacity B, with $C_{\mathbf{a}_t}$ being the current usage of bin $\mathbf{a}_t$.

F.2 Gym environments an Language Wrappers

We implemented the `WearableDevicesAssignment` environments as gymnasium environments Towers et al. (2024), while the `HeatAlerts` domain was already available in this format. We additionally created a `LanguageWrapper` Python class described in Table 2, which can be applied to any `gymnasium` environment. Our code implementations can be applied to any environment wrapped in this class.

F.3 RL implementations, hyperparameters and Settings

We implemented three main RL algorithms for the experiment sections: Attention-based SAC for `RBRL`, numeric PPO, and Finetuning-based PPO. We based our implementation on the single-file, high-quality implementations from the `cleanrl` project (Huang et al., 2022). For Attention-based SAC, we required significant changes to keep track of the rule-augmented state space, as described in

Method/Property	Type	Description
`task_text`	Property (Abstract)	Returns a description of the task that the environment is solving.
`action_space_text`	Property (Abstract)	Returns a description of the action space of the environment.
`state_descriptor(obs, info)`	Abstract Method	Converts the observation into a text description.
`step(action)`	Method	Wraps the step method of the environment adding the text representation to the state space.
`reset(seed, options)`	Method	Wraps the reset method of the environment adding the text representation to the state space.
`action_parser(s)`	Method	Parses an action string and converts it into an appropriate format for the environment's action space.
`(rule_examples`	Property (Optional)	Returns a list of string representation of rules.

Table 2: Methods and properties of the `LanguageWrapper` class

Section 4.4. Other major changes to the baseline SAC implementation (originally designed for Atari) were more frequent target network updates and updating the actor and critic four times per iteration. This was done to improve sample efficiency and cope with the slow generation by the LLM. Numeric PPO was used practically without modification.

For the Finetuning-based PPO, we used low-rank adaptation (LoRA) Hu et al. (2021) with the Transformers package and models hosted on Llama Hugging Face Wolf et al. (2020). We set the rank to $r = 1$ and the adaptation weight to 2, resulting in only 0.8% trainable parameters (still an order of magnitude larger than the Attention-based policy). Tables 3, 4, and 5 show the hyperparameters and settings used in these implementations.

F.4 Computing environment

SAC attention can run on a regular laptop since most of the computation happens in the cloud through API LLM calls, while the RL module is small and can run on personal CPUs. Nonetheless, the process is bottlenecked by the speed of generation from the APIs. A full run of 2 million environment steps, with parallelized API calls across four environments, took approximately four hours to complete. One training cycle did not exceed $10 in API costs. However, all the experiments and development incurred approximately $1,500 in API costs. As described in the main text, the LLM fine-tuning experiments used an Nvidia A100 40GB GPU for each seed, training on three seeds for 18 hours each. Computations were performed on a Slurm-based high-performance computing cluster.

Table 3: SAC Hyperparameters and Settings for `RBRL`.

Parameter	Default Value	Description
`num_envs`	4	Number of parallel environments
`total_timesteps`	500	Total number of environment steps
`gamma`	0.95	Discount factor γ
`tau`	1.0	Target smoothing coefficient
`batch_size`	16	Batch size of sample from the replay memory
`buffer_size`	4096	The replay memory buffer size
`max_episode_steps`	32	Episode truncation
`learning_starts`	256	Timestep to start learning
`policy_lr`	1×10^{-4}	Learning rate of policy network optimizer
`q_lr`	1×10^{-4}	Learning rate of Q-network optimizer
`actor_updates`	4	Number of actor updates per update cycle
`critic_updates`	4	Number of critic updates per update cycle
`target_network_frequency`	64	The frequency for the target network update
`alpha`	0.01	Initial entropy regularization coefficient
`autotune`	True	Automatic tuning of the entropy coefficient
`target_entropy_scale`	0.89	Coefficient for scaling the autotune entropy target
`dropout`	0.05	The dropout rate
`num_rules`	10	Number of rules for `RBRL`
`llm`	"gpt-4o-mini"	LLM for generation
`embedder_lm`	"m2-bert-80M-8k-retrieval"	The LLM to use for embeddings
`embed_dim`	768	Dimension of rule embeddings
`hidden_dim`	16	Hidden dimension of networks
`rule_reward_coef`	0.1	The reward coefficient for the rules
`num_self_attention_layers`	1	For the actor and critic
`num_cross_attention_layers`	1	For the actor and critic

Table 4: Numeric PPO Hyperparameters and Settings.

Parameter	Default Value	Description
total_timesteps	50000	Total timesteps of the experiments
learning_rate	2.5×10^{-4}	Learning rate of the optimizer
num_envs	4	Number of parallel environments
num_steps	512	Steps per policy rollout
anneal_lr	False	no learning rate annealing
gamma	0.95	Discount factor γ
gae_lambda	0.95	Lambda for Generalized Advantage Estimation
num_minibatches	4	Number of mini-batches
update_epochs	64	Number of update epochs
norm_adv	True	Whiten advantages
clip_coef	0.2	Surrogate clipping coefficient
clip_vloss	True	Clipped loss for value function
ent_coef	0.01	Coefficient of entropy term
vf_coef	0.5	Coefficient of value function
max_grad_norm	0.5	Maximum gradient clipping norm
target_kl	None	Target KL divergence threshold
hidden_dim	16	Hidden dimension of networks
num_hidden_layers	2	For policy and critic networks
max_episode_steps	32	Episode truncation

Table 5: LLM PPO Finetuning Hyperparameters and Settings.

Parameter	Default Value	Description
total_timesteps	500	Total number of timesteps
learning_rate	2.5×10^{-4}	Learning rate of optimizer
num_envs	4	Number of parallel game environments
num_steps	32	Steps per policy rollout
anneal_lr	True	Enable learning rate annealing
gamma	0.95	Discount factor γ
gae_lambda	0.95	Lambda for Generalized Advantage Estimation
update_epochs	4	Number of update epochs per cycle
norm_adv	True	Advantages whitening
clip_coef	0.2	Surrogate clipping coefficient
clip_vloss	True	Clipped loss for value function
ent_coef	0.01	Coefficient of entropy term
vf_coef	0.5	Coefficient of value function
kl_coef	0.05	KL divergence with reference model
max_grad_norm	0.5	Maximum gradient clipping norm
target_kl	None	Target KL divergence threshold
dropout	0.0	Dropout rate
llm	”meta-llama/Llama-3.1-8B-Instruct”	Model to fine-tune
train_dtype	”float16”	Training data type
gradient_accumulation_steps	16	Number of gradient accumulation steps
minibatch_size	1	Mini-batch size for fine-tuning
max_chunk_size	256	Maximum length sequence for the back propagation
max_episode_steps	32	Maximum number of steps per episode

G Additional Experimental Results

G.1 RBRL + Advanced Reasoning LLM

We conducted a new experiment using the advanced reasoning baseline “o3-mini” from ChatGPT, implemented within our MimicIII environment. This baseline has been trained with RL to solve reasoning and mathematical problems more effectively. The results are shown in Table 6 below.

Here, we observe that using RBRL with the advanced reasoning baseline helped to almost match the performance of the Oracle policy in very few environment steps, illustrating the compatibility of RBRL with reasoning-based LLMs. However, we remark that relying on o3-mini resulted in almost 10x higher cost and 5x slower inference than 4o-mini used for most experiments. As noted in the manuscript, our total cost for the experiments was approximately 2000, which would increase greatly with o3-mini. Hence, we focused on improving cost-effective LLMs.

Table 6: RBRL + Advanced Reasoning on MimicIII (Av. Reward)

Algorithm	@500 steps	@2.5k steps	@100k steps (working oracle)
PPO	-0.66 ± 0.12	-0.55 ± 0.07	-0.14 ± 0.06
SAC	-0.85 ± 0.12	-0.81 ± 0.15	–
RBRL (GPT–4o mini)	-0.70 ± 0.07	-0.33 ± 0.03	–
RBRL+ AdvancedReasoning (GPT–o3 mini)	-0.29 ± 0.05	-0.16 ± 0.05	–

G.2 Comparison of Other LLM Finetuning Methods for MDPs

In the main paper, we compared RBRL with token-wise LLM finetuning using LORA in the MimicIII domain. We further implemented the methodology described by Zhai et al. (2024) which provides a variation of the finetuning PPO methodology in which the log probability of an action is first aggregated across all tokens and no reference model reward (KL) divergence is used. The authors also introduce a weight coefficient λ_{cot} to reduce the weight of the COT tokens in the log-probability computation. The results are shown in Table 7. The best method was still RBRL, while the second best was the finetuning baseline at the action level with $\lambda_{\text{cot}} = 1.0$.

Table 7: New baseline on MIMICIII with Llama 3.1 8B* (Av. Rewardx

Baseline	Av. Return @ 2.5k
Finetuning – token level	-0.85 ± 0.21
Finetuning – action level ($\lambda_{\text{cot}} = 1.0$)	-0.46 ± 0.09
Finetuning – action level ($\lambda_{\text{cot}} = 0.3$)	-0.90 ± 0.12
RBRL*	$\mathbf{-0.36 \pm 0.05}$

G.3 Consistency of Explanations

A central claim of our work is that RBRL generates consistent explanations. Following the discussion in the literature (Lyu et al., 2024; Jacovi & Goldberg, 2020), we define a consistent explanation as one that is causally linked to the decision process. In RBRL, this consistency is achieved by design through a strict, decomposable pipeline: (`observation` $\rightarrow$ `rule` $\rightarrow$ `action`). Our key argument, which we evaluate empirically below, is that the state and the selected rule alone are sufficient to determine the action taken, regardless of the intermediate reasoning trace used to generate the rule. The rule itself serves as the true, functional explanation for the agent's decision.

Action selection comes from the rule, not the reasoning trace We show that including or not including the reasoning trace does not alter the action decision, which is based on the rule. For this purpose, we use a pretrained RBRL policy to collect tuples (`state, reasoning, rule, action`). Next we evaluate how much the action selection changes when including the reasoning trace in addition to the rule in the action prompt.

Metrics. We compute the agreement score as the fraction of times that the action selected without including the LLM reasoning in the action prompt is the same as that when including it. A higher value means the rule determines the action, not the LLM reasoning output. All experiments are with `gpt-4o-mini` as in the main paper. The scores are averaged over 100 environment steps with standard errors.

Table 8: Consistency of action selection

Environment	Agreement score $\pm$ SE
HeatAlerts	0.99 ± 0.010
Uganda	0.92 ± 0.027
Mimic III	0.92 ± 0.027
BinPacking	0.97 ± 0.017

Our results confirm that the action consistency comes mainly from the state and rule, not the LLM reasoning, further showing that LLM self-explanation is not the core of RBRL. In terms of Jacovi & Goldberg (2020), RBRL is more closely related to the class of methods where, *"the explanation [rule] is itself a model capable of making decisions (e.g., decision trees or rule lists)."*

G.4 Evaluating the Translation Fidelity from Rules to Explanations

Having established in our previous experiment that the rule is the true causal component of the decision, a critical next question arises: Does the final, user-facing natural language explanation translate all decision-relevant information from that rule? An ideal explanation should suffer no information loss in this translation, meaning it should be self-contained and sufficient for an observer to deduce the action that was taken.

To verify this, we conduct an action-prediction experiment. We provide an auxiliary LLM with the agent's full explanation but mask the final action (e.g., "...Thus, I concluded XXX."). The LLM's task is to predict the masked action. High prediction accuracy serves as a strong proxy for high-fidelity translation, indicating that the explanation successfully preserves the action-predictive information from the original rule.

An example of the experimental setup is shown in Table 9. Figure 7 presents the results of this evaluation across three domains: `HeatAlerts`, `Uganda`, and `MimicIII`. The plot compares the classification accuracy and F1 score of a binary classifier trained to predict the agent's actions solely from the masked explanations. We benchmark `RBRL` against two baselines: `CoT`, which produces naive rationale via chain-of-thought prompting, and `TBRL`, which incorporates optimization over toughts. Across all domains, `RBRL` consistently outperforms both baselines with higher accuracy and F1 scores. In particular, the large margin between `RBRL` and `CoT` highlights the importance of grounding explanations in decision-consistent rules rather than free-form text. The error bars represent 95% confidence intervals obtained via bootstrapping, demonstrating statistical robustness. These results confirm that `RBRL`'s explanations are more than just plausible narratives; they are reliable translations of the agent's underlying decision mechanism, preserving enough information to be considered truly consistent and trustworthy.

Table 9: Examples of Masked Explanations (HeatAlerts)

Agent	**Explanation**
CoT Reasoning	I observed a heat index of 100 F, a warning streak of 6, and a remaining budget for 2 warnings. Forecast shows high temperatures next week. I concluded XXX.
RBRL	I observed a heat index of 100 F with 6 warnings in 14 days. Reasoning that excessive warnings lead to fatigue, I applied a rule to not issue warnings if warnings in the last 7 days $\geq$ 5 and budget remained. Thus, I concluded XXX.

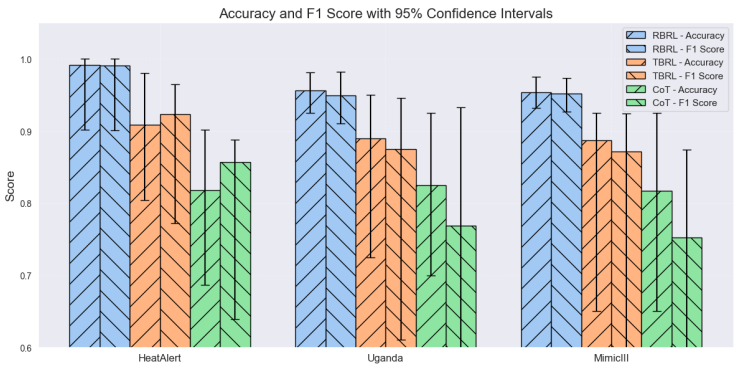


Figure 7: Consistency of explanation as measured by the action prediction task.

G.5 Study on Rule Diversity

As noted in Section E, we only require that at least one candidate rule yields the optimal actions with high probability; non-optimal actions are irrelevant. We conducted an experiment to assess this assumption in practice.

Setup. We use an auxiliary oracle numeric SAC policy trained for 100k steps per environment. The metric is the percentage of times the proxy optimal action is recommended by at least one rule. The experiment is repeated for 100 transitions.

Figure 8: Rule diversity verification.

Figure 8 illustrates the impact of increasing the number of rules on the match ratio between rule-based decisions and the reference RL agent actions across four environments: `HeatAlert`, `MimicIII`, `Uganda`, and `BinPacking`. The match ratio serves as a proxy for rule-to-action consistency, reflecting how well the selected rules align with the RL agent's behavior. Across all environments, increasing the number of candidate rules improves the match ratio, indicating that greater rule diversity enhances the likelihood of covering the correct action. The shaded regions represent variance across multiple runs, highlighting robustness trends in each domain. *In summary, as the number of rules generated increased, the probability of generating an optimal rule candidate increased, thereby increasing the performance guarantees of `RBRL` as discussed in section 5.*

G.6 Additional results for General RL domain

Table 10: Average Return on general RL domain: BabyAI

Baseline	**BabyAIGotoObj**	**BabyAIGotoLocal**
CoT (GPT-4o mini)	0.12 ± 0.12	0.12 ± 0.17
RBRL* @ 5k steps (GPT-4o mini)	$\mathbf{0.46 \pm 0.17}$	$\mathbf{0.32 \pm 0.11}$
SAC @ 5k steps	0.27 ± 0.22	0.16 ± 0.18

Besides the three aforementioned resource-constrained allocation domains, we further evaluate our `RBRL` in the more general standard RL settings. We applied our methodology directly to the widely used BabyAI domain Chevalier-Boisvert et al. (2018). The environment was originally proposed to investigate sample efficiency and generalization in RL. We run on two environments "BabyAIGotoObj", "BabyAIGotoLocal" environments using the standard wrapper for full observability are included in the official wrapper in the minigrid Python library. Improved performance for these two general RL domains can be found in Table 10. Numeric RL baselines can perfectly solve these tasks albeit requiring potentially millions of observations Chevalier-Boisvert et al. (2018). Here we focus on a small sample regime. We do not position our contribution as a general method for RL and nor claim any superiority over state of the art general-purpose RL.

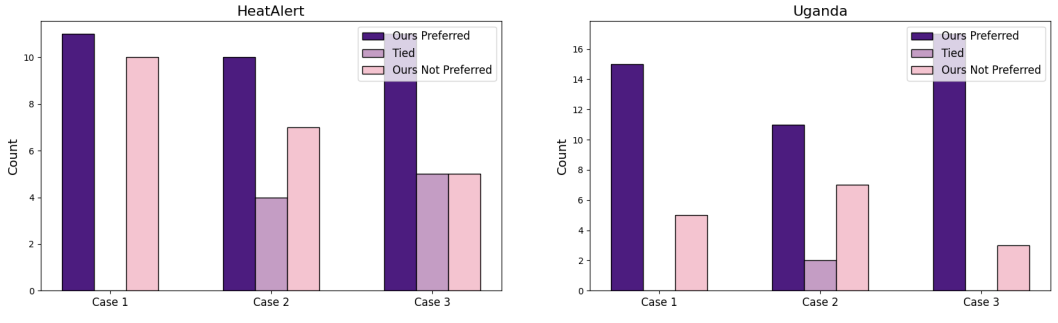
H Additional Survey Results

Figure 9 illustrates the results of a human survey conducted to evaluate the quality of explanations generated by our method compared to alternatives. A total of 21 valid responses were collected for the HeatAlert environment (Figure 9a), and 20 valid responses were gathered for the Uganda environment (Figure 9b). As shown in the figures, our method was favored by the majority of participants across all cases. In the HeatAlert environment, the preference for our approach is evident, although there is a small percentage of tied and "Not Preferred" responses. In contrast, the preference for our method is even more pronounced in the Uganda environment, with a significantly higher number of participants selecting "Ours Preferred." These results demonstrate the effectiveness of our approach in generating explanations that resonate better with human users, particularly in the Uganda domain.

Figure 10 illustrates the survey outcomes obtained by querying LLMs 20 times for each case in the HeatAlerts (Figure 10a) and Uganda (Figure 10b) environments. To ensure variability, the LLM's sampling temperature was controlled, enabling randomized responses for each trial. Similar to the human survey results, our method ("Ours Preferred") is overwhelmingly favored across all cases in both domains. Notably, the consistency of "Ours Preferred" responses highlights the effectiveness of our approach in generating explanations that align well with the LLM's evaluation criteria, further validating the robustness of our methodology.

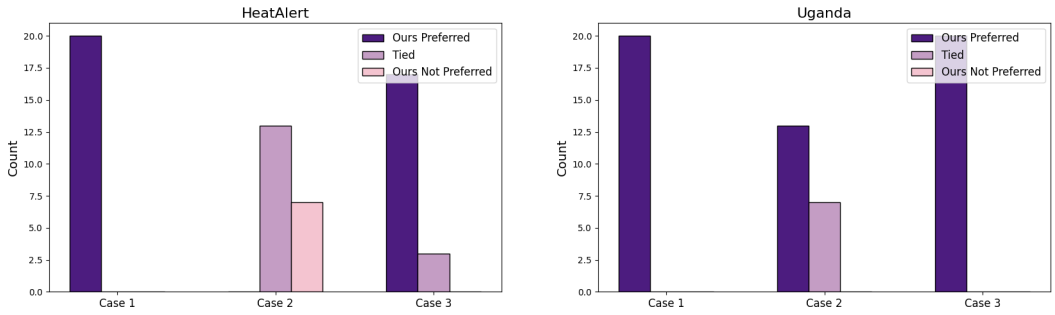
Figure 11 illustrates the survey results evaluating hallucination occurrences across two environments (Uganda and HeatAlert) for three explanation types: Chain of Thought (CoT), Rule-Bottleneck Reinforcement Learning (RBRL), and None (indicating no explanation).

In Figure 11a, the results from the human survey indicate that CoT-based explanations had a significant proportion of hallucinations, particularly in the Uganda environment, where it accounted for 42.4% of responses. RBRL explanations showed markedly fewer hallucinations in both domains, highlighting its robustness. A notable percentage of responses for None indicate scenarios where explanations were either absent or irrelevant. In Figure 11b, results from the LLM survey further emphasize the trends observed in the human survey. Hallucination rates for CoT were even higher in the Uganda environment (81.7%), whereas RBRL explanations exhibited almost no hallucinations across both domains. In the HeatAlert environment, the absence of explanations (None) led to the highest percentage of hallucinations, underlining the importance of well-structured, rule-based explanations like RBRL. Need to mention that the plausibility (human preference) of explanations of RBRL shown in the human survey was not conditional on seeing the CoT reasoning traces. These results collectively demonstrate that the RBRL framework significantly mitigates hallucinations, providing more accurate and reliable explanations compared to other methods.



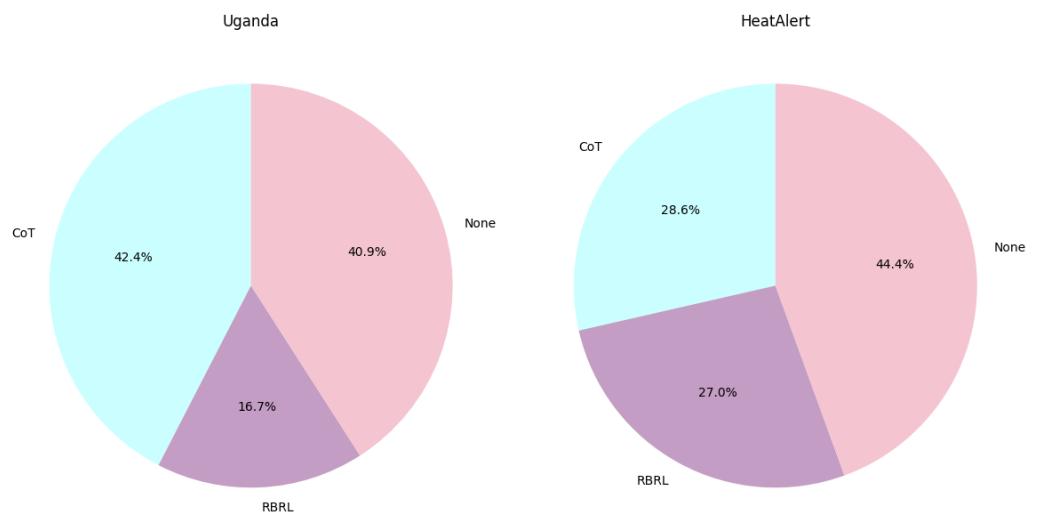
(a) HeatAlert (b) Uganda

Figure 9: Results from **human** surveys conducted in the HeatAlert (a) and Uganda (b) environments. 21 participants provided feedback for the HeatAlert domain, while 20 valid responses were collected for the Uganda domain. The results indicate that our method ("Ours Preferred") was favored by a majority of participants, particularly in the Uganda domain, where the preference is more pronounced.

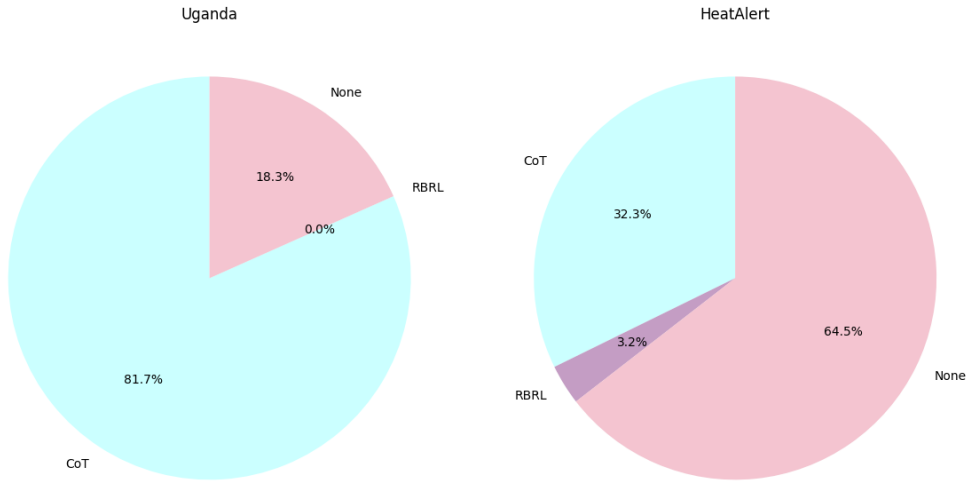


(a) `HeatAlert` (b) `Uganda`

Figure 10: Survey results generated by querying **LLMs** 20 times for each case in the `HeatAlert` (a) and `Uganda` (b) environments. By varying sampling temperatures, randomized responses were collected. The results show that our method ("Ours Preferred") consistently outperforms alternatives across all cases, highlighting its robustness and alignment with the evaluation criteria of the LLMs.



(a) Results from the **human** survey, showcasing the proportion of hallucination detected across three categories: `CoT`, `RBRL`, and `None`. In both domains, hallucinations were most frequently identified in `None`, with `RBRL` showing significantly fewer instances.



(b) Results from the **LLM**-based survey, where hallucination detection was assessed through multiple iterations of LLM evaluation. `CoT` exhibited higher hallucination rates in the `Uganda` domain, while `RBRL` demonstrated minimal hallucination occurrences in both domains.

Figure 11: Survey results for hallucination detection across the `HeatAlert` and `Uganda` environments.

I Prompt Templates and Rule Examples

I.1 Prompt Format

In this section, we illustrate the prompt format used in our `RBRL` for generating thoughts, rules, actions, rule scores, and explanations in Figure 12.

Generate Thoughts

First, reason about what elements should be considered when choosing the optimal action. Your response should consist of a single short paragraph that reflects on the consequences, benefits, and drawbacks of each action in the current state.

Generate Rules

Now, suggest {num_rules} rules that could be useful to make an optimal decision in the current state. For each rule, provide the explanation of why it is important to consider it at the given state. Each rule should be in machine-readable JSON Lines format. Each line should follow the following schema:

{'background' str, 'rule': str, 'state relevance': str, 'goal relevance': str}

- The 'background' should a brief introduction and motivation to the focus of the rule.
- The 'rule' should be a statement of the form '[do/select/prioritize] [if/when/condition]' where the condition must be relevant to the current state.
- The 'state relevance' should explain why the rule applies to the current problem state.
- The rule alone should be sufficient to deduce the optimal action that should be taken in the current problem state.
- Start each line with the character '```- {.

Generate Actions

Below is/are a prioritization rule/rules to make an optimal decision in the current state:

[selected rule/rules]

Now, choose the optimal action given the current problem state and this/these prioritization rule/rules. Your answer must consist exclusively of one of the following actions:

Possible actions:{description about action space}

You cannot refuse to respond. Do not provide additional information or context for your answer, only the action.

Generate Rule Scores

To make a decision in the current state, the following rule/rules was/were selected:

{rules}

You will now be given a question you need to answer with a simple 'yes' or 'no'.
q1 = "Is/are the rule/rules **alone** sufficient to understand the optimal action/decision that the system should take in current the problem state?"
q2 = "Is the condition in the rule/rules actionable and complete in the current problem state (containing sufficient detail about the current problem state without unnecessary information)?"
q3 = "Did the selected rule/rules sufficiently help to understand the previous decision without contradictions?"
Answer the following questions with a simple 'yes' or 'no' without additional information or justification. Your response should be a single word.

Generate Explanation

You chose action {outputs['action']} in the current problem state. Explain why you chose the optimal action based on the conversation and history. Your response should follow the template: "I observed... I used a rule stating... I concluded..."

Figure 12: Prompts template for generating thoughts, rules, actions, rule scores, and explanations.

I.2 Rule Example

In this section, we provide some rule examples for each domain in Figure 13.

Rule Examples

Example 1: Wearable Device Assignment

- **Rule1**: "Prioritize reallocating devices from patients who have worn the device for the least amount of time and have stable vital signs."
- **Rule2**: "Select to reallocate devices from patients with less critical vital signs to those with high pulse rates or low SPO2."
- **Rule3**: "Reallocate devices from patients whose mean vital signs are within normal ranges to those whose mean vital signs are abnormal."
- **Rule4**: "Select to reallocate devices from patients with lower standard deviation in vital signs, as they are less likely to experience sudden changes."
- **Rule5**: "When a new patient arrives, reallocate devices from patients who have shown the least improvement over time."

Example 2: Heat Alert Issuance

- **Rule1**: "If the current heat index is above the average heat index of the past week, issue a warning."
- **Rule2**: "If there have been 5 or more warnings in the last 14 days, do not issue a warning unless the heat index exceeds 100 F."
- **Rule3**: "If the heat forecast for the next day exceeds 98 F, issue a warning today."
- **Rule4**: "If today is a weekday and the heat index is above 95 F, do not issue a warning unless the forecasted heat index for the weekend exceeds 100 F."
- **Rule5**: "If the remaining number of warnings is 5 or fewer, prioritize issuing a warning when the heat index exceeds 98 F."

Example 3: Bin Packing Allocation

- **Rule1**: "If the current item is less than half the bin capacity, evaluate existing bins first before creating a new one."
- **Rule2**: "When possible, open a new bin instead of using a level that has already been used heavily."
- **Rule3**: "Select a bin such that the remaining space is closest to the bin capacity after placing the item."
- **Rule4**: "Always check if any bins are completely empty before making any decisions."
- **Rule5**: "If any bin can accommodate the item exactly without waste, prioritize that choice."

Figure 13: Rule examples for the considered two domains.

J Survey Example

In this section, we present a survey example from the `Wearable_Device_Assignment` domain. The survey for the `HeatAlert` domain follows the same format and can be easily adapted by substituting the task and corresponding actions. For brevity, we include only one example case from the `Wearable_Device_Assignment` domain.

Task: *You are tasked with optimizing the allocation of limited vital sign monitoring devices among patients. Devices improve vital signs and prevent abnormalities, but their limited availability requires reallocating them from stable patients to higher-risk incoming patients, who must always receive a device. The normal range of vital signs are provided in Figure 14. The goal is to minimize costs associated with abnormal vital signs, where costs are calculated exponentially based on deviations from predefined thresholds. Wearing a device improves abnormal vital signs with a 70% success rate.*

Possible actions: *Choose the id of the device that will be reallocated to the new incoming patient. Your answer should be a single integer i from 0 to 4 (the number of devices) such that*:

- *Always choose a free device if available*
- *If no free device is available, then choose device i whose current patient is at least risk or would benefit less from wearing the device.*

Vital Sign	Normal Range
Heart Rate	60–100 bpm
Respiratory Rate	12–20 breaths/min
SPO2 (Oxygen Saturation)	≥ 95%

Figure 14: Normal range of vital signs.

In the following, you will be presented with three cases. Each case includes two explanations. Please read the text for each case carefully and answer the questions provided.

Case 1: Current state of the decision problem:
Number of devices: 5
Number of free devices: 1
IDs of free devices: 3

Device 0: Device is currently assigned to a patient with the following description:
Timesteps wearing the device: 1
Pulse rate - Last value: 95.22 - Mean: 105.12 - Standard deviation/volatility: 10.56
Respiratory rate - Last value: 20.14 - Mean: 20.54 - Standard deviation/volatility: 0.64
SPO2 - Last value: 98.42 - Mean: 97.89 - Standard deviation/volatility: 0.88

Device 1: Device is currently assigned to a patient with the following description:
Timesteps wearing the device: 1
Pulse rate - Last value: 89.55 - Mean: 81.86 - Standard deviation/volatility: 8.55
Respiratory rate - Last value: 14.85 - Mean: 20.81 - Standard deviation/volatility: 3.40
SPO2 - Last value: 95.31 - Mean: 96.22 - Standard deviation/volatility: 1.38

Device 2: Device is currently assigned to a patient with the following description:
Timesteps wearing the device: 1
Pulse rate - Last value: 106.05 - Mean: 105.09 - Standard deviation/volatility: 2.91
Respiratory rate - Last value: 19.34 - Mean: 20.80 - Standard deviation/volatility: 2.69
SPO2 - Last value: 99.56 - Mean: 99.36 - Standard deviation/volatility: 0.27

Device 3: Device is currently free.

Device 4: Device is currently assigned to a patient with the following description:
Timesteps wearing the device: 1
Pulse rate - Last value: 80.02 - Mean: 79.03 - Standard deviation/volatility: 1.58
Respiratory rate - Last value: 22.71 - Mean: 21.31 - Standard deviation/volatility: 5.45
SPO2 - Last value: 99.61 - Mean: 99.86 - Standard deviation/volatility: 0.14

Explanation A: I chose to reallocate device 4 because the patient currently using it has a stable pulse rate (80.02) and a high SPO2 level (99.61), indicating they are less at risk and may not require continuous monitoring. In contrast, reallocating this device allows for an incoming patient, who likely has more urgent health needs, to receive the device, thereby optimizing the overall allocation of resources to those who require immediate attention.

Explanation B: I observed that device 3 is currently free, and there is an incoming patient who requires immediate monitoring to prevent potential deterioration in their health. I used a rule stating that all incoming patients must be assigned a device, with priority given to those at risk. I concluded that assigning 'device': 3 to the incoming patient ensures timely and necessary vital sign monitoring, aligning with the prioritization principle.

Q1. Do Explanation A and Explanation B appear the same or different to you?

☐ Same (Skip Q2 and go to Question Q3)
☐ Different

Q2. Which explanation do you find better?

☐ Explanation A
☐ Explanation B

Q3. Do the explanations contain any hallucinations?

☐ Both
☐ Only Explanation A
☐ Only Explanation B
☐ None