Neural Computing and Applications
https://doi.org/10.1007/s00521-022-07280-8

S.I.: HUMAN-ALIGNED REINFORCEMENT LEARNING FOR AUTONOMOUS AGENTS AND ROBOTS

Hierarchical goals contextualize local reward decomposition explanations

Finn Rietz[1,2] · Sven Magg[3] · Fredrik Heintz[4] · Todor Stoyanov[1] · Stefan Wermter[2] · Johannes A. Stork[1]

Received: 2 March 2022 / Accepted: 5 April 2022

Abstract

One-step reinforcement learning explanation methods account for individual actions but fail to consider the agent's future behavior, which can make their interpretation ambiguous. We propose to address this limitation by providing hierarchical goals as context for one-step explanations. By considering the current hierarchical goal as a context, one-step explanations can be interpreted with higher certainty, as the agent's future behavior is more predictable. We combine reward decomposition with hierarchical reinforcement learning into a novel explainable reinforcement learning framework, which yields more interpretable, goal-contextualized one-step explanations. With a qualitative analysis of one-step reward decomposition explanations, we first show that their interpretability is indeed limited in scenarios with multiple, different optimal policies—a characteristic shared by other one-step explanation methods. Then, we show that our framework retains high interpretability in such cases, as the hierarchical goal can be considered as context for the explanation. To the best of our knowledge, our work is the first to investigate hierarchical goals not as an explanation directly but as additional context for one-step reinforcement learning explanations.

Keywords Reinforcement learning · Explainable AI · Reward decomposition · Hierarchical goals · Local explanations

1 Introduction

The black-box-like characteristics of neural networks [1] and the increasing complexity of reinforcement learning (RL) agents [2] motivated numerous contributions in the explainable artificial intelligence and explainable reinforcement learning (XRL) fields [3–9]. However, by explaining individual transitions, most of these methods treat RL actions like isolated network decisions in supervised learning. Policy summarization methods [10–15], on the other hand, only explain the policy's overall strategy but do not provide explanations for concrete transitions. On their own, both of these approaches explain behavior only partially, which is not satisfactory [16, 17]. Such incomplete explanations are prone to misinterpretation and do not provide sufficient information to truly explain, validate and debug RL agents, as user studies have shown [18–20]. More powerful XRL methods are needed to unify these approaches and provide explanations for individual transitions while also considering the agent's overall behavior,

✉ Finn Rietz
finn.rietz@oru.se

Sven Magg
sven.magg@hitec-hamburg.de

Fredrik Heintz
fredrik.heintz@liu.se

Todor Stoyanov
todor.stoyanov@oru.se

Stefan Wermter
stefan.wermter@uni-hamburg.de

Johannes A. Stork
johannesandreas.stork@oru.se

[1] Science and Technology, Örebro University, Örebro, Sweden

[2] Department of Informatics, University of Hamburg, Hamburg, Germany

[3] Hamburger Informatik Technologie-Center, Universität Hamburg, Hamburg, Germany

[4] Department of Computer and Information Science, Linköping University, Linköping, Sweden

thereby explaining the agent on multiple levels of abstraction.

In this paper, we propose an XRL method that unifies one-step explanations and policy summarization by extending one-step explanations with a *context* that captures the policy's intermediate *goals*. We employ hierarchical reinforcement learning methods [21, 22] because they can learn a sequence of hierarchical goals that solve the overall RL problem. These hierarchical goals explicitly reveal which intermediate targets the policy is pursuing and hence can be exploited as context for local explanations, increasing their interpretability. Thus, we propose to integrate hierarchical RL and one-step XRL methods to thoroughly explain RL agents. Consider Fig. 1 for an illustration of our approach.[1]

We demonstrate that one-step explanations inherently depend on latent goals, even when no hierarchical architecture is employed. This implicit dependency on latent goals makes the interpretation of one-step explanations ambiguous when these goals are not directly observable. We demonstrate how hierarchical goals can be used as a context for one-step explanations to resolve their ambiguity in non-hierarchical settings. We find a strong effect of hierarchical goals on one-step explanations that yields insights beyond what is conveyed by the goal directly. The implication of our finding is that one-step RL explanations cannot be accurately interpreted *without* consideration of their context. This paper's main contribution is a method to obtain such a context, given slight adaptations of the problem definition that allows for a goal-based hierarchical partition.

2 Related work

Explainable artificial intelligence methods generally aim to make the behavior of AI systems more understandable to humans [4]. Unfortunately, important terms including explainability and interpretability are only vaguely defined in the explainable artificial intelligence field [8]. In this paper, we use the definitions from Puiutta and Veith's survey on XRL [8]: An *explanation* is an artifact that gives reasons for the occurrence of a phenomenon, while *interpretability* refers to the degree to which that explanation enables humans to predict the model's behavior. For example, the weights, biases, and activation functions of a deep neural network could be considered an explanation because this information directly determines the network's output. However, such an explanation would be uninterpretable because no human observer can predict the network's behavior from just the weights, biases, and activation functions. In the following sections, we generally refer to the output artifacts of any XRL method as an explanation and assess their different degrees of interpretability only qualitatively. Lastly, explanations of concrete action selections (one-step explanations) are referred to as *local* explanations, while *global* explanations aim to explain the entire model-generated behavior. In these terms, our work investigates the effect of hierarchical goals on the interpretability of local RL explanations.

2.1 Hierarchical reinforcement learning

Hierarchical reinforcement learning provides an extension to the classical RL problem that allows policies to operate on multiple timescales. Sutton et al. [21] originally introduced this extension in the *Options* framework to enable the application of RL algorithms to complex scenarios that require operations on different temporal scales and levels of abstraction [23]. Based on the Options framework, Kulkarni et al. [22] propose to use hierarchical RL for efficient exploration in complex environments with sparse reward signals. These methods, however, do not consider hierarchical RL for explainability purposes, which is the aim of our work.

Two previous works investigated hierarchical reinforcement learning for explainability. Beyret, -Shafti, and Faisal [24] employ hierarchical reinforcement learning to obtain an interpretable higher-level policy that generates sub-goals for a lower-level policy. Shu, X-iong, and Socher [25] annotated each sub-policy with natural language labels, which results in explainable task partitions. In both of these works, hierarchical RL and hierarchical goals are directly used for explanation. Although our approach is similar to [24], we propose to use hierarchical goals as *context* for local explanation methods and investigate the effect of hierarchical goals on local explanations, instead of relying directly on hierarchical reinforcement learning for explanations. Since hierarchical RL is a central element of our method, we describe hierarchical RL on a technical level in Sect. 3.2.

2.2 Local explanation methods

While local explanation methods offer a valid explanation for the current action selection, most of these methods have a drastic limitation because they disregard the agent's future behavior or only account for it implicitly (through the recursiveness of the Bellman equations). For example, saliency maps [6] are a popular local XRL method [19, 26–29] that reveals which areas of the input space cause the activations at the last layer of the policy network, offering insight into what guided the action selection. Intuitively, which part of the input is relevant for

[1] Code and data: https://github.com/frietz58/hdddqn.

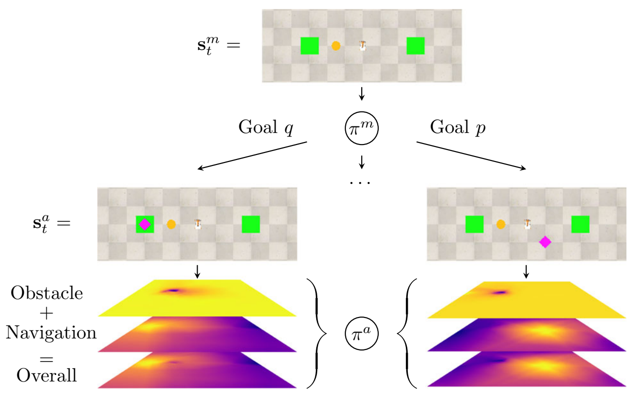


Fig. 1 Our hierarchical and locally explainable framework. The meta-controller π^m selects exemplary goals q or p for the state $\mathbf{s}_t^m$. The atomic-controller π^a learns to satisfy different goals, which are part of $\mathbf{s}_t^a$. The atomic-controller is locally explainable because it employs reward decomposition based on an obstacle and navigation component. The local explanations are contextualized by and can be interpreted with respect to the current goal

determining the best action depends on the agent's future behavior. For example, which pedestrians an autonomous vehicle has to consider at an intersection depends on its direction of travel. However, saliency maps do not take this information into account, which limits their explanation capabilities [18, 20, 30].

While saliency methods fully neglect the agent's future behavior, we also find that other local explanation methods account for it only implicitly. For example, in [31, 32] the authors provide local explanations by considering expected future states. In [33], action consequences are captured through forward-simulation on a learned environment model. Even though these methods consider the policy's behavior in future states, we argue that hierarchical goals would increase the interpretability of such methods as well. Considering again the autonomous navigation case, even when an explanation based on forward-simulation indicates no collision, we must still verify whether this explanation is indeed acceptable or whether intervention is required, considering the agent's heading.

Our work is also similar to Huber et al. [34], who propose the joint employment of local and global XRL methods by extending policy summaries with local saliency map explanations. We instead investigate how hierarchical goals can be used to extend and contextualize local reward decomposition explanations. In the next section, we introduce reward decomposition as the concrete local explanation method through which we investigate the effect of hierarchical goals on interpretability.

2.3 Reward decomposition for interpretability

Russell and Zimdars [35] introduce reward decomposition as an extension to the classical RL framework, where a number of sub-agents each provide different state-action values, accounting for smaller and different aspects of the overall problem. For example, consider a robotic agent that shall navigate to an arbitrary location in a dynamic environment. The agent has access to a static map of its environment, but this map does not contain dynamic obstacles, e.g. humans or lightweight furniture. This problem can be decomposed into two simpler problems: Navigating on the static map and dynamic obstacle avoidance based on sensor data, with one reward function for each problem. Reward decomposition allows the definition of multiple reward component functions, one for each sub-problem.

Juozapaitis et al. [36] propose to use reward decomposition for more interpretable RL and introduce the *drQ* algorithm, based on the intuition that a decomposed value function is more interpretable than a non-decomposed one. Although reward decomposition explanations result in significantly better mental models compared to solely observing agent behavior [37], reward decomposition produces one-step explanations and thus shares the previously mentioned limitation of local explanation methods. Thus, we propose hierarchical RL as an extension to reward decomposition and analyze the effect of hierarchical goals on the interpretability of local reward decomposition explanations. We describe reward decomposition on a technical level in Sect. 3.3.

3 Method

In this section, we describe how our method uses hierarchical goals to contextualize local explanations. We begin by establishing the classical RL problem formulation through a Markov Decision Process (MDP), then extend the MDP to a Semi-MDP to allow for hierarchical RL. We describe reward decomposition on a technical level and lastly describe how hierarchical RL and reward decomposition can be merged into a hierarchical XRL framework.

3.1 Reinforcement learning

Reinforcement learning is a computational framework for learning solutions to sequential decision-making problems, in which an agent interacts with an environment to maximize reward through trial-and-error search. The formal framework for sequential decision-making problems is a finite, fully observable, discrete-time MDP given by the 5-tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ is the probabilistic transition function, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function, and $\gamma \in [0,1]$ is a discount factor. $\mathcal{S}$ also contains terminal states which end an episode when the agent reaches those states. At any point in time t, the agent perceives the current state $s_t \in \mathcal{S}$ of the MDP and selects the action $a_t \in \mathcal{A}$ for the given state, according to its policy $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$. Upon execution of the action, the MDP transitions to the next state s_{t+1} based on its transition function $T(s_t, a_t, s_{t+1})$ and emits the reward $R(s_t, a_t, s_{t+1}) = r_t$, corresponding to the *desirability* of the transition (s_t, a_t, s_{t+1}), which numerically quantifies the cost or bonus of action a_t in s_t. The discounted *return* G_t of a policy π is defined as the discounted sum of future rewards $G_t^\pi = \sum_{k=0}^{\infty} \gamma_t^k r_{t+k+1}$ when following policy π from time t. Generally, the goal of RL is to find an optimal policy π^* that maximizes the expected discounted return.

While some methods learn policies directly [38, 39], value-based RL methods instead construct policies based on value functions. The state-action value function $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ of a policy π gives the value for taking action a in state s, thereafter following π:

$$\begin{aligned} Q^\pi(s,a) &= \mathbb{E}_\pi[G_t | s_t = s, a_t = a] \\ &= \sum_{s'} T(s'|s,a) \left[R(s,a,s') + \gamma \sum_{a'} \pi(a'|s') Q^\pi(s',a') \right] \end{aligned} \tag{1}$$

Thus, learning an estimate Q function for the optimal policy π^* enables the agent to always take the action with the highest value, which is the central idea behind the Q-learning algorithm [40]. Q-learning is a form of *temporal-difference* learning and employs the following update rule to estimate the Q-function of an optimal policy:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \underbrace{\left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right]}_{\text{temporal-difference error}} \tag{2}$$

Of particular relevance for this paper is Mnih et al.'s [41] DQN algorithm, which employs deep neural networks to learn an approximation of Q and provides the algorithmic basis for our method. In the next two sections, we elaborate on extensions to DQN that we ultimately combine into our hierarchical XRL framework.

3.2 Hierarchical reinforcement learning with goals

Hierarchical reinforcement learning provides an extension to the classical RL problem by introducing multiple time-scales into the framework, allowing policies to operate on different levels of abstraction. In such a framework, higher-level policies can learn a sequence of intermediate task *goals*, while lower-level policies learn to satisfy different goals. Based on this concept, Kulkarni et al. [22] propose hierarchical-DQN (h-DQN), extending Mnih et al.'s [41] DQN algorithm with hierarchical goals and policies, to boost exploration in complex environments with sparse rewards. In h-DQN, the higher-level policy is represented by a DQN, whose action space is a set of goals $g \in \mathcal{G}$. These goals are passed to a lower-level DQN, by concatenating the goal to the MDP's state representation, so that a lower-level DQN learns to implement different behaviors, depending on the goal. Thus, the state representation for the lower level policy is extended by additional dimensions for the hierarchical goal, so that $s_t = [s_t, g_t]$. Both the higher- and lower-level DQNs are updated according to Eq. 2, where the lower-level transitions are implicitly depending on the goal as part of the state, while the higher-level transitions are generated at a slower timescale.

Importantly, implementing h-DQN requires the definition of multiple reward functions, one for each level of the hierarchy. The agent's overall goal is to maximize the (global) reward at the highest level of the hierarchy, while an *intrinsic* reward function $R(s_t, a_t, s_{t+1}; g) = r_t^g \in \mathbb{R}$ is used to provide goal-dependent rewards for implementing the lower-level behaviors. Although h-DQN requires additional, manual design steps to transform the original problem into a hierarchical one, these steps are worthwhile for explanation purposes because they unlock goals as context for local explanations. Intuitively, the benefits that hierarchical RL affords in terms of providing a global context are independent of the choice of local XRL method and we assume that different XRL methods can be extended with hierarchical, goal-based context mechanisms. The other way around, existing hierarchical agents could be extended with local explanation methods to obtain more interpretable agents, which motivates investigating this approach. The key attribute of hierarchical RL is that lower-level policies depend on explicit goals, which can be exploited for increased interpretability, either directly (as in [24, 25]) or, as we propose, as context for local explanations.

3.3 Reinforcement learning with reward decomposition

Reward decomposition provides another extension to the classical RL problem that decomposes the reward function R into a number $C \in \mathbb{N}$ of smaller and simpler functions. Then, individual agents can be trained to maximize each individual reward function $R_c, 0 \leq c < C$ that accounts for a certain aspect of the overall reward function. The overall reward function R^+ is obtained from the sum of the individual reward functions such that $R^+(s,a) = \sum_{c=0}^{C} R_c(s,a)$. Russell and Zimdars [35] prove that it is possible to train individual RL agents on the individual reward functions R_c so that the summed action-value function yields globally optimal behavior. Juozapaitis et al. [36] exploit this idea for interpretability and propose drQ, which uses a vector-valued reward function $R : S \times A \mapsto \mathbb{R}^C$, that stores the C decomposed reward functions. Then, dedicated action-value functions $Q_c(s,a)$ can be learned for each decomposed reward function R_c. The global policy π^+, that selects actions to solve the overall problem, is implicitly defined by $Q_\pi^+(s,a) = \sum_{c=0}^{C} Q_c(s,a)$. As pointed out by Russell and Zimdars [35], the updates of the component value functions Q_c must reflect the global policy π^+ because otherwise the components maximize their local reward functions $R_c(s_t, a_t, s_{t+1})$ and their sum is not guaranteed to yield globally optimal behavior, a phenomenon referred to as *tragedy of the commons*. To account for this, drQ bootstraps the action a^+, which the global policy would select for the next state s_{t+1} by $a^+ = \arg\max_{a'} \sum_{c=0}^{C} Q_c(s_{t+1}, a')$, when calculating the TD-error. The update rule for drQ's component Q-functions is almost identical to the Q-learning update rule in Eq. 2, except the action for s_{t+1} is changed to the bootstrapped action a^+ of the global policy:

$$Q_c(s_t, a_t) \leftarrow Q_c(s_t, a_t) + \alpha\left[r_t^c + \gamma\, Q_c(s_{t+1}, a^+) - Q_c(s_t, a_t)\right] \tag{3}$$

Juozapaitis et al. [36] define the *Reward Difference Explanation* (RDX) as explanation metric. The RDX is simply the vectorized value difference between two actions and is given by $\Delta(s, a_1, a_2)^C = \mathbf{Q}^C(s, a_1) - \mathbf{Q}^C(s, a_2)$. The resulting vector contains positive or negative quantities that represent advantages or disadvantages of action a_1 over a_2 in state s, explaining the action selection in terms of trade-offs between the decomposed Q-functions. In our following analysis, we consider the decomposed Q-values and the RDX as explanations. Figure 4 provides examples for both. In the next section, we show how to merge reward decomposition and hierarchical RL to obtain more interpretable, goal contextualized local explanations.

3.4 Our hierarchical XRL framework

Our framework extends local drQ explanations with a hierarchical context, for increased interpretability. Following h-DQN, we employ a two-level hierarchy, where a higher-level DQN agent learns a sequence of goals that solve the task, while a lower-level DQN agent learns to satisfy the different goals. We refer to the higher-level agent as *meta-controller* π^m and the lower-level agent as *atomic-controller* π^a, since the former learns the task at a meta-level, while the latter implements atomic control actions.

To integrate reward decomposition via drQ into h-DQN, the meta- and atomic-controller are vertically decomposed, meaning each controller is obtained by joining multiple decomposed DQNs, according to drQ. Thus, at the atomic level, instead of training one agent, we train a number $C^a \in \mathbb{N}$ of agents, each still depending on the goals g selected by the meta-controller. For example, $Q_c^a(s, a; g)$ refers to the c-th component on the atomic level and learns the state-action values for the reward signal $R_c(s_t, a_t, s_{t+1}; g) = r_{c,t}^g$. The combined atomic-controller is obtained by summing up the local Q-functions on the atomic level $Q_+^a = \sum_{c=0}^{C^a} Q_c^a(s, a; g)$. Analogously, on the meta-level, we have a number $C^m \in \mathbb{N}$ of DQNs, one for each reward component on that level and their sum provides the overall meta-controller.

We only extend drQ by integrating hierarchical goals into the framework, which are part of the atomic-controller's state representation. Hierarchical goals remain fixed for each rollout of the atomic-controller and only ever change after the end of an atomic rollout and before the next one starts. Thus, the hierarchical goal can be thought of as a fixed index for a larger MDP during each atomic rollout. This index allows drQ to learn different policies for different values of the hierarchical goal, which is also illustrated in Fig. 1. Thus, our method simply learns many local drQ policies, and the convergence properties of drQ directly apply to our method. In practice, we apply a number of well-established methods to facilitate learning and stabilize convergence when training the above-mentioned DQNs. Firstly, we pretrain the atomic-controller with random goals before training the meta-controller, as suggested by Kulkarni et al. [22]. Furthermore, we employ target networks and experience replay when optimizing the individual DQNs through stochastic gradient descent, as described by Mnih et al. [41]. Lastly, we employ prioritized experience replay [42] for faster learning and double DQN [43] to combat the typical overestimation in Q-learning and DQN. Through this jointly hierarchical and decomposed architecture, each action selected by the atomic-controller can be explained locally through vertical reward decomposition, while that local explanation is contextualized by the currently given hierarchical goal g.

4 Experiments

With our experiments, we investigate and compare the interpretability of goal-contextualized local explanations with the interpretability of local explanations that do not have such a context. We hypothesize that a goal-based context increases the interpretability of local explanations. To evaluate our hypothesis, we train a classical and a hierarchically extended drQ agent so that we can qualitatively analyze and compare the interpretability of the two agents.

We illustrate our approach on a simple 2D navigation environment with a discrete action space. The agent has to reach two specific locations to successfully terminate the episode while avoiding a static obstacle. Episodes end when both locations have been visited or a number of time steps is exceeded. The order in which the agent visits the two locations does not matter, while close proximity with the obstacle yields increasingly negative rewards. The obstacle is mainly relevant for driving to the left side and largely irrelevant for driving to the right side, depending on the current position of the agent. To train a classical (flat, non-hierarchical) drQ agent and a jointly hierarchical and decomposed agent (our method), the environment features two slightly different versions that account for the differences between these two agents, which we describe in the following.

4.1 Non-hierarchical testbed

To solve the environment outline above, for the non-hierarchical agent, the state has 10 dimensions and is given by

$$\mathbf{s}_t = \left[i, j, \mathbf{a}, \mathbf{o}, \mathbf{l}, \mathbf{r}\right], \tag{4}$$

where i and j are binary indicator variables that switch from 0 to 1 once the agent was within a threshold distance of $d = 0.4$m of the respective target location, while $\mathbf{a}, \mathbf{o}, \mathbf{l}$, and $\mathbf{r}$ are the absolute 2D positions of the agent, the obstacle, and the left and right target locations. States in which $i = 1$ and $j = 1$ are terminal states. From this state representation, the non-hierarchical policy maps to a discrete action space of positional increments, which correspond to taking 0.25 m steps in the directions *North, South, East,* or *West* in a 12 m $\times$12 m arena. Although non-hierarchical, we are still training a reward-decomposed agent, thus we define several reward functions. In line with our example in Sect. 2.3, the first component $R_0 : \mathbb{R}^{10} \rightarrow \mathbb{R}$ rewards the agent for decreasing the distance between the two target locations and is given by

$$R_0(\mathbf{s}_{t+1}) = -10 - (1-i)||\mathbf{a} - \mathbf{l}|| - (1-j)||\mathbf{a} - \mathbf{r}||, \tag{5}$$

where $||\ldots||$ represents vector magnitude. R_0 only depends on $\mathbf{s}_{t+1}$ (instead of the entire transition) because s_{t+1} gives access to all symbols required to calculate the reward for the transition $(\mathbf{s}_t, a_t, \mathbf{s}_{t+1})$. Under R_0, the agent loses reward for each target location that has not yet been visited, proportional to the distance to that location. We add -10 as a constant cost so that an optimal agent should finish the episode with as few steps as possible, at the same time balancing the magnitude of reward component R_0 with respect to the second reward component R_1. The second reward component $R_1 : \mathbb{R}^{10} \rightarrow \mathbb{R}$ punishes the agent for being in close proximity of the obstacle and is given by

$$R_1(\mathbf{s}_{t+1}) = -20\frac{1}{2\pi\sigma^2}exp\left(-1\frac{(\mathbf{a}-\mathbf{o})^2}{2\sigma^2}\right), \tag{6}$$

which is a 2D normal distribution with constant standard deviation σ, for which we calculate the density at $\mathbf{a} - \mathbf{o}$, which is scaled by 20. This reward decomposition yields local explanations which reveal how the chosen components (R_0 and R_1) contribute to the policy's action selection, for example, when the obstacle has a strong effect. In accordance with drQ, the overall reward signal at time t corresponds to $\mathbf{r}_t = (R_0(\mathbf{s}_{t+1}), R_1(\mathbf{s}_{t+1}))^T$, depending on the transition $(\mathbf{s}_t, a_t, \mathbf{s}_{t+1})$ of the MDP.

4.2 Hierarchical environment adaptation

To obtain goals that are informative with respect to the local explanation, we manually select a domain-dependent hierarchy, for which we design the state- and action spaces. For our simple 2D navigation environment, we select a hierarchical task partition that yields intermediate navigation goals. These goals provide an informative context for local explanations because they reveal the target coordinate to which the agent will attempt to navigate.

For the meta-controller, we define the action space as discrete indices to a 2D, 11×11 grid of the environment, with $a_t^m \in \{0, 1, \ldots, 120\}$. The 2D coordinate indexed by a_t^m is the hierarchical goal $\mathbf{g}_t$ that is used in the atomic-controller and as context for local explanations in the following sections. To determine a_t^m, the meta-controller uses the same state representation as outlined in the previous section, so that $\mathbf{s}_t^m = \mathbf{s}_t$. The reward function for the meta-controller is scalar-valued with

$$R^m(\mathbf{s}_{t+1}^m) = \begin{cases} -10, & \text{if } ||\mathbf{a} - \mathbf{g}_t|| > d \\ -4, & \text{if } ||\mathbf{a} - \mathbf{g}_t|| < d \\ +2, & \text{if } ||\mathbf{a} - \mathbf{l}|| \wedge i = 0 \\ +2, & \text{if } ||\mathbf{a} - \mathbf{r}|| \wedge j = 0 \\ +10, & \text{if } i = 1 \wedge j = 1, \end{cases} \tag{7}$$

where $d = 0.4$ is a distance threshold scalar. Thus, the meta-controller is guided to select navigation goals that the atomic-controller can reach (within 10 steps), and that causes the agent to visit the two target locations. Given the meta-controller that selects goals, the atomic-controller's task is to drive to those goals. For this and following h-DQN, the state representation for the atomic-controller is obtained by concatenating the hierarchical goal to the state representation of the meta-controller, so that $\mathbf{s}_t^a = [\mathbf{s}_t^m, \mathbf{g}_t]$. We point out that other variables of the atomic controller's state can act as confounding factors with respect to the hierarchical goal and the atomic policy. The action space for the atomic-controller is identical to that of the non-hierarchical agent in the previous section, meaning it still selects small positional increments. For the atomic-controller, the reward decomposition is conceptually the same as for the non-hierarchical agent, except Eq. 5 changes to

$$R_0^a(\mathbf{s}_{t+1}) = -10 - ||\mathbf{a} - \mathbf{g}||, \tag{8}$$

as the atomic-controller is only concerned with satisfying the hierarchical goal given by the meta-controller, while the meta-controller must learn to select goals that solve the overall task modeled by the environment. The overall reward signal for the atomic-controller is given by $\mathbf{r}_t^a = (R_0^a(\mathbf{s}_{t+1}), R_1(\mathbf{s}_{t+1}))^T$, depending on the transition $(\mathbf{s}_t, a_t, \mathbf{s}_{t+1}; \mathbf{g}_t)$ of the MDP.

To summarize, our hierarchical environment version features state and action spaces for a two-level hierarchical agent, where the higher-level policy selects 2D navigation goals to reach terminal states, while the lower-level policy learns to drive to goals given by the higher-level policy. Additionally, the lower-level policy is reward-decomposed so that its action selections are locally explainable, while the hierarchical goal $\mathbf{g}_t$ provides a context for those local reward decomposition explanations. Images of the environment are provided in Fig. 2.

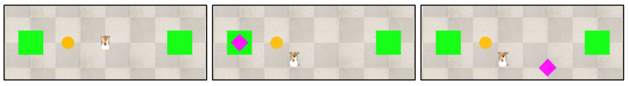
4.3 Results

We trained a classical and a hierarchically extended drQ agent on the respective environment versions in Sects. 4.1 and 4.2 until both reached a running success rate of 100%. We then compared and qualitatively analyzed the interpretability of these agents, which, as shown in the following sections, revealed two core results. Firstly, we found that purely local (one-step) explanations were insufficient to explain RL agents because they failed to account for the context of action selections. Secondly, we found that hierarchical goals could resolve this issue by providing an explicit context for the local explanation. In addition, we discuss properties of value functions that have problematic implications for value-based XRL methods.

4.3.1 One-step explanations are ambiguous

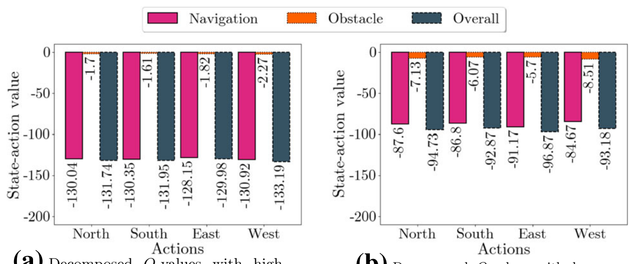
We argue that local one-step explanations, in the form of decomposed Q-values, do not provide sufficient information to evaluate or interpret RL agents because they do not consider the agent's future behavior. To illustrate this, we place the non-hierarchical drQ agent in the state of the environment shown in Fig. 2a. In this state, the agent is located at exactly the center of the arena, while both indicator variables i and j are zero, meaning both of the target locations must still be visited to successfully terminate the episode.

Given the agent is perfectly centered between the two target locations, an optimal policy should not have a clear preference for either side to drive to first. Indeed, the decomposed values in Fig. 3a only feature a marginal preference for driving *East* instead of *West*, with very low values under the obstacle component. However, the important thing to note is that the local explanation in Fig. 3a only accounts for the current transition, we cannot draw any conclusions regarding the agent's future behavior, which makes the interpretation of the explanation ambiguous. For example, we might assume that the agent will continue to drive east until it reaches the right target, which is why we see very low activations under the obstacle component. Without explicit knowledge about the agent's future behavior, however, we might also assume that, after taking one step away from the obstacle, the agent will drive to the left side. And if the agent would drive to the left side, should the obstacle not be more relevant for selecting the current action? When it is unclear how the



(a) State without hierarchical goal. **(b)** State with hierarchical goal. **(c)** State with different hierarchical goal.

Fig. 2 Cropped views of our simple 2D navigation environment. The green squares represent the locations the agent must visit. The orange circle represents the obstacle that must be avoided. If present, the magenta diamond represents the hierarchical goal selected by the meta-controller (Color figure online)



(a) Decomposed Q-values with high navigation values and low obstacle values in the initial state with $i, j = [0, 0]$. **(b)** Decomposed Q-values with lower navigation values and higher obstacle values with $i, j = [0, 1]$.

Fig. 3 Decomposed Q-values for the non-hierarchical agent and state in Fig. 2a

agent behaves in the near future and in close proximity of an obstacle, a human observer cannot infer whether intervention is required.

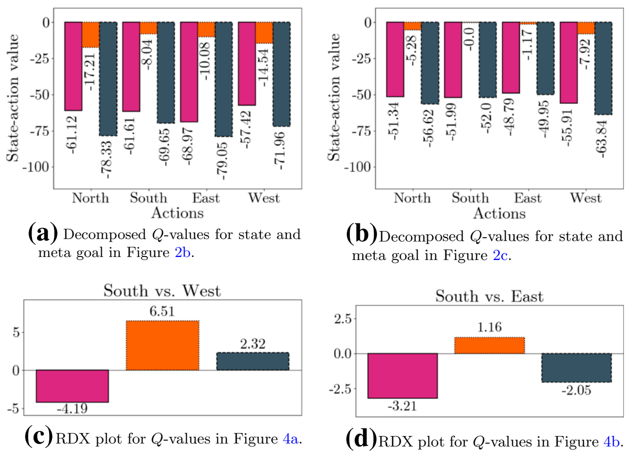
To underline this further, consider the decomposition in Fig. 3b, which is obtained by setting the state variable $j = 1$ (which indicates that the right side was already visited). Directly editing the state is not suitable in real systems and at runtime, but acceptable for illustration purposes in this example. The action-values under the obstacle component in Fig. 3b are noticeably higher than in Fig. 3a, which shows that the local explanation is, in fact, different depending on which side was already visited. Even when we assume that the agent will behave optimally and drive to the left side to complete the task, it remains impossible to interpret from the local explanation why the agent is selecting South over North or West. Thus, without having access to a goal-like context, local one-step explanations are prone to be misinterpreted even by domain experts, especially considering human (confirmation) biases and often unexpected but valid behaviors emitted by artificial agents.

Of course, in a simulated setting, we can simply continue the roll-out and solidify our understanding of the agent. However, this is not an option in real-world, potentially safety-critical scenarios where we must be able to intervene before observing the consequences for not intervening. Thus, as this example illustrates, local reward decomposition explanations can be ambiguous because they do not consider the agent's long-term behavior. This observation motivates us to consider a mechanism that reveals the agent's future behavior and provides a context for the interpretation of local explanations.

4.3.2 Hierarchical goals contextualize local explanations

In this section, we demonstrate that hierarchical goals can provide an informative context for local explanations that addresses the limitation outlined in the previous section. Consider the environment states in Fig. 2b and c, where the only difference between the two states is the location of the hierarchical goal. Contrary to the state representation of the non-hierarchical agent in Fig. 2a, we now have explicit knowledge about the navigational goal given by the meta-controller. Even without consideration of additional local explanations, we can already see how knowledge about the hierarchical goal improves the agent's interpretability, as we know explicitly where the agent attempts to navigate to. However, we argue that hierarchical goals *alone* are not sufficient explanations either, as they do not explain concrete action selections but only provide a general context.

For the two states in Fig. 2b and c, we obtain the respective decomposed Q-values shown in Fig. 4, where



(a) Decomposed Q-values for state and meta goal in Figure 2b.

(b) Decomposed Q-values for state and meta goal in Figure 2c.

(c) RDX plot for Q-values in Figure 4a.

(d) RDX plot for Q-values in Figure 4b.

Fig. 4 Different Q-value decompositions and respective RDX plots depending on the hierarchical goal

largely different Q-value decompositions are obtained and explained by the varying hierarchical goal. Where, in the case of the non-hierarchical agent, we had to form vague hypotheses about the agent's future behavior, now, our hierarchical framework directly provides the necessary information to interpret a given explanation. The context that the goal provides resolves the ambiguity in the obtained local explanation and allows us to determine whether the explanation is acceptable for a trained agent. For example, when the obstacle lies between the agent and the hierarchical goal, we expect the obstacle to affect the agent's behavior, which thus must be reflected in the explanation. Whereas, when the obstacle does not lie between the agent and the goal, the obstacle component should not affect the action selection. Indeed, such a characteristic is reflected in the explanations in Fig. 4. In Fig. 4a, action *South* has the highest value not because it brings the agent closer to the goal but because it avoids the larger punishments for driving closer to the obstacle, which is underlined by the RDX plot in Fig. 4c. In Fig. 4b, on the other hand, the action *East* has the highest value, independently of the values under the obstacle component, as the obstacle does not lie between the agent and the goal. This is underlined in the RDX plot in Fig. 4d, which shows that the advantage action *South* has over action *East* under the obstacle component does not outweigh the disadvantage of *South* over *East* under the navigation component. As this example shows, different predictions and explanations can occur depending on the hierarchical goal which, reversely, implies that local explanations are likely to be misinterpreted without an explicit context that indicates the agent's future behavior.

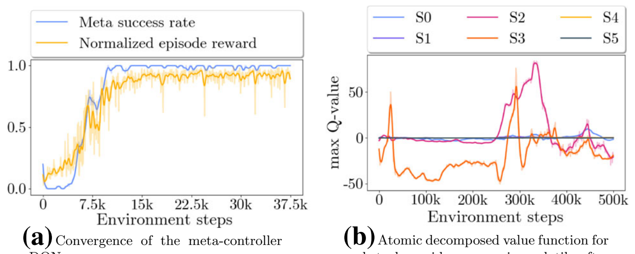
Thus, the context that hierarchical goals provide is clearly beneficial for local explanations, as it allows us to interpret the given explanation with respect to the current hierarchical goal. While some environments might have such small sets of optimal policies that the goal is implicitly apparent for every optimal policy (e.g. Cartpole, where the pole must always be in an upright orientation), whenever multiple optimal policies are equally valid (e.g. which side to visit first in our environment) this must be accounted for by explanation methods. Our method is one possible way to maintain the interpretability of local explanations given environments with multiple different optimal policies.

But our approach of exploiting hierarchical goals for improved interpretability still has a limitation: It is predicated on the choice of a task hierarchy. As such, the main insight that is conveyed through the hierarchical goal is predicated on this choice as well. For example, for our 2D-navigation environment, we chose a hierarchy that segments the episode into intermediate coordinates, which is thus the main information that is obtained from the hierarchical goal. However, our method yields insights beyond just the hierarchical goal because we also have access to local explanations and can capture the effect of different goals on the explanation. As we illustrated earlier, when reward decomposition is employed as explanation method, the goal also reveals the differences between local policies and the trade-off these policies make when satisfying different goals.

4.3.3 Value-based explanations are flawed

In this section, we outline fundamental flaws of value-based explanation methods, including reward decomposition. The key problem regarding the interpretation of value-based explanations is that they reflect the expected return of some (unknown) policy. The interpretation of value-based explanations only makes sense when we assume an optimal policy as otherwise the set of non-optimal policies that could generate the values in the explanation is near-limitless. For example, when the obstacle lies between the agent and the current goal, we might observe only very small activations under the obstacle component because the agent did not yet learn the negative rewards in close proximity of the obstacle. But it is also possible that the agent already learned to evade the obstacle perfectly and thus avoids the punishment altogether. Hence, the interpretation of value-based explanation strongly depends on the assumptions we impose on the policy (e.g. optimality). However, even when we assume an optimal policy, we do not know the value function for that optimal policy, hence it is difficult to access whether the value-based explanation is appropriate for the given state and the hierarchical goal.

The relation between the value-based explanation and the associated policy can also be observed in Fig. 4. The value of action *East* under the obstacle component is ten



(a) Convergence of the meta-controller DQN.

(b) Atomic decomposed value function for obstacle avoidance remains volatile after policy converged.

Fig. 5 Convergence behavior of policy compared to value function. Value estimations of the atomic-controller for states $S0$ to $S5$ keep on changing drastically long after the meta-controller converged to almost 100% success

times greater in Fig. 4a than in Fig. 4b, which might appear unintuitive at first, given that the action puts the agent in exactly the same position in both cases. The reason for this value difference is that the optimal policy for the two goals is different, hence the values are different. When we interpret the explanations in Fig. 4 we *assume* that the agent will behave optimally with respect to the different goals, in which case the different explanations seem adequate.

Additionally, it is well known that policies converge faster than value functions ([23], page 82). Thus, even if the policy achieves a 100% success rate, its value function might still contain exploration artifacts, see Fig. 5. We could have stopped the training at any point in time where the meta-controller's success was satisfactory and (especially in the second half of training) might have gotten very unexpected explanations from the atomic-controller, based on the volatile value function.

Thus, RL value-functions are poor candidates for explanations because they are associated with an unknown policy. Our method is not intended to fix these issues, although hierarchical goals partition the trajectory in smaller segments with local value functions that have expectations over shorter horizons which often have clear optimal policies. Although the relationship between policies and value functions is a basic, well-understood property, related work fails to point out the implications of this attribute for XRL. Independent of this, our finding that hierarchical goals can provide an important context for local RL explanations still holds true.

5 Conclusion

We propose to use hierarchical goals as context for local explanations for increased interpretability. We developed a jointly hierarchical and decomposed RL framework that allows us to interpret local reward decomposition explanations in the broader context of hierarchical goals, which are obtained from a higher-level policy. Our experiments show that local explanation methods, which only explain one-step decisions, are ambiguous and potentially misleading in environments that have multiple valid solutions. However, we also find that this shortcoming of local explanations can be mitigated by hierarchical goals because they resolve the ambiguity of local explanations by revealing which goal the agent is currently satisfying. This makes hierarchical goals (or other context-like mechanisms) a strictly necessary extension for local explanation methods when the environment features multiple different optimal policies. Lastly, we highlight significant flaws of value-based explanations for RL agents that have strong implications for the field.

We did not conduct an empirical evaluation of the level of interpretability of goal-contextualized local explanations in this paper and leave a user-study as the most important future work. A user study should validate that the goal context is beneficial for human observers to interpret the agent, especially for an industrial-level robustness and safety analysis. Additional future research could investigate approaches for learning the hierarchy and reward decomposition since, in this work, we relied on a handcrafted hierarchy as well as a handcrafted reward decomposition.

Author contributions All authors contributed to the method conception and design. The first draft of the manuscript was written by Finn Rietz. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open access funding provided by Örebro University. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. In addition, this work was supported by the Federal Ministry for Economic Affairs and Climate action on the basis of a decision by the German Bundestag as part of the research project "VeriKAS - Verification of learning AI applications in the aviation sector" (FKZ 20X1905A-D).

Data availability Not applicable.

Code availability https://github.com/frietz58/hdddqn

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose and declare that they have no conflict of interest.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication All authors agreed with the content and gave explicit consent to submit and obtained consent from the responsible authorities at the organization where the work has been carried out.

References

1. Olden JD, Jackson DA (2002) Illuminating the "black box'': a randomization approach for understanding variable contributions in artificial neural networks. Ecol Model 154(1–2):135–150. https://doi.org/10.1016/S0304-3800(02)00064-9
2. Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA (2017) Deep reinforcement learning: a brief survey. IEEE Signal Process Mag 34(6):26–38. https://doi.org/10.1109/MSP.2017.2743240
3. Heuillet A, Couthouis F, Rodríguez ND (2021) Explainability in deep reinforcement learning. Knowl Based Syst 214:106685. https://doi.org/10.1016/j.knosys.2020.106685
4. Molnar C (2022) Interpretable machine learning, 2nd edn. https://christophm.github.io/interpretable-ml-book/cite.html
5. Ribeiro MT, Singh S, Guestrin C (2016) "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, pp 1135–1144. https://doi.org/10.1145/2939672.2939778
6. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2020) Grad-cam: visual explanations from deep networks via gradient-based localization. Int J Comput Vis 128(2):336–359. https://doi.org/10.1007/s11263-019-01228-7
7. Wells L, Bednarz T (2021) Explainable AI and reinforcement learning - a systematic review of current approaches and trends. Front Artif Intell 4:550030. https://doi.org/10.3389/frai.2021.550030
8. Puiutta E, Veith EMSP (2020) Explainable reinforcement learning: a survey. Machine learning and knowledge extraction. Lecture notes in computer science, vol 12279. Springer, Cham, pp 77–95
9. Liu G, Schulte O, Zhu W, Li Q (2018) Toward interpretable deep reinforcement learning with linear model u-trees. In: Machine learning and knowledge discovery in databases - european conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, proceedings, Part II. Lecture notes in computer science, vol. 11052, pp. 414–429. https://doi.org/10.1007/978-3-030-10928-8_25
10. Amir O, Doshi-Velez F, Sarne D (2019) Summarizing agent strategies. Auton Agents Multi-Agent Syst 33(5):628–644. https://doi.org/10.1007/s10458-019-09418-w
11. Amir D, Amir O (2018) Highlights: summarizing agent behavior to people. In: Proceedings of the 17th International conference on autonomous agents and multiagent systems. AAMAS 2018. International foundation for autonomous agents and multiagent systems, Richland, SC, pp. 1168–1176. http://dl.acm.org/citation.cfm?id=3237869
12. Sequeira P, Gervasio MT (2020) Interestingness elements for explainable reinforcement learning: understanding agents'

capabilities and limitations. Artif Intell. https://doi.org/10.1016/j.artint.2020.103367

13. Topin N, Veloso M (2019) Generation of policy-level explanations for reinforcement learning. In: The Thirty-Third AAAI conference on artificial intelligence, pp. 2514–2521. https://doi.org/10.1609/aaai.v33i01.33012514
14. Huang SH, Held D, Abbeel P, Dragan AD (2019) Enabling robots to communicate their objectives. Auton Robots 43(2):309–326. https://doi.org/10.1007/s10514-018-9771-0
15. Zahavy T, Ben-Zrihem N, Mannor S (2016) Graying the black box: understanding dqns. In: Proceedings of the 33nd international conference on machine learning, ICML. JMLR Workshop and Conference Proceedings, vol. 48, pp. 1899–1908. http://proceedings.mlr.press/v48/zahavy16.html
16. Alharin A, Doan T, Sartipi M (2020) Reinforcement learning interpretation methods: a survey. IEEE Access 8:171058–171077. https://doi.org/10.1109/ACCESS.2020.3023394
17. Dazeley R, Vamplew P, Foale C, Young C, Aryal S, Cruz F (2021) Levels of explainable artificial intelligence for human-aligned conversational explanations. Artif Intell 299:103525. https://doi.org/10.1016/j.artint.2021.103525
18. Alqaraawi A, Schuessler M, Weiß P, Costanza E, Berthouze N (2020) Evaluating saliency map explanations for convolutional neural networks: a user study. In: IUI 2020: 25th international conference on intelligent user interfaces, Cagliari, Italy, March 17–20, 2020. ACM, New York, NY, USA, pp. 275–285 https://doi.org/10.1145/3377325.3377519
19. Iyer R, Li Y, Li H, Lewis M, Sundar R, Sycara KP (2018) Transparency and explanation in deep reinforcement learning neural networks. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018, pp. 144–150. ACM, New York, NY, USA. https://doi.org/10.1145/3278721.3278776
20. Atrey A, Clary K, Jensen DD (2020) Exploratory not explanatory: counterfactual analysis of saliency maps for deep reinforcement learning. In: 8th international conference on learning representations. https://openreview.net/forum?id=rkl3m1BFDB
21. Sutton RS, Precup D, Singh SP (1999) Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. Artif Intell 112(1–2):181–211. https://doi.org/10.1016/S0004-3702(99)00052-1
22. Kulkarni TD, Narasimhan K, Saeedi A, Tenenbaum J (2016) Hierarchical deep reinforcement learning: integrating temporal abstraction and intrinsic motivation. In: Advances in neural information processing systems, pp. 3675–3683. https://proceedings.neurips.cc/paper/2016/hash/f442d33fa06832082290ad8544a8da27-Abstract.html
23. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction, 2nd edn. The MIT Press, Cambridge, MA
24. Beyret B, Shafti A, Faisal AA (2019) Dot-to-dot: explainable hierarchical reinforcement learning for robotic manipulation. In: 2019 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp. 5014–5019. https://doi.org/10.1109/IROS40897.2019.8968488
25. Shu T, Xiong C, Socher R (2018) Hierarchical and interpretable skill acquisition in multi-task reinforcement learning. In: 6th international conference on learning representations, ICLR 2018 - conference track proceedings, vol. 3, pp. 1–14
26. Greydanus S, Koul A, Dodge J, Fern A (2018) Visualizing and understanding atari agents. In: Proceedings of the 35th international conference on machine learning. Proceedings of machine learning research, vol. 80, pp. 1787–1796. http://proceedings.mlr.press/v80/greydanus18a.html
27. Yang Z, Bai S, Zhang L, Torr PHS (2018) Learn to interpret atari agents, arXiv preprint arXiv:1812.11276
28. Annasamy RM, Sycara KP (2019) Towards better interpretability in deep q-networks. In: Proceedings of the AAAI conference on artificial intelligence, pp. 4561–4569. https://doi.org/10.1609/aaai.v33i01.33014561
29. Mott A, Zoran D, Chrzanowski M, Wierstra D, Rezende DJ (2019) Towards interpretable reinforcement learning using attention augmented agents. In: Advances in neural information processing systems 32: annual conference on neural information processing systems 2019, pp. 12329–12338. https://proceedings.neurips.cc/paper/2019/hash/e9510081ac30ffa83f10b68cde1cac07-Abstract.html
30. Olson ML, Khanna R, Neal L, Li F, Wong W (2021) Counterfactual state explanations for reinforcement learning agents via generative deep learning. Artif Intell 295:103455. https://doi.org/10.1016/j.artint.2021.103455
31. Yau H, Russell C, Hadfield S (2020) What did you think would happen? Explaining agent behaviour through intended outcomes. Adv Neural Inf Process Syst 33:18375–18386
32. Dodson T, Mattei N, Goldsmith J (2011) A natural language argumentation interface for explanation generation in markov decision processes. In: Algorithmic Decision Theory. Lecture Notes in Computer Science, vol. 6992, pp. 42–55. https://doi.org/10.1007/978-3-642-24873-3_4
33. van der Waa J, van Diggelen J, van den Bosch K, Neerincx MA (2018) Contrastive explanations for reinforcement learning in terms of expected consequences. In: IJCAI-18 Workshop on explainable AI arxiv:1807.08706
34. Huber T, Weitz K, André E, Amir O (2021) Local and global explanations of agent behavior: integrating strategy summaries with saliency maps. Artif Intell 301:103571. https://doi.org/10.1016/j.artint.2021.103571
35. Russell SJ, Zimdars A (2003) Q-decomposition for reinforcement learning agents. In: Proceedings of the 20th international conference on machine learning (ICML-03), pp. 656–663. http://www.aaai.org/Library/ICML/2003/icml03-086.php
36. Juozapaitis Z, Koul A, Fern A, Erwig M, Doshi-Velez F (2019) Explainable reinforcement learning via reward decomposition. In: Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence, pp. 47–53
37. Anderson A, Dodge J, Sadarangani A, Juozapaitis Z, Newman E, Irvine J, Chattopadhyay S, Fern A, Burnett M (2019) Explaining reinforcement learning to mere mortals: an empirical study. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19, pp 1328–1334. https://doi.org/10.24963/ijcai.2019/184
38. Sutton RS, Maei HR, Precup D, Bhatnagar S, Silver D, Szepesvári C, Wiewiora E (2009) Fast gradient-descent methods for temporal-difference learning with linear function approximation. In: Proceedings of the 26th annual international conference on machine learning, vol. 382. Association for Computing Machinery, New York, NY, USA, pp. 993–1000 https://doi.org/10.1145/1553374.1553501
39. Schulman J, Levine S, Abbeel P, Jordan MI, Moritz P (2015) Trust region policy optimization. In: Proceedings of the 32nd international conference on machine learning. JMLR Workshop and Conference Proceedings, vol. 37, pp. 1889–1897. http://proceedings.mlr.press/v37/schulman15.html
40. Watkins CJCH, Dayan P (1992) Technical note q-learning. Mach Learn 8:279–292. https://doi.org/10.1007/BF00992698
41. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller MA, Fidjeland A, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533. https://doi.org/10.1038/nature14236

42. Schaul T, Quan J, Antonoglou I, Silver D (2016) Prioritized experience replay. In: 4th International conference on learning representations, ICLR. http://arxiv.org/abs/1511.05952
43. van Hasselt H, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. In: Proceedings of the Thirtieth AAAI conference on artificial intelligence, pp. 2094–2100. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12389