UNRAVELING EXPLAINABLE REINFORCEMENT LEARNING USING BEHAVIOR TREE STRUCTURES

Kejia Wan[1]; Yuntao Liu[2]; Hengzhu Liu[1] ; Xinhai Xu[2]

[1]National University of Defense Technology
[2]Academy of Military Sciences

Introduction

While deep reinforcement learning (DRL) has shown remarkable proficiency in solving complex problems, its lack of interpretability hinders its safe and scalable application in real-world contexts. To address this issue, Explainable Reinforcement Learning (XRL) has emerged as a valuable approach to enhance the transparency and trustworthiness of machine-based decisions. One prominent approach within XRL is the incorporation of interpretable components, such as decision tree models (DT), to develop efficient and explainable models. However, existing decision tree models often overlook the sequential characteristics of RL tasks and fail to provide comprehensive insights into the decision-making process. This limitation calls for the exploration of alternative methods that can capture the temporal nature of policies.

To tackle these challenges, we propose an innovative framework called XRLBT, which introduces behavior tree structures (BT) to explainable reinforcement learning. XRLBT clusters the state space by aggregating temporally related states, enabling the construction of behavior tree structures aligned with the target DRL model. Unlike previous decision tree models, XRLBT incorporates an exploration technique specifically designed for temporal policies in DRL.

We acknowledges the emergence of the Behavior Tree structure as a fundamental component in capturing the sequential characteristics of RL tasks. We compare the interpretability offered by different methods, including DRL as a black box, decision trees providing consistent explanations, and BTs offering distinct explanations at each time slice, as shown in Figure.1.

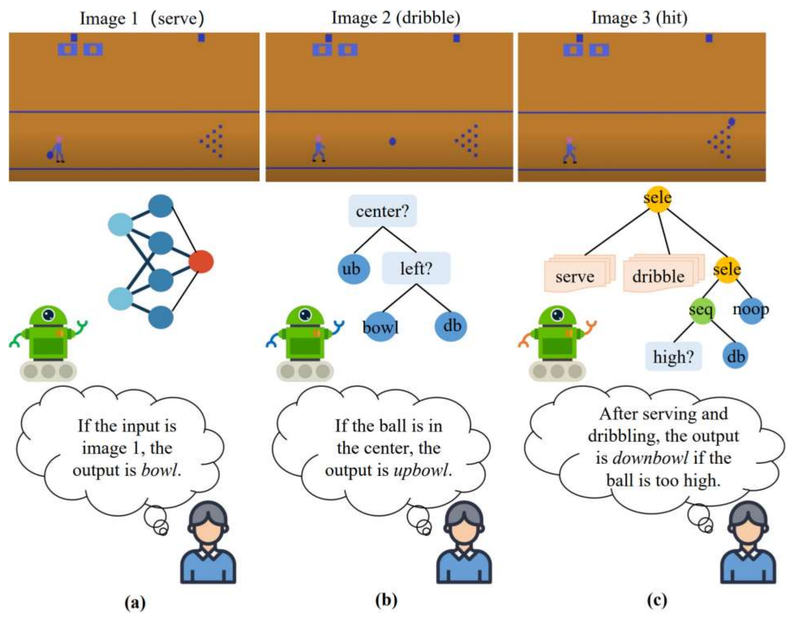

Figure 1. Illustration of explanations of three methods on the bowling. (a) The DRL functions as a black box, providing outputs solely based on the overall input. (b) The DT allows for the explanation based on certain input features, but this explanation remains consistent at every time step. (c) The BT provides distinct explanations at each time slice.

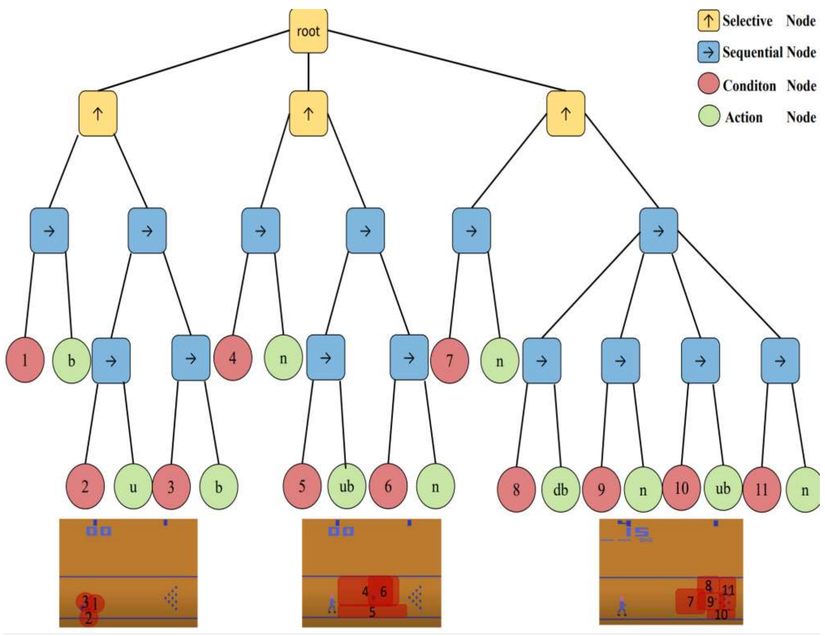

Figure 3. Illustration of a generated BT for the Atari game of BOWLING.

Method

This section outlines the key components and steps involved in the XRLBT framework, as shown in Figure.2.

We begin by introducing the Data Sampler component of XRLBT. We describe the RL agent that takes an N-dimensional state and produces an M-dimensional Q value. The action selector chooses the action corresponding to the largest Q value, and the agent receives a reward from the environment. To generate experiences for training, a buffer pool is used to store tuples comprising the state, action-value function, action, reward, and subsequent state. When the buffer pool reaches its capacity, batches of data are sampled from the buffer for further processing.

Next, we introduce the State Representor component, which addresses the challenge of representing continuous state data in a discrete behavior tree structure. We discuss the use of clustering techniques to identify patterns and relationships within the data. Instead of discretizing the state space via gridding, we propose using clustering based on the temporal dynamics of the data. We define similarity between clusters using the time derivatives of state features, capturing the change in state between timesteps. The K-MEANS function is employed to obtain cluster boundaries, yielding a set of states related to each cluster.

To construct the behavior tree nodes, we describe the process of incorporating the clustered states into the BT structure. We create condition nodes that represent the boundary values of each state class obtained from clustering. By combining these nodes, the behavior trees are formed. It is important to note that the state set used for constructing the behavior tree nodes must contain sequential states, capturing the chain of thoughts similar to human thinking. We discuss the evaluation and testing of the generated BTs, considering the trade-off between performance and tree size.

Result and Case

We conducted extensive experimentation across six benchmark environments to assess the effectiveness of XRLBT. The results showcased exceptional performance of the generated behavior tree structures, as shown in Table.1. The behavior tree structures, constructed based on clustered states, provided comprehensive insights and logical sequential decisions, addressing the challenges of explainability and performance in DRL applications. Comparisons with SOTA algorithms demonstrated the superiority of the proposed framework. The generated behavior tree structures outperformed existing interpretability methods for DRL models, such as decision tree models, in terms of transparency, trustworthiness, and performance.

We delves into the interpretation of a behavior tree (BT) in the context of a specific example, the bowling game.

A BT generated for the bowling game is presented in Figure 3. Upon analyzing the structure of the BT, it becomes apparent that the action nodes tend to cluster, forming three distinct subtrees interconnected by selection nodes. For example, within the right subtree, the left subtree is responsible for continuous movement as long as the bowling ball's position remains within a 10-pixel deviation from the horizontal center. The right subtree promptly adjusts the position downward when the ball's position exceeds the horizontal center by 10 pixels, maintaining this downward adjustment until the position falls below the horizontal center. When the position deviates by 10 pixels from the horizontal center, the subtree triggers an upward adjustment, sustaining it until the bowling pin is successfully hit.

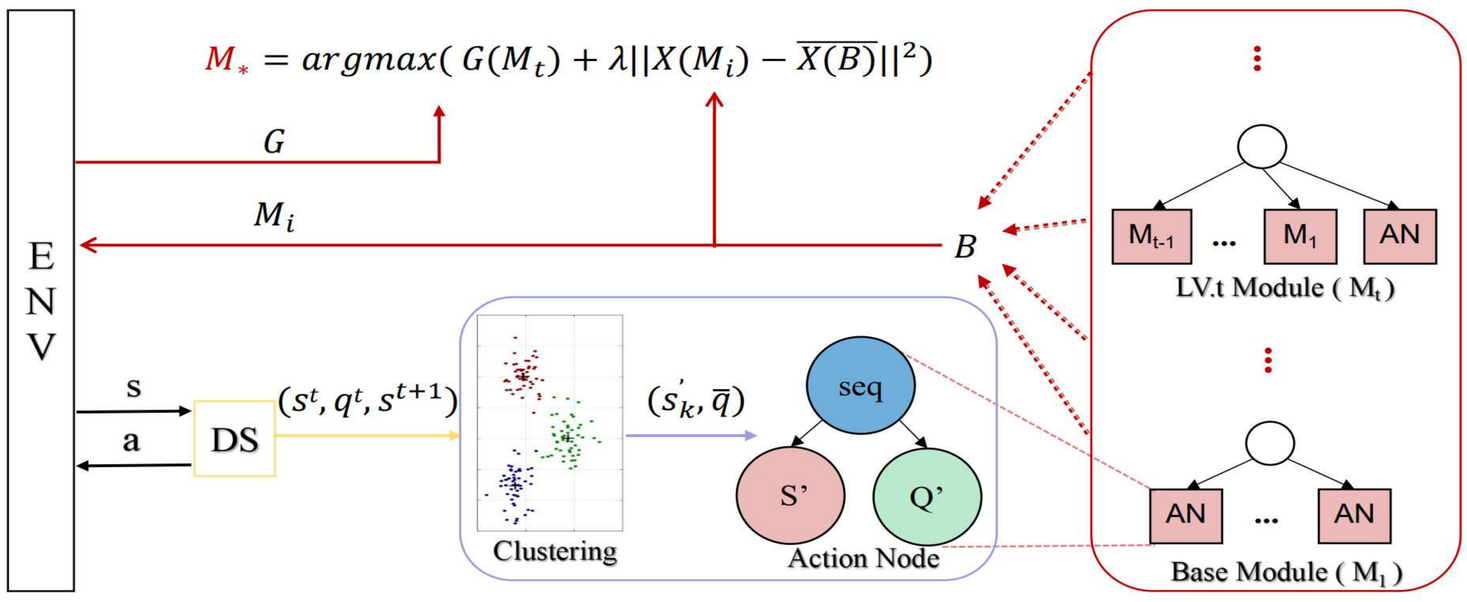

Figure 2. Algorithm overview. (Yellow) The Data Sampler interacts with the environment and stores track. (Purple) The State Representor expresses each state class as a action node by clustering. (Red) The BT Generator combines various modules and evaluates them.

Table 1. The performance comparison of DQN model and four different XRL algorithms based on tree structure. The optimal hyperparameters leading to the best results for the XRLBT method are included in the table.

Game	DQN	Viper	Q-BSP	Tripletree	XRLBT(ours)	Hyperparameters		
						K	W	λ
Pong	17.4	16.5	**17.9**	17.3	**17.9**	4	2	0.52
Freeway	28.9	27.7	29.1	28.7	**29.6**	5	3	0.47
Breakout	377.8	298.9	352.8	373.2	**378.1**	7	3	0.41
Space Invaders	1850	1450	1646	1794	**1825**	9	4	0.39
Bowling	40.6	28.4	35.2	37.8	**39.3**	11	4	0.32
Boxing	73.2	51.9	53.6	59.2	**63.9**	25	7	0.07

Contact

Kejia Wan
National University of Defense Technology
Email: wankejiaai@163.com