# Explainability for Safety in Reinforcement Learning

Tongtong Liu* Wake Forest University
Md Asifur Rahman† Wake Forest University
Joe McCalmon‡ Wake Forest University
Sarra Alqahtani§ Wake Forest University

## 1 Introduction

Reinforcement learning (RL) has shown promising potential for a wide range of real world applications including robotics over the years. However, due to the nature of exploratory learning involved in RL, it is vulnerable to adversarial threat which affects its usability in safety critical applications. Thus, in applications where safety is of utmost importance, there is a growing need for integrating RL with safety mechanisms. Moreover, RL agents are unable to *interpret* their behavior and most RL policy for complicated environment, in nature, are unintelligible to human, thus users lose confidence in its correctness. The black-box nature of RL's deep neural network (DNN) decisions decreases end-users' trust of the agent's behavior. The lack of *explainability* implies that the optimal strategies agents learn cannot be used to improve our understanding of their safety. Since safety-assurance approaches, in general, compromise system performance, we must ensure that human practitioners and users trust them, lest they ignore them and negate their effectiveness. The literature lacks a general framework that can automatically and interactively explain the features that assure agents' safety and performance individually and in teams.
In this paper, we demonstrate the usage of our explainable RL method, CAPS [5] that can be integrated with existing RL algorithms to improve their explainability with and without our proposed safety method. The interpretation and explainability of RL in general and safe RL in particular are still relatively unexplored, lacking well-established principles and best practices. This paper shows that we can bring the traditional software development life-cycle concept into RL safety assurance by allowing users to peek into its black- box model, understand its policies, then identify new safety requirements or refine existing ones. We use CAPS to explain the learned policies at the agent level with and without the safety method to increase users' trust in this method. Interpretable graphs generated from CAPS approximate the agent's behavior and can be further analyzed through manual inspection, model checkers, or statistical analysis to elucidate the behavior of the underlying RL systems. We hypothesize that RL safety will result if CAPS transparently offers enough explanations of agent's decisions and actions in terms of safety.

## 2 Reinforcement Learning

RL is suitable for solving Markov decision process (MDP) problem, which can be written as a tuple $\mathscr{M} = (\mathscr{S}, \mathscr{A}, \mathscr{P}, \mathscr{R})$. Here $\mathscr{S}$ denotes the state space, $\mathscr{A}$ denotes the action space, $\mathscr{P} : \mathscr{S} \times \mathscr{A}$ is the transition probability, and $\mathscr{R} : \mathscr{S} \times \mathscr{A} \times \mathscr{S}$ is the reward function. An RL agent chooses its actions according to its policy, $\pi(a_t|s_t)$ and adjust it to acquire the optimal policy $\pi^*$ that maximizes the state-action value function $Q_{\pi^*}(s,a); s \in \mathscr{S} and a \in \mathscr{A}$.

*e-mail: liut18@wfu.edu
†e-mail: rahmm21@wfu.edu
‡e-mail: mccalmonjoe@gmail.com
§e-mail: sarra-alqahtani@wfu.edu

## 3 Explainable RL: Safety as a Product of Explainability

One of the core challenges of making RL safe is making it explainable to the practitioners and end-users. Since safety assurance approaches, in general, compromise the system performance, it is critical to ensure that the human practitioners trust these approaches, lest they ignore them and negate their effectiveness in satisfying the safety requirements. Explainability builds such trust by allowing the practitioner to query the explanations for a given safety decision – in addition to the RL policy itself under this situation. This additional information helps make the safety decisions more actionable for the practitioner and helps build trust in the RL system. Hence, the safety in RL and AI ,in general, becomes a product of explainability and transparency. In this section, we briefly explain our safe and explainable methods for RL agents.

### 3.1 Safe RL Through Shielding and Adversarial Training

We built a safe method called Diversity for Adaptive Safety for RL (DAS-RL) using an adaptive shield mechanism to detect probable safety violations and switch between task policy and safety policy accordingly. The task policy for an optimal RL agent is trained to maximize its reward and thereby accomplish its task objective. Whereas the objective of the safety policy is to maximize agent's safety only without considering the objective function. Safety shield is a function that can identify potential unsafe states responsible for safety constraint violations and minimally manipulates agent's unsafe actions to ensure safety [1]. Unlike [6, 7] which use a safety critic shielding to prevent task policy from safety violation, meanwhile acquires a safety policy by optimizing a Lagrangian Relaxation (LR) function with safety critic approximation, we acquire the safety critic and the safety policy in a completely different manner by training an optimal adversary with respect to the victim agent. Since the complete environment dynamics can not be known prior execution, it is not possible to know how exactly the agents will behave in the real world. However, it is reasonable to assume that we can approximate their most likely behavior. Thus, by training an optimal adversary we can acquire the most likely behavior responsible for safety violations. DAS-RL learns its safety policy by using the optimal adversary's policy as the behavioral basis for the agent's safety violation nature and pushing its behavior away from the adversary's behavior. This allows us to define a safety policy that pushes the agent far away from the policy distribution of the adversary agent. During execution phase the shield mechanism can help the agent to adaptively switch between the task and safety policies. Since the critic function of the adversary actually captures the value of a state based on how likely it is going to violate any safety constraints, we defined our safety shield using the adversary's critic network.

### 3.2 Graphical Summaries for Explaining Learned Policy of RL Agent

In [5], we developed a tool, CAPS, for explaining individual RL policies. CAPS first collects simple natural language (NL) predicates from users that describe potential aspects of the agent's state. It then collects no more than 500 timesteps from the RL agent trajectories. To make the explanation process tractable, CAPS uses a clustering algorithm, CLTree [4], that abstracts the agent's states into a hierarchy of different configurations of clusters ($C$). Each cluster

groups similar states into one abstract state. A heuristic optimization technique selects the best configuration of the clusters, as determined by the accuracy of the state transitions and end-user interpretability. For each cluster, CAPS also identifies whether the agent considers the states in the cluster *safety-critical*. CAPS then forms the agent's policy ($\pi$) and transition function as a directed graph $G = (V, E)$, where the nodes $v$ are the clusters of states, forming abstract states, and the edges $E$ represent the actions chosen by $\pi$ as well as the probability of transitioning from one abstract state to the next. To enrich the graph with more semantic meaning, CAPS labels the abstract states (graph nodes) with concise NL explanations using the user-defined language predicates and Boolean algebra. By controlling the height of the CLTree, CAPS gives the end-user the choice of generating different policy graphs of different sizes, with each size corresponding to different levels of abstraction. Furthermore, we improved upon existing CAPS by computing two additional metric for each node — TS and FP. TS stands for timestep, which indicates the remaining timesteps the agent is expected to finish the episode starting from this abstract state. FP stands for failure probability, which indicates the estimated failure probability of the agent starting from this particular abstract state. In this paper, we use CAPS to explain the RL agent's policy with and without the safety method (DAS-RL) which will help the end-users and practitioners to; (1) understand the agent behavior in safe and unsafe situations, (2) identify the safety-critical states, and (3) identify new safety requirements or refine the existing ones.

## 4 Experiments

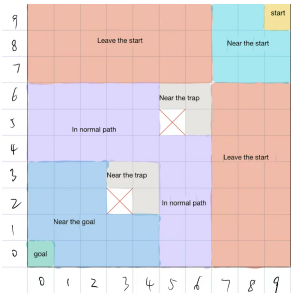
We demonstrate the explainability of RL agent policies using a gridworld environment under a white box attack: strategically-timed attack [3]. In this environment the agent's task is to safely navigate to the goal while avoiding getting into two fixed terminating traps. We divide the gridworld environment into different regions, labeled with human-interpretable predicates Fig.1a. We trained a standard soft actor critic (SAC) [2] algorithm to acquire the task policy. During our experiment we compared the standalone baseline SAC task policy against SAC task policy with DAS-RL safety under strategically-timed attack.

### 4.1 Behavioral Interpretation

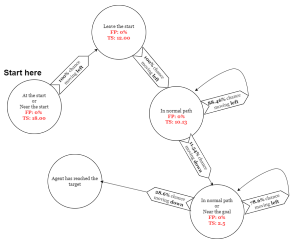
From the transition graph of the SAC optimal task policy without any attack (Fig. 1b) we can see the agent navigate from "start" to "goal" by moving through different "predicates", gradually decreasing the remaining timestep to finish this episodes. Since this is the optimal policy for agent, the failure probability is always 0%. However, the behavior of the same SAC task policy under strategically-timed attack changes significantly (Fig. 1d). In order to analyze the behavior of the agent with respect to external perturbation, we placed the agent at coordinates (4,4) in (Fig. 1a);a position closer to the trap which corresponds to "In normal path" node in CAPS graph. We can observe from the graph that under the attack, agent eventually move into the trap, with failure probability increases to 100%. Finally, from the transition graph of the SAC task policy equipped DAS-RL safety policy (Fig. 1f), we observe that for the same starting positions, the agent can successfully move away from the traps even under strategically-timed attack. As a result, the failure probability significantly decreases, and eventually the agent is able to navigate back to the goal state.

## 5 Conclusion

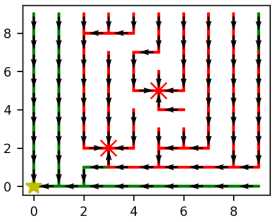
In this study, we use CAPS to study the behavioral pattern of the same SAC agent with and without our proposed DAS-RL safety algorithm under strategically-timed attack. The CAPS interpretation clearly shows the vulnerability of SAC agent without any safety measures which DAS-RL can clearly mitigate.



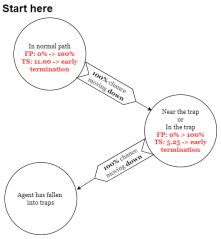

(a) Environment with human-interpretable predicates



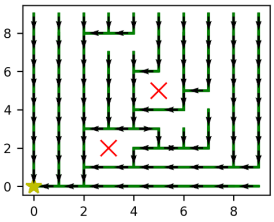

(b) CAPS transition graph of optimal SAC task policy without any attack



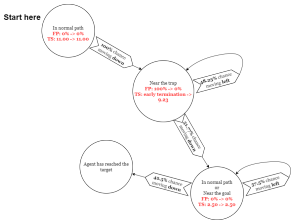
(c) SAC task policy trajectory under strategically-timed attack




(d) CAPS transition graph of optimal SAC task policy under strategically-timed attack



(e) SAC task policy + DAS-RL safety policy trajectory under strategically-timed attack




(f) CPAS transition graph of optimal SAC task policy+DAS-RL safety policy under strategically-timed attack

Figure 1: Experimental results. Environment's details and agents trajectory (left column); agent's transition graph (right column)

## References


[1] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

[3] Y. Lin, Z. Hong, Y. Liao, M. Shih, M. Liu, and M. Sun. Tactics of adversarial attack on deep reinforcement learning agents. *CoRR*, abs/1703.06748, 2017.

[4] B. Liu, Y. Xia, and P. S. Yu. Clustering via decision tree construction. 2004.

[5] J. McCalmon, T. Le, S. Alqahtani, and D. Lee. Caps: Comprehensible abstract policy summaries for explaining reinforcement learning agents. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pp. 889–897, 2022.

[6] K. Srinivasan, B. Eysenbach, S. Ha, J. Tan, and C. Finn. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

[7] B. Thananjeyan, A. Balakrishna, S. Nair, M. Luo, K. Srinivasan, M. Hwang, J. E. Gonzalez, J. Ibarz, C. Finn, and K. Goldberg. Recovery rl: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6(3):4915–4922, 2021.