From Explainability to Interpretability: Interpretable Reinforcement Learning Via Model Explanations

Anonymous authors
Paper under double-blind review

Keywords: Deep Reinforcement Learning, Interpretable Reinforcement Learning, Explainable Reinforcement Learning, Shapley Values.

Summary

Deep reinforcement learning (RL) has shown remarkable success in complex domains, however, the inherent black box nature of deep neural network policies raises significant challenges in understanding and trusting the decision-making processes. While existing explainable RL methods provide local insights, they fail to deliver a global understanding of the model, particularly in high-stakes applications. To overcome this limitation, we propose a novel model-agnostic framework that bridges the gap between explainability and interpretability by leveraging Shapley values to transform complex deep RL policies into transparent representations. The proposed approach offers two key contributions: a novel approach employing Shapley values to policy interpretation beyond local explanations, and a general framework applicable to off-policy and on-policy algorithms. We evaluate our approach with three existing deep RL algorithms and validate its performance in three classic control environments. The results demonstrate that our approach not only preserves the original models' performance but also generates more stable interpretable policies.

Contribution(s)

1. This paper presents an novel framework to derive interpretable policies from explainable methods.
   Context: Prior work focused on generate explanation in Reinforcement Learning without derive interpretable policy from it. (Beechey et al., 2023)
2. This framework generates highly transparent interpretable policies while maintaining model performance.
   Context: It overturns the conventional assumption that there must be a trade-off between interpretability and performance.
3. This model-agnostic framework is applicable to both off-policy and on-policy reinforcement learning algorithms
   Context: Prior works are mostly model-specific, limiting its ability to generalize across diverse RL scenarios.

From Explainability to Interpretability: Interpretable Reinforcement Learning Via Model Explanations

Anonymous authors
Paper under double-blind review

Abstract

Deep reinforcement learning (RL) has shown remarkable success in complex domains, however, the inherent black box nature of deep neural network policies raises significant challenges in understanding and trusting the decision-making processes. While existing explainable RL methods provide local insights, they fail to deliver a global understanding of the model, particularly in high-stakes applications. To overcome this limitation, we propose a novel model-agnostic framework that bridges the gap between explainability and interpretability by leveraging Shapley values to transform complex deep RL policies into transparent representations. The proposed approach offers two key contributions: a novel approach employing Shapley values to policy interpretation beyond local explanations, and a general framework applicable to off-policy and on-policy algorithms. We evaluate our approach with three existing deep RL algorithms and validate its performance in three classic control environments. The results demonstrate that our approach not only preserves the original models' performance but also generates more stable interpretable policies.

1 Introduction

Reinforcement learning (RL) is an important machine learning technique that learns to make decisions with the best outcomes defined by reward functions (Sutton & Barto, 2018). Recent advances in RL have shown remarkable performance when integrating RL with deep learning to solve challenging tasks with human-level or superior performance in, e.g., AlphaGo (Silver et al., 2017), Atari games (Mnih et al., 2015a), and robotics (Gu et al., 2017). These successes are largely due to the powerful function approximation capabilities of deep neural networks (DNNs), which excel at feature extraction and generalization. However, the use of DNNs also introduces significant challenges as these models are often considered "black boxes", making them difficult to interpret (Zahavy et al., 2016). They are often complex to train, computationally expensive, data-hungry, and susceptible to biases, unfairness, safety issues, and adversarial attacks (Henderson et al., 2018; Wu et al., 2024; Siddique et al., 2020). Thus, an open challenge is to provide quantitative explanations for these models such that they can be understood to gain trustworthiness.

Explainable reinforcement learning (XRL) has become an emerging topic that focuses on addressing the aforementioned challenges, aiming at explaining the decision-making processes of RL models to human users in high-stakes, real-world applications. XRL employs the concepts of interpretability and explainability, each with a distinct focus. Interpretability refers to the inherent clarity of a model's structure and functioning, often achieved through simpler models like decision trees (Bastani et al., 2018; Silva et al., 2020a) or linear functions that make a policy "self-explanatory" (Hein et al., 2018). On the other hand, explainability is related to the use of external, post-hoc methods to provide insights into the behavior of a trained model, aiming to clarify, justify, or rationalize its decisions. Examples include employing Shapley values to determine the importance of state features (Beechey et al., 2023) and counterfactual states to gain an understanding of agent behavior (Olson et al., 2021).

While explainability can provide valuable insights that build user trust, we argue that in high-stakes and real-world applications, explainability alone is insufficient. For instance, Shapley values (Shapley, 1953)—a well-known explainable model—provide local explanations by assigning numerical values that indicate the importance of individual features in specific states. Although such explanations can help users build trust by aligning with human intuition and prior knowledge when enough states are covered, they fail to enable users to fully reproduce or predict agent behavior. This is because these local explanations do not provide a comprehensive, global understanding of the model's functionality, leaving critical aspects of the decision-making process in the dark. In contrast, interpretability offers full transparency and intuitive understanding which is essential for critical applications where trust and comprehensibility are essential. However, the trade-off between simplicity and performance in interpretable models often results in reduced model performance.

Despite its limitations, explainability remains a valuable tool for uncovering insights into model behavior. It can facilitate the development of interpretable policies by abstracting key information from explanations and guiding policy formulation. In this paper, we propose a model-agnostic approach to generate interpretable policies by leveraging insights from explainability techniques in RL environments. This approach aims at balancing transparency and high performance, ensuring that the resulting models are both understandable and effective.

Contributions. In this paper, we present a novel approach that bridges the gap between explainable and interpretable reinforcement learning. Our main contribution is the development of an approach that leverages insights from explainable models to derive interpretable policies. In particular, instead of focusing on the local explanations provided by explainable models, the proposed model-agnostic approach aims to achieve highly transparent and interpretable policies without sacrificing model performance. Additional contributions include the application of the new approach to both off-policy and on-policy RL algorithms and the creation of three adaptations to deep RL methods that learn interpretable policies using insights from model explanation. Finally, we evaluate the effectiveness of our framework in three environments to demonstrate its effectiveness in generating interpretable policies.

2 Related Work

One popular approach used in explainable artificial intelligence (XAI) is to use Shapley values that provide a quantitative measure of the contributions of features to the output (Štrumbelj & Kononenko, 2010; 2014). In (Ribeiro et al., 2016), a method, called LIME, was proposed based on local surrogate models that approximate the predictions made by the original model. In (Wachter et al., 2017), the counterfactual is introduced into XAI by producing a perturbation input to change the original prediction to study the intrinsic causality of the model. In (Lundberg & Lee, 2017), the idea of SHAP was proposed to unify various existing feature attribution methods under a single theoretical framework based on Shapley values, providing consistent and theoretically sound explanations for a wide range of machine learning models.

Most existing explainable methods in RL adopt similar concepts from deep learning via framing the observation as input while the action or reward is the output. In (Beechey et al., 2023), on-manifold Shapley values were proposed to explain the value function and policy that offers more realistic and accurate explanations for RL agents. In (Olson et al., 2021), the counterfactual state explanations were developed to examine the impact of altering a state image in an Atari game to understand how these changes influence action selection. As RL possesses some unique challenges, such as sequential decision-making under a reward-driven framework, specialized methods have been considered for its explanation. For example, in (Juozapaitis et al., 2019), reward decomposition was proposed to break down a single reward into multiple meaningful components, providing insights into the factors influencing an agent's action preferences. Moreover, understanding the action selection in certain critical states of the entire sequence can enhance user trust (Huang et al., 2018). A summary of important yet not similar sets of states (trajectories) can provide a broader and more comprehensive view of agent behavior (Amir & Amir, 2018).

In contrast to the XRL, research in interpretable RL usually focuses on the transparency of the decision-making processes via, e.g., a simple representation of policies that are understandable to non-experts. The corresponding studies can be divided into direct and indirect approaches (Glanois et al., 2024). The direct approach aims to directly search a policy in the environment using the policy deemed interpretable by the designer or user. Examples of the direct methods include the use of decision tree (Silva et al., 2020b) or a simple closed-form formula (Hein et al., 2018) to represent the policy. The direct approach usually requires a prior expert knowledge for initialization to achieve good performance, often for small-scale problems. On the other hand, the indirect approach provides more flexibility by employing a two-step process: (1) train a non-interpretable policy with efficient RL algorithms, and (2) convert this non-interpretable policy into an interpretable one. For instance, Bastani et al. (2018) proposed VIPER, a method to learn high-fidelity decision tree policies from original DNN policies. Similarly, Verma et al. (2018) proposed PIRL, a method that presents a way to transform the neural network policy into a high-level programming language. Our proposed methods can be categorized into indirect interpretable approaches using Shapley values to transform original policies into simpler but rigorous closed-form function policies. Distinguishing ourselves from existing indirect interpretation approaches, we uniquely incorporate the Shapley value explanation method to generate more accurate and generalizable interpretable policy without relying on predefined interpretable structures.

3 Background

3.1 Reinforcement Learning

In Reinforcement Learning, an agent interacts with its environment, which is modeled as a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, d_0)$, where $\mathcal{S}$ is the set of states and $\mathcal{A}$ is the set of possible actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\gamma \in [0, 1]$ is discount factor, and $d_0 : \mathcal{S} \to [0, 1]$ specifies the initial state distribution. At time step t, the agent observes the current state $s_t \in \mathcal{S}$ and performs an action $a_t \in \mathcal{A}$. In response, the environment transitions to a new state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ and provides a reward r_{t+1}. The agent's objective is to learn a policy (i.e., strategy) π that maximizes the expected return $\mathbb{E}_\pi[G_t]$, where $G_t = \sum_{n=t}^{\infty} \gamma^n r_{n+1}$. In RL, policies can be deterministic $\pi : \mathcal{S} \to \mathcal{A}$ or stochastic $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$. consider an environment with n state features, where $\mathcal{S} = \mathcal{S}_1 \times ... \times \mathcal{S}_n$, and each state can be represented as an ordered set $s = \{s_i | s_i \in \mathcal{S}_i\}_{i=1}^n$. Using $N = \{1, ..., n\}$ to represent the set of all state features, a partial observation of the state can be denoted as the ordered set $s_C = \{s_i | i \in C\}$ where $C \subset N$.

3.2 Shapley Values in Reinforcement Learning

The *Shapley value* (Shapley, 1953) is a method from cooperative game theory that distributes credit for the total value $v(N)$ earned by a team N among its players. It is defined as

$$\phi_i(v) = \sum_{C \subseteq N \setminus \{i\}} \frac{|C|!(n - |C| - 1)!}{(n!)} [v(C \cup \{i\}) - v(C)], \tag{1}$$

where $v(C)$ represents the value generated by a coalition of players C. The Shapley value $\phi_i(v)$ is the average marginal contribution of player i when added to all possible coalitions C.

In RL, the state features $\{s_1, ..., s_n\}$ can be treated as players, and the policy output $\pi(s)$ can be viewed as the total value generated by their contributions. To compute the Shapley values of these players, it is essential to define a characteristic function $v(C)$ that reflects the model's output for a coalition of features $s_C \subseteq s_1, \ldots, s_n$.

As the trained policy is undefined for partial input s_C, it is important to correctly define the characteristic function for accurate Shapley values calculation. Following the *on-manifold* characteristic value function (Frye et al., 2021; Beechey et al., 2023), we account for feature correlations rather than assuming independence.

For a deterministic policy $\pi : S \to A$, which outputs actions, the characteristic function is defined as

$$v^{\pi}(C) := \pi_C(s) = \sum_{s' \in S} p^{\pi}(s'|s_c)\pi(s'), \tag{2}$$

where $s' = s_C \cup s'_{\bar{C}}$ and $p^{\pi}(s'|s_C)$ is the probability of being in state s' given the limited state features s_C is observed following policy π. Similarly, for a stochastic policy $\pi : S \times A \to [0, 1]$, which outputs action probabilities, the characteristic function is defined as

$$v^{\pi}(C) := \pi_C(a|s) = \sum_{s' \in S} p^{\pi}(s'|s_c)\pi(a|s'). \tag{3}$$

4 Method

In this section, we present our proposed methods in two main parts. First, Shapley vectors analysis focuses on extracting and capturing the underneath patterns provided by Shapley values. Secondly, interpretable policy formulation focuses on utilizing these patterns to construct interpretable policies with comparable performance. The complete algorithm is provided in Algorithm 1.

4.1 Shapley Vectors Analysis

Given a well-trained policy $\pi(s)$ (deterministic) or $\pi(a|s)$ (stochastic) in RL, Shapley values provide a way to explain the policy's behavior by quantifying the contributions of state features to the RL policy. Following the Shapley values methods (Beechey et al., 2023), we substitute (2) or (3) into the Shapley value formula, namely, (1), to compute $\phi_i(v^{\pi})$, *i.e.,* the contribution of feature i to the policy under state s.

The computed Shapley values $\phi_i(v^{\pi})$ provide insight into how each state feature i influences action selection. For example, in an environment with two discrete actions, $a_1 = -1$ and $a_2 = 1$. After computing the Shapley value $\phi_i(v^{\pi})$, a positive $\phi_i(v^{\pi})$ indicates that the feature i encourages the selection of a_2, while a negative value suggests a preference for a_1. Notably, Shapley values generalize across features; state features contributing equally to a decision will yield identical values, revealing symmetry in policy reasoning. In this paper, we take this property of Shapley values as their generalization ability.

To exploit this generalization, we represent each state s as a Shapley vector composed of contributions from all features given by

$$\Phi_s = (\phi_1, ..., \phi_n). \tag{4}$$

This enables us to cluster the states with similar action selection behavior which further gives insights into action-group boundaries.

4.1.1 Action K-Means Clustering.

To cluster states based on their Shapley vectors, we employ action K-means clustering. Given a set of states $(s_1, s_2, ..., s_m)$, where each state is represented by a n-dimensional Shapley vector $(\phi_1, \phi_2, ..., \phi_n)$, the algorithm partitions these states into k clusters $A = A_1, A_2, \ldots, A_k$, where k is the number of discrete actions in the environment. The clustering objective is to minimize inter-cluster variance given by

$$\arg\min_{A} \sum_{i=1}^{k} \sum_{\Phi_s \in A_i} \|\Phi_s - \boldsymbol{\mu}_i\|^2, \tag{5}$$

where $\boldsymbol{\mu}_i$ is the centroid of points in A_i, usually represented as $\boldsymbol{\mu}_i = \frac{1}{|A_i|} \sum_{\Phi_s \in A_i} \Phi_s$.

Algorithm 1 Shapley Vector Decision Boundary Algorithm

Input: Shapley vectors $(\Phi_{s_1}, \Phi_{s_2}, ..., \Phi_{s_m})$, Original states $(s_1, s_2, ..., s_m)$
Parameter: Action numbers k
Output: Decision Boundary functions $\{f_{ij}\}$ for each pair of actions (i, j)

Initialize empty set of boundary points $B = \{\}$
$A = \{A_1, ..., A_k\} \leftarrow$ Action KMeans$(\{\Phi_{s_i}\}_{i=1}^{m}, k)$
for $i = 1$ to k **do**
  $\boldsymbol{\mu}_i \leftarrow \frac{1}{|A_i|} \sum_{\Phi \in A_i} \Phi$
end for
for $i = 1$ to $k - 1$ **do**
  for $j = i + 1$ to k **do**
    $X_{ij} \leftarrow \arg\min_X (||X - \boldsymbol{\mu}_i||^2 - ||X - \boldsymbol{\mu}_j||^2)$
    $B \leftarrow B \cup \{X_{ij}\}$
    $s_{ij} \leftarrow \phi^{-1}(X_{ij})$
  end for
end for
for each pair of clusters (i, j) **do**
  $f_{ij}(s) \leftarrow$ Regression(s_{ij})
end for
return $\{f_{ij}\}$

4.1.2 Boundary Point Identification.

Once clusters are formed, the boundaries between action regions can be identified using boundary points. A *boundary point* X exists at the interface of two clusters A_i and A_j, where the policy is equally likely to select either action. This condition arises when the policy is not sure which action to take at the current state, and therefore can serve as a boundary decision. Formally, X is found by minimizing the difference between distances to cluster centroids

$$\arg\min_X \left(||X - \boldsymbol{\mu}_i||^2 - ||X - \boldsymbol{\mu}_j||^2 \right), \tag{6}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ are the centroid of points in A_i and A_j, respectively.

Property 1 (Existence and Uniqueness of Decision Boundaries)**.** *For a stationary deterministic policy π within an MDP, characterized by a fixed state distribution $d_\pi(s)$, there exists a unique boundary surface in the Shapley vector space such that*

1. *the boundary separates the Shapley vectors associated with distinct discrete actions; and*
2. *the Euclidean distance from any action's Shapley vector to this boundary remains constant across all states under the stationary policy.*

Proof. The efficiency property of Shapley values ensures that the sum of contributions from all features equals the difference between the policy's action value for state s and the expected action value across states, *i.e.*,

$$\sum_{i=1}^{n} \phi_i = \pi(s) - \mathbb{E}_S(\pi(S)). \tag{7}$$

For states s_p and s_q that lead to different action selection $\pi(s_p) = a_p$ and $\pi(s_q) = a_q$, where $a_p \neq a_q$, the difference between their action values defines a gap given by

$$|\pi(s_p) - \pi(s_q)| = |a_p - a_q| = \Delta a. \tag{8}$$

Given that the policy π is stationary with a fixed state distribution $\mu(s)$, the expected action value converges to a fixed scalar value given by

$$\mathbb{E}_{S\sim\mu}[\pi(S)] = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\pi(s) = \bar{a}. \tag{9}$$

By substituting (8) and (9) into the efficiency property (7), the Shapley value that sums for all states satisfy a gap

$$\left|\sum_{i=1}^{n}\phi_{i,s_p} - \sum_{i=1}^{n}\phi_{i,s_q}\right| = \Delta a, \forall s_p, s_q \in \mathcal{S}, \tag{10}$$

where $\pi(s_p) = a_p \neq \pi(s_q) = a_q$. This implies that the gap Δa exists between all states with different action selections. Consequently, we defined the boundary surface $\mathcal{B}$ in the Shapley vector space as

$$\mathcal{B} = \left\{\vec{v} \in \mathbb{R}^n \middle| \sum_{i=1}^{n} v_i = \bar{a} + \frac{\Delta a}{2}\right\}. \tag{11}$$

The distance from any Shapley vector plane Φ to this boundary surface $\mathcal{B}$ is given by

$$\text{dist}(\Phi_s, \mathcal{B}) = \frac{\sum_{i=1}^{n}\phi_i - \sum_{i=1}^{n} v_i}{\sqrt{n}}. \tag{12}$$

Therefore, for all states, $s_p, s_q \in \mathcal{S}$, the distances from their Shapley vectors to the boundary remain constant:

$$\text{dist}(\Phi_{s_p}, \mathcal{B}) = \text{dist}(\Phi_{s_q}, \mathcal{B}) \tag{13}$$

This proves the existence and uniqueness of the decision boundary in the Shapley vector space. □

The constant distance between the boundary surface and Shapley vector plane lays the foundation for an interpretable policy that maps each action region to its corresponding state region.

4.2 Interpretable Policy Formulation

With the decision boundary point's identification in the Shapley vector space, the next step is to map it back to the original state space to construct an interpretable policy.

4.2.1 Inverse Shapley Values.

To reconstruct the decision boundary in the state space, we model it as the *Inverse Shapley Value Problem* $\phi_i^{-1} : \phi_i(v) \to \{i\}$, where the goal is to recover the original state s corresponding to a given Shapley vector Φ_s. We address this problem by systematically storing the original states with their corresponding Shapley value vectors, enabling efficient inverse function operations. It allows us to map Shapley value vectors back to their original states directly, facilitating precise reconstruction of the decision boundary.

4.2.2 Decision Boundary Regression.

After the boundary state points s_{ij} are discovered using Shapley values, the decision can be drawn accordingly. While a variety of regression techniques can be used, we use linear regression due to its simplicity and interpretability. The resulting boundary functions f_{ij} define the action regions.

This policy is then reformulated by assigning actions based on the regions characterized by boundary functions. Specifically, for a given state s, the action a is determined by the cluster in which s resides relative to f_{ij}.

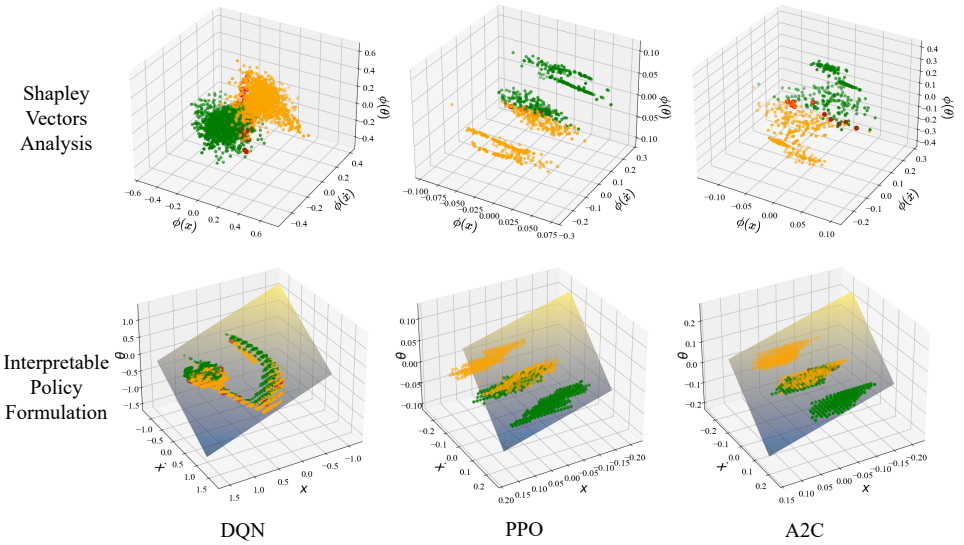


Figure 1: Visualization of Shapley values and interpretable policy formulation in the CartPole. The first row depicts the Shapley value vectors for DQN, PPO, and A2C, with clusters represented in different colors and boundary points highlighted in red. The second row illustrates the corresponding interpretable policy in the original state space, showing decision boundaries that separate the state space into distinct action regions. (Due to the limitations of dimensional plotting, only the first three features $x, \dot{x}, \theta$ are visualized in the figure)

Algorithm	**Decision Boundary**
DQN	$f_{01} = -0.5x - 0.687\dot{x} - 1.09\theta - \dot{\theta} - 0.018$
PPO	$f_{01} = -0.193x - 0.523\dot{x} - \theta - \dot{\theta} + 0.0014$
A2C	$f_{01} = -0.4875x - 0.9811\dot{x} - 1.09\theta - \dot{\theta}$

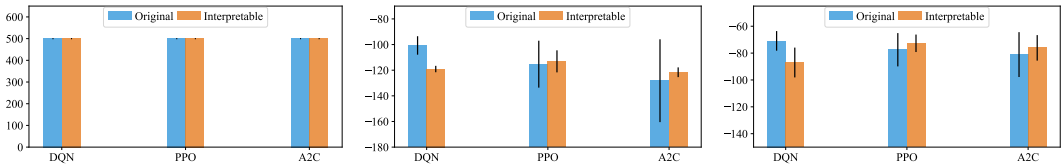
Table 1: CartPole interpretable policy boundary

5 Experiments

To evaluate the effectiveness of our proposed method, we performed experiments across three classical control environments from Gymnasium (Towers et al., 2024): CartPole and MountainCar. These environments were specifically chosen as they represent an important control problem where policy interpretability is crucial for real-world deployment. To demonstrate the generality of our framework, we applied it to both off-policy and on-policy deep RL algorithms. Specifically, we applied it to Deep Q-Network (DQN) (Mnih et al., 2015b) as an off-policy method, and Advantage Actor-Critic (A2C) (Mnih et al., 2016) and Proximal Policy Optimization (PPO) (Schulman et al., 2017) as on-policy methods. Our experimental results demonstrate that the interpretable policies generated by our method perform competitively to those of deep RL algorithms, and also exhibit better stability and broad applicability.

5.1 CartPole

The CartPole environment is a classic control problem in which an inverted pendulum is placed on the movable cart. The state space in this environment consists of four features: position of cart x, velocity of cart $\dot{x}$, angle between the pendulum and the vertical θ, and angular velocity of pendulum $\dot{\theta}$. The action space includes two discrete actions, where the first action 0 means push the cart to the



(a) CartPole (b) MountainCar (c) Acrobot

Figure 2: Performances of the interpretable policy with original algorithms—DQN, PPO, A2C in different environments.

left, and the second action 1 means push to the right. A reward of +1 is assigned for each timestep the pole remains upright. The goal in this environment is to balance the pendulum by applying forces in the left and right direction on the cart.

As explained in the method (Section 4), our goal is to obtain an interpretable policy for this problem. To achieve this, we first train three deep RL methods, namely DQN, PPO, and A2C to obtain the optimal policies. Once the models were trained, we evaluated their performance in the CartPole environment and sampled state distributions from 100 trajectories for each algorithm. For each sampled state, we computed the Shapley values of its features using Equation (1). With this step, we construct a Shapley value vector Φ_s that represents the contribution of state features to this policy's decision. The first row of Figure 1, illustrates the Shapley value vectors for DQN, PPO, and A2C, respectively. Using these Shapley values, we performed k-means clustering on the action space to identify cluster centroids, where each cluster represents a distinct action region. Each cluster is depicted in a different color. We then identified boundary points, which are shown in red in the first row of Figure 1. These boundary points indicate the transition between action regions.

Next, we reconstructed the decision boundary in the original state space using the boundary points identified in the Shapley vector space. The second row of Figure 1 shows these boundaries in the state space for each algorithm. Finally, as described in the methodology, we applied linear regression to derive an interpretable policy f_{ij}. The interpretable policies for DQN, PPO, and A2C are summarized in Table 1. These policies are obtained through their boundaries which separate the states into different action selection regions. In other words, the decision rule for these policies is: if $f_{01} > 0$, select action 0; otherwise, select action 1. This interpretable policy framework is fully transparent, enabling reproducibility and mitigating risks in high-stakes real-world applications.

To evaluate the performance of the interpretable policies, we tested them alongside the original deep RL policies over 10 episodes. The results, shown in Figure 2a, demonstrate that the interpretable policies consistently achieved the maximum reward of 500 across all algorithms. This indicates that our method preserves the performance of the original deep RL algorithms while providing interpretability. These results also highlight the generality and model-agnostic nature of the proposed framework.

5.2 MountainCar

The MountainCar environment is another classic control problem where a car is placed at the bottom of a sinusoidal valley. The state space for this environment consists of two features: car position along the x-axis x and the velocity of the car $\dot{x}$. The actions space contains two discrete actions: action 0 applies left acceleration on the car and action 1 applies right acceleration on the car. The goal of this environment is to accelerate the car to reach the goal state on top of the right hill. A reward of -1 is assigned for each timestep as punishment if the car fails to reach the goal state.

Following the proposed method (section 4), we perform the Shapley vectors analysis in three trained deep RL methods DQN, PPO, and A2C in the MountainCar environment. The result is shown in the first row of Figure 3. Each cluster represents a distinct action region, distinguished by a unique color and boundary points are highlighted in red. By mapping these boundary points back to the

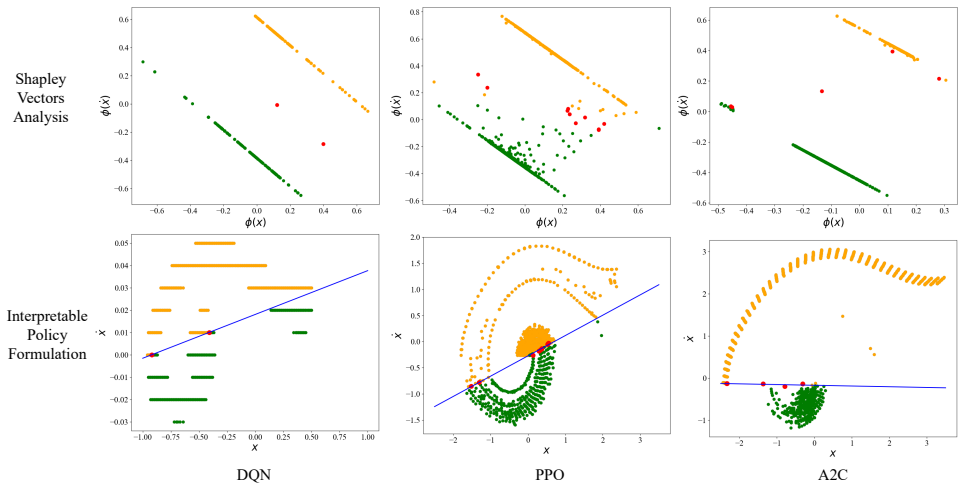


Figure 3: Visualization of Shapley values and interpretable policy formulation in the MountainCar. The first row depicts the Shapley value vectors for DQN, PPO, and A2C, with clusters represented in different colors and boundary points highlighted in red. The second row illustrates the corresponding interpretable policy in the original state space, showing decision boundaries that separate the state space into distinct action regions.

Algorithm	**Decision Boundary**
DQN	$f_{01} = 0.013x - \dot{x} + 0.0033$
PPO	$f_{01} = 0.35x - \dot{x} - 0.3$
A2C	$f_{01} = 0.003x - \dot{x} - 0.12$

Table 2: MountainCar interpretable policy boundary

original state space, we constructed the decision boundaries using linear regression, illustrated in the second row of Figure 3 as blue lines. The detailed interpretable policies for DQN, PPO, and A2C are in Table 2 and the decision rule is straightforward: when $f_{01} > 0$, action 0 is chosen, otherwise, action 1 is chosen.

Performance of the interpretable policies alongside the original algorithms was evaluated over 10 episodes, with results presented in Figure 2b. Interestingly, interpretable policies derived from PPO and A2C surprisingly outperformed their original algorithms, whereas the interpretable policy generated from DQN experienced a slight performance reduction. A notable observation is that all interpretable policies achieved significantly smaller standard deviations compared to their original counterparts, indicating more stable policy performance. This characteristic is particularly valuable in real-world applications where consistent and predictable behavior is crucial.

5.3 Acrobot

The Acrobot environment is a challenging classic control problem in which a two-link pendulum has its second joint actuated. The state space in this environment consists of six features: cosine of the first joint angle $\cos(\theta_1)$, sine of the first joint angle $\sin(\theta_1)$, cosine of the second joint angle $\cos(\theta_2)$, sine of the second joint angle $\sin(\theta_2)$, angular velocity of the first joint $\dot{\theta}_1$, and angular velocity of the second joint $\dot{\theta}_2$. The action space includes three discrete actions: action 0 for a torque of -1, action 1 for a torque of 0, and action 2 for a torque of +1. A reward of -1 is assigned for each timestep until the goal is achieved. The goal in this environment is to swing the end-effector of the

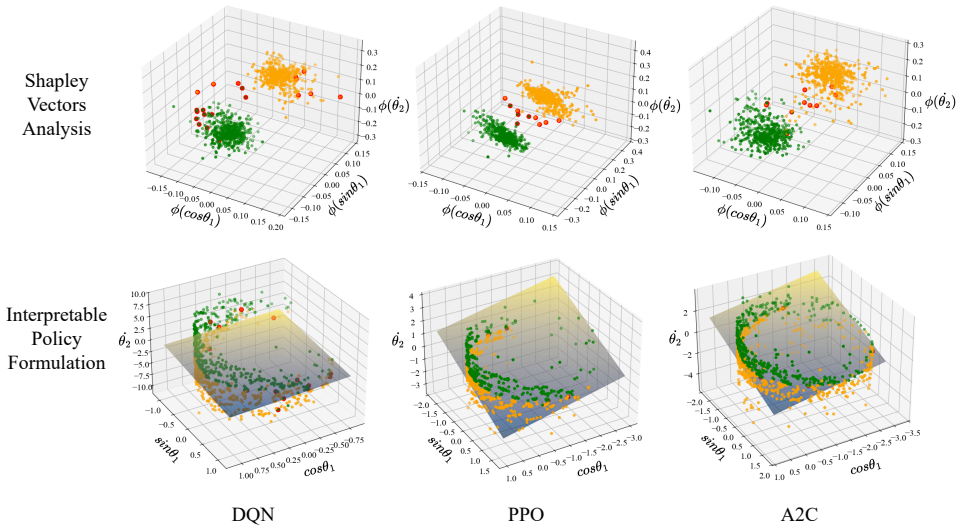


Figure 4: Visualization of Shapley values and interpretable policy formulation in the Acrobot. The first row depicts the Shapley value vectors for DQN, PPO, and A2C, with clusters represented in different colors and boundary points highlighted in red. The second row illustrates the corresponding interpretable policy in the original state space, showing decision boundaries that separate the state space into distinct action regions (Due to the limitations of dimensional plotting, only the first three features $\cos\theta_1, \sin\theta_1, \dot{\theta_2}$ are visualized in the figure.).

Algorithm	**Decision Boundary**
DQN	$f_{01} = 1.79cos\theta_1 - 1.86sin\theta_1 - 1.46cos\theta_2 + 0.28sin\theta_2 - 0.69\dot{\theta_1} - \dot{\theta_2} + 0.44$
PPO	$f_{01} = 0.06cos\theta_1 - 0.43sin\theta_1 - 0.04cos\theta_2 - 0.01sin\theta_2 - 0.36\dot{\theta_1} - \dot{\theta_2} - 0.01$
A2C	$f_{01} = 0.03cos\theta_1 - 0.19sin\theta_1 - 0.01cos\theta_2 - 0.54sin\theta_2 - 1.15\dot{\theta_1} - \dot{\theta_2} + 0.01$

Table 3: Acrobot interpretable policy boundary

pendulum to a height where the condition $-\cos(\theta_1) - \cos(\theta_1 + \theta_2) > 1.0$ is satisfied, at which point the episode terminates; otherwise, the episode ends after 500 steps.

Shapley vectors analysis is performed on three trained deep RL methods DQN, PPO and A2C in the Acrobot environment to find the action clusters, which is shown in the first row of Figure 4. We then constructed the decision boundary using linear regression, represented as the hyperplanes in the second row of Figure 4. The detailed interpretable policies are in Table 3 and the decision rule is straightforward: when $f_{01} > 0$, action 0 is chosen, otherwise, action 1 is chosen.

Performance of the interpretable policies and original algorithms is presented in Figure 2c. Interpretable policies derived from PPO and A2C outperformed the original algorithms with more stable performance across the 10 episodes evaluations, while interpretable policy obtained from DQN experienced a slight performance reduction.

5.4 Fidelity Score

To evaluate the behavior difference between interpretable policy and original policy, we introduce a straightforward fidelity function to quantify it:

$$F(\pi_{interp}, \pi_{orig}) = \frac{1}{|S|} \sum_{s \in S} \mathbb{1}\{\pi_{interp}(s) = \pi_{orig}(s)\}, \tag{14}$$

where the π_{interp} is the interpretable policy, π_{orig} represents the original policy and $\mathbb{1}\{\cdot\}$ is the indicator function. This fidelity score is equivalent to the accuracy when treating the original policy as the ground truth.

Fidelity Score (%)	**A2C**	**PPO**	**DQN**
Cartpole	76.75 ± 1.70	83.70 ± 0.53	50.08 ± 0.01
MountainCar	98.14 ± 0.30	98.75 ± 0.03	89.47 ± 0.25
Acrobot	99.15 ± 0.06	96.93 ± 0.09	78.62 ± 0.39

Table 4: Fidelity scores by environments and algorithms

The fidelity scores across all environments and algorithms are shown in Table 4. The fidelity scores highly correlated with the performance. For Cartpole environment, its intrinsic simplicity lowers the requirement of fidelity scores, which means low fidelity score can still yield high performance. For MountainCar and Acrobot, due to high complexity in these environment, only high fidelity scores can obtain high performance. In other words, interpretable policies derived from A2C and PPO perform better than that derived from DQN.

6 Conclusions and Future Work

In this paper, we formalized and addressed the unsolved problem of extracting interpretable policies from explainable methods in RL. We propose a novel approach that leverages Shapley values to generate transparent and interpretable policies for both off-policy and on-policy deep RL algorithms. Through comprehensive experiments conducted in three classic control environments using three deep RL algorithms, we demonstrated that our proposed method achieves comparable performance while generating interpretable and stable policies.

Potential future work includes: (1) extending the current approach to continuous action spaces by discretizing the action space, (2) conducting a scalability study of the proposed approach in more complex environments with high-dimensional state feature spaces, and (3) exploring performance differences across various regression methods.

References

Dan Amir and Ofra Amir. Highlights: Summarizing agent behavior to people. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pp. 1168–1176, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction. *Advances in neural information processing systems*, 31, 2018.

Daniel Beechey, Thomas MS Smith, and Özgür Şimşek. Explaining reinforcement learning with shapley values. In *International Conference on Machine Learning*, pp. 2003–2014. PMLR, 2023.

Christopher Frye, Damien de Mijolla, Tom Begley, Laurence Cowton, Megan Stanley, and Ilya Feige. Shapley explainability on the data manifold. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OPyWRrcjVQw.

Claire Glanois, Paul Weng, Matthieu Zimmer, Dong Li, Tianpei Yang, Jianye Hao, and Wulong Liu. A survey on interpretable reinforcement learning. *Machine Learning*, pp. 1–44, 2024.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation*, pp. 3389–3396. IEEE, 2017.

Daniel Hein, Steffen Udluft, and Thomas A Runkler. Interpretable policies for reinforcement learning by genetic programming. *Engineering Applications of Artificial Intelligence*, 76:158–169, 2018.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Sandy H Huang, Kush Bhatia, Pieter Abbeel, and Anca D Dragan. Establishing appropriate trust via critical states. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 3929–3936. IEEE, 2018.

Zoe Juozapaitis, Anurag Koul, Alan Fern, Martin Erwig, and Finale Doshi-Velez. Explainable reinforcement learning via reward decomposition. In *IJCAI/ECAI Workshop on explainable artificial intelligence*, 2019.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015a.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015b.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/mniha16.html.

Matthew L. Olson, Roli Khanna, Lawrence Neal, Fuxin Li, and Weng-Keen Wong. Counterfactual state explanations for reinforcement learning agents via generative deep learning. *Artificial Intelligence*, 295:103455, 2021. ISSN 0004-3702. DOI: https://doi.org/10.1016/j.artint.2021.103455. URL https://www.sciencedirect.com/science/article/pii/S0004370221000060.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. DOI: 10.1145/2939672.2939778. URL https://doi.org/10.1145/2939672.2939778.

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

Lloyd S Shapley. A value for n-person games. *Contribution to the Theory of Games*, 2, 1953.

Umer Siddique, Paul Weng, and Matthieu Zimmer. Learning fair policies in multi-objective (deep) reinforcement learning with average and discounted rewards. In *International Conference on Machine Learning*, pp. 8905–8915. PMLR, 2020.

Andrew Silva, Matthew Gombolay, Taylor Killian, Ivan Jimenez, and Sung-Hyun Son. Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 1855–1865. PMLR, 26–28 Aug 2020a. URL https://proceedings.mlr.press/v108/silva20a.html.

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, 2020b.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

Erik Štrumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11(1):1–18, 2010. URL http://jmlr.org/papers/v11/strumbelj10a.html.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41:647–665, 2014.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

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

Abhinav Verma, Vijayaraghavan Murali, Rishabh Singh, Pushmeet Kohli, and Swarat Chaudhuri. Programmatically interpretable reinforcement learning. In *International Conference on Machine Learning*, pp. 5045–5054. PMLR, 2018.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

Mingkang Wu, Umer Siddique, Abhinav Sinha, and Yongcan Cao. Offline reinforcement learning with failure under sparse reward environments. In *2024 IEEE 3rd International Conference on Computing and Machine Intelligence (ICMI)*, pp. 1–5. IEEE, 2024.

Tom Zahavy, Nir Ben-Zrihem, and Shie Mannor. Graying the black box: Understanding dqns. In *International conference on machine learning*, pp. 1899–1908. PMLR, 2016.