Predicate-based explanation of Reinforcement Learning

Léo Saulières ;*, Martin C. Cooper and Florence Dupin de Saint-Cyr

IRIT, University of Toulouse III, France

Abstract. For the purpose of understanding the impact of a Reinforcement Learning (RL) agent's decisions on the satisfaction of a given arbitrary predicate, we present a method based on the evaluation of the importance of actions. This highlights to the user the most important action(s) in a history of the agent's interactions with the environment. Having shown that calculating the action importance for a predicate to hold is #W[1]-hard, we propose a time-saving approximation. To do so, we use the most likely transitions in the environment. Experiments confirm the relevance of this approach.

1 Introduction

In the last decade, the use of neural networks has led to more powerful Artificial Intelligence models, but these were also complex, akin to black-boxes. To increase user confidence, a need for explanation arose, highlighted by researchers [3, 2] and legislators [5, 6]. Accordingly, Explainable Artificial Intelligence (XAI) is a field dedicated to the explanation of 'black box' models.

RL is a Machine Learning paradigm where an agent learns to make a sequence of actions within an environment. Given a state, the agent chooses an action at each time-step, arrives in a new state and receives a reward, determined by the environment dynamics. The agent's goal is to maximize its reward by learning an optimal policy.

EXplainable Reinforcement Learning (XRL) is a sub domain of XAI which aims to provide explainers for RL models. Our XRL method focuses on the explanation of past state-action sequences, which we call histories. The aim is to understand the impact of an action on the achievement of a given predicate. A predicate can reflect the agent's success, failure or problem-specific properties. To address this question, an importance score is computed for each action in the history. Then, the most important action(s) and corresponding state(s) are displayed to the user.

2 History-Explanation based on Predicates (HXP)

Before diving into the HXP method, we need to introduce some RL notation. A Markov Decision Process (MDP) describes an RL problem [9]. An MDP is a tuple $\langle S, A, R, p\rangle$ where S and A correspond respectively to the set of states and actions, $R : S \times A \rightarrow \mathbb{R}$ represents the reward function and $p : S \times A \rightarrow Pr(S)$ is the transition function of the environment. Given an action $a \in A$ and a state $s \in S$, $p(s'|s, a)$ is the probability to reach the state s' from s by doing a. In this paper, we restrict our explanations to deterministic policies $\pi : S \rightarrow A$, which map to each state s an action a. $\pi(s)$ denotes the action to do in s given the policy π.

Given a history reflecting the agent's behavior and a predicate d, our aim is to answer this question: *"Which actions were important to ensure that d was achieved, given the agent's policy π?"* by computing an importance score for each action in the history. The importance score represents the benefit (in terms of achieving the predicate) of performing an action a in a state s instead of another action $a' \in A(s)/\{a\}$ in s. Accordingly, the utility of doing an action a from a state s, relative to the achievement of predicate d, must be measured. As the impact of an action is not necessarily in the short term, our basic idea is: generate the set of length-k scenarios starting with action a in state s, then compute the probability of reaching a final state at horizon k in which d is valid. In this context, a scenario is a hypothetical state-action sequence generated in order to calculate the utility of the action and k is a constant (set, by default, to the history length whatever the position of the action in the history).

Scenarios are generated by using the agent's policy π at each time-step, and exploring each possible transition allowed by p. We use a recursive function, denoted $succ^k$, which returns the set of reachable states and corresponding probabilities at depth k from a given initial state. The probability of a state is the product of the probabilities along the current path to this state according to p.

Definition 1. *Given a transition function p, a set of (state, probability) pairs S_p and a policy π, $next_\pi$ is defined as follows:*

$$next_\pi(S_p, p) = \left\{ (s', pr \times p(s'|s,a)) \;\middle|\; \begin{array}{l} (s,pr) \in S_p \\ a = \pi(s) \\ p(s'|s,a) \neq 0 \end{array} \right\}$$

We can now define $succ^n_\pi$ as follows, where s_0 is the inital state:

$$succ^0_\pi(s_0, p) = \{(s_0, 1)\}$$
$$succ^{n+1}_\pi(s_0, p) = next_\pi(succ^n_\pi(s_0, p), p)$$

The utility of a set of (state, probability) pairs relative to a predicate d is simply the sum of the probabilities corresponding to the states where d holds.

Definition 2. *Given a predicate d, the utility u_d of a set of (state, probability) pairs S_p is defined as follows:*

$$u_d(S_p) = \sum_{(s,pr) \in S_p, s \models d} pr$$

The utility $u_d(succ^k_\pi(s_0, p))$ measures the probability that d is true after k steps of policy π from initial state s_0. Utility lies in the range $[0, 1]$. A utility close to 1 means a high probability of arriving, after k time steps, in a final state that satisfies d.

Definition 1 allows us to generate the final states obtained k steps after executing action a and compute its utility relative to a predicate

[*] Corresponding Author. Email: leo.saulieres@irit.fr.

d thanks to Definition 2. With this in mind, the *importance score* of an action a, from a state s in the history is the difference between the utility of a and the average utility of any other action $a' \in A(s)\backslash\{a\}$.

Definition 3. *Given a predicate d, an agent's policy π and a transition function p, the importance score of a from s at horizon k is defined by:*

$$imp_d(s, a, \pi, p, k) = u_d(succ_\pi^k(S_{(s,a)}, p)) - \underset{a' \in A(s)\backslash\{a\}}{\text{avg}} u_d(succ_\pi^k(S_{(s,a')}, p)) \quad (1)$$

where avg *is the average and $S_{(s,a)}$ is the set of reachable states (along with their probabilities) from s by performing action a. Formally, we have:*

$$S_{(s,a)} = \{(s', p(s'|s,a)) \mid p(s'|s,a) \neq 0\} \quad (2)$$

An importance score lies in the range $[-1, 1]$. The agent's action from a specific state is bad for achieving the predicate d if its importance score is negative. On the contrary, a positive score indicates a good action for achieving d in comparison with other available actions. Given a history of length k and a predicate d, HXP consists in calculating the importance scores for the k actions in the history and displaying to the user the most important action(s) for achieving d. For the calculation of importance scores, the combined use of *succ* and u is computationally hard, as we show now. For a predicate d, we write $d \in$ P to mean that it can be evaluated in time which is a polynomial function of the size of its input.

Proposition 1. *Given an initial state s_0 and a predicate $d \in P$, the problem of determining the probability that at the end of a length-k scenario, starting at s_0, the final state s_k satisfies the predicate d is $\#W[1]$-hard for parameter k.*

The proof of Proposition 1 can be found in [8]. Based on Proposition 1, it is easy to see that the computation of an importance score is #W[1]-hard. In this context, depending on k and the average number of transitions from a pair (s, a) in an RL problem, the importance score calculation can quickly become intractable. For this reason, we introduce a simple heuristic to reduce the computation time.

2.1 Approximate HXP

Approximate HXP refers to the approximate calculation of each action importance score. This approach consists in generating a large range of scenarios, avoiding the most unlikely ones. To do so, we could simply consider the most probable transition based on the transition function p at each time step. However, the result would be a single scenario, which is not informative enough. This is why we introduce a parameter $n \in \{1, \ldots, k-1\}$ for a scenario of length k to impose that the last n interactions with the environment are deterministic. Therefore, to generate diverse length-k scenarios, the exhaustive approach is applied for $k - n$ time-step(s) and the n last transition(s) are assumed to be the most likely ones. Accordingly, only a subset of scenarios is produced which is then used to compute the utility of an action from state s. This process is repeated for each feasible action from s to determine the importance score attached to an action a from s at horizon k.

Definition 4. *Given a predicate d, an agent policy π, a transition function p, a horizon k and n the number of deterministic transitions (with $1 \leq n < k$), the approximate importance score of a from s is defined by:*

$$imp_d^n(s, a, \pi, p, k) = u_d(succ_\pi^n(succ_\pi^{k-n}(S_{(s,a)}, p), p_{max})) - \underset{a' \in A(s)\backslash\{a\}}{\text{avg}} u_d(succ_\pi^n(succ_\pi^{k-n}(S_{(s,a')}, p)), p_{max})) \quad (3)$$

where p_{max} is a deterministic transition function mapping each pair (s, a) to only one transition corresponding to a most probable transition according to p.

This makes it possible to find the most important actions in a history in a reasonable time, parameterized by n. The larger n is, the fewer the number of scenarios, so the more approximate the importance score and greater the reduction in computation time.

2.2 Similarity Metric

The aim of the similarity metric is to compare the similarity between HXP and approximate HXP. This is done by comparing the two vectors of importance scores (corresponding to the k actions of the history). To do this, the L2 distance is used. A distance of 0 indicates identical importance scores for each action in the history. The worst-case distance occurs when importance scores are the opposite extrema (i.e. -1 and 1) for each of the k actions in the history, resulting in a distance of $2\sqrt{k}$. After normalization, the similarity score between two action importance vectors v_1, v_2 is thus defined as:

$$\text{similarity}(v_1, v_2) = 1 - \frac{L2(v_1, v_2)}{2\sqrt{k}}$$

where $L2(v_1, v_2)$ is the L2 norm of the difference between the two vectors and k is the vector dimension, i.e. the history length. The similarity score lies in the range $[0, 1]$ (for $k \geq 1$). Thus, a score of 1 indicates maximum similarity between the importance scores of the two vectors, and a score of 0 maximum dissimilarity.

3 Experimental Results

We tested (Approximate) HXP on different types of RL problems to verify its effectiveness and scope. The problems studied were: Frozen Lake (FL), Drone Coverage (DC) and Connect4 (C4) (described below). Training was carried out using the Q-learning algorithm [10] for the FL problem, and the Deep-Q-Network algorithm [4] for the C4 and DC problems. (The source code is available at: https://github.com/lsaulier/HXP). In the Tables, the exhaustive approach (denoted Exh) is compared with approximate ones in which the n Last transitions are deterministic (denoted nL).

3.1 Description of the problems

Frozen Lake In this problem, the agent moves in the surface of a frozen lake (a gridworld) with the aim to reach a specific position [1]. The agent loses if it falls into a hole. A state is the agent's position in the map, $S = \{1, \ldots, l \times c\}$ with, l, c the number of rows and columns in the map. The action space is $A = \{left, down, right, up\}$. The reward function is sparse: the agent receives a reward of 1 only by reaching the goal state.

The transition function p is stochastic: if the agent chooses a direction (e.g. *down*), it has 0.6 probability to go on this direction and 0.2 to go towards each remaining direction except the opposite one (here, 0.2 to go *left* and 0.2 to go *right*). Accordingly, the most likely transition used for approximate HXP is the one that is identical to the direction induced by the action.

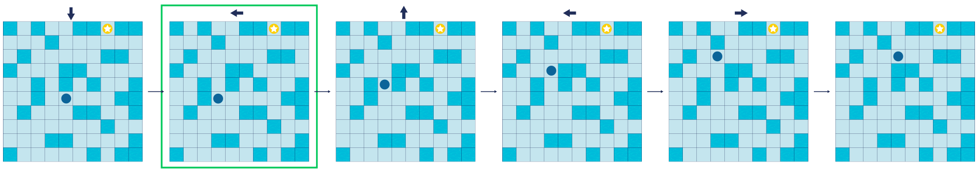

Figure 1. HXP for the *holes* predicate. The agent is symbolized by a blue dot, the dark blue cells are holes and the destination cell is marked by a star.

Drone Coverage The goal of this multi-agent problem [7] is that each drone has a perfect cover in a windy gridworld containing trees. The cover for a drone is the 3×3 square centered on its position. A cover is perfect if there are neither trees nor other drones. The action space is $A = \{left, down, right, up, stop\}$. The state features are the drone's position and it's vicinity (5×5 square centered on its position). The agent receives a reward of $+3$ if it has a perfect cover, and $+0.25 \times c$ otherwise, where c is the number of free cells (i.e. with no tree or drone) in its cover; it receives a *penalty* of -1 per drone in its 5×5 vicinity and a reward of -3 in case of a *crash*.

The stochastic transition function p, which represents the wind, pushes the agent to the *left, down, right, up* according to the following distribution: $[0.1, 0.2, 0.4, 0.3]$. After an agent's action, it moves to another position and then is impacted by the wind, unless the action is *stop* or the agent and wind directions are opposite. Since we are only interested in the actions of one agent to produce the HXP, when calculating the importance scores, we decided to limit the number of possible transitions by imposing the most probable transitions (i.e. the wind pushing the drones to the right) for the other agents[1].

Connect4 The objective of Connect4 is to reach a configuration where the player lines up 4 tokens in a row, column or diagonal. The state of a player is the whole board. The action space is $A = \{1, 2, 3, 4, 5, 6, 7\}$ where each number i is the action of dropping a token in the i^{th} column of the board. The reward function is sparse: the agent receives a reward only by reaching a terminal state (1, -1 and 0.5 for respectively a win, a loss and a draw).

The transition function p is stochastic since the player does not know the next opponent's move. To produce approximate HXP's, we need to define the most likely transition from each (s, a) pair. Player 1 and Player 2 have learnt by playing against each other. Accordingly, to study the behavior of Player 1, we assume that the most likely transition of its opponent is given by the policy of Player 2 (since the transition function of an *average* Connect4 player is unknown).

3.2 Results

As an HXP example for the FL problem, the *holes* predicate was used for the history shown in Figure 1. *holes* is a predicate that determines if the agent ends up in a hole. It is easy to see that the most important action for falling in a hole is the one at time-step 1. Indeed, with the action *left*, the agent has a chance of at least 60% of falling into the hole at coordinates (6,3) according to the transition function. Action importance scores in Table 1 confirm the assertion: whatever the approach, the action at time-step 1 stands out from the others.

For each problem, the average of similarity scores are shown in Table 2. All scores in the Table are based on 1000 length-5 histories for each predicate. In each history the predicate holds in the last state.

Table 1. Action importance scores of *holes* predicate in the history of Fig. 1

Time-step	0	1	2	3	4	Run time (s)
Exh.	-0.323	**0.315**	-0.262	-0.294	-0.119	0.025
1L	-0.34	**0.301**	-0.301	-0.303	-0.105	0.017
2L	-0.315	**0.379**	-0.317	-0.355	-0.109	0.014
3L	-0.387	**0.36**	-0.333	-0.373	-0.067	0.009
4L	-0.4	**0.467**	-0.467	-0.333	-0.067	0.008

Similarity scores of the FL, DC and C4 problems were calculated respectively for 3, 10 and 5 predicates. Due to lack of space, we do not describe these predicates (see [8]). Since the scores are close to 1, the approximate action importance scores are of good quality. We can note that the greater the number n (of final deterministic transitions), the lower the similarity to the exhaustive approach. Very logically, estimating the importance score based on fewer scenarios leads to a degradation of the approximation to the exhaustive approach.

Table 2. Average similarity scores of History-Explanation.

Problem	Exh-1L	Exh-2L	Exh-3L	Exh-4L
FL	0.992	0.983	0.971	0.954
DC	0.992	0.982	0.976	0.964
C4	0.995	0.979	0.955	0.918

The corresponding average times obtained by the different approaches for computing HXP's are shown in Table 3. The computation time decreases exponentially with the increase in the number of deterministic final transitions, which logically follows from the fact that less scenarios are explored.

Table 3. Average running time (in seconds) of History-Explanation.

Problem	Exh.	1L	2L	3L	4L
FL	0.006	0.005	0.003	0.002	0.001
DC	27.94	18.95	7.69	2.63	0.81
C4	21.51	20.49	6.51	1.58	0.33

4 Conclusion

Experiments showed that approximate HXP's were of good quality. Moreover, on average for all predicates and all values of n studied, including $n = k - 1$, approximate HXP highlights the same most important action as plain HXP in 86.91%, 87.75%, 76.19% of the histories for respectively the FL, DC and C4 problems. Of course, further experiments are needed to confirm scalability. One limit of the method is that the transition function is assumed known.

To sum up, we introduced History-eXplanation based on Predicates, which, from a history of the agent's interactions with the environment, exhibits to the user the important actions that are useful for achieving the predicate. To this end, we proposed an original method for computing action importance. The complexity of the latter being #W[1]-hard, we presented an approximate method to reduce the computation time. This method showed good results on short histories for a set of three problems from different settings.

[1] This assumption was necessary to limit the size of the search space, thus allowing us to compute HXP in an exhaustive way. Alternatively, one can consider that the HXP produced corresponds to a simpler version of the DC problem.

Acknowledgments

This work was supported by the AI Interdisciplinary Institute ANITI, funded by the French program "Investing for the Future – PIA3" under grant agreement no. ANR-19-PI3A-0004.

References

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba, 'Openai gym', *arXiv preprint arXiv:1606.01540*, (2016).

[2] Adnan Darwiche, 'Human-level intelligence or animal-like abilities?', *Commun. ACM*, **61**(10), 56–67, (2018).

[3] Zachary C. Lipton, 'The mythos of model interpretability', *Commun. ACM*, **61**(10), 36–43, (2018).

[4] 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**(7540), 529–533, (2015).

[5] S Nativi and S De Nigris, 'AI standardisation landscape: state of play and link to the EC proposal for an AI regulatory framework', (KJ-NA-30772-EN-N (online)), (2021).

[6] White House OSTP, 'Blueprint for an AI Bill of Rights', (October 2022).

[7] Léo Saulières, Martin C. Cooper, and Florence Bannay, 'Reinforcement learning explained via reinforcement learning: Towards explainable policies through predictive explanation', in *Proceedings of the 15th International Conference on Agents and Artificial Intelligence, ICAART 2023, Volume 2*, eds., Ana Paula Rocha, Luc Steels, and H. Jaap van den Herik, pp. 35–44. SCITEPRESS, (2023).

[8] Léo Saulières, Martin C Cooper, and Florence Dupin de Saint Cyr, 'Predicate-based explanation of a Reinforcement Learning agent via action importance evaluation', in *ECML/PKDD workshop on Advances in Interpretable Machine Learning and Artificial Intelligence*, p. to appear, Turin, Italy, (September 2023).

[9] Richard S Sutton and Andrew G Barto, *Reinforcement learning: An introduction*, MIT press, 2018.

[10] Christopher JCH Watkins and Peter Dayan, 'Q-learning', *Machine learning*, **8**(3), 279–292, (1992).