

A Survey of Explainable Reinforcement Learning: Targets, Methods and Needs

LÉO SAULIÈRES*, Univ Toulouse, INRAE, MIAT, France

The success of recent Artificial Intelligence (AI) models has been accompanied by the opacity of their internal mechanisms, due notably to the use of deep neural networks. In order to understand these internal mechanisms and explain the output of these AI models, a set of methods have been proposed, grouped under the domain of eXplainable AI (XAI). This paper focuses on a sub-domain of XAI, called eXplainable Reinforcement Learning (XRL), which aims to explain the actions of an agent that has learned by reinforcement learning. We propose an intuitive taxonomy based on two questions ‘What’ and ‘How’. The first question focuses on the target that the method explains, while the second relates to the way the explanation is provided. We use this taxonomy to provide a state-of-the-art review of over 250 papers. In addition, we present a set of domains close to XRL, which we believe should get attention from the community. Finally, we identify some needs for the field of XRL.

JAIR Track: Surveys

JAIR Associate Editor: Ofra Amir

JAIR Reference Format:

Léo Saulières. 2026. A Survey of Explainable Reinforcement Learning: Targets, Methods and Needs. *Journal of Artificial Intelligence Research* 87, Article 16 (October 2026), 88 pages. DOI: [10.1613/jair.1.20550](https://doi.org/10.1613/jair.1.20550)

1 Introduction

This paper presents a state of the art on explainable and transparent reinforcement learning. It brings together a set of relatively recent works that present new methods specific to Reinforcement Learning (RL) as well as papers using methods derived from the eXplainable Artificial Intelligence (XAI) domain, originally used to explain classifiers, such as LIME (Ribeiro et al. 2016) or SHAP (Lundberg and S. Lee 2017).

RL is a machine learning paradigm where an agent learns to make a sequence of actions within an environment. Given a set of information defined as a state, the agent chooses an action at each time-step, arrives in a new state and receives a reward, determined by the environment’s dynamics (transition and reward functions). The agent’s goal is to maximize its cumulative reward (also called return) by learning an optimal policy.

In addition to the eXplainable RL (XRL) methods, we briefly present a set of sub-domains of RL whose main motivation is the performance and generalisation of agent policies. Indeed, they can also be used to explain or make transparent the agent’s behavior. For example, Relational RL (RRL) (Dzeroski et al. 2001) consists of the agent reasoning on the basis of *relations* and *objects* rather than reasoning directly with raw data. The relationships on which the agent relies can then be used to explain its behavior.

This overview is based on a total of 12 states of the art (Alaarabiou et al. 2024; Dazeley et al. 2023; Gajcin and Dusparic 2024c; Glanois et al. 2024; Heuillet et al. 2021; Hickling et al. 2024; Krajna et al. 2022; Milani, Topin et al. 2024; Puiutta and Veith 2020; Vouros 2023; Wells and Bednarz 2021; Zelveder et al. 2021) and complementary

*Corresponding Author.

Author’s Contact Information: Léo Saulières, ORCID: [0000-0002-4800-9181](https://orcid.org/0000-0002-4800-9181), leo.saulieres@inrae.fr, Univ Toulouse, INRAE, MIAT, Castanet-Tolosan, France.



This work is licensed under a Creative Commons Attribution International 4.0 License.

© 2026 Copyright held by the owner/author(s).

DOI: [10.1613/jair.1.20550](https://doi.org/10.1613/jair.1.20550)

papers. The papers we present are those identified in the 12 surveys mentioned. Furthermore, we have selected papers in the field of XRL cited by the papers presented in the states of the art. In addition, in order to provide as comprehensive an overview as possible, we searched for papers via Google Scholar using the following keywords: ‘*Reinforcement Learning*’, ‘*Explainable*’, ‘*Interpretable*’, ‘*Understandable*’, ‘*Transparent*’. This additional selection was based solely on verifying that papers propose or use an XRL method, without considering the number of citations. The cutoff year for paper selection was 2024.

Our paper has four objectives:

- To reflect the methods designed to explain/make transparent agents that have learned by reinforcement.
- To present works that use model-agnostic methods.
- To briefly describe domains related to XRL on which researchers should focus.
- To draw a list of the needs for the XRL domain.

In this state of the art, we present methods aimed at *providing an explanation* as well as those aimed at *making the agent’s behavior transparent* by design. We believe it is important to cover both approaches in order to give the reader an overview of the methods used to make the agent’s behavior understandable to the user. By abuse of language, we refer to both approaches when talking about explanations or XRL in this paper.

Among researchers in the field of XRL, there is no consensus on the taxonomy to be used to classify the various works. To illustrate, Puiutta and Veith use the taxonomy of classifiers, Milani et al. separate the methods using categories that better reflect the aspects of RL and Dazeley et al. use the Causal Explanation Framework (Böhm and Pfister 2015) to propose a conceptual framework for XRL.

In the absence of a standard taxonomy, we propose in this paper a new taxonomy based simply on 2 questions: *What* and *How*. More specifically, the first question is used to determine what the method is trying to explain: “*What does the method want to explain? What does it want to make transparent?*”. We have identified three elements: the agent’s *policy*, a particular *sequence* of interactions between the agent and the environment, and a particular *action* by the agent in a given situation. The second question allows us to refine the taxonomy, by looking at the way in which the explanation is provided: “*How is it explained?*”. Thus, a set of ways of providing explanations are described in this state of the art, for each of the three above-mentioned elements. An overview of the taxonomy is presented in Figure 1, where an illustrative work is associated with each sub-category.

We believe that this taxonomy can help readers to quickly identify a body of work that is relevant to the question they want to answer (and also the way in which they want to answer it). This state of the art also makes it possible to determine trends, whether in terms of elements explained or approaches used.

The rest of the paper is structured as follows. Section 2 provides the background of reinforcement learning. Section 3 describes related work. Section 4 presents the methods explaining the agent’s policy. Section 5 describes the methods explaining the agent’s behavior in a sequence of interactions with the environment. Section 6 presents the methods explaining an agent’s choice of action from a state. Section 7 briefly describes domains related to XRL that we think researchers should pay attention to. Section 8 highlights the needs for the XRL domain according to the different surveys we studied.

2 Background

An RL problem is described by a Markov Decision Process (MDP), which is a tuple $\langle \mathcal{S}, \mathcal{A}, R, p \rangle$, where: $\mathcal{S}$ and $\mathcal{A}$ denotes respectively the state space and action space; $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function that provides a scalar feedback signal evaluating the immediate outcome of an action taken in a given state; $p : \mathcal{S} \times \mathcal{A} \rightarrow Pr(\mathcal{S})$ is the transition function that defines the probability distribution over next states given the current state and action. States are represented by a set of features $\{f_1, f_2, \dots, f_n\}$, where each feature f_i takes values in a domain D_i .

The agent’s goal is to learn a policy π that maximizes the expected cumulative reward, or return. A policy can be either deterministic, mapping each state to an action $\pi(s) = a$, or stochastic, mapping each state to a

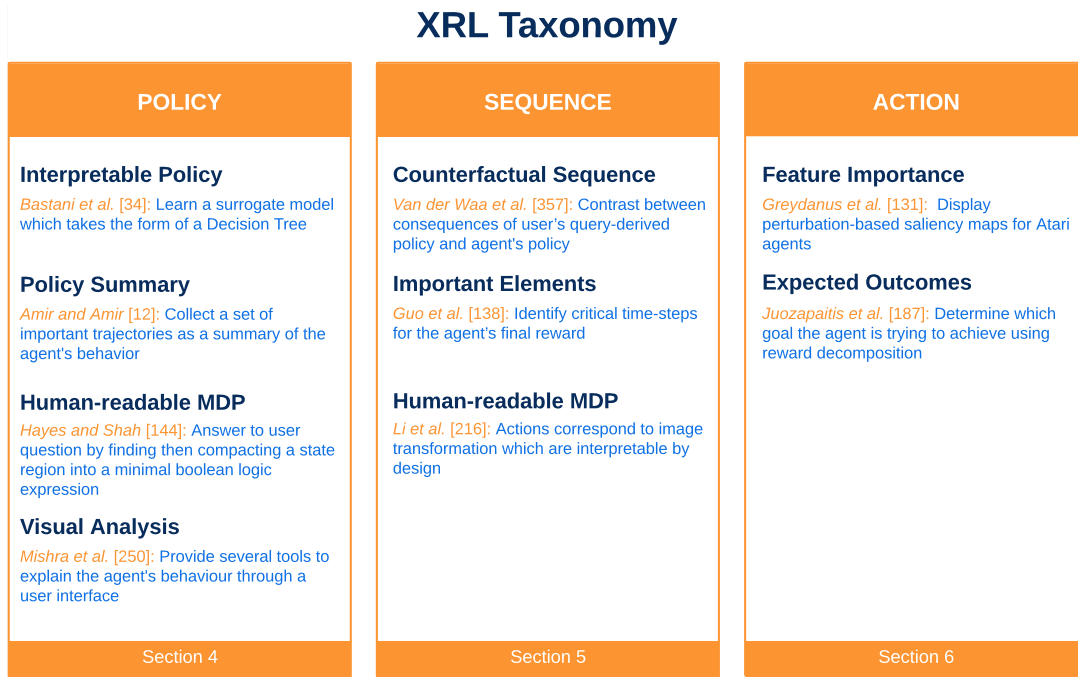


Fig. 1. Our taxonomy for Explainable Reinforcement Learning.

probability distributions over actions $\pi(a|s)$. To evaluate policies, value functions are defined: the state-value function $V^\pi(s)$ estimates the expected return from state s when following policy π , and the action-value function $Q^\pi(s, a)$ estimates the expected return when taking action a in state s and subsequently following π .

Q-learning is an RL algorithm that learns an approximation of the optimal action-value function $Q^*(s, a)$. It updates estimates iteratively using sampled transitions from interaction within the environment. The update rule is as follows: $Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$ where α is the learning rate, γ the discount factor, r the received reward, and s' the next state. The learned Q -values allow the agent to derive a policy by selecting actions with maximal expected value.

Several extensions of the standard MDP formulation have been proposed to address more complex settings, e.g. Partially Observable MDP (POMDP) and Multi-Agent RL (MARL). In MARL, multiple agents interact within the same environment and must learn while accounting for the behavior of other agents. In a POMDP, the agent does not have a full visibility of the state it is in. Instead, the agent receives partial information about the state, called an observation.

For complex problems with large or continuous state/action spaces, value functions or policies can be approximated using neural networks (NN). Common NN architectures include Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN) for spatial data such as images, and Recurrent Neural Network (RNN) for sequential data or partially observable settings. Combining RL with NN has led to the development of Deep Reinforcement Learning (DRL) algorithms such as Deep Q-Network (DQN) (Mnih, Kavukcuoglu, et al. 2015), which approximates the Q -function with a NN, and Asynchronous Advantage Actor-Critic (A3C) (Mnih, Badia, et al. 2016), which jointly learns a policy and a value function.

3 Related Work

Several surveys on XRL have been proposed in recent years, differing both in their scope and in the taxonomies used to organize the literature (Alaarabiou et al. 2024; Amitai and O. Amir 2024; Dazeley et al. 2023; Gajcin and Dusparic 2024c; Glanois et al. 2024; Heuillet et al. 2021; Hickling et al. 2024; Krajna et al. 2022; Milani, Topin et al. 2024; Puiutta and Veith 2020; Vouros 2023; Wells and Bednarz 2021; Zelveder et al. 2021).

There is no consensus on which taxonomy to adopt for XRL, as the following examples illustrate. Puiutta and Veith distinguishing explanations by their scope and the time at which they are extracted. In terms of scope, a *local* explanation explains an agent's choice of action, whereas a *global* explanation describes the agent's policy as a whole. In terms of time, an *intrinsic* explanation means that the model can be interpreted by itself, whereas a *post-hoc* explanation means that the explanation is provided after the agent's learning phase. Milani et al. separate the methods using categories that better reflect the aspects of RL. Works are separated into three categories: *feature importance*, which consists of identifying the features of a state that impact the agent's decision, *learning process and MDP*, which shows the past experiences or the components of the MDP that lead to the agent's behavior and *policy level*, which describes the agent's long-term behavior. As a final example, Dazeley et al. use the Causal Explanation Framework (Böhm and Pfister 2015) to propose a conceptual framework for XRL, called the Causal XRL Framework. A simplified version is used to categorise the methods into two types: *perception* and *action*. The first category includes methods that focus on the impact of the agent's perception on its actions and outcomes. The second category includes methods that focus on the choice of action and its impact on the outcome.

Most of the surveys either focus on a specific subset of explanation methods or illustrate the landscape of XRL approaches using a relatively limited number of representative works. Among the states of the art, Glanois et al. focuses on the interpretability of RL, Amitai and O. Amir focuses on global explanations, and Gajcin and Dusparic focuses on counterfactual explanations whereas the others are not specific to one type of explanation. A counterfactual explanation is used to determine which part of the input (e.g. a state in RL) must be modified in order to change the model's output (e.g. the agent's policy). It is interesting to note that this type of method is under-represented in XRL (Gajcin and Dusparic).

In contrast to these surveys, this work aims to provide a broad overview of the XRL literature. To this end, we review more than 250 papers and introduce a taxonomy designed to organize the different approaches according to two complementary dimensions: the *target of the explanation* and the *way the explanation is produced*. This structure allows groups of related works to be quickly identified by jointly considering what target is explained and how the explanation is generated. Within this taxonomy, commonly used distinctions from the XAI literature fall within the 'How' dimension. In particular, *intrinsic* explanations correspond to methods in which the policy is interpretable by design (e.g. taking the form of a set of rules), whereas *post-hoc* explanations are generated after the learning phase without taking into account the interpretability of the policy to be explained. In the RL context, methods that make use of an MDP that is interpretable by design and those that transform an MDP into an interpretable MDP can be regarded as *intrinsic* and *post-hoc* methods respectively, given their aim of explaining the agent's policy. Similarly, methods can be distinguished according to whether they are *model-specific*, based on a specific type of policy (e.g. work focused on explaining DQN and RNN policies presented in this survey), or *model-agnostic*, treating the agent as a black box and deriving explanations from its observed behavior or through surrogate models.

The proposed taxonomy aims to facilitate the exploration of the XRL literature by providing a simple structure that highlights the relationships between explanation targets and explanation ways. By explicitly separating the explanation target from the way of explanation, it becomes easier to compare methods that explain similar targets using different techniques, or conversely methods that rely on similar way of explanation but different

targets. This perspective helps reveal connections between works that are often presented independently in the literature and provides a clearer overview of the XRL domain.

4 Policy-level Methods

Before delving into the state of the art, we briefly explain how to interpret Figures 2, 11, 14, which abstractly represent the taxonomy used for each target, as well as the order in which the groups of methods are presented in the paper. Each figure represents a left-to-right hierarchy of the taxonomy. In each figure, the leftmost element denotes the target of the explanation. Moving to the right, blue nodes indicate high-level ways of explanation. These ways are refined into orange nodes which are then refined into green nodes, corresponding to more specific approaches of explanation. Branching indicates that a way of explanation can be divided into sub-groups of work. In the remainder of the paper, for each target, the various groups of work are ordered in descending order according to the number of works identified per group.

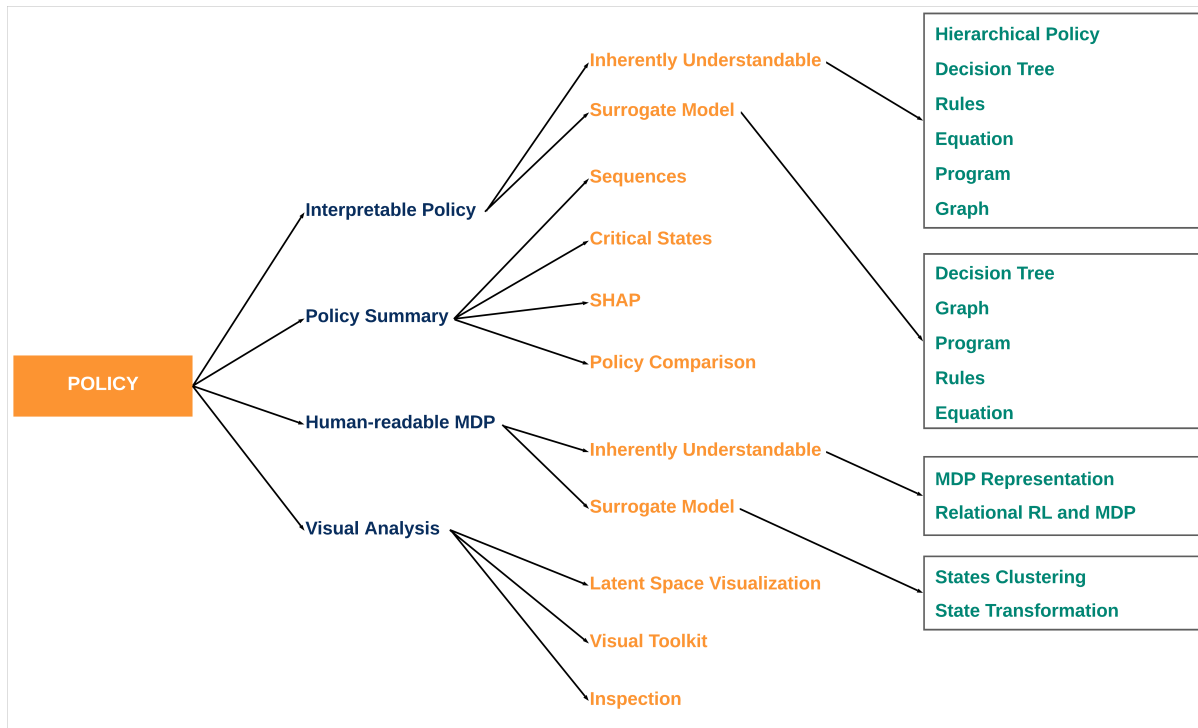


Fig. 2. Detailed taxonomy for policy-level methods.

As described in Figure 2, policy explanation is carried out using four ways of explanation. Constructing *Interpretable Policies* makes the agent’s reasoning transparent. *Summarising Policies* provides an overview of the agent’s capabilities. The use of *Human-readable MDP* allows the user to understand the information available to the agent, its actions and the dynamics of the environment. *Visual analysis* helps the user to study the agent’s policy visually, using various tools and metrics. *Interpretable Policies* are divided into two groups of work: learning an interpretable surrogate model of the policy and directly learning an interpretable policy. In this work, the policy can take different forms, such as a decision tree or an equation. A policy summary can be composed of

a set of state-action sequences or critical states, resulting from the use of SHAP, and can be used to compare policies. *Human-readable MDPs* are divided into two groups of work: Surrogate Model, which transforms the MDP to make it interpretable, and Inherently Understandable, which proposes an MDP that is interpretable by design. *Visual Analysis* brings together specific analysis of DQN latent spaces, visual interfaces and agent capacity inspection.

4.1 Interpretable Policy

A way of making the agent's behavior understandable to the user is to construct interpretable policies, thereby making the agent's decision-making transparent. To do this, the agent's policy is directly learned in such a way as to be interpretable, or an interpretable surrogate model is learned to imitates the agent's opaque decision-making. To clarify, although these methods enable us to explain the agent's decision making (i.e. its choice of action in a particular situation), we choose to classify this type of approach as policy explanation, because the main result of these methods is an interpretable form of the policy, and the action explanation is simply derived from it.

4.1.1 Inherently Understandable. A policy is said to be inherently understandable when it is interpretable by its form. In other words, each decision made by the agent can be explained by looking at the policy, which takes the form of a set of rules or an equation, for example. Unlike the surrogate model approach, which requires learning a second understandable policy that best imitates the first, which is opaque, the inherently understandable approach brings together a set of works that learn a single interpretable policy. We categorise these works according to the form of the policy, which can be a hierarchical policy, rules, a decision tree, a graph, an equation or a program. These different policy formats are presented below, along with some illustrative works.

Hierarchical Policy. Hierarchical RL (HRL) is a sub-domain of RL consisting in learning a policy which is structured in a hierarchical manner, generally comprising 2 levels, so as to break down the task to be performed into a set of sub-tasks. For each sub-task, a *low-level* policy is learned. A *high-level* policy is learned using the various *low-level* policies as an action. This approach was not originally designed to explain a policy, but to improve the performance and generalisation of policies on tasks that are decomposable. Therefore, in this section we do not pretend to provide an overview of HRL, but rather to present work that has been designed for explainability or that seems relevant to this purpose.

Of the 20 works in this group (cf. Table 1), 12 focus on the interpretability of the policies provided. (Andreas et al. 2017; Y. Jiang et al. 2019; Rossum et al. 2021; Shu et al. 2018) propose that low-level or high-level policies are explicitly represented using human-understandable symbolic labels or natural language descriptions. Model-based RL methods use the dynamics of the environment (i.e. the transition function and the reward function) to obtain an agent's policy. In a model-based HRL context, D. Xu and Fekri (2021) learn symbolic logical transition rules between sub-goals and B. Wu et al. (2019) learn multiple imperfect environment models and use a gating controller to compose modular interpretable policies. Three works (Chang et al. 2020; Jin et al. 2022; Lyu et al. 2019) combine RL with planning, where a high-level policy composed of symbolic actions is generated by a planner, and low-level policies are learned for each symbolic action. Rietz et al. (2023) combines HRL with reward decomposition (which will be introduced in Section 6.2.2) to obtain both policy-level and action-level explanations. Beyret et al. (2019) proposes an approach where the high-level agent iteratively generates sub-goals to guide the low-level agent, and Tasse et al. (2020) draws inspiration from a slightly different approach to HRL to obtain a new skill without learning, through the use of Boolean algebra on tasks.

In this paragraph, we mention some HRL methods or methods close to the domain which are not focused on explainability, but which are nonetheless worth mentioning. (Furelos-Blanco et al. 2021; Leonetti et al. 2016; Sridharan et al. 2019) use logic programming, e.g. Furelos-Blanco et al. mix Inductive Logic Programming (ILP) and RL by learning an automaton composed of edges which represent a sub-goal by a logic formula. (Eysenbach

Table 1. Interpretable Policy works.

Type		Refs
Inherently Understandable (54)	Hierarchical Policy (20)	Y. Jiang et al. 2019; Shu et al. 2018; D. Xu and Fekri 2021 Chang et al. 2020; Lyu et al. 2019; B. Wu et al. 2019 Beyret et al. 2019; Jin et al. 2022; Rossum et al. 2021 Andreas et al. 2017; Rietz et al. 2023; Tasse et al. 2020 Furelos-Blanco et al. 2021; Leonetti et al. 2016; Sridharan et al. 2019 Eysenbach et al. 2019; F. Yang, Lyu, et al. 2018; A. Zhang et al. 2018 Driessens and Blockeel 2001; Kulkarni et al. 2016
	Rules (12)	Akrour et al. 2019; Hein, Hentschel, et al. 2017; J. Huang et al. 2020 Z. Jiang and Luo 2019; Ou et al. 2024; L. Zhang et al. 2021 Hazra and Raedt 2023; Ma et al. 2021; Payani and Fekri 2020 Dong et al. 2019; Song et al. 2022; Zimmer et al. 2021
	Decision Tree (10)	Roth et al. 2019; Silva et al. 2019; Topin, Milani, et al. 2021 Cole et al. 2003; Custode and Iacca 2023; Paleja et al. 2022 Gupta et al. 2015; Likmeta et al. 2020 Das et al. 2020; Ernst et al. 2005
	Graph (5)	Inala et al. 2020; Y. Lin et al. 2022; T. Wang et al. 2018 Hasanbeig et al. 2021; Varma 2021
	Equation (4)	Hein, Udluft, et al. 2019; Landajuela et al. 2021 Ault et al. 2020; Maes et al. 2012
	Program (3)	Gu et al. 2024; Trivedi et al. 2021; Wilson et al. 2018
Surrogate Model (34)	Decision Tree (17)	Coppens et al. 2019; Dai, Q. Chen, et al. 2022; W. Guo and Wei 2022 Bastani, Pu, et al. 2018; Dhebar, Deb, et al. 2020; Ding et al. 2020 Milani, Z. Zhang, et al. 2022; Sieusahai and Guzdial 2021; Vasic et al. 2022 Acharya et al. 2020; Cichosz and Pawelczak 2014; G. Liu, Sun, et al. 2021 Carbone 2020; Løver et al. 2021; Skirzynski et al. 2021 Jhunjhunwala et al. 2020; G. Liu, Schulte, et al. 2018
	Graph (9)	Domenech i Vila et al. 2024; McCalmon et al. 2022 Kazhdan et al. 2020; Koul et al. 2019; Vinzent et al. 2022 Danesh et al. 2021; Hüyük et al. 2023; J. H. Lee 2019 Topin and M. Veloso 2019
	Program (5)	Kholer, Delfosse, et al. 2024; Verma, Murali et al. 2018; H. Zhu et al. 2019 Burke et al. 2019; Verma, H. M. Le, et al. 2019
	Rules (2)	Nagesh Rao et al. 2019; Soares et al. 2021
	Equation (2)	Kubalik et al. 2017; H. Zhang et al. 2020

et al. 2019; A. Zhang et al. 2018) build a graph to obtain high-level policies, and (Driessens and Blockeel 2001; Kulkarni et al. 2016) use variants of Q-learning in a hierarchical context.

We refer the reader to Appendix A.1.1 for more detailed summaries of these works.

Rules. The following works represent the policy of an agent by a set of rules. The 12 works identified (cf. Table 1) can be divided according to the type of rules used as policy. These rules are either fuzzy rules or logic-based rules.

A fuzzy rule describes a decision using membership functions that represent the degree to which a state satisfies a condition. A total of four works (Akrouir et al. 2019; Hein, Hentschel, et al. 2017; J. Huang et al. 2020; Ou et al. 2024) represent policies as if-then rules, where states belong to fuzzy sets through membership functions. To give an example, a model-based approach is presented in Hein, Hentschel, et al., entitled Fuzzy Particle Swarm RL. The aim is to learn a set of fuzzy rules offline with particle swarm optimization (Kennedy and Eberhart 1995), using NN's to represent world models. The produced rules take the form: 'if s is m then o ' where s is a state, m a Gaussian membership function and o a real number. The rules are visualised by presenting the membership functions and an example state. The learning of world models has an impact on the production of efficient policies: models that do not approximate the real dynamics correctly can cause the agent to learn the wrong policy.

A first-order logic rule describes a decision using symbolic predicates and logical relations. It takes the form a clause $a \leftarrow a_1, a_2, \dots, a_n$ where a is the head atom (i.e. the action to be performed) and $a_1, a_2, \dots, a_n$ the body atoms (i.e. the preconditions to be met). A total of eight works (Dong et al. 2019; Hazra and Raedt 2023; Z. Jiang and Luo 2019; Ma et al. 2021; Payani and Fekri 2020; Song et al. 2022; L. Zhang et al. 2021; Zimmer et al. 2021) represent policies as first-order logic if-then rules. Most of the works (Hazra and Raedt; Z. Jiang and Luo; Payani and Fekri; L. Zhang et al.) is based on DILP (Evans and Grefenstette 2018) to produce interpretable policies. This neuro-symbolic approach makes it possible to learn a set of symbolic rules weighted by a confidence score, while still having the advantages of using NNs. As an example, Deep Explainable Relational RL (Hazra and Raedt) represents an interpretable policy by a set of if-then rules. The objective is to find a policy such that given a state and background knowledge, an action is predicted. An NN is learned to produce rules and semantic constraints are proposed to avoid redundancy between rules.

Refer to Appendix A.1.1 for more detailed summaries of the mentioned works.

Decision Tree. The interpretable policy learnt in the following works takes the form of a Decision Tree (DT). A DT is a hierarchical structure composed of nodes and branches that allow instances to be classified. In our context, instances are the agent's states, and classes correspond in most works to actions. Thus, an internal node in a DT allows the state space to be split into two sub-regions, and a terminal node, called a leaf, associates a state with an action. The main difference in the work lies in the output, i.e. the leaves of the DT.

(Custode and Iacca 2023; Roth et al. 2019; Silva et al. 2019) methods represent Q-values in the leaves, making the DT a structured approximation of the Q-function, from which the policy is derived. (Gupta et al. 2015; Likmeta et al. 2020; Silva et al. 2019; Topin, Milani et al. 2021) methods approximate the policy directly, with leaves representing action distributions or actions. As an example, Differentiable DT (DDT) is used in Silva et al. The DDT can be used for Q-learning, where each leaf contains the Q-values of the actions, or for a gradient-based approach, where each leaf contains the distribution of the actions. The tree is then discretised to obtain an interpretable DT (an example for the CartPole environment (Brockman et al. 2016) is shown in Figure 3). In addition, a decision list can be extracted. The results show that these trees have a good performance compared to an MLP, while being more interpretable, according to the user study carried out. Alongside this work, Paleja et al. (2022) is learning about sparse linear controllers (i.e. based on a few features).

Among the 10 papers in this section (cf. Table 1), the following four works do not aim for interpretability, but we think it is interesting to mention them, because of their representation of the Q-function as a tree. Only a few works are presented, to give an overview. Ernst et al. (2005) approximates the Q-function via ensemble tree methods, (Cole et al. 2003; Das et al. 2020) approximates it by leveraging relational predicates and background knowledge (in a Relational MDP setting, described in more detail in Section 4.3.1), and K. S. Ng (2004) transforms the approximated Q-function into a compact Boolean representation of the policy.

For more details on this line of work, we refer the reader to Appendix A.1.1.

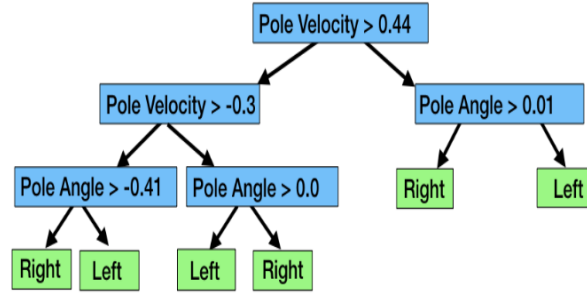


Fig. 3. A discrete decision tree for the CartPole environment (Silva et al. 2019). Each internal node of the tree (in blue) represents a condition on a feature of the agent's state, and each leaf of the tree (in green) corresponds to an action.

Graph. Five works represent policy using a graph (which is not a decision tree). A graph is a structure composed of a set of nodes and edges, allowing relationships between nodes to be represented via edges. Among these works, (Y. Lin et al. 2022; T. Wang et al. 2018) represent the policy using a graph structure reflecting the physical or relational structure of the task (robot skeleton or task entities), with interpretability obtained through graph analysis or visualization of learned representations. (Hasanbeig et al. 2021; Inala et al. 2020) represent the agent's policy as an explicit state machine (automaton), assimilated to a graph, where nodes are akin to sub-policies and transitions encode switching conditions or task progression, yielding interpretable control logic. Varma (2021) replaces neural networks with a Tsetlin Machine (Granmo 2018) that learns policies as sets of logical clauses. Although symbolic in nature, scalability issues (many clauses required) reduce interpretability. We refer the reader to Appendix A.1.1 for more details on this line of work.

Equation. The policy takes the form of an equation. This form is used when the agent's actions can be expressed as a function of the features of the agent's state. A total of four works fall into this group (cf. Table 1) and are briefly described below. In a model-based RL setting, the work of Hein et al. 2019 uses genetic programming to generate policy function trees, evaluating interpretability via a node-weighted complexity measure. Landalueja et al. 2021 propose a policy generator-evaluator framework: the RNN-based generator produces expression trees for continuous-control policies, trained via reward feedback from the policy evaluator. In Maes et al. (2012), policy search is framed as selecting the best small-sized formula from clustered candidates in a multi-armed bandit setting. Assuming that monotonicity over state features ensures interpretability in traffic control tasks, Ault et al. (2020) jointly learns Q-values and a monotonic (regulatable) precedence function. For more details, refer to Appendix A.1.1.

Program. A policy in the form of a program is proposed in the three following works (cf. Table 1). For the remainder of the survey, a program is defined as a symbolic, executable, and understandable policy representation expressed in a programming (or environment-specific) language, composed of control-flow constructs (e.g. conditionals, loops, sequencing) and interpretable primitives.

In order to obtain an interpretable policy based on a domain-specific language, Trivedi et al. (2021) learns a latent embedding space of programs, then searches this space (via Cross Entropy Method (Rubinstein 1997)) a latent program solving a given task. In Wilson et al. (2018), policies are obtained via Cartesian Genetic Programming (Miller and P. Thomson 2000), where programs are represented as directed graphs of predefined function nodes. A program expressed in a domain-specific language is constructed in Gu et al. (2024) using Monte Carlo Tree Search and refined via Bayesian optimization (Snoek et al. 2012), resulting in small and interpretable programs. Refer to Appendix A.1.1 for a more detailed summary of this work.

4.1.2 Surrogate Model. A surrogate model is an interpretable substitute for an agent's opaque policy. It is used to understand the agent's behavior. This section organises the works according to the form taken by the surrogate model.

Decision Tree. The following works propose a surrogate model in the form of a DT. This surrogate model format is the most widely used in the XRL literature, with a total of 17 works identified (cf. Table 1).

Most of the work approximates the agent policy with a DT (i.e., the output of the DT is an action), whereas (Jhunjunwala et al. 2020; G. Liu, Schulte, et al. 2018) are interested in approximating Q values (the output of the DT is a Q value). As an example, Jhunjunwala et al. proposes a new structure, called Q-BSP Tree, which models the Q-value in an interpretable way and an Ordered Sequential Monte Carlo algorithm which learns to approximate the Q-value of the DQN. For a problem comprising n actions, the Q-BSP forest is composed of n Q-BSP trees, each learning the Q-value of one action.

In order to make nodes interpretable, two approaches stand out. The first approach consists of representing nodes by linear functions on state features, thereby enabling an understanding of the influence of features on the action choice (Carbone 2020; Coppens et al. 2019; Dai, Q. Chen, et al. 2022; W. Guo and Wei 2022; Jhunjunwala et al.; G. Liu, Schulte, et al.; Løver et al. 2021). As an example, Dai et al. propose an *information gain rate weighted oblique DT* (IGR-WODT) to extract an interpretable policy from a DRL agent. Based on three defined metrics, IGR-WODT provides a policy that requires a much smaller number of parameters (only 1.1% of the number of DRL model parameters). An *oblique DT* is composed of internal nodes that separate data based on linear combinations of features (i.e. of the form $w_1f_1 + w_2f_2 + \dots + w_nf_n > 0$ where w_i are weights, f_i are state features and n is the number of features). The WODT (B. Yang et al. 2019), on which IGR-WODT is based, uses a logistic regression model to determine the probability of an instance belonging to the left or right child node. The extension provided by IGR-WODT consists of using a different objective function inspired by the information gain (Quinlan 1986). The second approach to making nodes interpretable consists of modifying the representation of states so as to learn a DT based on interpretable features rather than raw features (Acharya et al. 2020; G. Liu, Sun, et al. 2021; Sieusahai and Guzdial 2021; Skirzynski et al. 2021). These interpretable features can take the form of objects or sprites extracted from an image (G. Liu, Sun, et al.; Sieusahai and Guzdial) or predicates (Acharya et al.; Skirzynski et al.). To illustrate, the method proposed by Sieusahai et al. involves learning a DT based not on the image but on a set of *sprites* associated with their position in the image. A sprite is an image representing a character or object within a game. To learn the DT, a function transformation is used to extract a set of sprites with their coordinates from the image representing the state of the DRL agent. The DT learns to approach the DRL agent based on this more compact representation of the agent's state, which makes the internal nodes of the tree interpretable.

Among the works, only one focuses on the multi-agent aspect (Milani, Z, Zhang, et al. 2022).

For a more detailed summary of the 17 works, please refer to Appendix A.1.2.

Graph. The surrogate models described in this section are an abstract policy represented by a graph. In most of the work in this section, a node of the graph represents an (abstract) state of the environment, and an edge between two nodes corresponds to an agent's action coupled with an environment transition. The aim of the nine works (Danesh et al. 2021; Domenech i Vila et al. 2024; Hüyük et al. 2023; Kazhdan et al. 2020; Koul et al. 2019; J. H. Lee 2019; McCalmon et al. 2022; Topin and M. Veloso 2019; Vinzent et al. 2022) (cf. Table 1) is to build a graph that zooms out the agent's policy.

Five of these works (Domenech i Vila et al.; Kazhdan et al.; McCalmon et al.; Topin and M. Veloso; Vinzent et al.) propose to calculate abstract states to build a graph modelling the agent's high-level policy. The method used to produce abstract states differs: Topin and M. Veloso groups states according to the action performed by the agent from a state and the importance of the state features; McCalmon et al. calculates abstract states using a clustering algorithm (B. Liu et al. 2005) and then describes them using a natural language sentence; (Domenech i Vila et al.; Kazhdan et al.; Vinzent et al.) are based on a set of predefined predicates to determine abstract states.

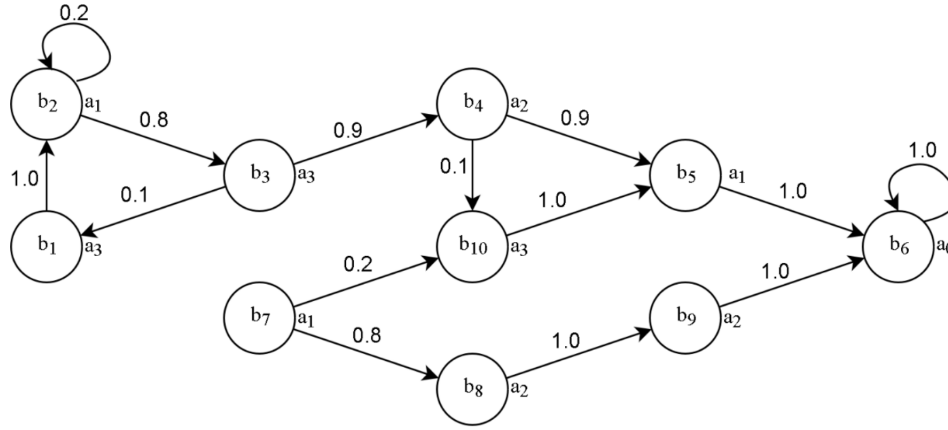


Fig. 4. Example of Abstract Policy Graph (Topin and M. Veloso 2019). Each vertex is an abstract state with an associated action, and the edges between two vertices represent transitions (weighted by their probability of occurring).

To illustrate, Topin and M. Veloso presents *abstracted policy graphs* (APG), which are Markov chains of abstract states. In an APG, edges are transitions labelled by their probability of occurring, nodes are abstract states labelled by the agent action performed from that state. An abstract state represents a grouping of similar states that is constructed with respect to the actions and feature importance of the states, calculated via FIRM (Zien et al. 2009). The transition probability from node S to node S' is an average of the transition probabilities on the states represented by S and S' . To build the APG, a policy, a value function and a set of data on the agent's interactions within the environment are required. An example of APG is described in Figure 4.

Two works (Danesh et al.; Koul et al.) focus on transforming a policy that takes the form of a Recurrent NN (RNN) into a Moore Machine (MM) (Moore 1956), which is a finite state machine whose output depends only on the current state and where each state is labelled by an output value (an action in this context).

While most work focuses on explaining policies in a single-agent context, (Domenech i Vila et al.; Kazhdan et al.) focuses on the multi-agent context.

For further details on the works of this section, we refer the reader to Appendix A.1.2.

Program. Using an environment-specific language, the program is produced based directly on DRL policies in (Verma, H. M. Le, et al. 2019; Verma, Murali, et al. 2018; H. Zhu et al. 2019). Verma, H. M. Le, et al. iteratively search for a policy represented by a neural network using policy-gradient methods and to project the policy into the space of programmatic policies (i.e. policies representable by programs). Based on a fixed DRL agent's policy, (Verma, Murali, et al.; H. Zhu et al.) constrain the program's search space in order to provide an interpretable policy within a reasonable time.

An intermediate step is used in (Burke et al. 2019; Kohler, Delfosse, et al. 2024) to obtain a program expressed via Python programming language: Kohler, Delfosse, et al. proposes learning a policy by imitation in the form of a regularised oblique tree (Murthy et al. 1994), then converting it into a program, and Burke et al. calculates a sequence expressed by interpretable low-level motion primitives in a robot manipulation task context, then uses program induction to simplify the sequence into a program.

Refer to Appendix A.1.2 for a more detailed description of this work. Moreover, for a detailed overview of program synthesis for interpreting DRL agents, we recommend Bastani, Inala, et al. (2020) to the interested reader.

Rules. Two works (cf. Table 1) present a set of if-then rules that act as a surrogate model. In this work, an if-then rule associates an action or a model that returns an action with a region of the state space. The aggregation of these rules makes it possible to approximate the policy of the DRL agent. In Nagesh Rao et al. (2019), a learned rule represents a region of the state space and outputs an action via a linear model, whereas in Soares et al. (2021), a learned rule associates an action with each state similar to a prototype.

For more details, refer to Appendix A.1.2.

Equation. The substitute policy takes the form of equations in the following two works (cf. Table 1). It is in a context where the agent's action is determined as a function of continuous features of the agent's state. Genetic algorithms are used to find such policies: H. Zhang et al. (2020) use the evolution feature synthesis (Arnaldo et al. 2015) and Kubalík et al. (2017) a variant of Single Node Genetic Programming (Jackson 2012). To obtain concise equations, Kubalík et al. determines a complexity limit (defined by the number of features used in the equation) and H. Zhang et al. proposes a method based on the complexity-performance trade-off.

We refer the reader to Appendix A.1.2 for more details on this work.

Table 1 lists *Interpretable Policy* works. With a total of 88 works mentioned (including 34 in *Surrogate Model* and 54 in *Inherently Understandable*), *Interpretable Policy* is the most widely used ways of explaining policies. The most commonly used and least commonly used policy forms in the works are decision trees and equations, respectively.

4.2 Policy Summary

In the blue sky papers (O. Amir et al. 2018, 2019), O. Amir *et al.* propose a framework for the Strategy Summarization problem that can be decomposed into three parts: World States Representation, Intelligent States Extraction and Strategy Summary Interface. World States Representation is concerned with encoding the state space to make it comprehensible to the user, for example by grouping states into abstract states. Intelligent States Extraction presents different directions for extracting states that are useful for the user's purpose. The summary should be of reasonable length but still provide sufficient information to the user. Strategy Summary Interface provides considerations for user interface design. In addition, this work presents ideas for methods of evaluating summaries, such as domains or metrics.

Table 2. Policy Summary works.

Type	Refs
Sequences (13)	D. Amir and O. Amir 2018; Huber, Weitz, et al. 2021; Septon et al. 2023 Amitai, Septon, et al. 2024; Feit et al. 2022; Sequeira and Gervasio 2020 S. H. Huang, Held, et al. 2019; Lage et al. 2019a,b; Sequeira and Gervasio 2023 Altmann et al. 2024; Frost et al. 2022; Gottesman et al. 2020
Critical States (8)	Dao, Huff, et al. 2021; S. H. Huang, Bhatia, et al. 2018; Rupprecht et al. 2020 Dao, I. Mishra, et al. 2018; Dinu et al. 2020, I. Mishra et al. 2018 Kazak et al. 2019; Sreedharan, Srivatsava, and Kambhampati 2020
SHAP (6)	Sequeira and Gervasio 2023; Y. Wang et al. 2020; K. Zhang et al. 2022 Angelotti and Díaz-Rodríguez 2023; Beechey et al. 2023; Heuillet et al. 2022
Policy Comparison (2)	Amitai and O. Amir 2022; Gajcin, Nair, et al. 2021

4.2.1 Sequences. The 13 works (cf. Table 2) in this section provide the user with a summary in the form of a set of sequences, composed of state-action pairs. These sequences are the result of the agent's interaction (via its policy) within the environment and thus reflect its behavior. The main objective of (Altmann et al. 2024; D. Amir and O. Amir 2018; Amitai, Septon, et al. 2024; Feit et al. 2022; Frost et al. 2022; S. H. Huang, Held, et al. 2019; Huber, Weitz, et al. 2021; Lage et al. 2019a,b; Septon et al. 2023; Sequeira and Gervasio 2020, 2023) is to provide a limited number of informative sequences for the user, with the aim of producing an overview of the strengths and/or weaknesses of the agent's policy.

As an example, the HIGHLIGHTS algorithm (D. Amir and O. Amir) provides a set of sequences that summarise the agent's behavior. To select sequences, D. Amir and O. Amir focus on important states using the notion of *state importance* (Clouse 1996) (which uses the agent's Q-values). From the important states, sequences are created to provide the user with more context. Thus, a sequence includes states that precede the important state, the state itself, and states that follow it. In the same work, the HIGHLIGHTS-DIV algorithm is proposed to provide more varied sequences as a summary.

For a more detailed summary of the various works on policy summaries via sequences, refer to Appendix A.2.1.

4.2.2 Critical States. An agent's policy can be summarised simply as a set of so-called critical states in the agent's interactions with the environment. The definition of a critical state varies according to the 8 works identified (cf. Table 2), but the overall idea is a state that has a significant impact on the agent's behavior and allows the policy to be understood through examples. Critical states are identified by constructing a dense reward function (Arjona-Medina et al. 2019), using Q-values (S. H. Huang, Bhatia, et al. 2018; Rupprecht et al. 2020), via compilation and then constructing a sequence of states characterising the policy (Sreedharan, Srivastava, and Kambhampati 2020).

To illustrate this line of work, Sreedharan, Srivastava, and Kambhampati propose a policy summary in the form of milestones, called landmarks, that the agent seeks to reach. The summary is a partial order of landmarks. For the exploding blocks-world problem (Younes 2005), an ordering of landmarks is described in Figure 5. The identification of landmarks is done using compilation procedures and the ordering between these landmarks using methods for reasoning about task hierarchies in MDP's.

Other work focuses on summarising learning by identifying the states that had the greatest impact on agent training (Dao, Huff, et al. 2021; Dao, I. Mishra, et al. 2018; I. Mishra et al. 2018), and Kazak et al. (2019) proposes a state to the user as a counterexample of an unverified requirement in the context of formal NN verification. We refer the reader to Appendix A.2.2 for more details on work summarising policy by critical states.

4.2.3 SHAP. This work is based on the model agnostic method, called SHAP (Lundberg and S. Lee 2017). It is interesting to note that this method is mainly used to explain classifiers. SHAP consists of determining the impact of each feature of an input on the output of the model. In our context, the input is a state, the output an action, and the model the agent's policy. This method is an approximation to the computation of Shapley values (Shapley et al. 1953). Originally from the field of game theory, Shapley values are used to determine the extent to which a member of a coalition contributes to the final value obtained. To determine Shapley values, it is necessary to calculate for each player m of the game, the set of possible coalitions c containing m , and to see the impact of its removal from c on the final value. SHAP proposes different approximations to Shapley values, which consist in limiting the number of coalition samples. Two approximations have been used in the work presented in this survey: Kernel SHAP, a model-agnostic estimation, and DeepSHAP, an estimation specific to neural networks. Kernel SHAP is based on Linear LIME (Ribeiro et al. 2016) (this method will be described in Section 6.1.2) and DeepSHAP on DeepLIFT (Shrikumar et al. 2016).

As this section is concerned with the explanation of policies, SHAP is used globally: it identifies the impact of each feature on the agent's policy. To clarify, SHAP values are calculated for each state of an episode, or even a set of episodes of interactions between the agent and the environment. These values are then usually provided to

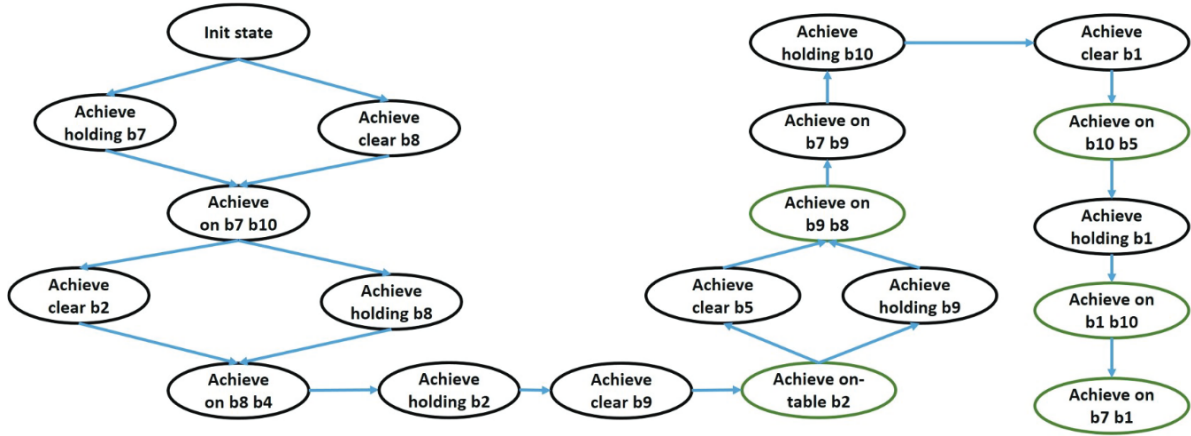


Fig. 5. Landmarks and their ordering obtained using TLdR (Sreedharan, Srivastava, and Kambhampati 2020) for the exploding blocks-world problem (Younes 2005). Vertices are landmarks and each blue edge defines an order between two landmarks. The green nodes are goals that the agent must reach.

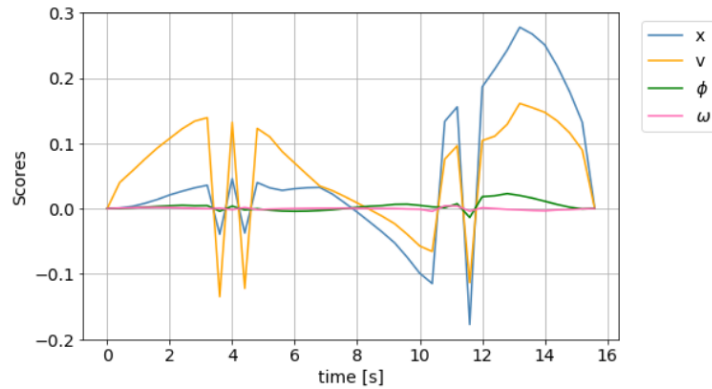


Fig. 6. SHAP for a set of states along an episode for a machine control use case (Y. Wang et al. 2020). The evolution of the importance score of each feature is represented by a colored curve in the chart, this importance varying over time.

the user in plots, such as the one shown in Figure 6 from Y. Wang et al. (2020). Y. Wang et al.; K. Zhang et al. (2022) use the method as is (K. Zhang et al. use DeepSHAP and Y. Wang et al. use Kernel SHAP), while other works (Angelotti and Díaz-Rodríguez 2023; Beechey et al. 2023; Heuillet et al. 2022; Sequeira and Gervasio 2023) calculate SHAP values differently. For example, in Sequeira and Gervasio, SHAP values are calculated using XGBoost machines (T. Chen and Guestrin 2016) to simplify the analysis, and in Heuillet et al., they are calculated via Monte Carlo sampling in a cooperative MARL context. For a more detailed summary of the 6 works (cf. Table 2), refer to Appendix A.2.3.

4.2.4 Policy Comparison. Two works have been found to compare two policies. This type of method makes it possible to evaluate two agents by identifying the differences in their behavior. Amitai and O. Amir (2022)

identifies important states in terms of performance differences between two policies, while Gajcin, Nair, et al. (2021) identifies them in terms of preference differences. We refer the reader to Appendix A.2.4 for more details.

Table 2 lists the Policy Summary works. Among the 29 works, the majority propose summaries composed of state-action sequences, and only two works focused on policy comparisons.

4.3 Human-readable MDP

This section brings together a set of works explaining policy using an environment that is (partially) interpretable by a user. Since the MDP is a tuple composed of a state space, an action space, a transition function and a reward function, the works focus on making one or more elements of the MDP interpretable, either directly (i.e. the component is interpretable by design) or via a surrogate model.

Table 3. Human-readable MDP works.

Type		Refs
Inherently Understandable (33)	MDP Representation (20)	Bica et al. 2021; Jenner and Gleave 2022; Saldiran et al. 2024 Devidze et al. 2021; Srinivasan and Doshi-Velez 2020 Bewley and Lécué 2022; Camacho et al. 2019 Gaon and Brafman 2020; Icarte, Klassen, et al. 2018 Kaiser et al. 2019; Kasenberg and Scheutz 2017; X. Li et al. 2017 Degris et al. 2006; Goel et al. 2018; Metzen 2013 Finkelstein et al. 2022; Jacq et al. 2022; Tabrez and Hayes 2019 Icarte, Waldie, et al. 2019; Z. Xu et al. 2020
	Relational RL and MDP (13)	Diuk et al. 2008; Marom and Rosman 2018; Scholz et al. 2014 Kansky et al. 2017; Veerapaneni et al. 2019; Walker et al. 2007 Martínez et al. 2017; Walsh 2010; G. Zhu, Z. Huang, et al. 2018 Agnew and Domingos 2018; G. Zhu, J. Wang, et al. 2020 Adjodah et al. 2018; Zambaldi et al. 2018
Surrogate Model (16)	States Clustering (8)	Bekkemoen and Langseth 2023; Hayes and Shah 2017 Climent et al. 2021; Iucci et al. 2021 Booth et al. 2019; Wollenstein-Betech et al. 2020 Bewley and Lawry 2021; Bewley, Lawry, and Richards 2022
	State Transformation (8)	Garnelo et al. 2016; Raffin et al. 2018 Dai, Ouyang, et al. 2023; Garcez et al. 2018 François-Lavet et al. 2019; A. Zhang et al. 2018 J. Chen et al. 2022; Gajcin and Dusparic 2022

4.3.1 Inherently Understandable. The following works directly represent the MDP in order for it to be (partially) interpretable. In comparison with the next section, the MDP is interpretable from its definition. The following works focus on the reward function, the transition function and the state space.

MDP Representation. This section groups works that represents the MDP, especially the dynamics of the environment (i.e. the reward function and the transition function) in an interpretable way. 11 of the 20 works (cf. Table 3) do not have explainability as their primary objective, but are mentioned because of the interpretable form of the reward and/or transition function proposed.

In (Bewley and Lécué 2022; Bica et al. 2021; Camacho et al. 2019; Devidze et al. 2021; Gaon and Brafman 2020; Icarte, Klassen, et al. 2018; Icarte, Waldie, et al. 2019; Jenner and Gleave 2022; Kasenberg and Scheutz 2017; X. Li et al. 2017; Saldiran et al. 2024; Srinivasan and Doshi-Velez 2020; Tabrez and Hayes 2019; Z. Xu et al. 2020), interpretability is based solely on the reward function. Decomposition methods (Bica et al.; Saldiran et al.) propose to express reward as (weighted) interpretable components (reward objectives (Saldiran et al.) or state features (Bica et al.)) whose relative importance can be visualized or analyzed. To simplify dense rewards into sparse, interpretable ones while preserving optimal policies, transformation methods are proposed (Devidze et al.; Jenner and Gleave). Tree-based reward models (Bewley and Lécué; Degris et al. 2006; Srinivasan and Doshi-Velez) represent (part) of the reward function through tree structures. Preference-based RL (PbRL) consists of solving a problem modelled by an MDP where the reward function is not provided. A fitness function reflecting the user's preference on pairs of trajectories is used to learn a reward function upstream of the agent's policy. As an example of tree-based reward model, Bewley and Lécué propose a PbRL algorithm that represents the interpretable reward function as a binary tree structure where each internal node is a test on a state-action pair and each leaf returns a reward. An example is shown in Figure 7. Although not directly focused on explainability, some works propose representing rewards in an interpretable manner, such as a finite state automaton (Camacho et al.; Gaon and Brafman; Icarte, Klassen, et al.; Icarte, Waldie, et al.; Z. Xu et al.) or a logical specification (Kasenberg and Scheutz; X. Li et al.) capturing the temporal structure of the objective.

The transition function is interpretable in Degris et al.; Goel et al. 2018; Kaiser et al. 2019; Metzen 2013. Kaiser et al. proposes to learn a transition function using a Bayesian approach where the causal sources influencing transitions are explicit, and Goel et al. proposes to learn a transition function in terms of the movement of objects on states that are images. Precisely, Motion-Oriented REinforcement Learning (MOREL) (Goel et al. 2018) improves the interpretability and, depending on the problem, the sample complexity of the RL algorithm used, by tracking moving objects in states. This approach is tested for a set of 59 Atari Games, where the state of an agent takes the form of an image. First, an unsupervised method is applied to learn to predict object and camera motions as well as object masks using a random policy to generate the data. Then, the agent uses and refines the previously learned model to learn its policy. Users can view the object masks to understand the agent's policy. Metzen and Degris et al. represent the function respectively by a graph and a DT. Finkelstein et al. (2022) proposes a contrastive explanation based on a sequence of MDP transformations, and Jacq et al. (2022) proposes a particular MDP that allows for obtaining a sparse policy.

We refer the reader to Appendix A.3.1 for more detailed summaries of this work.

Relational RL and MDP. The following works provide an interpretable representation of states and transitions and are based on works presenting relational MDPs (Guestrin et al. 2003) and Relational RL (Dzeroski et al. 2001). As a reminder, the objective is to be able to learn a generalizable and efficient policy based on a representation of the environment by a set of objects and relations between them. In this section, we present only an overview of work in these domains. For more details, we recommend reading the state of the art Otterlo (2012) dating from 2012, which does not therefore include recent work.

Out of the 13 works in this group (cf. Table 3), 7 are not proposed as methods for explainability, although their approach makes the MDP interpretable. Object-Oriented MDPs (Diuk et al. 2008) and their extensions (Marom and Rosman 2018; Scholz et al. 2014) explicitly encode states as objects and relations and define transitions relationally, improving abstraction and generalisation. In line with this idea, (Kansky et al. 2017; Veerapaneni et al. 2019; Walker et al. 2007; Walsh 2010) propose learning a transition function based on objects (Kansky et al.; Veerapaneni et al.; Walker et al.) or relational action schemas (Walsh). To illustrate, in a model-based RL context, Veerapaneni et al. approximate the observation model which represents an image (i.e. agent state) by a set of semantic masks representing objects and the dynamics model which describes the temporal evolution of the different objects for planning.

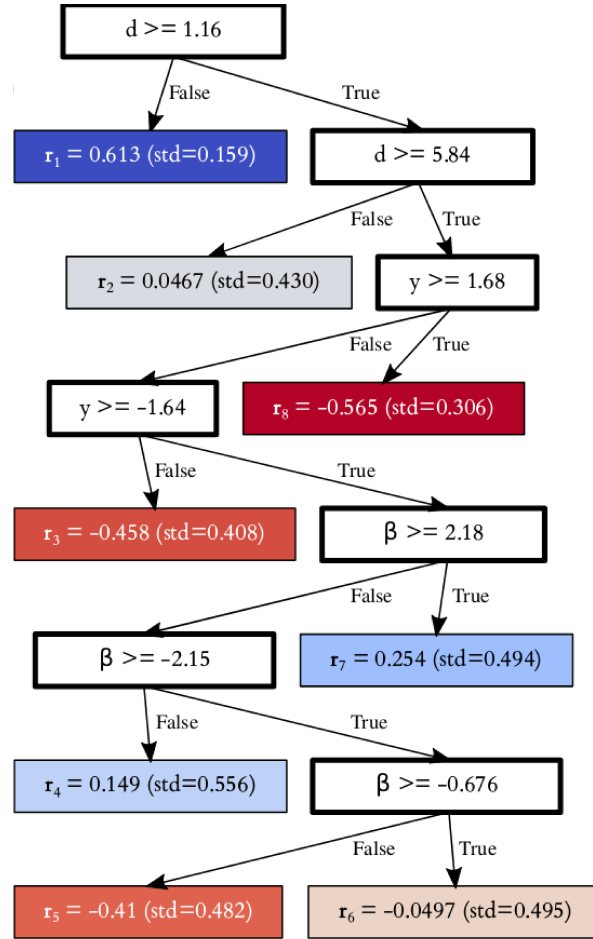


Fig. 7. A reward function as a binary tree structure for the original RoboCar environment consisting of a 4-wheeled vehicle whose goal is to reach a certain area while avoiding obstacles (Bewley and Lécué 2022). In this reward function, each internal node represents a test on a feature of an agent state, and each leaf a reward. The color of the leaves reflects the amount of reward the agent can obtain (blue means a high amount, red a low one).

Among other works (Adjodah et al. 2018; Agnew and Domingos 2018; Martínez et al. 2017; Zambaldi et al. 2018; G. Zhu, Z. Huang, et al. 2018; G. Zhu, J. Wang, et al. 2020), (Agnew and Domingos; Zambaldi et al.; G. Zhu, Z. Huang, et al.; G. Zhu, J. Wang, et al.) also learn to extract from a raw state (usually an image) a set of objects and relations between them in order to learn the transition function. For example, G. Zhu, J. Wang, et al. proposes a three-level learning architecture for learning model dynamics from the most to the least abstract. To aid learning, the output of an abstract level is provided to the level below. In order from most to least abstract, motion detection detects regions containing dynamic objects from an image sequence, instance segmentation produces coarse masks of dynamic objects and dynamics learning learns the transition function based on agent action and object relationships.

For more details on the works mentioned, refer to Appendix A.3.1.

4.3.2 Surrogate Model. The following works propose to obtain, from a non-interpretable component of the MDP, in this case the state space, a substitute model of the MDP that is understandable to the user. The 16 works in this section (cf. Table 3) focus on obtaining interpretable states by performing *state clustering* (8 works) or *state transformation* (8 works).

States Clustering. The following work clusters the states to make the MDP interpretable. The aim of these methods is to obtain, from the state space, groups of states called abstract states that are easy for a user to describe and understand. The work uses different approaches to derive these abstract states. For example, Bekkemoen and Langseth (2023) is based on attention maps (a concept described in more detail in Section 6.1.1) and Bewley and Lawry (2021) uses a combination of agent action, value function and temporal dynamics. Depending on the work, an abstract state takes the form of a natural language description (Booth et al. 2019; Climent et al. 2021; Hayes and Shah 2017; Wollenstein-Betech et al. 2020), a set of region limits of a state space (Bewley and Lawry), a representative state and an associated action (Bekkemoen and Langseth), or can be grouped by a tree diagram (Bewley, Lawry, and Richards 2022).

As an example of this line of work, Bekkemoen and Langseth propose the ASAP method, which groups states into clusters representing abstract states. The explanation consists of two parts: a display of a representative state and its associated action for each abstract state identified and a modelling of a policy graph, which is a Markov chain showing the agent's behavior in the abstract state space. Abstract states are identified using attention maps (which can be likened to saliency maps). ASAP is illustrated with the Mountain Car environment (Brockman et al. 2016) in Figure 8.

For more details on the work on states clustering, refer to Appendix A.3.2.

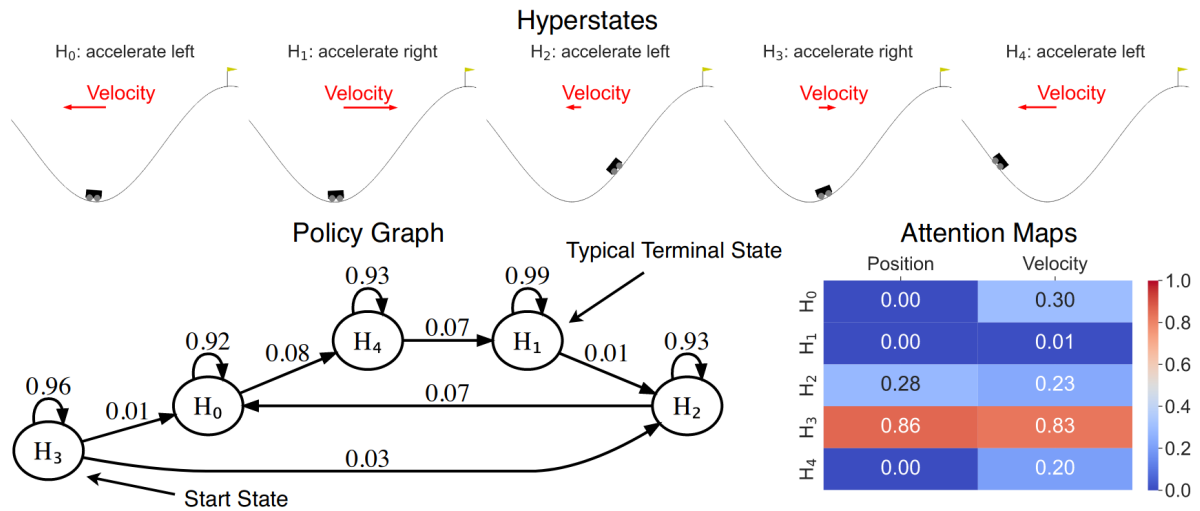


Fig. 8. Explanation via ASAP (Bekkemoen and Langseth 2023) of the agent's policy for the Mountain Car environment. The first line shows a representative state for each hyperstate with its associated action. The second line is divided into two parts. On the left, a policy graph models the policy based on the hyperstates. Each vertex is a hyperstate and each edge represents a transition between two hyperstates (weighted by its probability of occurrence). On the right, the attention maps of each hyperstate are displayed (where an attention score is associated with each feature, determining the impact of the features on the agent's decision-making).

State Transformation. A state transformation is a function mapping an original state, observation or set of observations into an alternative representation whose structure, dimensionality, or semantics improve the interpretability. The field of State Representation Learning (SRL) consists of learning a low-dimensional representation of the state, which evolves over time and reflects the change induced by the agent’s actions. As stated in the survey Lesort et al. (2018), in addition to the learning efficiency aspect, methods in SRL enable low-dimensional states composed of understandable features to be obtained (see Raffin et al. 2018 for a toolbox of SRL methods).

In line with this field, François-Lavet et al. (2019) propose to combine the model-based and model-free approaches of RL algorithms to learn an abstract state representation. A specific loss is added for learning the representation of an abstract state so that features of the abstract state are impacted by the agent’s actions. This makes the representation interpretable. The method is applied in a deterministic environment, but can be adapted for stochastic environments.

The other works convert raw states (which are generally images) into object-level states (Garcez et al. 2018; Garnelo et al. 2016), binary attribute states (A. Zhang et al. 2018) or symbolic image states (J. Chen et al. 2022). Dai, Ouyang, et al. (2023) stands out for its use of prior knowledge to improve interpretability, and Gajcin and Dusparic (2022) for its use of state transformation for the detection of causal confusion. For more information, refer to Appendix A.3.2.

Table 3 lists *Human-readable MDP* works. Of the 49 works mentioned, the 16 *Surrogate Model* methods focus exclusively on modifying the MDP state space, while the 33 *Inherently Understandable* methods focus on the use of transition functions, reward functions, and state spaces that are interpretable by design.

4.4 Visual Analysis

This section brings together a body of work focusing on the extraction of agent policy information to be displayed visually to the user during or after the learning phase.

Table 4. Visual Analysis works.

Type	Refs
Visual Toolkit (5)	Jaunet et al. 2020; A. Mishra et al. 2021; J. Wang et al. 2019 W. He et al. 2020; McGregor et al. 2017
Latent Space Visualization (3)	Annamay and Sycara 2019; Ben-Zrihem et al. 2016; Zahavy et al. 2016
Inspection (2)	Dethise et al. 2019; Ullauri et al. 2022

4.4.1 Visual Toolkit. A user interface including a set of tools for visual analysis of the agent’s behavior is proposed in W. He et al. (2020); Jaunet et al. (2020); McGregor et al. (2017); A. Mishra et al. (2021); J. Wang et al. (2019).

To illustrate, A. Mishra et al. propose a total of 8 tools to be used to understand the agent’s policy. This interface, which is described in Figure 9, requires the features describing a state to be interpretable. PolicyExplainer provides a global view of the frequency of actions and rewards, the policy, the value function (cf. panels A-C in Figure 9) and can be used to look at particular states and their associated Q-value, or a particular trajectory (cf. panels D-G in Figure 9). In addition, the ‘Why?’, ‘Why not?’ and ‘When?’ questions are answered by providing the state features that have an impact on the choice of action (cf. panel H in Figure 9).

The other visualisation interfaces (W. He et al.; Jaunet et al.; McGregor et al.; J. Wang et al.) are designed for experts, to help them with debugging. For more details on these interfaces, we refer the reader to Appendix A.4.1.

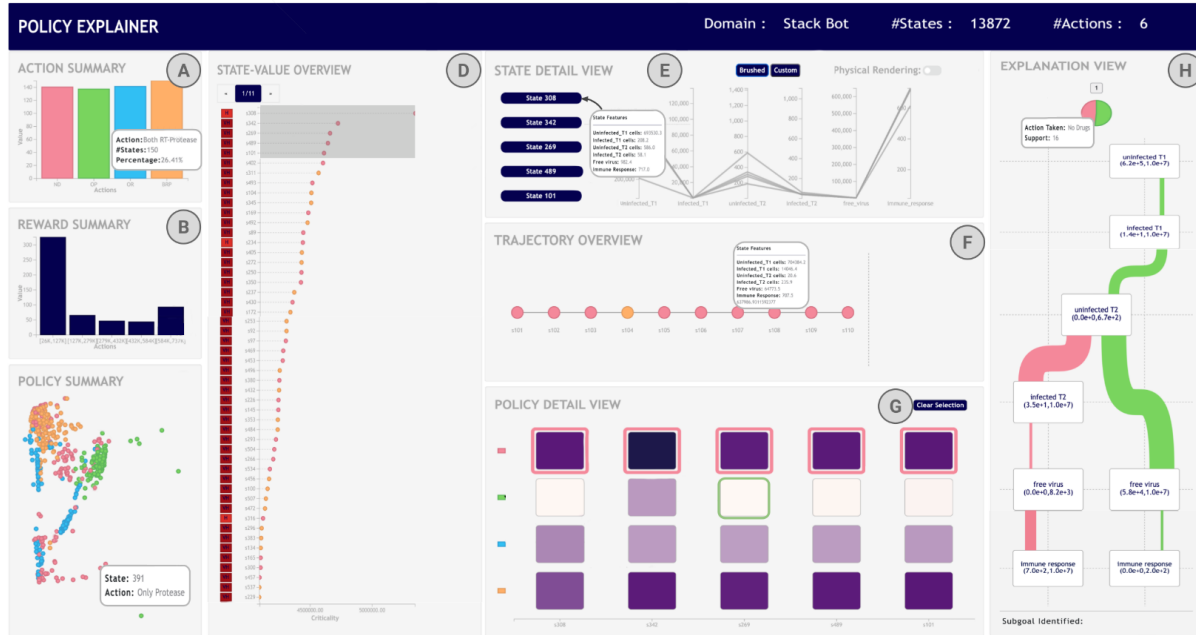


Fig. 9. PolicyExplainer visual interface (A. Mishra et al. 2021). The different tools are divided into three parts. Panels A-C provide a global view of the frequency of actions and rewards, the policy, the value function. Panels D-G provide a detailed analysis of particular states and their associated Q-value, or a particular trajectory. Panel H answers the questions ‘Why?’, ‘Why not?’ and ‘When?’ using state features.

4.4.2 Latent Space Visualization. Three works are interested in visually analysing a DQN that has learned to play Atari games in order to understand its internal representation of states. To do this, the main idea is to cluster the internal representations of states so that they can then be analysed and displayed to the user.

The first work of this group, Zahavy et al. (2016), propose to visualise the activations of the DQN by applying t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and G. Hinton 2008) to reduce the dimension. Before visualisation, a set of data is collected, including hand-crafted features such as the agent’s position, so that clusters can be identified and analysed. In addition, saliency maps (a concept described in more detail in Section 6.1.1) are also generated on the different states collected. By analysing the dynamics between clusters, the authors identified that the DQN has learned a hierarchical aggregation of the state space.

In addition to identifying clusters, Ben-Zrihem et al. (2016) constructs a semi-aggregated MDP that allows transitions between clusters to be obtained and studied, as shown in Figure 10. For more information on (Annasamy and Sycara 2019; Ben-Zrihem et al.), refer to Appendix A.4.2.

4.4.3 Inspection. The next two works simply propose to study the agent’s behavior by analysing the metrics and displaying them in the form of plots. The ETemoX framework (Ullauri et al. 2022) tracks the overall reward, the exploration/exploitation trade-off, and a feature specific to the environment under study during training, and Dethise et al. (2019) focuses, among other metrics, on the probability of action choice and confidence in the choice of actions. For more details, we refer the reader to Appendix A.4.3.

Table 4 summarises the *Visual Analysis* works. With only 10 works, this way of explanation is the least studied for explaining policies among the ways identified via our taxonomy.

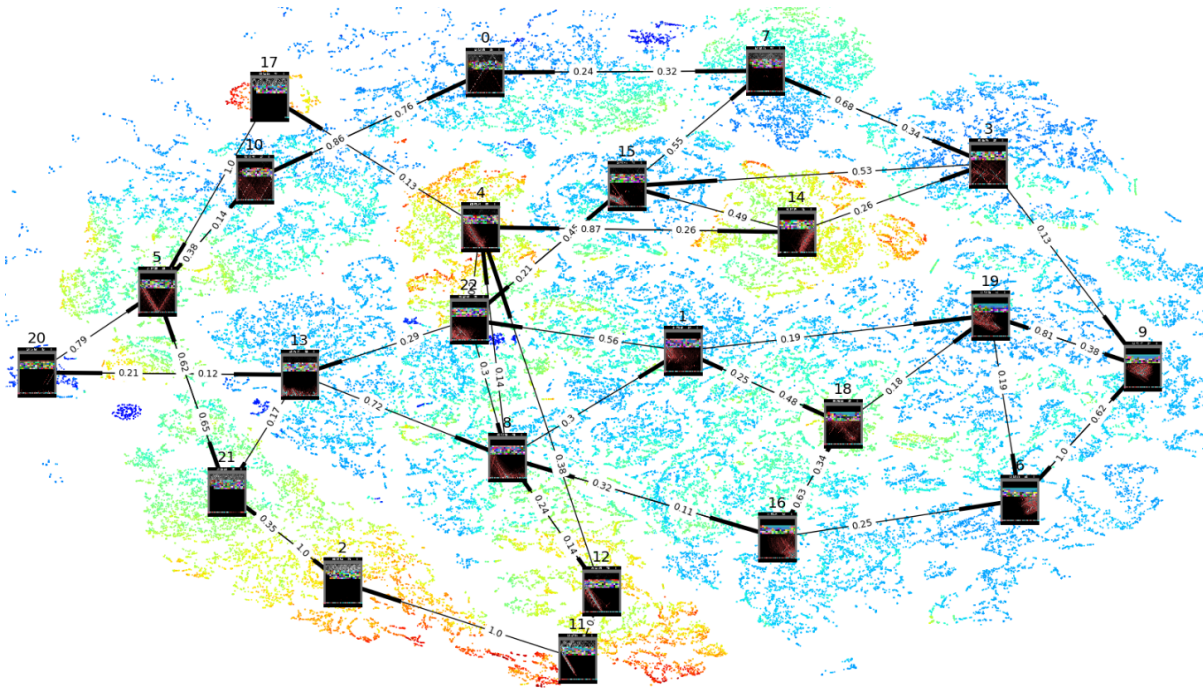


Fig. 10. Semi-MDP built on top of a t-SNE map for the Atari 2600 Breakout environment (Ben-Zrihem et al. 2016). The t-SNE map is the set of points on which the Semi-MDP is based. It is made up of states (or clusters of MDP states), linked by transitions weighted by their probability of occurrence. Each cluster is modelled by an image of the game where the position of the ball in each state of the cluster is displayed in red.

5 Sequence-level Methods

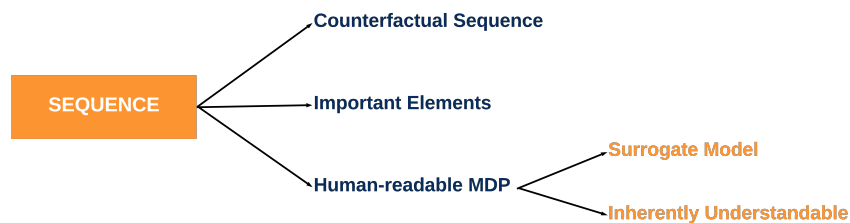


Fig. 11. Detailed taxonomy for sequence-level methods.

The methods described in this section explain sequences of interaction between the agent and the environment. As shown in Figure 11, sequences are explained in three ways. *Counterfactual Sequences* allow a comparison of the agent’s behavior with a sequence that uses an alternative policy, and thus determine the agent’s strengths and weaknesses. The *Important Elements* of a sequence are those that have the greatest impact on the agent’s ability to achieve a given objective. The use of *Human-readable MDP* makes it possible to understand, in the context of a sequence, the information available to the agent, the actions and the dynamics of the environment.

As with the policy explanation category, *Human-readable MDP* are divided into two groups (cf. Figure 11): the use of surrogate models to make MDP component(s) interpretable, and the use of MDP component(s) interpretable by design.

Table 5. Sequence-level works.

Type		Refs
Counterfactual Sequence (6)		Alabdulkarim and Riedl 2022; Waa et al. 2018 Gajcin and Dusparic 2024a; Tsirtsis, De, et al. 2021 Stefik et al. 2021; Tsirtsis and Rodriguez 2023
Important Elements (4)		W. Guo, X. Wu, et al. 2021; H. Liu et al. 2023 Saulières et al. 2024, 2023a
Human-readable MDP (3)	Surrogate Model (2)	Soni 2024; Sreedharan, Soni, et al. 2022
	Inherently Understandable (1)	W. Li et al. 2020

5.1 Counterfactual Sequence

A total of six works (cf. Table 5) aims to compare the state-action sequence of an agent with a counterfactual sequence where one or more actions differ from the agent's policy (Alabdulkarim and Riedl 2022; Gajcin and Dusparic 2024a; Stefik et al. 2021; Tsirtsis, De, et al. 2021; Tristsis and Rodriguez 2023; Waa et al. 2018).

This counterfactual sequence can be produced based on the user's question (Alabdulkarim and Riedl; Waa et al.), using an evolutionary algorithm (Gajcin and Dusparic) or an algorithm based on dynamic programming (Tsirtsis, De, et al.; Tristsis and Rodriguez). The explanation consists of comparing the agent's sequence with the counterfactual sequence according to the rewards obtained (Alabdulkarim and Riedl; Gajcin and Dusparic; Tsirtsis, De, et al.; Tristsis and Rodriguez; Waa et al.), but also the occurrence of actions (Alabdulkarim and Riedl; Waa et al.), their cost (Alabdulkarim and Riedl; Waa et al.) or even the types of risks (Stefik et al.).

To illustrate this line of work, in a context where the environment includes different reward classes, Alabdulkarim and Riedl train a set of influence predictor models for each reward type during agent training. The counterfactual sequence is generated by considering the action (or sequence of actions) proposed by the user, then using the agent's policy. The sequences are compared using influence predictors (i.e. a calculation of the average influence of each type of reward in each sequence), the cost of the actions and the final reward obtained. An example showing the interest of this approach is presented in Figure 12.

For further details on the work of the *counterfactual sequence* approach, we refer the reader to Appendix B.1.

5.2 Important Elements

The aim of these methods is to determine, within a given sequence, the most important elements for achieving an objective. Depending on the work, these important elements are time-steps (W. Guo, X. Wu, et al. 2021), states (H. Liu et al. 2023) or actions (Saulières et al. 2024, 2023a). To provide these elements, (W. Guo, X. Wu, et al.; H. Liu et al.) learn a model upstream and (Saulières et al.; Saulières et al.) use the environment functions.

As an example of this line of work, the EDGE algorithm is a self-explainable model that predicts, for a given episode, the final agent's reward (W. Guo, X. Wu, et al.). The provided explanation takes the form of a set of important time-steps within the agent's interaction episode with the environment. An explanation of a game of Atari 2600 Pong (Bellemare et al. 2013) is shown in Figure 13.

For more details, we refer the reader to Appendix B.2.

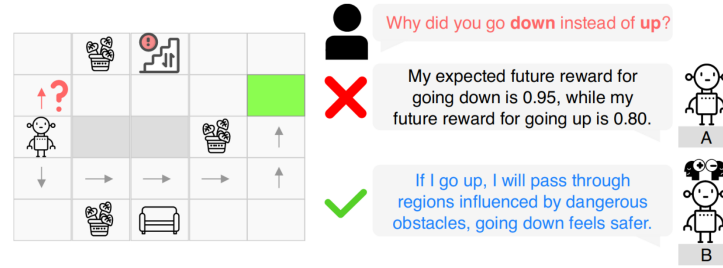


Fig. 12. Possible explanations for the sequence of actions that the agent (represented by a robot in cell (3,1)) performs in a 2D environment to reach the green cell (Alabdulkarim and Riedl 2022). The user’s question is written in red. Explanation A (in black and marked by a red cross) is based solely on the agent’s policy, while explanation B (in blue marked with a correct tick) uses influence predictors. According to authors, explanation B is of better quality.

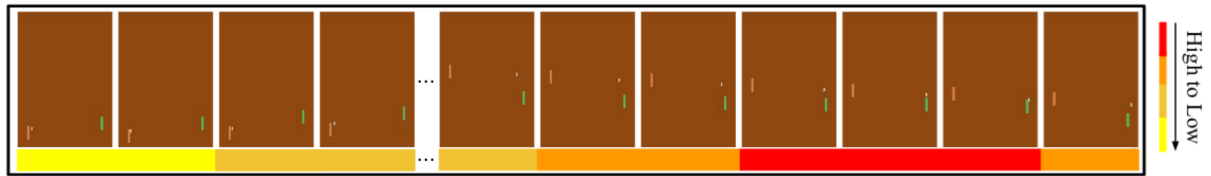


Fig. 13. An explanation of an episode from a game of Atari 2600 Pong (W. Guo, X. Wu, et al. 2021). The objective is to determine the important time-steps for the defeat of the agent controlling the green pad. The colored bar below each state of the episode represents its level of importance (yellow represents a low level, orange a medium level and red a high level).

5.3 Human-readable MDP

5.3.1 Surrogate Model. (Soni 2024; Sreedharan, Soni, et al. 2022) propose to obtain, from a non-interpretable element of the MDP, in this case the action space, an interpretable surrogate model. The idea is to represent actions symbolically as a triplet comprising the preconditions, cost and effects of the action. In Soni, an additional task is to determine the type of user that the explainer is dealing with in order to provide a more appropriate explanation. For more information on these two works, we refer the reader to Appendix B.3.

5.3.2 Inherently Understandable. In a context of sequential reconstruction of magnetic resonance images, Li et al. (2020) use RL to successively apply interpretable pixel-wise operations. In this environment, the action space is directly interpretable because each action corresponds to common Computer Vision filters, such as the Sobel filter or the Gaussian filter.

Table 5 summarizes the various works cited for *sequence-level* methods. This target receives the least attention from the XRL community, with only 13 works identified. Among these methods, the most common ways of explanation is *Counterfactual Sequences*.

6 Action-level Methods

As described in Figure 14, the explanation of actions is carried out using two ways of explanation: *Feature Importance* which determines the features of the agent’s state that led it to choose an action, and *Expected Outcomes* which justifies the agent’s decision by providing its future potential impact. *Feature Importance* is

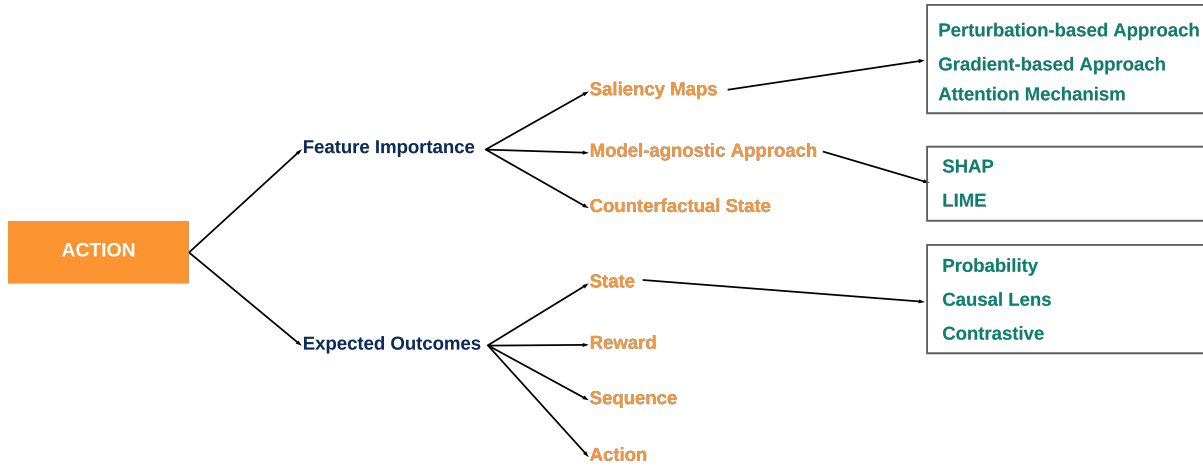


Fig. 14. Detailed taxonomy for action-level methods.

divided into three groups of work: feature importance is calculated using saliency maps, with the use of model-agnostic approaches (SHAP (Lundberg and S. Lee 2017) and LIME (Ribeiro et al. 2016)) or via the computation of a counterfactual state. Saliency maps are calculated using three distinct approaches: perturbation-based, gradient-based and attention mechanism. The nature of *Expected Outcomes* varies according to the work identified (cf. Figure 14): it can be a state, a reward, a state-action sequence or an action.

6.1 Feature Importance

To explain the choice of an action a from a state s , the following works describe the importance of the features of s in the agent's choice. Most of this work falls into three categories: saliency maps, model-agnostic approaches and counterfactual states.

6.1.1 Saliency Maps. Saliency maps are used when the state of an agent is represented by an image. This method makes it possible to show pixels, or groups of pixels, that are essential in the agent's decision-making. In other words, the explanation takes the form of an image (the agent's state) overlaid with a heat map, or saliency maps, representing the (groups of) pixels in the image that are most important for the agent's decision. The major different ways of calculating them are: gradient-based, perturbation-based and attention mechanisms approaches.

As this type of approach is fashionable, we do not intend to provide an exhaustive description of the work related to saliency maps in the XRL context. For a detailed analysis of the use of saliency maps for RL agents, see Atrey et al. (2020). Most of the experiments on saliency maps were carried out on Atari2600 games using the Arcade Learning Environment (Bellemare et al. 2013). Note that saliency maps can also be referred to as attention maps or attention masks in this section.

Attention Mechanism. This section focuses on methods providing saliency maps based on attention mechanisms. Attention mechanisms are modules used to make the agent focus on certain parts of the image. The use of these mechanisms to produce explanations has been identified in 15 works (see Table 6).

Unlike the vast majority of methods that construct the attention mask using an attention block positioned at the beginning of the NN, (Huber, Schiller, et al. 2019; Y. Liu et al. 2022) propose constructing it using the entire NN. To illustrate, the Action Region Scoring (ARS) module proposed in Y. Liu et al. is used both to explain the

Table 6. Feature Importance works.

Type	Refs
Saliency Maps (32)	Attention Mechanism (15) Itaya et al. 2021; Mott et al. 2019; Nikulin et al. 2019 Y. Liu et al. 2022; Ou et al. 2024; Shi et al. 2022 Bao et al. 2021; Josef and Degani 2020; Y. Tang et al. 2020 X. Wang, Yuan, et al. 2019; Q. Yang et al. 2024; Z. Yang et al. 2018 Huber, Schiller, et al. 2019; Z. Wang, and S. Wang 2022 Q. Zhang 2021
	Perturbation-based (10) Douglas et al. 2019; Greydanus et al. 2018; S. Guo et al. 2021 Huber, Limmer, et al. 2022; Puri et al. 2020; Yan 2023 Iyer et al. 2018; Y. Li et al. 2017 Anderson et al. 2020; Persiani and Hellström 2022
	Gradient-based (7) Dao, Huff, et al. 2021; Joo and K. Kim 2019; Nie et al. 2019 Hilton et al. 2020; Zahavy et al. 2016 L. He et al. 2021; Weitkamp et al. 2019
Model-agnostic Approach (15)	SHAP (13) Carbone 2020; Løver et al. 2021; Rizzo et al. 2019; Y. Wang et al. 2020 Remman, Strümke, et al. 2022; Theumer et al. 2022; K. Zhang et al. 2022 Liessner et al. 2021; Remman and Lekkas 2021; Schreiber et al. 2021 Beechey et al. 2023; Sequeira and Gervasio 2023; Xiong et al. 2024
	LIME (3) Dethise et al. 2019; Kotevska et al. 2020; H. Li et al. 2022
Counterfactual State (5)	Huber, Demmler, et al. 2023; Olson et al. 2021; Samadi et al. 2024 Druce et al. 2021; Gajcin and Dusparic 2024b
Others (2)	Davoodi and Komeili 2021; Elizalde et al. 2009

choice of action and to improve learning. In the CNN, at the output of each convolutional layer to which a ReLU activation has been applied, an ARS module is used to identify the important regions and then combined with the current image representation. This helps learning by indicating the regions to focus on. For the explanation part, the saliency map is produced by retrieving the outputs of the ARS modules and combining them.

(Y. Tang et al. 2020; Z. Yang et al. 2018) stand out from other works by identifying important regions (groups of pixels) rather than individual pixels, and (Ou et al. 2024; X. Wang, Yuan, et al. 2019) by adding a complementary explanation to the visual explanation (i.e. the saliency map). As an example, the approach proposed by X. Wang, Yuan, et al. produces a natural language explanation in addition to an attention map. The architecture of the algorithm presented breaks down into three parts: an encoder, which encodes the image into a set of features, an attention mechanism, which returns the salient features and the decoder, which generates a verbal explanation from the salient features.

For a more detailed summary of other work using attention mechanisms to provide saliency maps, refer to Appendix C.1.1.

Perturbation-based Approach. Works in this section uses a perturbation-based approach to generate saliency maps. Perturbation-based approaches perturb the input image and evaluate its impact on the agent's policy.

Some works (Anderson et al. 2020; Iyer et al. 2018; Y. Li et al. 2017) use this approach on groups of pixels identified as objects in the image, as shown in Figure 15.

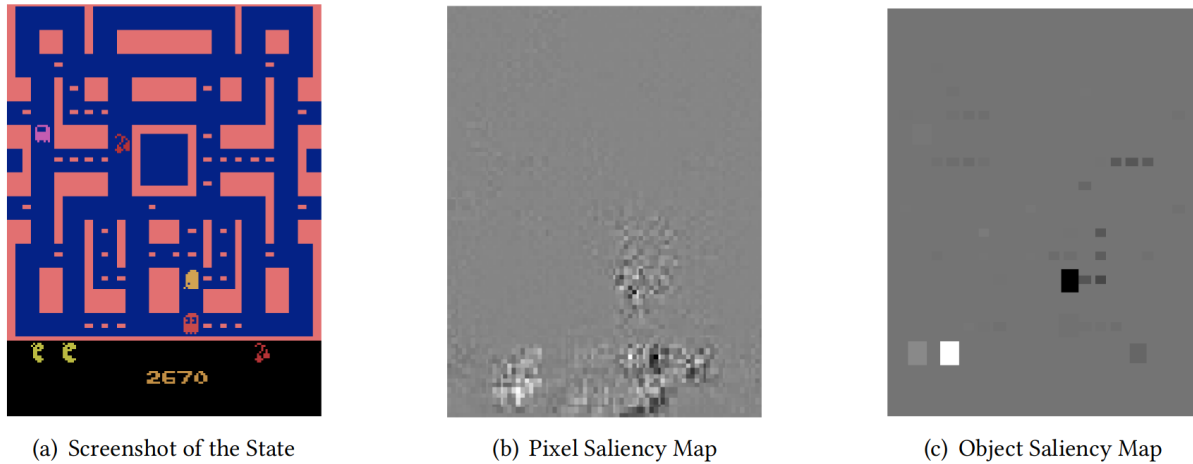


Fig. 15. Two saliency maps for a state-action pair from the Ms PacMan environment (Iyer et al. 2018). The action is ‘move right’ and the sub-figure ‘a’ is the state from which a pixel saliency map (sub-figure ‘b’) and an object saliency map (sub-figure ‘c’) are generated. The saliency of an element is represented using a grey scale, where the darker the element, the more salient it is. In sub-figure ‘c’, we can see that the agent has focused on Ms PacMan and the pills in the right path to choose the ‘move right’ action. Conversely, it did not focus on the part of the image that symbolises 1 life for the player, as shown by the white square in the object saliency map.

Various methods are used to perturb images, such as Gaussian blur around a pixel (Douglas et al. 2019; Greydanus et al. 2018; S. Guo et al. 2021; Persiani and Hellström 2022; Yan 2023) or object masking (Anderson et al.; Iyer et al.; Y. Li et al.). To evaluate the salience of a pixel or object, methods use the value function and policy (Douglas et al.; Greydanus et al.; S. Guo et al.; Persiani and Hellström), the Q function (Anderson et al.; Iyer et al.; Y. Li et al.; Puri et al. 2020) or the advantage function (Yan). As an example of this line of work, the method in Greydanus et al. involves adding spatial uncertainty around the pixel at coordinates (i, j) . This is done by interpolating the original image with a Gaussian blur applied around (i, j) . In practice, instead of applying this perturbation for each pixel, it was applied per group of 5 pixels, which reduces the computational cost and produces good saliency maps. Using the A3C learning algorithm, saliency maps are generated and displayed jointly for the agent’s policy and for its value function V (i.e. this captures the regions that are important for the agent’s choice of action and evaluation of the image). Three works based on this method (Douglas et al.; S. Guo et al.; Persiani and Hellström). We refer the reader to Appendix C.1.1 for more information on the other methods of this line of work.

Gradient-based approach. This section focuses on methods providing saliency maps using a gradient-based approach. A total of seven works were identified in this line of work (cf. Table 6). Based on the image, gradient-based approaches use the gradient of a class (i.e. action) in the last layer of the neural network and recover the information with back-propagation.

An example of a well-known method used in certain works (e.g. Joo and K. Kim 2019; Nie et al. 2019) is Grad-CAM (Selvaraju et al. 2017). This method can be used on CNN’s without having to modify the architecture or re-train the model, in order to create saliency maps. To find out the influence of pixels on a class c for a given image, a forward pass is performed, the gradient of class c is calculated, then the signal is propagated with a backward pass. Importance weights of the feature maps (where a feature map is the output of a convolutional layer of a CNN) are calculated and then the saliency map is generated by applying a ReLU operation on the

weighted linear combination of the feature maps. This method has been used in several works to explain RL agents (Dao, Huff, et al. 2021, Joo and K. Kim; Nie et al.; Weitkamp et al. 2019). To illustrate, Joo and K. Kim propose an architecture combining A3C (Mnih, Badia, et al. 2016) with Grad-CAM in order to visualise the salient parts of the image for the agent's choice of action.

For a more detailed summary of the work using Grad-CAM (Dao, Huff, et al., Joo and K. Kim; Nie et al.; Weitkamp et al.) and others (L. He et al. 2021; Hilton et al. 2020; Zahavy et al. 2016), refer to Appendix C.1.1.

6.1.2 Model-agnostic Approach. This section focuses on the use of SHAP (Lundberg and S. Lee 2017) and LIME (Ribeiro et al. 2016) and their variants to determine the importance of state features in the agent's choice of action. As a reminder, SHAP is an approximation of shapley values (Shapley et al. 1953) performed using either Kernel SHAP or DeepSHAP. LIME estimates the influence that each feature of input x has on the model's output. In our context, the input is a state of the agent, the model the agent's policy and the output an action. To calculate the influence of features, a transparent surrogate model is learned locally around x . This surrogate model is a linear model. To train it, a set of points is generated by perturbing x and weighted as a function of distance from x . After training, the weights of the linear model are used to explain the influence of each feature of x on the output of the model.

SHAP. A total of 13 works were identified to provide explanations for the action performed by the agent via SHAP (cf. Table 6), using either Kernel SHAP (Carbone 2020; Løver et al. 2021; Remman, Strümke, et al. 2022; Rizzo et al. 2019; Y. Wang et al. 2020) or DeepSHAP (Liessner et al. 2021; Remman and Lekkas 2021; Schreiber et al. 2021; Theumer et al. 2022; K. Zhang et al. 2022). For example, Rizzo et al. use Kernel SHAP for a traffic light control problem and K. Zhang et al. use DeepSHAP for a power system emergency control problem.

Other variants of SHAP have been used in (Remman, Strümke, et al.; Xiong et al. 2024) to provide explanations, namely Causal SHAP (Heskes et al. 2020), TreeSHAP (Lundberg, Erion, et al. 2018), and TabularSHAP (Xiong et al.). For a detailed description of the work based on SHAP, refer to Appendix C.1.2.

LIME. Only three works have been identified that use LIME to provide action explanations (Dethise et al. 2019; Kotevska et al. 2020; H. Li et al. 2022).

Dethise et al. uses LIME to explain an agent's decision, or the agent's global behavior. LIME is also used in H. Li et al. to add an explanatory layer to an advanced persistent threats defense mechanism and in Kotevska et al. to analyse the behavior of an agent on a heating ventilation and air-conditioning control task.

We refer the reader to Appendix C.1.2 for further details.

6.1.3 Counterfactual State. A body of work aims to provide an answer to the question 'Why does the agent perform action a from state s rather than action a' ?' by proposing a counterfactual state s' , close to s , in which the agent would have chosen a' .

Among the five works reviewed (cf. Table 6), three of them (Huber, Demmler, et al. 2023; Olson et al. 2021; Samadi et al. 2024) use GANs (Goodfellow et al. 2014) to generate s' . This architecture consists of jointly learning a generative model G that creates data and a discriminative model D that determines whether the input data is generated or comes from the training dataset. After learning, only the generative model is used. As an example of work, in order to provide a counterfactual explanation where the state is an image, Olson et al. propose to learn, in addition to these two models, an encoder which encodes the state s in order to omit information from s related to the action choice. D is trained to determine the action distribution based on the encoded state. G is trained to reconstruct s based on the action distribution and the encoded state. For training, it is necessary to get an initial dataset of state-action pairs using the agent to be explained. This method, tested on Atari2600 games, generates a state s' , close to s , in which the agent performs a different action.

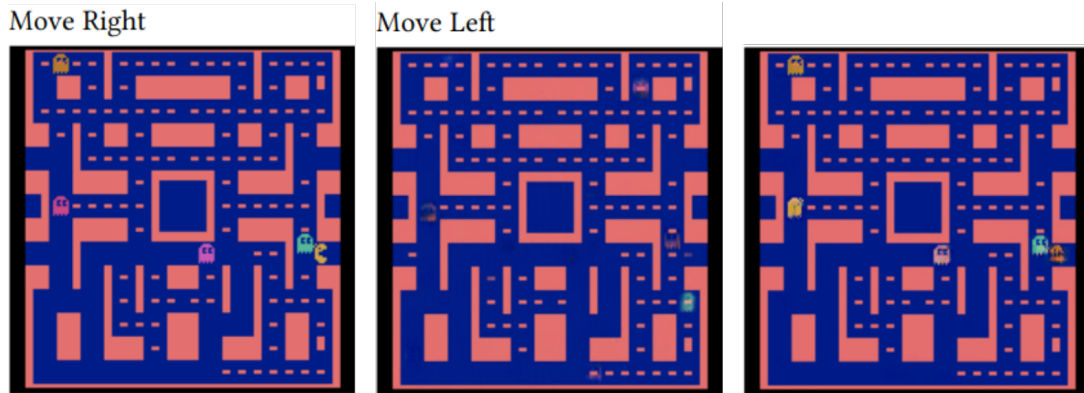


Fig. 16. Comparison of counterfactual states in the Atari 2600 Ms PacMan environment (Bellemare et al. 2013) generated on the basis of a state-action pair (on the left) (Huber, Demmler, et al. 2023). The agent's action is 'move right' and the counterfactual one is 'move left'. The state in the middle, where Ms PacMan is missing, is generated using (Olson et al. 2021) and the state on the right, where Ms PacMan is safe in the left part of the map, is generated using Huber, Demmler, et al. The first counterfactual state is not a convincing one as it shows that the agent chooses 'move left' while Ms PacMan does not even appear, whereas in the second one, Ms PacMan is (partially) visible.

In Druce et al. (2021), the counterfactual state s' is obtained via predefined changes in s and in Gajcin and Dusparic (2024b) it is obtained via the computation of a sequence of actions leading to s' , providing recourse to the user.

For more details on the work addressing the question 'Why does the agent perform action a from state s rather than action a' ?', we refer the reader to Appendix C.1.3.

6.1.4 Others. Two works do not fall into the previously identified work groups (cf. Table 6). Davoodi and Komeili (2021) explain the impact of features in reaching risky states by learning a linear model that estimate feature importance and Elizalde et al. (2009) display to the user the most relevant feature according to the value function and provide an associated verbal explanation based on a knowledge-base dataset.

For further details, refer to Appendix C.1.4.

Table 6 groups together *Feature Importance* works. Of the 54 works, the two most commonly used ways of explaining actions are *saliency maps*, followed by *model-agnostic approaches*. Among these two ways of explanation, the majority of works uses attention mechanisms and SHAP to calculate feature importance. Only five methods propose counterfactual explanations via *counterfactual states*.

6.2 Expected Outcomes

The work presented below focuses on explaining the agent's choice of action by giving and analysing the agent's expected outcomes. These works have been categorised according to the type of information provided to the user. This information can be a (partial) state, a reward, a state-action sequence, or even an action.

6.2.1 State. This line of work is interested in providing the future state(s), or features of the state(s), resulting from an action a from a state s , as an explanation. With 17 works identified, this line of research is the most substantial in terms of number of papers (cf. Table 7). The works explain the choice of an action a from a state s in different ways, such as by calculating the causal link between the pair (s, a) and the features of the next state,

Table 7. Expected Outcomes works.

	Type	Refs
State (16)	Causal Lens (5)	Madumal et al. 2020a,b; Volodin 2021 Herlau and Larsen 2022; Yu et al. 2023
	Probability (4)	Cruz, Dazeley, and Vamplew 2019; Cruz, Dazeley, Vamplew, and Moreira 2023 Dodson et al. 2011; Khan et al. 2009
	Contrastive (4)	Barros et al. 2020; Z. Lin et al. 2021; Stein 2021; Tsuchiya et al. 2023
	Others (3)	Fukuchi et al. 2017a,b; Pan et al. 2019
Reward (10)		Erwig et al. 2018; Iucci et al. 2021; Juozapaitis et al. 2019, Saldiran et al. 2024 Anderson et al. 2020; Feit et al. 2022; Rietz et al. 2023 Amitai, Septon, et al. 2024; Kampik et al. 2019; Towers et al. 2024
Sequence (5)		Amitai, Septon, et al. 2024; Rouveirol et al. 2023; Yau et al. 2020 Luss et al. 2023; Saulières et al. 2023b
Action (1)		Dodson et al. 2011

by providing the probability of reaching a certain state, and by comparing a with a contrastive action (proposed by the user).

Causal Lens. The five works (Herlau and Larsen 2022; Madumal et al. 2020a,b; Volodin 2021; Yu et al. 2023) propose using causal models to explain the impact of the agent’s choice of action on, in particular, the features of the state reached.

Among these models, (Madumal et al. 2020a,b; Yu et al.) use an Action Influence Model (AIM) which is a structural causal model (Halpern and Pearl 2001) with the addition of action (introduced in Madumal et al. 2020b). As an example, Yu et al. extend the use of AIM to problems with a continuous action space, without the need of prior knowledge of the environment causal structure. Indeed, to represent the dynamics of the environment, the causal model is learned based on the agent’s interactions with the environment. An example of a learned causal model for the LunarLander environment (Brockman et al. 2016) is shown in Figure 17.

In this line of work, Madumal et al. 2020a,b use a manually defined causal model, whereas (Herlau and Larsen; Volodin; Yu et al.) learn a sparse model.

For more details on this work, refer to Appendix C.2.1.

Probability. The four works (Cruz, Dazeley, and Vamplew 2019; Cruz, Dazeley, Vamplew, and Moreira 2023; Dodson et al. 2011; Khan et al. 2009) (cf. Table 7) in this group explain the agent’s choice of action with the probability of reaching a certain state or type of states.

To illustrate this line of work, the Memory-based eXplainable Reinforcement Learning approach (Cruz, Dazeley, and Vamplew) provides the probability of success (i.e. reaching a goal state) and the number of transitions to reach the goal state. When the agent is trained, the success probabilities P and the number T of transitions leading to a goal state are calculated and updated for each state. The agent can then provide answers to the questions ‘Why does the agent choose action a ?’, ‘Why does the agent choose action a and not b ?’ or ‘What is the probability of reaching a goal state in 8 steps from s ?’.

(Dodson et al.; Khan et al.) compute the probability of reaching certain types of states and provide explanations in natural language, using predefined templates.

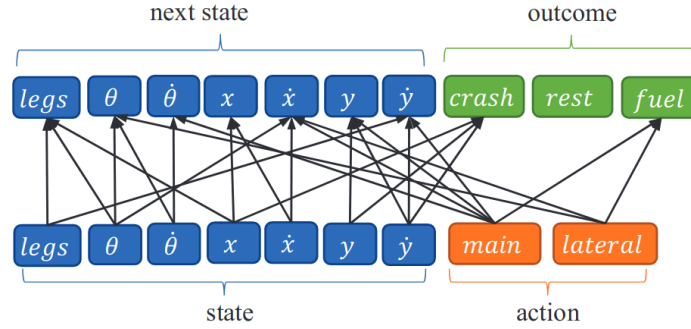


Fig. 17. Causal model for the LunarLander environment (Yu et al. 2023). The features of a state are in blue, the actions in orange and the outcomes in green. An arrow between two elements describes a causal relation.

For further details on (Cruz, Dazeley, and Vamplew; Cruz, Dazeley, Vamplew, and Moreira; Dodson et al.; Khan et al.), refer to Appendix C.2.1.

Contrastive. This line of work proposes comparing the action performed by the agent with a contrasting action that the agent could have performed. This comparison is made based on the cost of the actions and the probability of reaching a terminal state (Stein 2021), the confidence of reaching it (Barros et al. 2020), the type of state that the agent can reach (Tsuchiya et al. 2023), or the features impacted by the actions (Z. Lin et al. 2021).

As an example, Tsuchiya et al. present a method similar to reward decomposition (which will be described in the next section), but focused on states. Additional neural networks are learned to estimate the Q-value of each state type. The different state types are provided in advance by the user. A contrastive explanation between the agent’s choice of action and that of the user is performed by displaying and then comparing the different Q-values decomposed by state type.

We refer the reader to Appendix C.2.1 for more details.

Others. Three works do not fall into the previously identified work groups (cf. Table 7). In Fukuchi et al. (2017a,b), the explanation is the impact of the action on the value of the state features n time steps after performing it, which is associated with an appropriate expert instruction previously used for training the agent. Image semantic segmentation consists of labelling each pixel of an image. The agent learns to predict an image semantic segmentation of the future state, which acts as an explanation, in Pan et al. (2019).

For further details, refer to Appendix C.2.1.

6.2.2 Reward. The agent’s choice of action can be explained in terms of the reward that the agent is seeking to obtain. 8 out of 10 works (cf. Table 7) uses the principle of reward decomposition (Amitai, Septon, et al. 2024; Anderson et al. 2020; Erwig et al. 2018; Feit et al. 2022; Iucci et al. 2021; Juozapaitis et al. 2019; Rietz et al. 2023; Saldiran et al. 2024).

Reward decomposition is used to represent the reward function in an interpretable way: it is decomposed into a set of comprehensible reward types. Thus, to understand the agent’s choice of action, it is sufficient to look at the type of reward it seeks to maximise. Based on this method, (Erwig et al.; Juozapaitis et al.) propose Reward Difference eXplanations (RDX) and Minimal Sufficient eXplanation (MSX). RDX simply compares 2 actions from a state by comparing their Q-values for each type of reward, while MSX provides a more compact explanation based on RDX. An example of RDX, from Iucci et al., is shown in Figure 18.

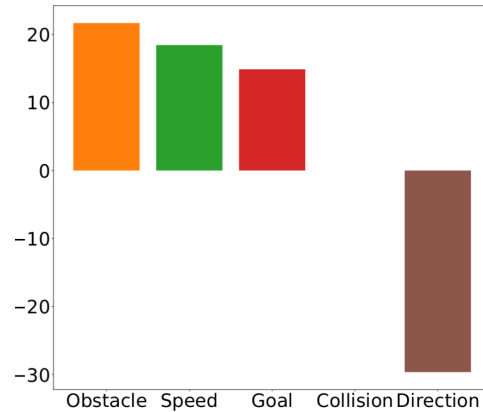


Fig. 18. RDX for a Human-Robot collaboration task (Iucci et al. 2021). Two actions (related to robot speed) are compared across four different reward types (on the x-axis of the chart). The y-axis shows the loss and gain for each type of reward when performing the agent's action rather than the other one.

Several works (Amitai, Septon, et al.; Anderson et al.; Feit et al.; Iucci et al.; Rietz et al.; Saldiran et al.) combine reward decomposition (by simply displaying the different Q-values or by using RDX, MSX) with other methods, e.g. autonomous policy explanation (Iucci et al.).

Towers et al. (2024) defines a new estimator allowing the agent to predict the n future expected rewards, which is then used to provide explanations, and Kampik et al. (2019) proposes several ways to explain actions in a human-agent cooperation context, which could be transposed to a cooperative MARL context.

We refer the reader to Appendix C.2.2 for further details on this line of work.

6.2.3 Sequence. The expected outcome of an action here takes the form of a sequence, or a summary of sequences, which the agent can reach by performing an action a from a state s .

Three of the five works (cf. Table 7) propose summarising future sequences (Luss et al. 2023; Rouveirol et al. 2023; Saulières et al. 2023b): Rouveirol et al. groups them using a rule-based classifier that allows each group to be represented by a logical formula, Saulières et al. summarises them into three sequences, which are the best-case, worst-case and most-probable case scenarios according to the reward and transition function of the environment, and Luss et al. groups future states into clusters, or sub-goals that the agent seeks to achieve.

In Yau et al. (2020), a belief map is learned, to show the user the states that the agent aims to reach by performing action a from state s , as shown in Figure 19 in the Taxi environment. In Amitai, Septon, et al. (2024), the proposed approach shows the user a counterfactual sequence where the agent would have performed a' from s , to compare it with the actual sequence.

For more details, refer to Appendix C.2.3.

6.2.4 Action. Only one work explains the choice of an action by indicating future actions that could be taken.

Dodson et al. (2011) provide a natural language explanation for advising students on their choice of courses. This problem, represented by a MDP, is explained using two modules, one of which is called Model-Based Explanation. This module extracts information from the MDP about the usefulness of performing the recommended action in terms of future actions. As an example, if the student takes course x , this will enable him to take courses y, z in the future.

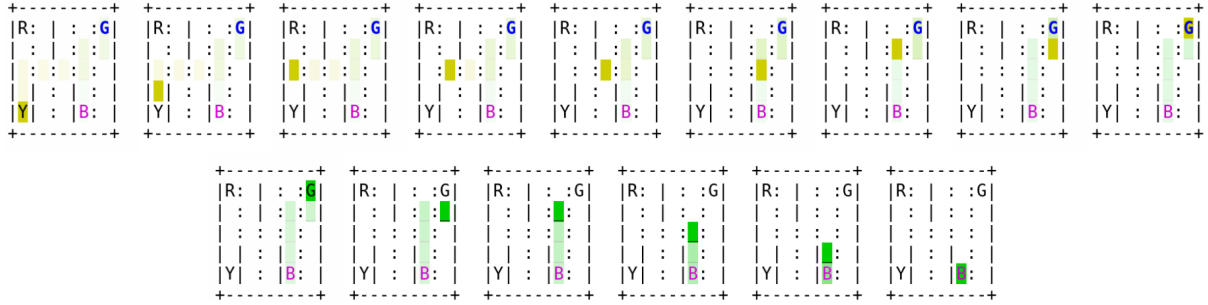


Fig. 19. Explanation of each action in an episode in the Taxi environment (Yau et al. 2020). The episode is described on two lines, starting on the left of the first line. In each state, the agent's future positions according to the belief map are displayed by colored cells. The color intensity reflects the agent's confidence in accessing the position (the more intense the colour, the more confident the agent). In this episode, the Taxi picks up the passenger at the position marked with a 'G' and drops him off at the position marked with a 'B'.

Table 7 groups together *Expected Outcomes* works. With 31 works, this way of explaining action is the least common. Among these works, the majority focuses on expected outcomes taking the form of state (16 works), followed by reward (10 works), particularly obtained via reward decomposition.

In addition to the various works mentioned for action explanation, we have identified four works that use both types of explanation (i.e. *Feature Importance* and *Expected Outcomes*). To illustrate, an approach is presented in N. Wang et al. (2016) for directly extracting explanations based on the POMDP. The idea is to retrieve the various types of information from the POMDP model and translate them into natural language sentences using predefined templates. For example, the agent can use its expected reward outcome to justify its choice of action, the transition function to describe the probability of achieving different outcomes, or its observation to clarify it to the user. Among other works, Neerincx et al. (2018) proposes a general framework for providing perceptual and cognitive explanations based on the agent's confidence, goals, and beliefs, and (Ehsan, Harrison, et al. 2018; Ehsan, Tambwekar, et al. 2019) first build a dataset in which each state-action pair is associated with an explanation provided by a user, and then train a model on this dataset to provide an explanation. Refer to Appendix C.2.4 for further details.

7 Related Domains

Explainable Planning. Planning proposes a set of methods for providing a plan as a solution to a problem given the initial configuration, the goal configuration and a set of actions (or operators) defined by preconditions and effects. Planning and RL being relatively close domains, explainability methods for planning, grouped under the name XAIP, are of particular interest. We believe that this area should be further explored by XRL researchers. As a first step in this domain, we will briefly describe some XAIP surveys and methods.

A survey and roadmap for XAIP is described by Fox et al. (2017) in order to obtain more efficient methods. Different questions are posed to guide the search, such as 'Why can't you do that?' or 'Why do I need to replan at this point?' in the respective cases where a plan is not found by the algorithm and a failure of the current plan occurs. Chakraborti et al. (2020) categorise the different XAIP works according to the target of the explanation and the properties of the explanation. The method may explain the planning algorithm, the problem model or the plan. As examples of properties, XAIP methods can use abstractions and/or contrastive approaches. A brief overview is given in Hoffmann and Magazzeni (2019), with a focus on contrastive explanations. For these explanations, several types of question are interesting to study, such as 'Why action a instead of action b?' or

‘Why does the current plan satisfy property p rather than q ?’. Cashmore et al. (2019) describe a framework for answering a set of contrastive questions and the different challenges for effective XAIP methods.

Plan debugging is carried out by finding states s called bugs where there is a difference between the evaluation of the value of s based on the current plan and the optimal plan (Steinmetz et al. 2022). This comparison can be performed according to two distinct criteria: a quantitative criterion that considers the cost of the plan, and a qualitative criterion that considers the resolution of the problem. The unsolvability of a planning problem is explained by a hierarchical abstraction of sub-goals (Sreedharan, Srivastava, Smith, et al. 2019). These sub-goals are necessary to solve the problem but are for the most part unreachable. The proposed method consecutively identifies the appropriate level of abstraction for the explanation, the sequence of sub-goals to be achieved and the first unreachable one in the sequence.

The Explaining Robot Action (Lomas et al. 2012) answers a user question in the form of a sentence. To do this, the method determines the information required, selects the template for the answer and sends requests to the world model and the planner to fill in the template.

In a problem where not all goals are feasible, Eifler et al. (2020) propose two types of explanations based on plan-property entailments. In this work, a plan property p , expressed as a Boolean function, entails q , which means that all plans satisfied by p are satisfied by q . A local explanation for a ‘Why not p ?’ question is to provide a set of undesirable plan properties that would be satisfied by satisfying p . A global explanation is a graph that returns all plan-property entailments. This method is used in Eifler and Hoffmann (2020) in an iterative planning context where the user can, via a user interface, ask questions about the different plan-properties.

Seegebarth et al. (2012) construct an axiomatic system composed of first-order logic formulae and then use it to explain the plan steps and the ordering constraints between plan steps. The explanations take the form of a sequence of applications of the axioms allowing the question about the plan to be answered.

In a problem where the user’s mental model differs from that of the agent, the explanation consists of a set of model changes so that the plan is optimal for both the agent and the user (Sreedharan, Chakraborti, et al. 2018). The aim is to provide the user with the minimum number of model changes. This approach provides explanations for an uncertain user model or a several-users model.

RADAR-X (Valmeekam et al. 2021) is a user interface that allows the user to ask contrastive questions about the plan in an interactive way. The user describes a plan, and the explanation provides a reduced set of information, based on Chakraborti, Sreedharan, Y. Zhang, et al. (2017), to explain the interest of the plan compared to the one proposed.

Given a set of positive and negative plans, Kim et al. (2019) generate a set of LTL formulas as an explanation with a Bayesian inference-based method that is robust to noise. In this context, noise corresponds to swapping plans from the two sets. The explanations describe the positive plans and not the negative ones.

In the context of multi-objective probabilistic planning, Sukkerd et al. (2018, 2020) propose to compare an optimal policy with alternative policies based on the different objectives. This contrastive approach takes the form of a verbal explanation where the values of the policies for the different objectives are compared. An algorithm is presented to generate alternative policies that are Pareto optimal according to a certain objective.

Two explanations are described in Edmonds et al. (2019) for a robotic task. This task is solved with a symbolic action planner and a haptic prediction model. The explanation extracted from the symbolic action planner is the symbolic action sequence performed by the robot and the one extracted from the haptic prediction model is a visualisation of the effects of the previous action.

Model Checking. This domain brings together a set of approaches that analyse a program according to correctness properties (for a survey of the domain, see Jhala and Majumdar 2009). Probabilistic (or statistical) model checking is a domain that could be of greater interest to XRL (for a survey, see Agha and Palmiskog 2018). It covers a range of methods for checking properties expressed in stochastic temporal logic. These approaches are

based on sampling, and return confidence scores associated with the respect of the properties studied. In the following, we briefly describe a few works that are more or less close to XRL and that use methods from these domains to provide explanations.

In a MARL context, Boggess et al. (2023) propose an algorithm that verifies the user request expressed in a probabilistic Computational Tree Logic formula (Aziz et al. 1995) using probabilistic model checking. The user request corresponds to a (partial) plan proposed by the user. The verification is performed on an abstraction of the multi-agent policy, which is updated if the request is not verified. If, despite this, the request is still not verified, the algorithm generates an explanation as to why it has not been respected.

TraceVis (Gros, Groß, et al. 2020) makes it possible to combine visualization methods with model checking, carried out with Deep Statistical Model Checking (Gros, Hermanns, et al. 2020). This approach makes it possible to analyse a policy represented by a NN.

Li et al. (2020) use a probabilistic model checker to provide a synthesised explanation to the user when necessary in a human-on-the-loop approach. In this work, an explanation is a content, effect and cost triplet.

Model checking and statistical model checking would make it possible to study the agent's policy concerning respecting a set of properties in order to analyse in more detail what the agent has learned. In addition to properties linked to the agent's performance, it would be interesting to consider more varied properties based on, for example, the use of a certain action (or strategy) that is deemed dangerous or costly by the user, or access to certain regions of the state space. Such diversity would help the user decide whether or not to use an agent's policy.

Algorithmic Recourse. This domain is generally applied to classification or regression models and aims to propose a set of approaches explaining a prediction in a contrastive way. The aim is to answer the question ‘*Why a rather than b from instance x?*’ and to provide recommendations for obtaining *b*. These recommendations can be seen as a set of actions to be performed from instance *x* to reach an instance *x'* where the output of the model would be *b*. For an overview of the different methods in this domain, we recommend Karimi, Barthe, et al. (2020).

As an example, the work of Karimi et al. (2021) presents a formulation of the problem that consists of generating counterfactual instances by minimizing not the change in features of the instance to be explained, but the cost of the actions that lead to a counterfactual instance. This reformulation makes it possible to avoid obtaining counterfactuals whose recommendations are sub-optimal or even infeasible. More recent work in XAI (Pfeifer et al. 2025; Rozajac et al. 2025) proposes approaches that are close to algorithmic recourse. Rozajac et al. propose a generative framework for producing counterfactual explanations in 3D semantic segmentation tasks. Pfeifer et al. introduce a model-agnostic method that generates global explanations of feature importance through counterfactual paths in knowledge-graph-based domains.

Algorithmic recourse seems to be an interesting area to investigate in order to provide XRL methods. Indeed, this approach could be used to explain the choice of one action over another (or the choice of one sequence of actions over another), and provide recommendations for reaching a state (resp. set of states), allowing the contrastive action (resp. sequence of actions) to be performed. This explanation could simply target the understanding of actions or outcomes (e.g. state, reward, respected predicate). Several works come close to this domain by providing *Counterfactual States* (Druce et al. 2021; Gajcin and Dusparic 2024b; Huber, Demmler, et al. 2023; Olson et al. 2021; Samadi et al. 2024) or *Counterfactual Sequences* (Alabdulkarim and Riedl 2022; Gajcin and Dusparic 2024a; Stefik et al. 2021; Tsirtsis, De, et al. 2021; Tsirtsis and Rodriguez 2023; Waa et al. 2018) but do not provide recommendations. Properties (e.g. recourse, sparsity) and ideas for counterfactual explanations specific to RL are proposed in Gajcin and Dusparic (2024c). One work has been identified as being directly inspired by the domain: Gajcin and Dusparic (2024b) describes a method for providing a counterfactual state reachable by agent actions from the base state.

RL Sub-domains. HRL (cf. Section 4.1.1), RRL and relational MDP (cf. Section 4.3.1) have been the subject of brief method overviews in this state of the art. These areas bring interpretability by default, hence are interesting to look at in more detail.

Similarly, Causal RL brings together a body of work that combines causality with RL. The aim of this combination is to improve the data efficiency of RL problems while providing interpretability on the causal relations between states, actions, etc. Although some of the work in this survey falls into this category (Druce et al. 2021; Herlau and Larsen 2022; Madumal et al. 2020a,b; Volodin 2021; Yu et al. 2023), we recommend the survey Zeng et al. (2023) which presents a general overview of the advances in this domain by categorising the work into two categories: approaches which must learn the causal information of an environment before the agent is trained, and those which already have this information.

RL as an Explanation Framework. The following section mentions different works found during our research that provide explanations for AI models using RL. To clarify, the idea behind this framework is to model (in part) the problem of finding explanations as an RL problem. Different AI models can be explained using this framework, such as classification or regression models (Z. Chen et al. 2022; Kohler, Akrou, et al. 2023; Lash 2022; Nguyen et al. 2021; Samoilescu 2021), recommendation systems (X. Wang, Y. Chen, et al. 2018) or policies learned via RL (Saulières et al. 2023b).

As an example of work, to answer in a counterfactual fashion the question ‘*Why predict a rather than b from instance x?*’, Lash proposes to describe the problem of generating a counterfactual instance of a classification or regression model as a decision-making problem, then solve it using a DRL algorithm.

For more details on the other works mentioned, we refer the reader to Appendix C.2.4.

We believe that the domains discussed are of real interest in the future design of XRL methods. We encourage researchers to read the various surveys highlighted: Agha and Palmskog (2018); Cashmore et al. (2019); Chakraborti, Sreedharan, and Kambhampati (2020); Fox et al. (2017); Hoffmann and Magazzeni (2019); Jhala and Majumdar (2009); Karimi, Barthe, et al. (2020); Otterlo (2012); Pateria et al. (2022); Zeng et al. (2023). The explainability of agents that have learned by reinforcement is a flourishing domain that requires particular attention on several points, or needs. We will list these needs below, which could also be considered for the XAI domain in general.

8 Needs for XRL

Compare Methods. Of the various works cited in this survey, a relatively small proportion compare their approach with other XRL methods. Indeed, of the 238 papers presented in this paper that fall directly within the field of XRL, only 55 works offer a comparison in terms of explainability with other XRL methods. As stated in the surveys (Milani, Topin, et al. 2024; Vouros 2023), it is necessary to compare XRL methods. In this way, for a given target (e.g. action) and for the same type of method (e.g. feature importance), developers could determine which method is best suited to the problem considered. To give a few examples, the works (Barceló et al. 2020; Pierson et al. 2024; Pocius et al. 2019; Xiong et al. 2024) tend to compare explainable methods in different ways.

For classifiers, Barceló et al. compare methods by studying of the computational complexity of generating explanations. The evaluation of XRL methods can either compare several types of explanation (Pierson et al.; Pocius et al.) or focus on a specific group of methods (Atrey et al. 2020; Xiong et al.).

For instance, Pierson et al. compare multiple XRL methods for explaining an agent’s policy through a user study. These methods include HIGHLIGHTS (D. Amir and O. Amir 2018), a combination of HIGHLIGHTS with saliency maps, graphs summarizing the agent’s behavior, and a hybrid approach combining saliency maps and graphs. Pocius et al. propose three evaluation tasks based on the StarCraft II environment (Vinyals et al. 2017) to assess XRL methods.

Conversely, (Atrey et al.; Xiong et al.) focus on evaluating a specific group of XRL methods. Xiong et al. introduce a benchmark dedicated to *Feature Importance* approaches, including a set of environments, implemented XRL

methods, and evaluation metrics such as fidelity, stability, and computation time. More specifically, Atrey et al. develop a dedicated methodology for evaluating saliency maps based on image interventions, allowing them to analyze the reliability of such explanations.

Provide Metrics. One of the shortcomings of XRL is the lack of a unified way of comparing methods, whatever their type or target. For a high level overview of metrics used in XRL, we refer the reader to Milani, Topin, et al. (2024). We believe that a unified assessment of methods is important, but also a more diverse set of metrics for each type of method in order to measure the quality of the explanations. In the following, we present different ways of measuring the quality of explanations and categorizing the evaluation methods in the XAI domain.

Zhou et al. (2021) present a survey of methods and metrics for evaluating XAI methods. A set of metrics are categorised according to the type of explanation evaluated and the properties of explanations taken into account by the metric (e.g. clarity, parsimony, soundness).

Doshi-Velez and Kim (2017) divide the methods for evaluating the interpretability of a model into three parts. Application-grounded evaluation groups together methods that rely on end-task experts to determine its interpretability. For human-grounded evaluations, this is determined using lay persons. Functionally-grounded evaluations are a set of methods that use functions defined by certain properties that act as interpretability metrics, and therefore do not require human experiments.

An overview of evaluation methods for interpretable machine learning models is given in F. Yang, Du, et al. (2019). Three properties are considered in the evaluation methods: generalizability, which determines the extent to which the explanations are specific to the instance, fidelity, which determines the extent to which the explainer's output matches the model's decision-making process, and persuasibility, which determines user satisfaction.

A set of measures is presented in Hoffman et al. (2018) for XAI. These are grouped into four sets: methods that evaluate the quality of explanations, user satisfaction, user understanding of the model and performance according to their mental model.

4 metrics are set out in the blue-sky paper (Rosenfeld 2021). D quantifies the difference in performance between the agent's opaque model and its surrogate model. R measures the simplicity of the proposed model, F the number of elements useful for generating the explanation and S the stability of the explainer.

Amgoud and Ben-Naim (2022) present a total of 10 axioms that explainers should satisfy. These axioms are limited to explainers whose explanation is a subset of the features of an instance. Among the axioms, irreducibility defines that an explainer should only contain useful features and feasibility defines that an explainer is a subset of at least one instance of the classification problem.

Depending on the dataset, Amiri et al. (2020) propose to generate a ground truth of explanations, which allows comparison with the explanations produced by LIME (Ribeiro et al. 2016).

In Wäldchen et al. (2022), several methods based on the notion of feature importance are compared using an auxiliary task. This task consists of guiding an agent playing Connect4 to win. In this context, the agent has only a partial observation of the board, which is given by a feature importance method. The idea is to measure, over a set of games, whether the agent manages to win based solely on the cells on the board indicated as important by an explainer.

Sokol and Flach (2020) present a set of elements to be considered when evaluating XAI methods, which are grouped according to 5 dimensions: functional, operational, usability, safety and validation.

Perform User Studies. Of the various works cited in this survey, a relatively small proportion validate their approach with a user study (which is also noted in Milani, Topin, et al. 2024; Wells and Bednarz 2021). Indeed, of the 238 papers presented in this paper that fall directly within the field of XRL, only 55 works conducted a user study to evaluate the proposed XRL method. As consumers of explanations of agents' behavior, it makes sense that any XRL method should at least be evaluated by a user study. We believe that taking end-users into

consideration in the development process of an explainable method is essential, whether the end-users are domain experts, developers or lay persons. Furthermore, with sufficiently in-depth user studies, it would be possible to identify the strengths and weaknesses of the approach, and then use this feedback to improve the approach.

Develop User Interface. In line with the above need, we believe that it is important to propose, in addition to an XRL method, a dedicated user interface. The construction of a toolbox is suggested as a future direction in Vouras (2023). An interface should be intuitive and ergonomic so that the user gets the most out of the explainability method. In user studies, a more or less basic interface is proposed. The various works explaining the agent's policy through *Visual Toolkit* (W. He et al. 2020; Jaunet et al. 2020; McGregor et al. 2017; A. Mishra et al. 2021; J. Wang et al. 2019) is a good example of a way of thinking about explainability. In our opinion, it is as important to provide a good XRL method as it is to provide a user interface that improves its usefulness from a user's point of view.

9 Conclusion

This paper has categorised recent work in the field of XRL using two questions: 'What?' and 'How?'. 3 targets for explanations have been identified, namely the agent's policy, the state-action sequence resulting from the agent's interaction within the environment and the agent's choice of action.

This state of the art allows us to highlight the low interest of researchers in explaining sequences of agent interactions: 176 works use methods which are categorised as *Policy-level methods*, 90 as *action-level methods* and only 13 as *sequence-level methods*. We assume that this is due to the fact that XAI for classifiers influences XRL. Indeed, the vast majority of classifier explanation methods simply explain a decision locally or the model globally. On the other hand, many methods have been proposed to explain a simple agent decision or policy. Among the different approaches identified, the majority of methods that explain an agent's decision by *Feature Importance* do so using saliency maps, although this approach is limited to agents whose state is an image. For methods based on expected outcomes, the majority return states or features of states as explanations. For policy, the majority of papers fall into the *Interpretable Policy* category, with a predominance of methods proposing directly interpretable policies.

This state of the art proposes an intuitive taxonomy that allows us to quickly identify a set of works related to the target we want to explain and the way we want to do it. This paper has also enabled us to highlight the types of work done to explain or make interpretable one of the targets, to show the use of methods initially intended for the explanation of classifiers in the context of XRL. Moreover, it briefly describes a set of domains that we consider relevant to explore in order to propose new XRL methods and lists several needs for this topic.

Acknowledgments

We thank the reviewer for their valuable feedback and suggestions that helped improve this survey.

References

- K. D. Abeyrathna, O. Granmo, L. Jiao, and M. Goodwin. 2019. "The Regression Tsetlin Machine: A Tsetlin Machine for Continuous Output Problems." In: *Progress in Artificial Intelligence, 19th EPIA Conference on Artificial Intelligence, EPIA 2019, Vila Real, Portugal, September 3-6, 2019, Proceedings, Part II* (Lecture Notes in Computer Science). Ed. by P. M. Oliveira, P. Novais, and L. P. Reis. Vol. 11805. Springer, 268–280. doi:[10.1007/978-3-030-30244-3_23](https://doi.org/10.1007/978-3-030-30244-3_23).
- A. Acharya, R. L. Russell, and N. R. Ahmed. 2020. "Explaining Conditions for Reinforcement Learning Behaviors from Real and Imagined Data." *CoRR*, abs/2011.09004. <https://arxiv.org/abs/2011.09004> arXiv: 2011.09004.
- J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim. 2018. "Sanity Checks for Saliency Maps." In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 9525–9536. <https://proceedings.neurips.cc/paper/2018/hash/294a8ed24b1ad22ec2e7efea049b8737-Abstract.html>.

- D. Adjodah, T. Klinger, and J. Joseph. 2018. "Symbolic relation networks for reinforcement learning." In: *Proceedings of the Workshop on Relational Representation Learning in Conference on Neural Information Processing Systems (NeurIPS)*. <https://r2learning.github.io/assets/papers/CameraReadySubmission%203.pdf>.
- G. Agha and K. Palmiskog. 2018. "A Survey of Statistical Model Checking." *ACM Trans. Model. Comput. Simul.*, 28, 1, 6:1–6:39. doi:10.1145/3158668.
- W. Agnew and P. Domingos. 2018. "Unsupervised object-level deep reinforcement learning." In: *NeurIPS Workshop on Deep RL*. <https://homes.cs.washington.edu/~pedrod/papers/drl18.pdf>.
- R. Akrou, D. Tateo, and J. Peters. 2019. "Towards reinforcement learning of human readable policies." In: *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases: The 1st Workshop on Deep Continuous-Discrete Machine Learning*. <https://www.ias.informatik.tu-darmstadt.de/uploads/Team/RiadAkrou/decodeml2019.pdf>.
- M. Alaarabiou, N. Delestre, and L. Vercoeur. 2024. "Explicabilité en Apprentissage par Renforcement: vers une Taxinomie Unifiée." *JIAF-JFPDA*, 90. https://pfia2024.univ-lr.fr/assets/files/Conf%5C%C3%5C%A9rence-JIAF/JIAF2024_paper_11.pdf.
- A. Alabdulkarim and M. O. Riedl. 2022. "Experiential Explanations for Reinforcement Learning." *CoRR*, abs/2210.04723. arXiv: 2210.04723. doi:10.48550/ARXIV.2210.04723.
- P. Altmann, C. Davignon, M. Zorn, F. Ritz, C. Linnhoff-Popien, and T. Gabor. 2024. "REACT: Revealing Evolutionary Action Consequence Trajectories for Interpretable Reinforcement Learning." *CoRR*, abs/2404.03359. arXiv: 2404.03359. doi:10.48550/ARXIV.2404.03359.
- L. Amgoud and J. Ben-Naim. 2022. "Axiomatic Foundations of Explainability." In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*. Ed. by L. D. Raedt. *ijcai.org*, 636–642. doi:10.24963/IJCAI.2022/90.
- D. Amir and O. Amir. 2018. "HIGHLIGHTS: Summarizing Agent Behavior to People." In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS*. Ed. by E. André, S. Koenig, M. Dastani, and G. Sukthankar. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 1168–1176. <http://dl.acm.org/citation.cfm?id=3237869>.
- O. Amir, F. Doshi-Velez, and D. Sarne. 2018. "Agent Strategy Summarization." In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*. Ed. by E. André, S. Koenig, M. Dastani, and G. Sukthankar. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 1203–1207. <http://dl.acm.org/citation.cfm?id=3237877>.
- O. Amir, F. Doshi-Velez, and D. Sarne. 2019. "Summarizing agent strategies." *Auton. Agents Multi Agent Syst.*, 33, 5, 628–644. doi:10.1007/S10458-019-09418-W.
- S. S. Amiri, R. O. Weber, P. Goel, O. Brooks, A. Gandley, B. Kitchell, and A. Zehm. 2020. "Data Representing Ground-Truth Explanations to Evaluate XAI Methods." *CoRR*, abs/2011.09892. <https://arxiv.org/abs/2011.09892> arXiv: 2011.09892.
- Y. Amitai and O. Amir. 2022. "I Don't Think So": Summarizing Policy Disagreements for Agent Comparison." In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelfth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 5269–5276. doi:10.1609/AAAI.V36I5.20463.
- Y. Amitai and O. Amir. 2024. "A survey of global explanations in reinforcement learning." In: *Explainable agency in artificial intelligence*. CRC Press, 21–42.
- Y. Amitai, Y. Septon, and O. Amir. 2024. "Explaining Reinforcement Learning Agents through Counterfactual Action Outcomes." In: *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*. Ed. by M. J. Wooldridge, J. G. Dy, and S. Natarajan. AAAI Press, 10003–10011. doi:10.1609/AAAI.V38I9.28863.
- A. Anderson et al.. 2020. "Mental Models of Mere Mortals with Explanations of Reinforcement Learning." *ACM Trans. Interact. Intell. Syst.*, 10, 2, 15:1–15:37. doi:10.1145/3366485.
- J. Andreas, D. Klein, and S. Levine. 2017. "Modular Multitask Reinforcement Learning with Policy Sketches." In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (Proceedings of Machine Learning Research). Ed. by D. Precup and Y. W. Teh. Vol. 70. PMLR, 166–175. <http://proceedings.mlr.press/v70/andreas17a.html>.
- G. Angelotti and N. Díaz-Rodríguez. 2023. "Towards a more efficient computation of individual attribute and policy contribution for post-hoc explanation of cooperative multi-agent systems using Myerson values." *Knowledge-Based Systems*, 260, 110189. <https://doi.org/10.1016/j.knsys.2022.110189>.
- P. P. Angelov and D. P. Filev. 2004. "An approach to online identification of Takagi-Sugeno fuzzy models." *IEEE Trans. Syst. Man Cybern. Part B*, 34, 1, 484–498. doi:10.1109/TSMCB.2003.817053.
- R. M. Annasamy and K. P. Sycara. 2019. "Towards Better Interpretability in Deep Q-networks." In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 4561–4569. doi:10.1609/AAAI.V33I01.33014561.
- J. A. Arjona-Medina, M. Gillhofer, M. Widrich, T. Unterthiner, J. Brandstetter, and S. Hochreiter. 2019. "RUDDER: Return Decomposition for Delayed Rewards." In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*

- 2019, *NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, 13544–13555. <https://proceedings.neurips.cc/paper/2019/hash/16105fb9cc614fc29e1bda00dab60d41-Abstract.html>.
- I. Arnaldo, U. O'Reilly, and K. Veeramachaneni. 2015. "Building Predictive Models via Feature Synthesis." In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11-15, 2015*. Ed. by S. Silva and A. I. Esparcia-Alcázar. ACM, 983–990. doi:10.1145/2739480.2754693.
- K. J. Åström. 1965. "Optimal control of Markov processes with incomplete state information I." *Journal of mathematical analysis and applications*, 10, 174–205. <https://lucris.lub.lu.se/ws/portalfiles/portal/5323668>.
- A. Atrey, K. Clary, and D. D. Jensen. 2020. "Exploratory Not Explanatory: Counterfactual Analysis of Saliency Maps for Deep Reinforcement Learning." In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=rkl3m1BFDB>.
- J. Ault, J. P. Hanna, and G. Sharon. 2020. "Learning an Interpretable Traffic Signal Control Policy." In: *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*. Ed. by A. E. F. Seghrouchni, G. Sukthankar, B. An, and N. Yorke-Smith. International Foundation for Autonomous Agents and Multiagent Systems, 88–96. doi:10.5555/3398761.3398777.
- A. Aziz, V. Singhal, and F. Balarin. 1995. "It Usually Works: The Temporal Logic of Stochastic Systems." In: *Computer Aided Verification, 7th International Conference, Liège, Belgium, July, 3-5, 1995, Proceedings* (Lecture Notes in Computer Science). Ed. by P. Wolper. Vol. 939. Springer, 155–165. doi:10.1007/3-540-60045-0_48.
- S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. 2015. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." *PloS one*, 10, 7, e0130140. <https://doi.org/10.1371/journal.pone.0130140>.
- W. Bao, Q. Yu, and Y. Kong. 2021. "DRIVE: Deep Reinforced Accident Anticipation with Visual Explanation." In: *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 7599–7608. doi:10.1109/ICCV48922.2021.00752.
- P. Barceló, M. Monet, J. Pérez, and B. Subercaux. 2020. "Model Interpretability through the lens of Computational Complexity." In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. <https://proceedings.neurips.cc/paper/2020/hash/b1adda14824f50ef24ff1c05bb66faf3-Abstract.html>.
- P. V. A. Barros, A. Tanevska, F. Cruz, and A. Sciutti. 2020. "Moody Learners - Explaining Competitive Behaviour of Reinforcement Learning Agents." In: *Joint IEEE 10th International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2020, Valparaíso, Chile, October 26-30, 2020*. IEEE, 1–8. doi:10.1109/ICDL-EPIROB48136.2020.9278125.
- O. Bastani, J. P. Inala, and A. Solar-Lezama. 2020. "Interpretable, Verifiable, and Robust Reinforcement Learning via Program Synthesis." In: *xxAI - Beyond Explainable AI - International Workshop, Held in Conjunction with ICML 2020, July 18, 2020, Vienna, Austria, Revised and Extended Papers* (Lecture Notes in Computer Science). Ed. by A. Holzinger, R. Goebel, R. Fong, T. Moon, K. Müller, and W. Samek. Vol. 13200. Springer, 207–228. doi:10.1007/978-3-031-04083-2_11.
- O. Bastani, Y. Pu, and A. Solar-Lezama. 2018. "Verifiable Reinforcement Learning via Policy Extraction." In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 2499–2509. <https://proceedings.neurips.cc/paper/2018/hash/e6d8545daa42d5ced125a4bf747b3688-Abstract.html>.
- D. Beechey, T. M. S. Smith, and Ö. Simsek. 2023. "Explaining Reinforcement Learning with Shapley Values." In: *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA* (Proceedings of Machine Learning Research). Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett. Vol. 202. PMLR, 2003–2014. <https://proceedings.mlr.press/v202/beechey23a.html>.
- Y. Bekkemoen and H. Langseth. 2023. "ASAP: Attention-Based State Space Abstraction for Policy Summarization." In: *Asian Conference on Machine Learning, ACML 2023, 11-14 November 2023, Istanbul, Turkey* (Proceedings of Machine Learning Research). Ed. by B. Yanikoglu and W. L. Buntine. Vol. 222. PMLR, 137–152. <https://proceedings.mlr.press/v222/bekkemoen24a.html>.
- M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. "The Arcade Learning Environment: An Evaluation Platform for General Agents." *J. Artif. Intell. Res.*, 47, 253–279. doi:10.1613/JAIR.3912.
- N. Ben-Zrihem, T. Zahavy, and S. Mannor. 2016. "Visualizing Dynamics: from t-SNE to SEMI-MDPs." *CoRR*, abs/1606.07112. <http://arxiv.org/abs/1606.07112> arXiv: 1606.07112.
- D. P. Bertsekas. 2011. "Approximate policy iteration: A survey and some new methods." *Journal of Control Theory and Applications*, 9, 3, 310–335. <https://doi.org/10.1007/s11768-011-1005-3>.
- T. Bewley and J. Lawry. 2021. "TripleTree: A Versatile Interpretable Representation of Black Box Agents and their Environments." In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 11415–11422. doi:10.1609/AAAI.V35I13.17360.
- T. Bewley, J. Lawry, and A. Richards. 2022. "Summarising and Comparing Agent Dynamics with Contrastive Spatiotemporal Abstraction." *CoRR*, abs/2201.07749. <https://arxiv.org/abs/2201.07749> arXiv: 2201.07749.

- T. Bewley and F. Lécué. 2022. "Interpretable Preference-based Reinforcement Learning with Tree-Structured Reward Functions." In: *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. Ed. by P. Faliszewski, V. Mascardi, C. Pelachaud, and M. E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 118–126. doi:[10.5555/3535850.3535865](https://doi.org/10.5555/3535850.3535865).
- B. Beyret, A. Shafti, and A. A. Faisal. 2019. "Dot-to-Dot: Explainable Hierarchical Reinforcement Learning for Robotic Manipulation." In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019, Macau, SAR, China, November 3-8, 2019*. IEEE, 5014–5019. doi:[10.1109/IROS40897.2019.8968488](https://doi.org/10.1109/IROS40897.2019.8968488).
- I. Bica, D. Jarrett, A. Hüyük, and M. van der Schaar. 2021. "Learning "What-if" Explanations for Sequential Decision-Making." In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=h0de3QWtGG>.
- K. Boggess, S. Kraus, and L. Feng. 2023. "Explainable Multi-Agent Reinforcement Learning for Temporal Queries." In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*. ijcai.org, 55–63. doi:[10.24963/IJCAI.2023/7](https://doi.org/10.24963/IJCAI.2023/7).
- G. Böhm and H.-R. Pfister. 2015. "How people explain their own and others' behavior: a theory of lay causal explanations." *Frontiers in psychology*, 6, 109763. <https://doi.org/10.3389/fpsyg.2015.00139>.
- S. Booth, C. Muise, and J. Shah. 2019. "Evaluating the Interpretability of the Knowledge Compilation Map: Communicating Logical Statements Effectively." In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by S. Kraus. ijcai.org, 5801–5807. doi:[10.24963/IJCAI.2019/804](https://doi.org/10.24963/IJCAI.2019/804).
- G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. 2016. *Openai gym*. (2016). <https://arxiv.org/abs/1606.01540>.
- M. Burke, S. Penkov, and S. Ramamoorthy. 2019. "From Explanation to Synthesis: Compositional Program Induction for Learning from Demonstration." In: *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*. Ed. by A. Bicchi, H. Kress-Gazit, and S. Hutchinson. doi:[10.15607/RSS.2019.XV.015](https://doi.org/10.15607/RSS.2019.XV.015).
- A. Camacho, R. T. Icarte, T. Q. Klassen, R. A. Valenzano, and S. A. McIlraith. 2019. "LTL and Beyond: Formal Languages for Reward Function Specification in Reinforcement Learning." In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by S. Kraus. ijcai.org, 6065–6073. doi:[10.24963/IJCAI.2019/840](https://doi.org/10.24963/IJCAI.2019/840).
- N. B. Carbone. 2020. "Explainable AI for path following with Model Trees." Master's thesis. NTNU. <https://hdl.handle.net/11250/2780917>.
- M. Cashmore, A. Collins, B. Krarup, S. Krivic, D. Magazzeni, and D. E. Smith. 2019. "Towards Explainable AI Planning as a Service." *CoRR*, abs/1908.05059. [http://arxiv.org/abs/1908.05059](https://arxiv.org/abs/1908.05059) arXiv: 1908.05059.
- T. Chakraborti, S. Sreedharan, and S. Kambhampati. 2020. "The Emerging Landscape of Explainable Automated Planning & Decision Making." In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*. Ed. by C. Bessiere. ijcai.org, 4803–4811. doi:[10.24963/IJCAI.2020/669](https://doi.org/10.24963/IJCAI.2020/669).
- T. Chakraborti, S. Sreedharan, Y. Zhang, and S. Kambhampati. 2017. "Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy." In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. Ed. by C. Sierra. ijcai.org, 156–163. doi:[10.24963/IJCAI.2017/23](https://doi.org/10.24963/IJCAI.2017/23).
- W. Chang, W. Lizhen, Y. Chao, W. Zhichao, L. Han, and Y. Chao. 2020. "Coactive design of explainable agent-based task planning and deep reinforcement learning for human-UAVs teamwork." *Chinese Journal of Aeronautics*, 33, 11, 2930–2945. <https://doi.org/10.1016/j.cja.2020.05.001>.
- J. Chen, S. E. Li, and M. Tomizuka. 2022. "Interpretable End-to-End Urban Autonomous Driving With Latent Deep Reinforcement Learning." *IEEE Trans. Intell. Transp. Syst.*, 23, 6, 5068–5078. doi:[10.1109/TITS.2020.3046646](https://doi.org/10.1109/TITS.2020.3046646).
- T. Chen and C. Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi. ACM, 785–794. doi:[10.1145/2939672.2939785](https://doi.org/10.1145/2939672.2939785).
- Z. Chen, F. Silvestri, J. Wang, H. Zhu, H. Ahn, and G. Tolomei. 2022. "ReLAX: Reinforcement Learning Agent Explainer for Arbitrary Predictive Models." In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*. Ed. by M. A. Hasan and L. Xiong. ACM, 252–261. doi:[10.1145/3511808.3557429](https://doi.org/10.1145/3511808.3557429).
- Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo. 2018. "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation." In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 8789–8797. doi:[10.1109/CVPR.2018.00916](https://doi.org/10.1109/CVPR.2018.00916).
- P. Cichosz and L. Pawelczak. 2014. "Imitation learning of car driving skills with decision trees and random forests." *Int. J. Appl. Math. Comput. Sci.*, 24, 3, 579–597. doi:[10.2478/AMCS-2014-0042](https://doi.org/10.2478/AMCS-2014-0042).
- A. Climent, D. Gnatyshak, and S. Álvarez-Napagao. 2021. "Applying and Verifying an Explainability Method Based on Policy Graphs in the Context of Reinforcement Learning." In: *Artificial Intelligence Research and Development - Proceedings of the 23rd International Conference of the Catalan Association for Artificial Intelligence, CCIA 2021, Virtual Event, 20-22 October, 2021* (Frontiers in Artificial Intelligence and Applications). Ed. by M. Villaret, T. Alsinet, C. Fernández, and A. Valls. Vol. 339. IOS Press, 455–464. doi:[10.3233/FAIA210166](https://doi.org/10.3233/FAIA210166).

- J. A. Clouse. 1996. *On integrating apprentice learning and reinforcement learning*. University of Massachusetts Amherst. <https://www.proquest.com/openview/b76443e456463ddae17fcec0a16f8b8c/>.
- K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman. 2019. “Quantifying Generalization in Reinforcement Learning.” In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (Proceedings of Machine Learning Research). Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. PMLR, 1282–1289. <http://proceedings.mlr.press/v97/cobbe19a.html>.
- J. Cole, J. Lloyd, and K. S. Ng. 2003. “Symbolic learning for adaptive agents.” In: *Proceedings of the Annual Partner Conference, Smart Internet Technology Cooperative Research Centre*. <https://www.researchgate.net/publication/2898449>.
- Y. Coppens, K. Efthymiadis, T. Lenaerts, A. Nowé, T. Miller, R. Weber, and D. Magazzeni. 2019. “Distilling deep reinforcement learning policies in soft decision trees.” In: *Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence*, 1–6. <https://cris.vub.be/ws/portalfiles/portal/46718934/>.
- P. T. Costa Jr and R. R. McCrae. 1996. “Mood and personality in adulthood.” In: *Handbook of emotion, adult development, and aging*. Elsevier, 369–383. <http://dx.doi.org/10.1016/B978-012464995-8/50021-2>.
- F. Cruz, R. Dazeley, and P. Vamplew. 2019. “Memory-Based Explainable Reinforcement Learning.” In: *AI 2019: Advances in Artificial Intelligence - 32nd Australasian Joint Conference, Adelaide, SA, Australia, December 2-5, 2019, Proceedings* (Lecture Notes in Computer Science). Ed. by J. Liu and J. Bailey. Vol. 11919. Springer, 66–77. doi:10.1007/978-3-030-35288-2_6.
- F. Cruz, R. Dazeley, P. Vamplew, and I. Moreira. 2023. “Explainable robotic systems: understanding goal-driven actions in a reinforcement learning scenario.” *Neural Comput. Appl.*, 35, 25, 18113–18130. doi:10.1007/S00521-021-06425-5.
- L. L. Custode and G. Iacca. 2023. “Evolutionary Learning of Interpretable Decision Trees.” *IEEE Access*, 11, 6169–6184. doi:10.1109/ACCESS.2023.33236260.
- Y. Dai, H. Ouyang, H. Zheng, H. Long, and X. Duan. 2023. “Interpreting a deep reinforcement learning model with conceptual embedding and performance analysis.” *Appl. Intell.*, 53, 6, 6936–6952. doi:10.1007/S10489-022-03788-7.
- Y. Dai, Q. Chen, et al.. 2022. “Enhanced oblique decision tree enabled policy extraction for deep reinforcement learning in power system emergency control.” *Electric Power Systems Research*, 209, 107932. <https://doi.org/10.1016/j.epsr.2022.107932>.
- M. H. Danesh, A. Koul, A. Fern, and S. Khorram. 2021. “Re-understanding Finite-State Representations of Recurrent Policy Networks.” In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event* (Proceedings of Machine Learning Research). Ed. by M. Meila and T. Zhang. Vol. 139. PMLR, 2388–2397. <http://proceedings.mlr.press/v139/danesh21a.html>.
- G. Dao, W. H. Huff, and M. Lee. 2021. “Learning Sparse Evidence- Driven Interpretation to Understand Deep Reinforcement Learning Agents.” In: *IEEE Symposium Series on Computational Intelligence, SSCI 2021, Orlando, FL, USA, December 5-7, 2021*. IEEE, 1–7. doi:10.1109/SSCI50451.2021.9660192.
- G. Dao, I. Mishra, and M. Lee. 2018. “Deep Reinforcement Learning Monitor for Snapshot Recording.” In: *17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018, Orlando, FL, USA, December 17-20, 2018*. Ed. by M. A. Wani, M. M. Kantardzic, M. S. Mouchaweh, J. Gama, and E. Lughofer. IEEE, 591–598. doi:10.1109/ICMLA.2018.00095.
- S. Das, S. Natarajan, K. Roy, R. Parr, and K. Kersting. 2020. “Fitted Q-learning for Relational Domains.” *CoRR*, abs/2006.05595. <https://arxiv.org/abs/2006.05595>.
- O. Davoodi and M. Komeili. 2021. “Feature-Based Interpretable Reinforcement Learning based on State-Transition Models.” In: *2021 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2021, Melbourne, Australia, October 17-20, 2021*. IEEE, 301–308. doi:10.1109/SMC52423.2021.9658917.
- R. Dazeley, P. Vamplew, and F. Cruz. 2023. “Explainable reinforcement learning for broad-XAI: a conceptual framework and survey.” *Neural Comput. Appl.*, 35, 23, 16893–16916. doi:10.1007/S00521-023-08423-1.
- T. Degris, O. Sigaud, and P. Willemin. 2006. “Learning the structure of Factored Markov Decision Processes in reinforcement learning problems.” In: *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006* (ACM International Conference Proceeding Series). Ed. by W. W. Cohen and A. W. Moore. Vol. 148. ACM, 257–264. doi:10.1145/1143844.1143877.
- A. Dethise, M. Canini, and S. Kandula. 2019. “Cracking Open the Black Box: What Observations Can Tell Us About Reinforcement Learning Agents.” In: *Proceedings of the 2019 Workshop on Network Meets AI & ML, NetAI@SIGCOMM 2019, Beijing, China, August 23, 2019*. ACM, 29–36. doi:10.1145/3341216.3342210.
- R. Devidze, G. Radanovic, P. Kamalaruban, and A. Singla. 2021. “Explicable Reward Design for Reinforcement Learning Agents.” In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 20118–20131. <https://proceedings.neurips.cc/paper/2021/hash/a7f0d2b95c60161b3f3c82f764b1d1c9-Abstract.html>.
- Y. D. Dhebar and K. Deb. 2021. “Interpretable Rule Discovery Through Bilevel Optimization of Split-Rules of Nonlinear Decision Trees for Classification Problems.” *IEEE Trans. Cybern.*, 51, 11, 5573–5584. doi:10.1109/TCYB.2020.3033003.
- Y. D. Dhebar, K. Deb, S. Nagesh Rao, L. Zhu, and D. P. Filev. 2020. “Interpretable-AI Policies using Evolutionary Nonlinear Decision Trees for Discrete Action Systems.” *CoRR*, abs/2009.09521. <https://arxiv.org/abs/2009.09521> arXiv: 2009.09521.
- E. W. Dijkstra. 1959. “A note on two problems in connexion with graphs.” *Numerische Mathematik*, 1, 269–271. doi:10.1007/BF01386390.

- Z. Ding, P. Hernandez-Leal, G. W. Ding, C. Li, and R. Huang. 2020. “CDT: Cascading Decision Trees for Explainable Reinforcement Learning.” *CoRR*, abs/2011.07553. <https://arxiv.org/abs/2011.07553> arXiv: 2011.07553.
- M. Dinu, M. Hofmarcher, V. P. Patil, M. Dorfer, P. M. Blies, J. Brandstetter, J. A. Arjona-Medina, and S. Hochreiter. 2020. “XAI and Strategy Extraction via Reward Redistribution.” In: *xxAI - Beyond Explainable AI - International Workshop, Held in Conjunction with ICML 2020, July 18, 2020, Vienna, Austria, Revised and Extended Papers* (Lecture Notes in Computer Science). Ed. by A. Holzinger, R. Goebel, R. Fong, T. Moon, K. Müller, and W. Samek. Vol. 13200. Springer, 177–205. doi:10.1007/978-3-031-04083-2_10.
- C. Diuk, A. Cohen, and M. L. Littman. 2008. “An object-oriented representation for efficient reinforcement learning.” In: *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008* (ACM International Conference Proceeding Series). Ed. by W. W. Cohen, A. McCallum, and S. T. Roweis. Vol. 307. ACM, 240–247. doi:10.1145/1390156.1390187.
- T. Dodson, N. Mattei, and J. Goldsmith. 2011. “A Natural Language Argumentation Interface for Explanation Generation in Markov Decision Processes.” In: *Algorithmic Decision Theory - Second International Conference, ADT 2011, Piscataway, NJ, USA, October 26-28, 2011. Proceedings* (Lecture Notes in Computer Science). Ed. by R. I. Brafman, F. S. Roberts, and A. Tsoukiàs. Vol. 6992. Springer, 42–55. doi:10.1007/978-3-642-24873-3_4.
- M. Domenech i Vila, D. Gnatyshak, A. Tormos, V. Gimenez-Abalos, and S. Alvarez-Napagao. 2024. “Explaining the Behaviour of Reinforcement Learning Agents in a Multi-Agent Cooperative Environment Using Policy Graphs.” *Electronics*, 13, 3, 573. <http://dx.doi.org/10.3390/electronics13030573>.
- H. Dong, J. Mao, T. Lin, C. Wang, L. Li, and D. Zhou. 2019. “Neural Logic Machines.” *CoRR*, abs/1904.11694. <http://arxiv.org/abs/1904.11694> arXiv: 1904.11694.
- F. Doshi-Velez and B. Kim. 2017. *Towards a rigorous science of interpretable machine learning*. (2017). <https://doi.org/10.48550/arXiv.1702.08608>.
- N. Douglas, D. Yim, B. Kartal, P. Hernandez-Leal, F. Maurer, and M. E. Taylor. 2019. “Towers of Saliency: A Reinforcement Learning Visualization Using Immersive Environments.” In: *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces, ISS 2019, Daejeon, South Korea, November 10-13, 2019*. Ed. by B. Lee, G. Lee, S. D. Scott, M. Tory, and J. Kim. ACM, 339–342. doi:10.1145/3343055.3360747.
- K. Driessens and H. Blockeel. 2001. “Learning digger using hierarchical reinforcement learning for concurrent goals.” In: *Proceedings of the European Workshop on Reinforcement Learning*. CKI Utrecht University, 11–12.
- J. Druce, M. Harradon, and J. Tittle. 2021. “Explainable Artificial Intelligence (XAI) for Increasing User Trust in Deep Reinforcement Learning Driven Autonomous Systems.” *CoRR*, abs/2106.03775. <https://arxiv.org/abs/2106.03775> arXiv: 2106.03775.
- S. Dzeroski, L. D. Raedt, and K. Driessens. 2001. “Relational Reinforcement Learning.” *Mach. Learn.*, 43, 1/2, 7–52. doi:10.1023/A:1007694015589.
- M. Edmonds et al.. 2019. “A tale of two explanations: Enhancing human trust by explaining robot behavior.” *Sci. Robotics*, 4, 37. doi:10.1126/SCIROBOTICS.AAY4663.
- U. Ehsan, B. Harrison, L. Chan, and M. O. Riedl. 2018. “Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations.” In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*. Ed. by J. Furman, G. E. Marchant, H. Price, and F. Rossi. ACM, 81–87. doi:10.1145/3278721.3278736.
- U. Ehsan, P. Tambwekar, L. Chan, B. Harrison, and M. O. Riedl. 2019. “Automated rationale generation: a technique for explainable AI and its effects on human perceptions.” In: *Proceedings of the 24th International Conference on Intelligent User Interfaces, IUI 2019, Marina del Ray, CA, USA, March 17-20, 2019*. Ed. by W. Fu, S. Pan, O. Brdiczka, P. Chau, and G. Calvary. ACM, 263–274. doi:10.1145/3301275.3302316.
- R. Eifler, M. Cashmore, J. Hoffmann, D. Magazzeni, and M. Steinmetz. 2020. “A New Approach to Plan-Space Explanation: Analyzing Plan-Property Dependencies in Oversubscription Planning.” In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 9818–9826. doi:10.1609/AAAI.V34I06.6534.
- R. Eifler and J. Hoffmann. 2020. “Iterative Planning with Plan-Space Explanations: A Tool and User Study.” *CoRR*, abs/2011.09705. <https://arxiv.org/abs/2011.09705> arXiv: 2011.09705.
- F. Elizalde, L. E. Sucar, J. Noguez, and A. Reyes. 2009. “Generating Explanations Based on Markov Decision Processes.” In: *MICAI 2009: Advances in Artificial Intelligence, 8th Mexican International Conference on Artificial Intelligence, Guanajuato, Mexico, November 9-13, 2009. Proceedings* (Lecture Notes in Computer Science). Ed. by A. H. Aguirre, R. M. Borja, and C. A. R. García. Vol. 5845. Springer, 51–62. doi:10.1007/978-3-642-05258-3_5.
- D. Ernst, P. Geurts, and L. Wehenkel. 2005. “Tree-Based Batch Mode Reinforcement Learning.” *J. Mach. Learn. Res.*, 6, 503–556. <http://jmlr.org/papers/v6/ernst05a.html>.
- M. Erwig, A. Fern, M. Murali, and A. Koul. 2018. “Explaining deep adaptive programs via reward decomposition.” In: *IJCAI/ECAP workshop on explainable artificial intelligence*. <https://par.nsf.gov/biblio/10096985>.
- R. Evans and E. Grefenstette. 2018. “Learning Explanatory Rules from Noisy Data.” *J. Artif. Intell. Res.*, 61, 1–64. doi:10.1613/JAIR.5714.
- B. Eysenbach, R. Salakhutdinov, and S. Levine. 2019. “Search on the Replay Buffer: Bridging Planning and Reinforcement Learning.” In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, 15220–15231. <https://proceedings.neurips.cc/paper/2019/hash/5c48ff18e0a47baaf81d8b8ea51eec92-Abstract.html>.

- F. Feit, A. Metzger, and K. Pohl. 2022. “Explaining Online Reinforcement Learning Decisions of Self-Adaptive Systems.” In: *IEEE International Conference on Autonomic Computing and Self-Organizing Systems, ACSOS 2022, Virtual, CA, USA, September 19-23, 2022*. Ed. by R. Casadei et al. IEEE, 51–60. doi:[10.1109/ACSOS55765.2022.00023](https://doi.org/10.1109/ACSOS55765.2022.00023).
- M. Finkelstein, L. Liu, Y. Kolumbus, D. C. Parkes, J. S. Rosenshein, and S. Keren. 2022. “Reinforcement Learning Explainability via Model Transforms (Student Abstract).” In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 12943–12944. doi:[10.1609/AAAI.V36I11.21608](https://doi.org/10.1609/AAAI.V36I11.21608).
- M. Fox, D. Long, and D. Magazzeni. 2017. “Explainable Planning.” *CoRR*, abs/1709.10256. <http://arxiv.org/abs/1709.10256> arXiv: [1709.10256](https://arxiv.org/abs/1709.10256).
- V. François-Lavet, Y. Bengio, D. Precup, and J. Pineau. 2019. “Combined Reinforcement Learning via Abstract Representations.” In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 3582–3589. doi:[10.1609/AAAI.V33I01.33013582](https://doi.org/10.1609/AAAI.V33I01.33013582).
- N. Frosst and G. E. Hinton. 2017. “Distilling a Neural Network Into a Soft Decision Tree.” In: *Proceedings of the First International Workshop on Comprehensibility and Explanation in AI and ML 2017 co-located with 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017), Bari, Italy, November 16th and 17th, 2017* (CEUR Workshop Proceedings). Ed. by T. R. Besold and O. Kutz. Vol. 2071. CEUR-WS.org. https://ceur-ws.org/Vol-2071/CEXAIIA%5C_2017%5C_paper%5C_3.pdf.
- J. Frost, O. Watkins, E. Weiner, P. Abbeel, T. Darrell, B. A. Plummer, and K. Saenko. 2022. “Explaining Reinforcement Learning Policies through Counterfactual Trajectories.” *CoRR*, abs/2201.12462. <https://arxiv.org/abs/2201.12462> arXiv: [2201.12462](https://arxiv.org/abs/2201.12462).
- J. Fu, K. Luo, and S. Levine. 2017. “Learning Robust Rewards with Adversarial Inverse Reinforcement Learning.” *CoRR*, abs/1710.11248. <http://arxiv.org/abs/1710.11248> arXiv: [1710.11248](https://arxiv.org/abs/1710.11248).
- Y. Fukuchi, M. Osawa, H. Yamakawa, and M. Imai. 2017a. “Application of Instruction-Based Behavior Explanation to a Reinforcement Learning Agent with Changing Policy.” In: *Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part I* (Lecture Notes in Computer Science). Ed. by D. Liu, S. Xie, Y. Li, D. Zhao, and E. M. El-Alfy. Vol. 10634. Springer, 100–108. doi:[10.1007/978-3-319-70087-8_11](https://doi.org/10.1007/978-3-319-70087-8_11).
- Y. Fukuchi, M. Osawa, H. Yamakawa, and M. Imai. 2017b. “Autonomous Self-Explanation of Behavior for Interactive Reinforcement Learning Agents.” In: *Proceedings of the 5th International Conference on Human Agent Interaction, HAI 2017, Bielefeld, Germany, October 17 - 20, 2017*. Ed. by B. Wrede, Y. Nagai, T. Komatsu, M. Hanheide, and L. Natale. ACM, 97–101. doi:[10.1145/3125739.3125746](https://doi.org/10.1145/3125739.3125746).
- D. Furelos-Blanco, M. Law, A. Jonsson, K. Broda, and A. Russo. 2021. “Induction and Exploitation of Subgoal Automata for Reinforcement Learning.” *J. Artif. Intell. Res.*, 70, 1031–1116. doi:[10.1613/JAIR.1.12372](https://doi.org/10.1613/JAIR.1.12372).
- J. Gajcin and I. Dusparic. 2024a. “ACTER: Diverse and Actionable Counterfactual Sequences for Explaining and Diagnosing RL Policies.” *CoRR*, abs/2402.06503. arXiv: [2402.06503](https://arxiv.org/abs/2402.06503). doi:[10.48550/ARXIV.2402.06503](https://doi.org/10.48550/ARXIV.2402.06503).
- J. Gajcin and I. Dusparic. 2024b. “RACCER: Towards Reachable and Certain Counterfactual Explanations for Reinforcement Learning.” In: *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024, Auckland, New Zealand, May 6-10, 2024*. Ed. by M. Dastani, J. S. Sichman, N. Alechina, and V. Dignum. ACM, 632–640. doi:[10.5555/3635637.3662915](https://doi.org/10.5555/3635637.3662915).
- J. Gajcin and I. Dusparic. 2022. “ReCCoVER: Detecting Causal Confusion for Explainable Reinforcement Learning.” In: *Explainable and Transparent AI and Multi-Agent Systems - 4th International Workshop, EXTRAAMAS 2022, Virtual Event, May 9-10, 2022, Revised Selected Papers* (Lecture Notes in Computer Science). Ed. by D. Calvaresi, A. Najjar, M. Winikoff, and K. Främling. Vol. 13283. Springer, 38–56. doi:[10.1007/978-3-031-15565-9_3](https://doi.org/10.1007/978-3-031-15565-9_3).
- J. Gajcin and I. Dusparic. 2024c. “Redefining Counterfactual Explanations for Reinforcement Learning: Overview, Challenges and Opportunities.” *ACM Comput. Surv.*, 56, 9, 219:1–219:33. doi:[10.1145/3648472](https://doi.org/10.1145/3648472).
- J. Gajcin, R. Nair, T. Pedapati, R. Marinescu, E. Daly, and I. Dusparic. 2021. “Contrastive Explanations for Comparing Preferences of Reinforcement Learning Agents.” *CoRR*, abs/2112.09462. <https://arxiv.org/abs/2112.09462> arXiv: [2112.09462](https://arxiv.org/abs/2112.09462).
- M. Gaon and R. I. Brafman. 2020. “Reinforcement Learning with Non-Markovian Rewards.” In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 3980–3987. doi:[10.1609/AAAI.V34I04.5814](https://doi.org/10.1609/AAAI.V34I04.5814).
- A. S. d’Avila Garcez, A. R. R. Dutra, and E. Alonso. 2018. “Towards Symbolic Reinforcement Learning with Common Sense.” *CoRR*, abs/1804.08597. <http://arxiv.org/abs/1804.08597> arXiv: [1804.08597](https://arxiv.org/abs/1804.08597).
- M. Garnelo, K. Arulkumaran, and M. Shanahan. 2016. “Towards Deep Symbolic Reinforcement Learning.” *CoRR*, abs/1609.05518. <http://arxiv.org/abs/1609.05518> arXiv: [1609.05518](https://arxiv.org/abs/1609.05518).
- C. Glanois, P. Weng, M. Zimmer, D. Li, T. Yang, J. Hao, and W. Liu. 2024. “A survey on interpretable reinforcement learning.” *Mach. Learn.*, 113, 8, 5847–5890. doi:[10.1007/S10994-024-06543-W](https://doi.org/10.1007/S10994-024-06543-W).
- M. Göbelbecker, T. Keller, P. Eyerich, M. Brenner, and B. Nebel. 2010. “Coming Up With Good Excuses: What to do When no Plan Can be Found.” In: *Proceedings of the 20th International Conference on Automated Planning and Scheduling, ICAPS 2010, Toronto, Ontario, Canada*,

- May 12-16, 2010. Ed. by R. I. Brafman, H. Geffner, J. Hoffmann, and H. A. Kautz. AAAI, 81–88. <http://www.aaai.org/ocs/index.php/ICAPS/ICAPS10/paper/view/1453>.
- V. Goel, J. Weng, and P. Poupart. 2018. “Unsupervised Video Object Segmentation for Deep Reinforcement Learning.” In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 5688–5699. <https://proceedings.neurips.cc/paper/2018/hash/96f2b50b5d3613adf9c27049b2a888c7-Abstract.html>.
- I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. 2014. “Generative Adversarial Networks.” CoRR, abs/1406.2661. <http://arxiv.org/abs/1406.2661> arXiv: 1406.2661.
- O. Gottesman, J. Futoma, Y. Liu, S. Parbhoo, L. A. Celi, E. Brunskill, and F. Doshi-Velez. 2020. “Interpretable Off-Policy Evaluation in Reinforcement Learning by Highlighting Influential Transitions.” In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research)*. Vol. 119. PMLR, 3658–3667. <http://proceedings.mlr.press/v119/gottesman20a.html>.
- O. Granmo. 2018. “The Tsetlin Machine - A Game Theoretic Bandit Driven Approach to Optimal Pattern Recognition with Propositional Logic.” CoRR, abs/1804.01508. <http://arxiv.org/abs/1804.01508> arXiv: 1804.01508.
- S. Greydanus, A. Koul, J. Dodge, and A. Fern. 2018. “Visualizing and Understanding AtariAgents.” In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research)*. Ed. by J. G. Dy and A. Krause. Vol. 80. PMLR, 1787–1796. <http://proceedings.mlr.press/v80/greydanus18a.html>.
- T. P. Gros, D. Groß, S. Gumhold, J. Hoffmann, M. Klauck, and M. Steinmetz. 2020. “TraceVis: Towards Visualization for Deep Statistical Model Checking.” In: *Leveraging Applications of Formal Methods, Verification and Validation: Tools and Trends - 9th International Symposium on Leveraging Applications of Formal Methods, ISOFA 2020, Rhodes, Greece, October 20-30, 2020, Proceedings, Part IV (Lecture Notes in Computer Science)*. Ed. by T. Margaria and B. Steffen. Vol. 12479. Springer, 27–46. doi:10.1007/978-3-030-83723-5_3.
- T. P. Gros, H. Hermanns, J. Hoffmann, M. Klauck, and M. Steinmetz. 2020. “Deep Statistical Model Checking.” In: *Formal Techniques for Distributed Objects, Components, and Systems - 40th IFIP WG 6.1 International Conference, FORTE 2020, Held as Part of the 15th International Federated Conference on Distributed Computing Techniques, DisCoTec 2020, Valletta, Malta, June 15-19, 2020, Proceedings (Lecture Notes in Computer Science)*. Ed. by A. Gotsman and A. Sokolova. Vol. 12136. Springer, 96–114. doi:10.1007/978-3-030-50086-3_6.
- Y. Gu, K. Zhang, Q. Liu, W. Gao, L. Li, and J. Zhou. 2024. “ π -Light: Programmatic Interpretable Reinforcement Learning for Resource-Limited Traffic Signal Control.” In: *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*. Ed. by M. J. Wooldridge, J. G. Dy, and S. Natarajan. AAAI Press, 21107–21115. doi:10.1609/AAAI.V38I19.30103.
- C. Guestrin, D. Koller, C. Gearhart, and N. Kanodia. 2003. “Generalizing Plans to New Environments in Relational MDPs.” In: *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Ed. by G. Gottlob and T. Walsh. Morgan Kaufmann, 1003–1010. <http://ijcai.org/Proceedings/03/Papers/144.pdf>.
- S. Guo, R. Zhang, B. Liu, Y. Zhu, D. H. Ballard, M. M. Hayhoe, and P. Stone. 2021. “Machine versus Human Attention in Deep Reinforcement Learning Tasks.” In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 25370–25385. <https://proceedings.neurips.cc/paper/2021/hash/d58e2f0776f0f4de9cd7963c857f2534-Abstract.html>.
- W. Guo and P. Wei. 2022. “Explainable deep reinforcement learning for aircraft separation assurance.” In: *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*. IEEE, 1–10. <http://dx.doi.org/10.1109/DASC55683.2022.9925786>.
- W. Guo, X. Wu, U. Khan, and X. Xing. 2021. “EDGE: Explaining Deep Reinforcement Learning Policies.” In: *NeurIPS*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 12222–12236. <https://proceedings.neurips.cc/paper/2021/hash/65c89f5a9501a04c073b354f03791b1f-Abstract.html>.
- U. D. Gupta, E. Talvitie, and M. Bowling. 2015. “Policy Tree: Adaptive Representation for Policy Gradient.” In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. Ed. by B. Bonet and S. Koenig. AAAI Press, 2547–2553. doi:10.1609/AAAI.V29I1.9613.
- D. C. Halbert. 1984. *Programming by example*. University of California, Berkeley.
- J. Y. Halpern and J. Pearl. 2001. “Causes and Explanations: A Structural-Model Approach - Part II: Explanations.” In: *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*. Ed. by B. Nebel. Morgan Kaufmann, 27–34. <http://dx.doi.org/10.1093/bjps/axi148>.
- M. Hasanbeig, N. Y. Jeppu, A. Abate, T. Melham, and D. Kroening. 2021. “DeepSynth: Automata Synthesis for Automatic Task Segmentation in Deep Reinforcement Learning.” In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 7647–7656. doi:10.1609/AAAI.V35I9.16935.
- A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi. 2021. “Escaping the Big Data Paradigm with Compact Transformers.” CoRR, abs/2104.05704. <https://arxiv.org/abs/2104.05704> arXiv: 2104.05704.

- B. Hayes and J. A. Shah. 2017. “Improving Robot Controller Transparency Through Autonomous Policy Explanation.” In: *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, HRI 2017, Vienna, Austria, March 6-9, 2017*. Ed. by B. Mutlu, M. Tscheligi, A. Weiss, and J. E. Young. ACM, 303–312. doi:[10.1145/2909824.3020233](https://doi.org/10.1145/2909824.3020233).
- R. Hazra and L. D. Raedt. 2023. “Deep Explainable Relational Reinforcement Learning: A Neuro-Symbolic Approach.” In: *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part IV* (Lecture Notes in Computer Science). Ed. by D. Koutra, C. Plant, M. G. Rodriguez, E. Baralis, and F. Bonchi. Vol. 14172. Springer, 213–229. doi:[10.1007/978-3-031-43421-1_13](https://doi.org/10.1007/978-3-031-43421-1_13).
- L. He, N. Aouf, and B. Song. 2021. “Explainable Deep Reinforcement Learning for UAV autonomous path planning.” *Aerospace science and technology*, 118, 107052. <http://dx.doi.org/10.1016/j.ast.2021.107052>.
- W. He, T. Lee, J. van Baar, K. Wittenburg, and H. Shen. 2020. “DynamicsExplorer: Visual Analytics for Robot Control Tasks involving Dynamics and LSTM-based Control Policies.” In: *2020 IEEE Pacific Visualization Symposium, PacificVis 2020, Tianjin, China, June 3-5, 2020*. IEEE, 36–45. doi:[10.1109/PACIFICVIS48177.2020.7127](https://doi.org/10.1109/PACIFICVIS48177.2020.7127).
- D. Hein, A. Hentschel, T. A. Runkler, and S. Udluft. 2017. “Particle swarm optimization for generating interpretable fuzzy reinforcement learning policies.” *Eng. Appl. Artif. Intell.*, 65, 87–98. doi:[10.1016/J.ENGAPPAL.2017.07.005](https://doi.org/10.1016/J.ENGAPPAL.2017.07.005).
- D. Hein, S. Udluft, and T. A. Runkler. 2019. “Generating interpretable reinforcement learning policies using genetic programming.” In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO 2019, Prague, Czech Republic, July 13-17, 2019*. Ed. by M. López-Ibáñez, A. Auger, and T. Stützle. ACM, 23–24. doi:[10.1145/3319619.3326755](https://doi.org/10.1145/3319619.3326755).
- T. Herlau and R. Larsen. 2022. “Reinforcement Learning of Causal Variables Using Mediation Analysis.” In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 6910–6917. doi:[10.1609/AAAI.V36I6.20648](https://doi.org/10.1609/AAAI.V36I6.20648).
- T. Heskes, E. Sijben, I. G. Bucur, and T. Claassen. 2020. “Causal Shapley Values: Exploiting Causal Knowledge to Explain Individual Predictions of Complex Models.” In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. <https://proceedings.neurips.cc/paper/2020/hash/32e54441e6382a7fbacbbaf3c450059-Abstract.html>.
- M. Hessel et al. 2018. “Rainbow: Combining Improvements in Deep Reinforcement Learning.” In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by S. A. McIlraith and K. Q. Weinberger. AAAI Press, 3215–3222. doi:[10.1609/AAAI.V32I1.11796](https://doi.org/10.1609/AAAI.V32I1.11796).
- A. Heuillet, F. Couthouis, and N. D. Rodríguez. 2022. “Collective eXplainable AI: Explaining Cooperative Strategies and Agent Contribution in Multiagent Reinforcement Learning With Shapley Values.” *IEEE Comput. Intell. Mag.*, 17, 1, 59–71. doi:[10.1109/MCI.2021.3129959](https://doi.org/10.1109/MCI.2021.3129959).
- A. Heuillet, F. Couthouis, and N. D. Rodríguez. 2021. “Explainability in deep reinforcement learning.” *Knowl. Based Syst.*, 214, 106685. doi:[10.1016/J.KNOSYS.2020.106685](https://doi.org/10.1016/J.KNOSYS.2020.106685).
- T. Hickling, A. Zenati, N. Aouf, and P. Spencer. 2024. “Explainability in Deep Reinforcement Learning: A Review into Current Methods and Applications.” *ACM Comput. Surv.*, 56, 5, 125:1–125:35. doi:[10.1145/3623377](https://doi.org/10.1145/3623377).
- J. Hilton, N. Cammarata, S. Carter, G. Goh, and C. Olah. 2020. “Understanding RL vision.” *Distill*, 5, 11, e29. <http://dx.doi.org/10.23915/distill.00029>.
- S. Hochreiter and J. Schmidhuber. 1997. “Long Short-Term Memory.” *Neural Comput.*, 9, 8, 1735–1780. doi:[10.1162/NECO.1997.9.8.1735](https://doi.org/10.1162/NECO.1997.9.8.1735).
- R. R. Hoffman, S. T. Mueller, G. Klein, and J. Litman. 2018. “Metrics for Explainable AI: Challenges and Prospects.” *CoRR*, abs/1812.04608. <http://arxiv.org/abs/1812.04608> arXiv: 1812.04608.
- J. Hoffmann and D. Magazzeni. 2019. “Explainable AI Planning (XAIP): Overview and the Case of Contrastive Explanation (Extended Abstract).” In: *Reasoning Web. Explainable Artificial Intelligence - 15th International Summer School 2019, Bolzano, Italy, September 20-24, 2019, Tutorial Lectures* (Lecture Notes in Computer Science). Ed. by M. Krötzsch and D. Stepanova. Vol. 11810. Springer, 277–282. doi:[10.1007/978-3-030-31423-1_9](https://doi.org/10.1007/978-3-030-31423-1_9).
- J. Huang, P. P. Angelov, and C. Yin. 2020. “Interpretable policies for reinforcement learning by empirical fuzzy sets.” *Eng. Appl. Artif. Intell.*, 91, 103559. doi:[10.1016/J.ENGAPPAL.2020.103559](https://doi.org/10.1016/J.ENGAPPAL.2020.103559).
- S. H. Huang, K. Bhatia, P. Abbeel, and A. D. Dragan. 2018. “Establishing Appropriate Trust via Critical States.” In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*. IEEE, 3929–3936. doi:[10.1109/IROS.2018.8593649](https://doi.org/10.1109/IROS.2018.8593649).
- S. H. Huang, D. Held, P. Abbeel, and A. D. Dragan. 2019. “Enabling robots to communicate their objectives.” *Auton. Robots*, 43, 2, 309–326. doi:[10.1007/S10514-018-9771-0](https://doi.org/10.1007/S10514-018-9771-0).
- T. Huber, M. Demmler, S. Mertes, M. L. Olson, and E. André. 2023. “GANterfactual-RL: Understanding Reinforcement Learning Agents’ Strategies through Visual Counterfactual Explanations.” In: *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023*. Ed. by N. Agmon, B. An, A. Ricci, and W. Yeoh. ACM, 1097–1106. doi:[10.5555/3545946.3598751](https://doi.org/10.5555/3545946.3598751).

- T. Huber, B. Limmer, and E. André. 2022. “Benchmarking Perturbation-Based Saliency Maps for Explaining Atari Agents.” *Frontiers Artif. Intell.*, 5. doi:[10.3389/FRAI.2022.903875](https://doi.org/10.3389/FRAI.2022.903875).
- T. Huber, D. Schiller, and E. André. 2019. “Enhancing Explainability of Deep Reinforcement Learning Through Selective Layer-Wise Relevance Propagation.” In: *KI 2019: Advances in Artificial Intelligence - 42nd German Conference on AI, Kassel, Germany, September 23-26, 2019, Proceedings* (Lecture Notes in Computer Science). Ed. by C. Benz Müller and H. Stuckenschmidt. Vol. 11793. Springer, 188–202. doi:[10.1007/978-3-030-30179-8_16](https://doi.org/10.1007/978-3-030-30179-8_16).
- T. Huber, K. Weitz, E. André, and O. Amir. 2021. “Local and global explanations of agent behavior: Integrating strategy summaries with saliency maps.” *Artif. Intell.*, 301, 103571. doi:[10.1016/J.ARTINT.2021.103571](https://doi.org/10.1016/J.ARTINT.2021.103571).
- A. Hüyük, D. Jarrett, and M. van der Schaar. 2023. “Explaining by Imitating: Understanding Decisions by Interpretable Policy Learning.” *CoRR*, abs/2310.19831. arXiv: [2310.19831](https://arxiv.org/abs/2310.19831). doi:[10.48550/ARXIV.2310.19831](https://doi.org/10.48550/ARXIV.2310.19831).
- R. T. Icarte, T. Q. Klassen, R. A. Valenzano, and S. A. McIlraith. 2018. “Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning.” In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10-15, 2018* (Proceedings of Machine Learning Research). Ed. by J. G. Dy and A. Krause. Vol. 80. PMLR, 2112–2121. <http://proceedings.mlr.press/v80/icarte18a.html>.
- R. T. Icarte, E. Waldie, T. Q. Klassen, R. A. Valenzano, M. P. Castro, and S. A. McIlraith. 2019. “Learning Reward Machines for Partially Observable Reinforcement Learning.” In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, 15497–15508. <https://proceedings.neurips.cc/paper/2019/hash/532435c44bec236b471a47a88d63513d-Abstract.html>.
- J. P. Inala, O. Bastani, Z. Tavares, and A. Solar-Lezama. 2020. “Synthesizing Programmatic Policies that Inductively Generalize.” In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=S1l8oANFDH>.
- H. Itaya, T. Hirakawa, T. Yamashita, H. Fujiyoshi, and K. Sugiura. 2021. “Visual Explanation using Attention Mechanism in Actor-Critic-based Deep Reinforcement Learning.” In: *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*. IEEE, 1–10. doi:[10.1109/IJCNN52387.2021.9534363](https://doi.org/10.1109/IJCNN52387.2021.9534363).
- A. Iucci, A. Hata, A. Terra, R. Inam, and I. Leite. 2021. “Explainable Reinforcement Learning for Human-Robot Collaboration.” In: *20th International Conference on Advanced Robotics, ICAR 2021, Ljubljana, Slovenia, December 6-10, 2021*. IEEE, 927–934. doi:[10.1109/ICAR53236.2021.9659472](https://doi.org/10.1109/ICAR53236.2021.9659472).
- R. Iyer, Y. Li, H. Li, M. Lewis, R. Sundar, and K. P. Sycara. 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*. Ed. by J. Furman, G. E. Marchant, H. Price, and F. Rossi. ACM, 144–150. doi:[10.1145/3278721.3278776](https://doi.org/10.1145/3278721.3278776).
- D. Jackson. 2012. “A New, Node-Focused Model for Genetic Programming.” In: *Genetic Programming - 15th European Conference, EuroGP 2012, Málaga, Spain, April 11-13, 2012. Proceedings* (Lecture Notes in Computer Science). Ed. by A. Moraglio, S. Silva, K. Krawiec, P. Machado, and C. Cotta. Vol. 7244. Springer, 49–60. doi:[10.1007/978-3-642-29139-5_5](https://doi.org/10.1007/978-3-642-29139-5_5).
- R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. 1991. “Adaptive Mixtures of Local Experts.” *Neural Comput.*, 3, 1, 79–87. doi:[10.1162/NECO.1991.3.1.79](https://doi.org/10.1162/NECO.1991.3.1.79).
- A. Jacq, J. Ferret, O. Pietquin, and M. Geist. 2022. “Lazy-MDPs: Towards Interpretable Reinforcement Learning by Learning When to Act.” *CoRR*, abs/2203.08542. arXiv: [2203.08542](https://arxiv.org/abs/2203.08542). doi:[10.48550/ARXIV.2203.08542](https://doi.org/10.48550/ARXIV.2203.08542).
- T. K. Jain, D. S. Kushwaha, and A. K. Misra. 2008. “Optimization of the Quine-McCluskey Method for the Minimization of the Boolean Expressions.” In: *Fourth International Conference on Autonomic and Autonomous Systems, ICAS 2008, 16-21 March 2008, Gosier, Guadeloupe*. IEEE Computer Society, 165–168. doi:[10.1109/ICAS.2008.11](https://doi.org/10.1109/ICAS.2008.11).
- T. Jaunet, R. Vuillemot, and C. Wolf. 2020. “DRLViz: Understanding Decisions and Memory in Deep Reinforcement Learning.” *Comput. Graph. Forum*, 39, 3, 49–61. doi:[10.1111/CGF.13962](https://doi.org/10.1111/CGF.13962).
- E. Jenner and A. Gleave. 2022. “Preprocessing Reward Functions for Interpretability.” *CoRR*, abs/2203.13553. arXiv: [2203.13553](https://arxiv.org/abs/2203.13553). doi:[10.48550/ARXIV.2203.13553](https://doi.org/10.48550/ARXIV.2203.13553).
- R. Jhala and R. Majumdar. 2009. “Software model checking.” *ACM Comput. Surv.*, 41, 4, 21:1–21:54. doi:[10.1145/1592434.1592438](https://doi.org/10.1145/1592434.1592438).
- A. Jhunjunwala, J. Lee, S. Sedwards, V. Abdelzad, and K. Czarnecki. 2020. “Improved Policy Extraction via Online Q-value Distillation.” In: *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 1–8. doi:[10.1109/IJCNN48605.2020.9207648](https://doi.org/10.1109/IJCNN48605.2020.9207648).
- Y. Jiang, S. Gu, K. Murphy, and C. Finn. 2019. “Language as an Abstraction for Hierarchical Deep Reinforcement Learning.” In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, 9414–9426. <https://proceedings.neurips.cc/paper/2019/hash/0af787945872196b42c9f73ead2565c8-Abstract.html>.
- Z. Jiang and S. Luo. 2019. “Neural Logic Reinforcement Learning.” In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (Proceedings of Machine Learning Research). Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. PMLR, 3110–3119. <http://proceedings.mlr.press/v97/jiang19a.html>.

- M. Jin, Z. Ma, K. Jin, H. H. Zhuo, C. Chen, and C. Yu. 2022. "Creativity of AI: Automatic Symbolic Option Discovery for Facilitating Deep Reinforcement Learning." In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 7042–7050. doi:[10.1609/AAAILV36I6.20663](https://doi.org/10.1609/AAAILV36I6.20663).
- H. Joo and K. Kim. 2019. "Visualization of Deep Reinforcement Learning using Grad-CAM: How AI Plays AtariGames?" In: *IEEE Conference on Games, CoG 2019, London, United Kingdom, August 20-23, 2019*. IEEE, 1–2. doi:[10.1109/CIG.2019.8847950](https://doi.org/10.1109/CIG.2019.8847950).
- S. Josef and A. Degani. 2020. "Deep Reinforcement Learning for Safe Local Planning of a Ground Vehicle in Unknown Rough Terrain." *IEEE Robotics Autom. Lett.*, 5, 4, 6748–6755. doi:[10.1109/LRA.2020.3011912](https://doi.org/10.1109/LRA.2020.3011912).
- Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, and F. Doshi-Velez. 2019. "Explainable reinforcement learning via reward decomposition." In: *IJCAI/ECAL Workshop on explainable artificial intelligence*.
- M. Kaiser, C. Otte, T. A. Runkler, and C. H. Ek. 2019. "interpretable dynamics models for data-efficient reinforcement learning." In: *27th European Symposium on Artificial Neural Networks, ESANN 2019, Bruges, Belgium, April 24-26, 2019*. <https://www.esann.org/sites/default/files/proceedings/legacy/es2019-98.pdf>.
- T. Kampik, J. C. Nieves, and H. Lindgren. 2019. "Explaining Sympathetic Actions of Rational Agents." In: *Explainable, Transparent Autonomous Agents and Multi-Agent Systems - First International Workshop, EXTRAAMAS 2019, Montreal, QC, Canada, May 13-14, 2019, Revised Selected Papers (Lecture Notes in Computer Science)*. Ed. by D. Calvaresi, A. Najjar, M. Schumacher, and K. Främling. Vol. 11763. Springer, 59–76. doi:[10.1007/978-3-030-30391-4_4](https://doi.org/10.1007/978-3-030-30391-4_4).
- K. Kansky et al. 2017. "Schema Networks: Zero-shot Transfer with a Generative Causal Model of Intuitive Physics." In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (Proceedings of Machine Learning Research). Ed. by D. Precup and Y. W. Teh. Vol. 70. PMLR, 1809–1818. <http://proceedings.mlr.press/v70/kansky17a.html>.
- S. Karakovskiy and J. Togelius. 2012. "The Mario AI Benchmark and Competitions." *IEEE Trans. Comput. Intell. AI Games*, 4, 1, 55–67. doi:[10.1109/TCIAIG.2012.2188528](https://doi.org/10.1109/TCIAIG.2012.2188528).
- A. Karimi, G. Barthe, B. Schölkopf, and I. Valera. 2020. "A survey of algorithmic recourse: definitions, formulations, solutions, and prospects." *CoRR*, abs/2010.04050. <https://arxiv.org/abs/2010.04050> arXiv: 2010.04050.
- A. Karimi, B. Schölkopf, and I. Valera. 2021. "Algorithmic Recourse: from Counterfactual Explanations to Interventions." In: *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*. Ed. by M. C. Elish, W. Isaac, and R. S. Zemel. ACM, 353–362. doi:[10.1145/3442188.3445899](https://doi.org/10.1145/3442188.3445899).
- D. Kasenberg and M. Scheutz. 2017. "Interpretable apprenticeship learning with temporal logic specifications." In: *56th IEEE Annual Conference on Decision and Control, CDC 2017, Melbourne, Australia, December 12-15, 2017*. IEEE, 4914–4921. doi:[10.1109/CDC.2017.8264386](https://doi.org/10.1109/CDC.2017.8264386).
- Y. Kazak, C. W. Barrett, G. Katz, and M. Schapira. 2019. "Verifying Deep-RL-Driven Systems." In: *Proceedings of the 2019 Workshop on Network Meets AI & ML, NetAI@SIGCOMM 2019, Beijing, China, August 23, 2019*. ACM, 83–89. doi:[10.1145/3341216.3342218](https://doi.org/10.1145/3341216.3342218).
- D. Kazhdan, Z. Shams, and P. Liò. 2020. "MARLeME: A Multi-Agent Reinforcement Learning Model Extraction Library." In: *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 1–8. doi:[10.1109/IJCNN48605.2020.9207564](https://doi.org/10.1109/IJCNN48605.2020.9207564).
- J. Kennedy and R. Eberhart. 1995. "Particle swarm optimization." In: *Proceedings of International Conference on Neural Networks (ICNN'95), Perth, WA, Australia, November 27 - December 1, 1995*. IEEE, 1942–1948. doi:[10.1109/ICNN.1995.488968](https://doi.org/10.1109/ICNN.1995.488968).
- O. Z. Khan, P. Poupart, and J. P. Black. 2009. "Minimal Sufficient Explanations for Factored Markov Decision Processes." In: *Proceedings of the 19th International Conference on Automated Planning and Scheduling, ICAPS 2009, Thessaloniki, Greece, September 19-23, 2009*. Ed. by A. Gerevini, A. E. Howe, A. Cesta, and I. Refanidis. AAAI <http://aaai.org/ocs/index.php/ICAPS/ICAPS09/paper/view/726>.
- J. Kim, C. Muise, A. Shah, S. Agarwal, and J. Shah. 2019. "Bayesian Inference of Linear Temporal Logic Specifications for Contrastive Explanations." In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by S. Kraus. ijcai.org, 5591–5598. doi:[10.24963/IJCAI.2019/776](https://doi.org/10.24963/IJCAI.2019/776).
- H. Kohler, R. Akrou, and P. Preux. 2023. "Optimal Interpretability-Performance Trade-off of Classification Trees with Black-Box Reinforcement Learning." *CoRR*, abs/2304.05839. arXiv: 2304.05839. doi:[10.48550/ARXIV.2304.05839](https://doi.org/10.48550/ARXIV.2304.05839).
- H. Kohler, Q. Delfosse, R. Akrou, K. Kersting, and P. Preux. 2024. "Interpretable and Editable Programmatic Tree Policies for Reinforcement Learning." *CoRR*, abs/2405.14956. arXiv: 2405.14956. doi:[10.48550/ARXIV.2405.14956](https://doi.org/10.48550/ARXIV.2405.14956).
- O. Kotevska, J. Munk, K. R. Kurte, Y. Du, K. Amasyali, R. W. Smith, and H. Zandi. 2020. "Methodology for Interpretable Reinforcement Learning Model for HVAC Energy Control." In: *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10-13, 2020*. Ed. by X. Wu et al. IEEE, 1555–1564. doi:[10.1109/BIGDATA50022.2020.9377735](https://doi.org/10.1109/BIGDATA50022.2020.9377735).
- A. Koul, A. Fern, and S. Greydanus. 2019. "Learning Finite State Representations of Recurrent Policy Networks." In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. <https://openreview.net/forum?id=S1gOpsCctm>.
- A. Krajna, M. Brcic, T. Lipic, and J. Doncevic. 2022. "Explainability in reinforcement learning: perspective and position." *CoRR*, abs/2203.11547. arXiv: 2203.11547. doi:[10.48550/ARXIV.2203.11547](https://doi.org/10.48550/ARXIV.2203.11547).

- J. Kubalík, E. Alibekov, and R. Babuška. 2017. “Optimal control via reinforcement learning with symbolic policy approximation.” *IFAC-PapersOnLine*, 50, 1, 4162–4167. <http://dx.doi.org/10.1016/j.ifacol.2017.08.805>.
- T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. 2016. “Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation.” In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, 3675–3683. <https://proceedings.neurips.cc/paper/2016/hash/f442d33fa06832082290ad8544a8da27-Abstract.html>.
- I. Lage, D. Lifschitz, F. Doshi-Velez, and O. Amir. 2019a. “Exploring Computational User Models for Agent Policy Summarization.” In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by S. Kraus. ijcai.org, 1401–1407. doi:[10.24963/IJCAI.2019/194](https://doi.org/10.24963/IJCAI.2019/194).
- I. Lage, D. Lifschitz, F. Doshi-Velez, and O. Amir. 2019b. “Toward Robust Policy Summarization.” In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*. Ed. by E. Elkind, M. Veloso, N. Agmon, and M. E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems, 2081–2083. <http://dl.acm.org/citation.cfm?id=3332017>.
- M. Landajuela, B. K. Petersen, S. Kim, C. P. Santiago, R. Glatt, T. N. Mundhenk, J. F. Pettit, and D. M. Faissol. 2021. “Discovering symbolic policies with deep reinforcement learning.” In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research)*. Ed. by M. Meila and T. Zhang. Vol. 139. PMLR, 5979–5989. <http://proceedings.mlr.press/v139/landajuela21a.html>.
- M. T. Lash. 2022. “HEX: Human-in-the-loop Explainability via Deep Reinforcement Learning.” *CoRR*, abs/2206.01343. arXiv: [2206.01343](https://arxiv.org/abs/2206.01343). doi:[10.48550/ARXIV.2206.01343](https://doi.org/10.48550/ARXIV.2206.01343).
- J. H. Lee. 2019. “Complementary reinforcement learning towards explainable agents.” *CoRR*, abs/1901.00188. <http://arxiv.org/abs/1901.00188> arXiv: [1901.00188](https://arxiv.org/abs/1901.00188).
- M. Leonetti, L. Iocchi, and P. Stone. 2016. “A synthesis of automated planning and reinforcement learning for efficient, robust decision-making.” *Artif. Intell.*, 241, 103–130. doi:[10.1016/J.ARTINT.2016.07.004](https://doi.org/10.1016/J.ARTINT.2016.07.004).
- T. Lesort, N. D. Rodríguez, J. Goudou, and D. Filliat. 2018. “State representation learning for control: An overview.” *Neural Networks*, 108, 379–392. doi:[10.1016/J.NEUNET.2018.07.006](https://doi.org/10.1016/J.NEUNET.2018.07.006).
- H. Li, J. Wu, H. Xu, G. Li, and M. Guizani. 2022. “Explainable Intelligence-Driven Defense Mechanism Against Advanced Persistent Threats: A Joint Edge Game and AI Approach.” *IEEE Trans. Dependable Secur. Comput.*, 19, 2, 757–775. doi:[10.1109/TDSC.2021.3130944](https://doi.org/10.1109/TDSC.2021.3130944).
- N. Li, S. Adepu, E. Kang, and D. Garlan. 2020. “Explanations for human-on-the-loop: a probabilistic model checking approach.” In: *SEAMS '20: IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, Seoul, Republic of Korea, 29 June - 3 July, 2020*. Ed. by S. Honiden, E. D. Nitto, and R. Calinescu. ACM, 181–187. doi:[10.1145/3387939.3391592](https://doi.org/10.1145/3387939.3391592).
- W. Li, X. Feng, H. An, X. Y. Ng, and Y. Zhang. 2020. “MRI Reconstruction with Interpretable Pixel-Wise Operations Using Reinforcement Learning.” In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 792–799. doi:[10.1609/AAAI.V34I01.5423](https://doi.org/10.1609/AAAI.V34I01.5423).
- X. Li, C. I. Vasile, and C. Belta. 2017. “Reinforcement learning with temporal logic rewards.” In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*. IEEE, 3834–3839. doi:[10.1109/IROS.2017.8206234](https://doi.org/10.1109/IROS.2017.8206234).
- Y. Li, K. P. Sycara, and R. Iyer. 2017. “Object-sensitive Deep Reinforcement Learning.” In: *GCAI 2017, 3rd Global Conference on Artificial Intelligence, Miami, FL, USA, 18-22 October 2017 (EPiC Series in Computing)*. Ed. by C. Benz Müller, C. L. Lisetti, and M. Theobald. Vol. 50. EasyChair, 20–35. doi:[10.29007/XTGM](https://doi.org/10.29007/XTGM).
- R. Liessner, J. Dohmen, and M. A. Wiering. 2021. “Explainable Reinforcement Learning for Longitudinal Control.” In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence, ICAART 2021, Volume 2, Online Streaming, February 4-6, 2021*. Ed. by A. P. Rocha, L. Steels, and H. J. van den Herik. SCITEPRESS, 874–881. doi:[10.5220/0010256208740881](https://doi.org/10.5220/0010256208740881).
- V. Lifschitz. 2008. “What Is Answer Set Programming?” In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. Ed. by D. Fox and C. P. Gomes. AAAI Press, 1594–1597. <http://www.aaai.org/Library/AAAI/2008/aaai08-270.php>.
- A. Likmeta, A. M. Metelli, A. Tirinzoni, R. Giol, M. Restelli, and D. Romano. 2020. “Combining reinforcement learning with rule-based controllers for transparent and general decision-making in autonomous driving.” *Robotics Auton. Syst.*, 131, 103568. doi:[10.1016/J.ROBOT.2020.103568](https://doi.org/10.1016/J.ROBOT.2020.103568).
- T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. 2016. “Continuous control with deep reinforcement learning.” In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. <http://arxiv.org/abs/1509.02971>.
- Y. Lin, A. S. Wang, E. Undersander, and A. Rai. 2022. “Efficient and Interpretable Robot Manipulation With Graph Neural Networks.” *IEEE Robotics Autom. Lett.*, 7, 2, 2740–2747. doi:[10.1109/LRA.2022.3143518](https://doi.org/10.1109/LRA.2022.3143518).
- Z. Lin, K. Lam, and A. Fern. 2021. “Contrastive Explanations for Reinforcement Learning via Embedded Self Predictions.” In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=Ud3DSz72nYR>.

- B. Liu, Y. Xia, and P. S. Yu. 2005. "Clustering via decision tree construction." *Foundations and advances in data mining*, 97–124. http://dx.doi.org/10.1007/11362197_5.
- G. Liu, O. Schulte, W. Zhu, and Q. Li. 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). Ed. by M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Iffrim. Vol. 11052. Springer, 414–429. doi:10.1007/978-3-030-10928-8_25.
- G. Liu, X. Sun, O. Schulte, and P. Poupart. 2021. "Learning Tree Interpretation from Object Representation for Deep Reinforcement Learning." In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 19622–19636. <https://proceedings.neurips.cc/paper/2021/hash/a35fe7f7fe8217b4369a0af4244d1fca-Abstract.html>.
- H. Liu, M. Zhuge, B. Li, Y. Wang, F. Faccio, B. Ghanem, and J. Schmidhuber. 2023. "Learning to Identify Critical States for Reinforcement Learning from Videos." In: *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 1955–1965. doi:10.1109/ICCV51070.2023.00187.
- Y. Liu, X. Wang, Y. Chang, and C. Jiang. 2022. "Towards Explainable Reinforcement Learning Using Scoring Mechanism Augmented Agents." In: *Knowledge Science, Engineering and Management - 15th International Conference, KSEM 2022, Singapore, August 6-8, 2022, Proceedings, Part II* (Lecture Notes in Computer Science). Ed. by G. Memmi, B. Yang, L. Kong, T. Zhang, and M. Qiu. Vol. 13369. Springer, 547–558. doi:10.1007/978-3-031-10986-7_44.
- M. Lomas, R. Chevalier, E. V. C. II, R. C. Garrett, J. Hoare, and M. Kopack. 2012. "Explaining robot actions." In: *International Conference on Human-Robot Interaction, HRI'12, Boston, MA, USA - March 05 - 08, 2012*. Ed. by H. A. Yanco, A. Steinfeld, V. Evers, and O. C. Jenkins. ACM, 187–188. doi:10.1145/2157689.2157748.
- J. Løyer, V. B. Gjørsum, and A. M. Lekkass. 2021. "Explainable AI methods on a deep reinforcement learning agent for automatic docking." *IFAC-PapersOnLine*, 54, 16, 146–152. <http://dx.doi.org/10.1016/j.ifacol.2021.10.086>.
- D. C. Luckham and B. Frasca. 1998. "Complex event processing in distributed systems." *Computer Systems Laboratory Technical Report CSL-TR-98-754*. Stanford University, Stanford, 28, 16.
- S. M. Lundberg, G. G. Erion, and S. Lee. 2018. "Consistent Individualized Feature Attribution for Tree Ensembles." *CoRR*, abs/1802.03888. <http://arxiv.org/abs/1802.03888> arXiv: 1802.03888.
- S. M. Lundberg and S. Lee. 2017. "A Unified Approach to Interpreting Model Predictions." In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, 4765–4774. <https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html>.
- R. Luss, A. Dhurandhar, and M. Liu. 2023. "Local Explanations for Reinforcement Learning." In: *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*. Ed. by B. Williams, Y. Chen, and J. Neville. AAAI Press, 9002–9010. doi:10.1609/AAAI.V37I7.26081.
- D. Lyu, F. Yang, B. Liu, and S. Gustafson. 2019. "SDRL: Interpretable and Data-Efficient Deep Reinforcement Learning Leveraging Symbolic Planning." In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2970–2977. doi:10.1609/AAAI.V33I01.33012970.
- Z. Ma, Y. Zhuang, P. Weng, H. H. Zhuo, D. Li, W. Liu, and J. Hao. 2021. "Learning Symbolic Rules for Interpretable Deep Reinforcement Learning." *CoRR*, abs/2103.08228. <https://arxiv.org/abs/2103.08228> arXiv: 2103.08228.
- P. Madumal, T. Miller, L. Sonenberg, and F. Vetere. 2020a. "Distal Explanations for Explainable Reinforcement Learning Agents." *CoRR*, abs/2001.10284. <https://arxiv.org/abs/2001.10284> arXiv: 2001.10284.
- P. Madumal, T. Miller, L. Sonenberg, and F. Vetere. 2020b. "Explainable Reinforcement Learning through a Causal Lens." In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2493–2500. doi:10.1609/AAAI.V34I03.5631.
- F. Maes, R. Fonteneau, L. Wehenkel, and D. Ernst. 2012. "Policy Search in a Space of Simple Closed-form Formulas: Towards Interpretability of Reinforcement Learning." In: *Discovery Science - 15th International Conference, DS 2012, Lyon, France, October 29-31, 2012. Proceedings* (Lecture Notes in Computer Science). Ed. by J. Ganschcia, P. Lenca, and J. Petit. Vol. 7569. Springer, 37–51. doi:10.1007/978-3-642-33492-4_6.
- O. Marom and B. Rosman. 2018. "Zero-Shot Transfer with Deictic Object-Oriented Representation in Reinforcement Learning." In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 2297–2305. <https://proceedings.neurips.cc/paper/2018/hash/df0aab058ce179e4f7ab135ed4e641a9-Abstract.html>.
- S. Marsland, J. Shapiro, and U. Nehmzow. 2002. "A self-organising network that grows when required." *Neural Networks*, 15, 8-9, 1041–1058. doi:10.1016/S0893-6080(02)00078-3.

- D. M. Martínez, G. Alenyà, and C. Torras. 2017. “Relational reinforcement learning with guided demonstrations.” *Artif. Intell.*, 247, 295–312. doi:[10.1016/J.ARTINT.2015.02.006](https://doi.org/10.1016/J.ARTINT.2015.02.006).
- J. McCalmon, T. Le, S. M. Alqahtani, and D. Lee. 2022. “CAPS: Comprehensible Abstract Policy Summaries for Explaining Reinforcement Learning Agents.” In: *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. Ed. by P. Faliszewski, V. Mascardi, C. Pelachaud, and M. E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 889–897. doi:[10.5555/3535850.3535950](https://doi.org/10.5555/3535850.3535950).
- S. McGregor, H. Buckingham, T. G. Dietterich, R. Houtman, C. A. Montgomery, and R. A. Metoyer. 2017. “Interactive visualization for testing Markov Decision Processes: MDPVIS.” *J. Vis. Lang. Comput.*, 39, 93–106. doi:[10.1016/J.JVLC.2016.10.007](https://doi.org/10.1016/J.JVLC.2016.10.007).
- J. H. Metzen. 2013. “Learning Graph-Based Representations for Continuous Reinforcement Learning Domains.” In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part I* (Lecture Notes in Computer Science). Ed. by H. Blockeel, K. Kersting, S. Nijssen, and F. Zelezny. Vol. 8188. Springer, 81–96. doi:[10.1007/978-3-642-40988-2_6](https://doi.org/10.1007/978-3-642-40988-2_6).
- S. Milani, N. Topin, M. Veloso, and F. Fang. 2024. “Explainable Reinforcement Learning: A Survey and Comparative Review.” *ACM Comput. Surv.*, 56, 7, 168:1–168:36. doi:[10.1145/3616864](https://doi.org/10.1145/3616864).
- S. Milani, Z. Zhang, N. Topin, Z. R. Shi, C. A. Kamhoua, E. E. Papalexakis, and F. Fang. 2022. “MAVIPER: Learning Decision Tree Policies for Interpretable Multi-agent Reinforcement Learning.” In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2022, Grenoble, France, September 19-23, 2022, Proceedings, Part IV* (Lecture Notes in Computer Science). Ed. by M. Amini, S. Canu, A. Fischer, T. Guns, P. K. Novak, and G. Tsoumakas. Vol. 13716. Springer, 251–266. doi:[10.1007/978-3-031-26412-2_16](https://doi.org/10.1007/978-3-031-26412-2_16).
- J. F. Miller and P. Thomson. 2000. “Cartesian Genetic Programming.” In: *Genetic Programming, European Conference, Edinburgh, Scotland, UK, April 15-16, 2000, Proceedings* (Lecture Notes in Computer Science). Ed. by R. Poli, W. Banzhaf, W. B. Langdon, J. F. Miller, P. Nordin, and T. C. Fogarty. Vol. 1802. Springer, 121–132. doi:[10.1007/978-3-540-46239-2_9](https://doi.org/10.1007/978-3-540-46239-2_9).
- A. Mishra, U. Soni, J. Huang, and C. Bryan. 2021. “Why? Why not? When? Visual Explanations of Agent Behavior in Reinforcement Learning.” *CoRR*, abs/2104.02818. <https://arxiv.org/abs/2104.02818> arXiv: 2104.02818.
- I. Mishra, G. Dao, and M. Lee. 2018. “Visual Sparse Bayesian Reinforcement Learning: A Framework for Interpreting What an Agent Has Learned.” In: *IEEE Symposium Series on Computational Intelligence, SSCI 2018, Bangalore, India, November 18-21, 2018*. IEEE, 1427–1434. doi:[10.1109/SSCI.2018.8628887](https://doi.org/10.1109/SSCI.2018.8628887).
- V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. 2016. “Asynchronous Methods for Deep Reinforcement Learning.” In: *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (JMLR Workshop and Conference Proceedings). Ed. by M. Balcan and K. Q. Weinberger. Vol. 48. JMLR.org, 1928–1937. <http://proceedings.mlr.press/v48/mniha16.html>.
- V. Mnih, K. Kavukcuoglu, et al.. 2015. “Human-level control through deep reinforcement learning.” *Nat.*, 518, 7540, 529–533. doi:[10.1038/NATURE14236](https://doi.org/10.1038/NATURE14236).
- E. F. Moore. 1956. “Gedanken-experiments on sequential machines.” *Automata studies*, 34, 129–153. <http://dx.doi.org/10.1515/9781400882618-006>.
- K. R. Mopuri, U. Garg, and R. V. Babu. 2019. “CNN Fixations: An Unraveling Approach to Visualize the Discriminative Image Regions.” *IEEE Trans. Image Process.*, 28, 5, 2116–2125. doi:[10.1109/TIP.2018.2881920](https://doi.org/10.1109/TIP.2018.2881920).
- A. Mott, D. Zoran, M. Chrzanowski, D. Wierstra, and D. J. Rezende. 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, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, 12329–12338. <https://proceedings.neurips.cc/paper/2019/hash/e9510081ac30ffa83f10b68cde1cac07-Abstract.html>.
- L. M. de Moura and N. S. Bjørner. 2008. “Z3: An Efficient SMT Solver.” In: *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings* (Lecture Notes in Computer Science). Ed. by C. R. Ramakrishnan and J. Rehof. Vol. 4963. Springer, 337–340. doi:[10.1007/978-3-540-78800-3_24](https://doi.org/10.1007/978-3-540-78800-3_24).
- S. K. Murthy, S. Kasif, and S. Salzberg. 1994. “A System for Induction of Oblique Decision Trees.” *J. Artif. Intell. Res.*, 2, 1–32. doi:[10.1613/JAIR.63](https://doi.org/10.1613/JAIR.63).
- R. B. Myerson. 1977. “Graphs and Cooperation in Games.” *Math. Oper. Res.*, 2, 3, 225–229. doi:[10.1287/MOOR.2.3.225](https://doi.org/10.1287/MOOR.2.3.225).
- S. Nagesh Rao, B. Costa, and D. P. Fílev. 2019. “Interpretable Approximation of a Deep Reinforcement Learning Agent as a Set of If-Then Rules.” In: *18th IEEE International Conference On Machine Learning And Applications, ICMLA 2019, Boca Raton, FL, USA, December 16-19, 2019*. Ed. by M. A. Wani, T. M. Khoshgoftaar, D. Wang, H. Wang, and N. Seliya. IEEE, 216–221. doi:[10.1109/ICMLA.2019.00041](https://doi.org/10.1109/ICMLA.2019.00041).
- M. A. Neerincx, J. van der Waa, F. Kaptein, and J. van Diggelen. 2018. “Using Perceptual and Cognitive Explanations for Enhanced Human-Agent Team Performance.” In: *Engineering Psychology and Cognitive Ergonomics - 15th International Conference, EPCE 2018, Held as Part of HCI International 2018, Las Vegas, NV, USA, July 15-20, 2018, Proceedings* (Lecture Notes in Computer Science). Ed. by D. Harris. Vol. 10906. Springer, 204–214. doi:[10.1007/978-3-319-91122-9_18](https://doi.org/10.1007/978-3-319-91122-9_18).

- A. Y. Ng and S. Russell. 2000. "Algorithms for Inverse Reinforcement Learning." In: *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, Stanford University, Stanford, CA, USA, June 29 - July 2, 2000. Ed. by P. Langley. Morgan Kaufmann, 663–670. http://www.eecs.harvard.edu/cs286r/courses/spring06/papers/ngRuss_irl00.pdf.
- K. S. Ng. 2004. "Alkemy: A learning system based on an expressive knowledge representation formalism." *submitted for publication*. <https://www.researchgate.net/profile/Kee-Siong-Ng/publication/228801223>.
- T. M. Nguyen, T. P. Quinn, T. Nguyen, and T. Tran. 2021. "Counterfactual Explanation with Multi-Agent Reinforcement Learning for Drug Target Prediction." *CoRR*, abs/2103.12983. <https://arxiv.org/abs/2103.12983> arXiv: 2103.12983.
- X. Nie, M. Hiraga, and K. Ohkura. 2019. "Visualizing Deep Q-learning to Understanding Behavior of Swarm Robotic System." In: *Proceedings of the 23rd Asia Pacific Symposium on Intelligent and Evolutionary Systems*, Tottori, Japan, December 6-8, 2019. Ed. by H. Sato, S. Iwanaga, and A. Ishii. Springer, 118–129. doi:10.1007/978-3-030-37442-6_11.
- D. Nikulin, A. Ianina, V. Aliev, and S. I. Nikolenko. 2019. "Free-Lunch Saliency via Attention in AtariAgents." In: *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*. IEEE, 4240–4249. doi:10.1109/ICCVW.2019.00522.
- M. Oberst and D. A. Sontag. 2019. "Counterfactual Off-Policy Evaluation with Gumbel-Max Structural Causal Models." In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (Proceedings of Machine Learning Research). Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. PMLR, 4881–4890. <http://proceedings.mlr.press/v97/oberst19a.html>.
- M. L. Olson, R. Khanna, L. Neal, F. Li, and W. Wong. 2021. "Counterfactual state explanations for reinforcement learning agents via generative deep learning." *Artif. Intell.*, 295, 103455. doi:10.1016/J.ARTINT.2021.103455.
- M. van Otterlo. 2012. "Solving Relational and First-Order Logical Markov Decision Processes: A Survey." In: *Reinforcement Learning. Adaptation, Learning, and Optimization*. Vol. 12. Ed. by M. A. Wiering and M. van Otterlo. Springer, 253–292. doi:10.1007/978-3-642-27645-3_8.
- L. Ou, Y. Chang, Y. Wang, and C. Lin. 2024. "Fuzzy Centered Explainable Network for Reinforcement Learning." *IEEE Trans. Fuzzy Syst.*, 32, 1, 203–213. doi:10.1109/TFUZZ.2023.3295055.
- R. R. Paleja, Y. Niu, A. Silva, C. Ritchie, S. Choi, and M. C. Gombolay. 2022. "Learning Interpretable, High-Performing Policies for Autonomous Driving." In: *Robotics: Science and Systems XVIII, New York City, NY, USA, June 27 - July 1, 2022*. Ed. by K. Hauser, D. A. Shell, and S. Huang. doi:10.15607/RSS.2022.XVIII.068.
- X. Pan, X. Chen, Q. Cai, J. F. Canny, and F. Yu. 2019. "Semantic Predictive Control for Explainable and Efficient Policy Learning." In: *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*. IEEE, 3203–3209. doi:10.1109/ICRA.2019.8794437.
- K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. "Bleu: a Method for Automatic Evaluation of Machine Translation." In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*. ACL, 311–318. doi:10.3115/1073083.1073135.
- S. Pateria, B. Subagdjia, A. Tan, and C. Quek. 2022. "Hierarchical Reinforcement Learning: A Comprehensive Survey." *ACM Comput. Surv.*, 54, 5, 109:1–109:35. doi:10.1145/3453160.
- A. Payani and F. Fekri. 2020. "Incorporating Relational Background Knowledge into Reinforcement Learning via Differentiable Inductive Logic Programming." *CoRR*, abs/2003.10386. <https://arxiv.org/abs/2003.10386> arXiv: 2003.10386.
- A. Payani and F. Fekri. 2019. "Inductive Logic Programming via Differentiable Deep Neural Logic Networks." *CoRR*, abs/1906.03523. <http://arxiv.org/abs/1906.03523> arXiv: 1906.03523.
- M. Persiani and T. Hellström. 2022. "The Mirror Agent Model: A Bayesian Architecture for Interpretable Agent Behavior." In: *Explainable and Transparent AI and Multi-Agent Systems - 4th International Workshop, EXTRAAMAS 2022, Virtual Event, May 9-10, 2022, Revised Selected Papers* (Lecture Notes in Computer Science). Ed. by D. Calvaresi, A. Najjar, M. Winikoff, and K. Främling. Vol. 13283. Springer, 111–123. doi:10.1007/978-3-031-15565-9_7.
- V. Petsiuk, A. Das, and K. Saenko. 2018. "RISE: Randomized Input Sampling for Explanation of Black-box Models." In: *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*. BMVA Press, 151. <http://bmvc2018.org/contents/papers/1064.pdf>.
- B. Pfeifer, M. Krzyzinski, H. Baniecki, A. Holzinger, and P. Biecek. 2025. "Explaining and visualizing black-box models through counterfactual paths." *Pattern Analysis and Applications*, 28, 3, 154.
- B. D. Pierson, D. Arendt, J. Miller, and M. E. Taylor. 2024. "Comparing explanations in RL." *Neural Comput. Appl.*, 36, 1, 505–516. doi:10.1007/S00521-023-08696-6.
- R. Pocius, L. Neal, and A. Fern. 2019. "Strategic Tasks for Explainable Reinforcement Learning." In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 10007–10008. doi:10.1609/AAAI.V33I01.330110007.
- E. Puiutta and E. M. S. P. Veith. 2020. "Explainable Reinforcement Learning: A Survey." In: *Machine Learning and Knowledge Extraction - 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25-28, 2020, Proceedings* (Lecture Notes in Computer Science). Ed. by A. Holzinger, P. Kieseberg, A. M. Tjoa, and E. R. Weippl. Vol. 12279. Springer, 77–95. doi:10.1007/978-3-030-57321-8_5.

- N. Puri, S. Verma, P. Gupta, D. Kayastha, S. V. Deshmukh, B. Krishnamurthy, and S. Singh. 2020. "Explain Your Move: Understanding Agent Actions Using Specific and Relevant Feature Attribution." In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=SJgzLkBKPB>.
- J. R. Quinlan. 1986. "Induction of Decision Trees." *Mach. Learn.*, 1, 1, 81–106. doi:10.1023/A:1022643204877.
- A. Raffin, A. Hill, R. Traoré, T. Lesort, N. D. Rodríguez, and D. Filliat. 2018. "S-RL Toolbox: Environments, Datasets and Evaluation Metrics for State Representation Learning." *CoRR*, abs/1809.09369. <http://arxiv.org/abs/1809.09369> arXiv: 1809.09369.
- S. B. Remman and A. M. Lekkas. 2021. "Robotic Lever Manipulation using Hindsight Experience Replay and Shapley Additive Explanations." In: *2021 European Control Conference, ECC 2021, Virtual Event / Delft, The Netherlands, June 29 - July 2, 2021*. IEEE, 586–593. doi:10.23919/ECC54610.2021.9654850.
- S. B. Remman, I. Strümke, and A. M. Lekkas. 2022. "Causal versus Marginal Shapley Values for Robotic Lever Manipulation Controlled using Deep Reinforcement Learning." In: *American Control Conference, ACC 2022, Atlanta, GA, USA, June 8-10, 2022*. IEEE, 2683–2690. doi:10.23919/ACC53348.2022.9867807.
- M. T. Ribeiro, S. Singh, and C. Guestrin. 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, San Francisco, CA, USA, August 13-17, 2016*. Ed. by B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi. ACM, 1135–1144. doi:10.1145/2939672.2939778.
- F. Rietz, S. Magg, F. Heintz, T. Stoyanov, S. Wermter, and J. A. Stork. 2023. "Hierarchical goals contextualize local reward decomposition explanations." *Neural Comput. Appl.*, 35, 23, 16693–16704. doi:10.1007/S00521-022-07280-8.
- S. G. Rizzo, G. Vantini, and S. Chawla. 2019. "Reinforcement Learning with Explainability for Traffic Signal Control." In: *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019, Auckland, New Zealand, October 27-30, 2019*. IEEE, 3567–3572. doi:10.1109/ITSC.2019.8917519.
- O. Ronneberger, P. Fischer, and T. Brox. 2015. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III* (Lecture Notes in Computer Science). Ed. by N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi. Vol. 9351. Springer, 234–241. doi:10.1007/978-3-319-24574-4_28.
- A. Rosenfeld. 2021. "Better Metrics for Evaluating Explainable Artificial Intelligence." In: *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*. Ed. by F. Dignum, A. Lomuscio, U. Endriss, and A. Nowé. ACM, 45–50. doi:10.5555/3463952.3463962.
- C. van Rossum, C. Feinberg, A. A. Shumays, K. Baxter, and B. Bartha. 2021. "A Novel Approach to Curiosity and Explainable Reinforcement Learning via Interpretable Sub-Goals." *CoRR*, abs/2104.06630. <https://arxiv.org/abs/2104.06630> arXiv: 2104.06630.
- A. M. Roth, N. Topin, P. Jamshidi, and M. Veloso. 2019. "Conservative Q-Improvement: Reinforcement Learning for an Interpretable Decision-Tree Policy." *CoRR*, abs/1907.01180. <http://arxiv.org/abs/1907.01180> arXiv: 1907.01180.
- C. Rouveiro, M. K. Aoual, H. Soldano, and V. Ventos. 2023. "Explaining Optimal Trajectories." In: *Rules and Reasoning - 7th International Joint Conference, RuleML+RR 2023, Oslo, Norway, September 18-20, 2023, Proceedings* (Lecture Notes in Computer Science). Ed. by A. Fensel, A. Ozaki, D. Roman, and A. Soylu. Vol. 14244. Springer, 206–221. doi:10.1007/978-3-031-45072-3_15.
- D. Rozjanc, N. Lukač, S. Schweng, C. Gollob, A. Nothdurft, K. Stampfer, J. Del Ser, and A. Holzinger. 2025. "Explaining 3D Semantic Segmentation Through Generative AI-Based Counterfactuals." *Expert Systems*, 42, 12, e70163.
- R. Y. Rubinstein. 1997. "Optimization of computer simulation models with rare events." *European Journal of Operational Research*, 99, 1, 89–112. [http://dx.doi.org/10.1016/S0377-2217\(96\)00385-2](http://dx.doi.org/10.1016/S0377-2217(96)00385-2).
- C. Rupprecht, C. Ibrahim, and C. J. Pal. 2020. "Finding and Visualizing Weaknesses of Deep Reinforcement Learning Agents." In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=rylvYaNYDH>.
- C. Ryan, J. J. Collins, and M. O'Neill. 1998. "Grammatical Evolution: Evolving Programs for an Arbitrary Language." In: *Genetic Programming, First European Workshop, EuroGP'98, Paris, France, April 14-15, 1998, Proceedings* (Lecture Notes in Computer Science). Ed. by W. Banzhaf, R. Poli, M. Schoenauer, and T. C. Fogarty. Vol. 1391. Springer, 83–96. doi:10.1007/BFB0055930.
- E. Saldiran, M. Hasanazade, G. Inalhan, and A. Tsourdos. 2024. "Towards Global Explainability of Artificial Intelligence Agent Tactics in Close Air Combat." *Aerospace*, 11, 6, 415. <http://dx.doi.org/10.3390/aerospace11060415>.
- A. Samadi, K. Koufos, K. Debattista, and M. Dianati. 2024. "SAFE-RL: Saliency-Aware Counterfactual Explainer for Deep Reinforcement Learning Policies." *CoRR*, abs/2404.18326. arXiv: 2404.18326. doi:10.48550/ARXIV.2404.18326.
- R. Samoilescu, A. V. Looveren, and J. Klaise. 2021. "Model-agnostic and Scalable Counterfactual Explanations via Reinforcement Learning." *CoRR*, abs/2106.02597. <https://arxiv.org/abs/2106.02597> arXiv: 2106.02597.
- L. Saulières, M. C. Cooper, and F. D. de Saint-Cyr. 2024. "Backward Explanations via Redefinition of Predicates." In: *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)* (Frontiers in Artificial Intelligence and Applications). Ed. by U. Endriss, F. S. Melo, K. Bach, A. J. B. Diz, J. M. Alonso-Moral, S. Barro, and F. Heintz. Vol. 392. IOS Press, 786–793. doi:10.3233/FAIA240563.
- L. Saulières, M. C. Cooper, and F. D. de Saint-Cyr. 2023a. "Predicate-Based Explanation of a Reinforcement Learning Agent via Action Importance Evaluation." In: *Machine Learning and Principles and Practice of Knowledge Discovery in Databases - International Workshops*

- of ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Revised Selected Papers, Part I (Communications in Computer and Information Science). Ed. by R. Meo and F. Silvestri. Vol. 2133. Springer, 21–37. doi:[10.1007/978-3-031-74630-7_2](https://doi.org/10.1007/978-3-031-74630-7_2).
- L. Saulières, M. C. Cooper, and F. D. de Saint-Cyr. 2023b. “Predictive Explanations for and by Reinforcement Learning.” In: *Agents and Artificial Intelligence - 15th International Conference, ICAART 2023, Lisbon, Portugal, February 22-24, 2023, Revised Selected Papers* (Lecture Notes in Computer Science). Ed. by A. P. Rocha, L. Steels, and H. J. van den Herik. Vol. 14546. Springer, 115–140. doi:[10.1007/978-3-031-55326-4_6](https://doi.org/10.1007/978-3-031-55326-4_6).
- J. Scholz, M. Levihn, C. L. I. Jr., and D. Wingate. 2014. “A Physics-Based Model Prior for Object-Oriented MDPs.” In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014* (JMLR Workshop and Conference Proceedings). Vol. 32. JMLR.org, 1089–1097. <http://proceedings.mlr.press/v32/scholz14.html>.
- L. Schreiber, G. d. O. Ramos, and A. L. Bazzan. 2021. “Towards explainable deep reinforcement learning for traffic signal control.” In: *Proc. of LatinX in AI Workshop @ ICML 2021, LXAI. LXIA*. <http://dx.doi.org/10.52591/lxai2021072414>.
- J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. 2015. “Trust Region Policy Optimization.” In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015* (JMLR Workshop and Conference Proceedings). Ed. by F. R. Bach and D. M. Blei. Vol. 37. JMLR.org, 1889–1897. <http://proceedings.mlr.press/v37/schulman15.html>.
- B. Seegebarth, F. Müller, B. Schattenberg, and S. Biundo. 2012. “Making Hybrid Plans More Clear to Human Users - A Formal Approach for Generating Sound Explanations.” In: *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS 2012, Atibaia, São Paulo, Brazil, June 25-19, 2012*. Ed. by L. McCluskey, B. C. Williams, J. R. Silva, and B. Bonet. AAAI. <http://www.aaai.org/ocs/index.php/ICAPS/ICAPS12/paper/view/4691>.
- F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber. 2008. “Policy Gradients with Parameter-Based Exploration for Control.” In: *Artificial Neural Networks - ICANN 2008, 18th International Conference, Prague, Czech Republic, September 3-6, 2008, Proceedings, Part I* (Lecture Notes in Computer Science). Ed. by V. Kurková, R. Neruda, and J. Koutník. Vol. 5163. Springer, 387–396. doi:[10.1007/978-3-540-87536-9_40](https://doi.org/10.1007/978-3-540-87536-9_40).
- R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. 2017. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization.” In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 618–626. doi:[10.1109/ICCV.2017.74](https://doi.org/10.1109/ICCV.2017.74).
- Y. Septon, T. Huber, E. André, and O. Amir. 2023. “Integrating Policy Summaries with Reward Decomposition for Explaining Reinforcement Learning Agents.” In: *Advances in Practical Applications of Agents, Multi-Agent Systems, and Cognitive Mimetics. The PAAMS Collection - 21st International Conference, PAAMS 2023, Guimarães, Portugal, July 12-14, 2023, Proceedings* (Lecture Notes in Computer Science). Ed. by P. Mathieu, F. Dignum, P. Novais, and F. de la Prieta. Vol. 13955. Springer, 320–332. doi:[10.1007/978-3-031-37616-0_27](https://doi.org/10.1007/978-3-031-37616-0_27).
- P. Sequeira and M. T. Gervasio. 2020. “Interestingness elements for explainable reinforcement learning: Understanding agents’ capabilities and limitations.” *Artif. Intell.*, 288, 103367. doi:[10.1016/J.ARTINT.2020.103367](https://doi.org/10.1016/J.ARTINT.2020.103367).
- P. Sequeira and M. T. Gervasio. 2023. “IxDR: A Novel Explainable Deep Reinforcement Learning Toolkit Based on Analyses of Interestingness.” In: *Explainable Artificial Intelligence - First World Conference, xAI 2023, Lisbon, Portugal, July 26-28, 2023, Proceedings, Part I* (Communications in Computer and Information Science). Ed. by L. Longo. Vol. 1901. Springer, 373–396. doi:[10.1007/978-3-031-44064-9_20](https://doi.org/10.1007/978-3-031-44064-9_20).
- L. S. Shapley et al. 1953. “A value for n-person games.” *Contributions to the Theory of Games, Volume II*. <http://dx.doi.org/10.1515/9781400881970-018>.
- W. Shi, G. Huang, S. Song, Z. Wang, T. Lin, and C. Wu. 2022. “Self-Supervised Discovering of Interpretable Features for Reinforcement Learning.” *IEEE Trans. Pattern Anal. Mach. Intell.*, 44, 5, 2712–2724. doi:[10.1109/TPAMI.2020.3037898](https://doi.org/10.1109/TPAMI.2020.3037898).
- A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje. 2016. “Not Just a Black Box: Learning Important Features Through Propagating Activation Differences.” *CoRR*, abs/1605.01713. <http://arxiv.org/abs/1605.01713> arXiv: 1605.01713.
- T. Shu, C. Xiong, and R. Socher. 2018. “Hierarchical and Interpretable Skill Acquisition in Multi-task Reinforcement Learning.” In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=SJJQVZW0b>.
- A. Sieusahai and M. Guzdial. 2021. “Explaining Deep Reinforcement Learning Agents in the Atari Domain through a Surrogate Model.” In: *Proceedings of the Seventeenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2021, virtual, October 11-15, 2021*. Ed. by D. Thue and S. G. Ware. AAAI Press, 82–90. <https://ojs.aaai.org/index.php/AIIDE/article/view/18894>.
- A. Silva, T. Killian, I. D. J. Rodriguez, S.-H. Son, and M. Gombolay. 2019. *Optimization methods for interpretable differentiable decision trees in reinforcement learning*. (2019). <https://doi.org/10.48550/arXiv.1903.09338>.
- J. Skrzynski, F. Becker, and F. Lieder. 2021. “Automatic discovery of interpretable planning strategies.” *Mach. Learn.*, 110, 9, 2641–2683. doi:[10.1007/S10994-021-05963-2](https://doi.org/10.1007/S10994-021-05963-2).
- D. Smilkov, N. Thorat, B. Kim, F. B. Viégas, and M. Wattenberg. 2017. “SmoothGrad: removing noise by adding noise.” *CoRR*, abs/1706.03825. <http://arxiv.org/abs/1706.03825> arXiv: 1706.03825.
- J. Snoek, H. Larochelle, and R. P. Adams. 2012. “Practical Bayesian Optimization of Machine Learning Algorithms.” In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Ed. by P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, 2960–2968. <https://proceedings.neurips.cc/paper/2012/hash/05311655a15b75fab86956663e1819cd-Abstract.html>.

- E. A. Soares, P. P. Angelov, B. Costa, M. Castro, S. Nagesh Rao, and D. P. Filev. 2021. "Explaining Deep Learning Models Through Rule-Based Approximation and Visualization." *IEEE Trans. Fuzzy Syst.*, 29, 8, 2399–2407. doi:[10.1109/TFUZZ.2020.2999776](https://doi.org/10.1109/TFUZZ.2020.2999776).
- K. Sokol and P. A. Flach. 2020. "Explainability fact sheets: a framework for systematic assessment of explainable approaches." In: *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*. Ed. by M. Hildebrandt, C. Castillo, L. E. Celis, S. Ruggieri, L. Taylor, and G. Zanfir-Fortuna. ACM, 56–67. doi:[10.1145/3351095.3372870](https://doi.org/10.1145/3351095.3372870).
- Z. Song, Y. Jiang, J. Zhang, P. Weng, D. Li, W. Liu, and J. Hao. 2022. "An interpretable deep reinforcement learning approach to autonomous driving." In: *IJCAI Workshop on Artificial Intelligence for Autonomous Driving*. https://learn-to-race.org/workshop-ai4ad-ijcai2022/assets/papers/paper_14.pdf.
- U. Soni. 2024. *Towards More Accessible Human-AI Interactions in Sequential Decision-making Tasks*. Tech. rep. Arizona State University. <https://keep.lib.asu.edu/items/193894>.
- S. Sreedharan, T. Chakraborti, and S. Kambhampati. 2018. "Handling Model Uncertainty and Multiplicity in Explanations via Model Reconciliation." In: *Proceedings of the Twenty-Eighth International Conference on Automated Planning and Scheduling, ICAPS 2018, Delft, The Netherlands, June 24-29, 2018*. Ed. by M. de Weerd, S. Koenig, G. Röger, and M. T. J. Spaan. AAAI Press, 518–526. <https://aaai.org/ocs/index.php/ICAPS/ICAPS18/paper/view/17783>.
- S. Sreedharan, U. Soni, M. Verma, S. Srivastava, and S. Kambhampati. 2022. "Bridging the Gap: Providing Post-Hoc Symbolic Explanations for Sequential Decision-Making Problems with Inscrutable Representations." In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. <https://openreview.net/forum?id=o-1v9hdSult>.
- S. Sreedharan, S. Srivastava, and S. Kambhampati. 2020. "TLdR: Policy Summarization for Factored SSP Problems Using Temporal Abstractions." In: *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020*. Ed. by J. C. Beck, O. Buffet, J. Hoffmann, E. Karpas, and S. Sohrabi. AAAI Press, 272–280. <https://ojs.aaai.org/index.php/ICAPS/article/view/6671>.
- S. Sreedharan, S. Srivastava, D. E. Smith, and S. Kambhampati. 2019. "Why Can't You Do That HAL? Explaining Unsolvability of Planning Tasks." In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by S. Kraus. ijcai.org, 1422–1430. doi:[10.24963/IJCAI.2019/197](https://doi.org/10.24963/IJCAI.2019/197).
- M. Sridharan, M. Gelfond, S. Zhang, and J. L. Wyatt. 2019. "REBA: A Refinement-Based Architecture for Knowledge Representation and Reasoning in Robotics." *J. Artif. Intell. Res.*, 65, 87–180. doi:[10.1613/JAIR.1.11524](https://doi.org/10.1613/JAIR.1.11524).
- S. Srinivasan and F. Doshi-Velez. 2020. "Interpretable batch IRL to extract clinician goals in ICU hypotension management." *AMIA Summits on Translational Science Proceedings*, 2020, 636. <https://pubmed.ncbi.nlm.nih.gov/32477686/>.
- M. Stefik et al. 2021. "Explaining autonomous drones: An XAI journey." *Applied AI Letters*, 2, 4, e54. <http://dx.doi.org/10.1002/ail.2.54>.
- G. J. Stein. 2021. "Generating High-Quality Explanations for Navigation in Partially-Revealed Environments." In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 17493–17506. <https://proceedings.neurips.cc/paper/2021/hash/926ec030f29f83ce5318754fdb631a33-Abstract.html>.
- M. Steinmetz et al. 2022. "Debugging a Policy: Automatic Action-Policy Testing in AI Planning." In: *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. Ed. by A. Kumar, S. Thiébaux, P. Varakantham, and W. Yeoh. AAAI Press, 353–361. <https://ojs.aaai.org/index.php/ICAPS/article/view/19820>.
- R. Sukkerd, R. G. Simmons, and D. Garlan. 2018. "Towards explainable multi-objective probabilistic planning." In: *Proceedings of the 4th International Workshop on Software Engineering for Smart Cyber-Physical Systems, ICSE 2018, Gothenburg, Sweden, May 27, 2018*. Ed. by T. Bures, J. S. Fitzgerald, B. R. Schmerl, and D. Weyns. ACM, 19–25. doi:[10.1145/3196478.3196488](https://doi.org/10.1145/3196478.3196488).
- R. Sukkerd, R. G. Simmons, and D. Garlan. 2020. "Tradeoff-Focused Contrastive Explanation for MDP Planning." In: *29th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2020, Naples, Italy, August 31 - September 4, 2020*. IEEE, 1041–1048. doi:[10.1109/RO-MAN47096.2020.9223614](https://doi.org/10.1109/RO-MAN47096.2020.9223614).
- M. Sundararajan, A. Taly, and Q. Yan. 2017. "Axiomatic Attribution for Deep Networks." In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (Proceedings of Machine Learning Research). Ed. by D. Precup and Y. W. Teh. Vol. 70. PMLR, 3319–3328. <http://proceedings.mlr.press/v70/sundararajan17a.html>.
- R. S. Sutton, J. Modayil, M. Delp, T. Degris, P. M. Pilarski, A. White, and D. Precup. 2011. "Horde: a scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction." In: *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*. Ed. by L. Sonenberg, P. Stone, K. Tumer, and P. Yolum. IFAAMAS, 761–768. <http://portal.acm.org/citation.cfm?id=2031726%5C&CFID=54178199%5C&CFTOKEN=61392764>.
- R. S. Sutton, D. Precup, and S. Singh. 1999. "Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning." *Artif. Intell.*, 112, 1-2, 181–211. doi:[10.1016/S0004-3702\(99\)00052-1](https://doi.org/10.1016/S0004-3702(99)00052-1).
- A. Tabrez and B. Hayes. 2019. "Improving Human-Robot Interaction Through Explainable Reinforcement Learning." In: *14th ACM/IEEE International Conference on Human-Robot Interaction, HRI 2019, Daegu, South Korea, March 11-14, 2019*. IEEE, 751–753. doi:[10.1109/HRI.2019.8673198](https://doi.org/10.1109/HRI.2019.8673198).
- H. Tang, H. Liu, D. Xu, P. H. S. Torr, and N. Sebe. 2023. "AttentionGAN: Unpaired Image-to-Image Translation Using Attention-Guided Generative Adversarial Networks." *IEEE Trans. Neural Networks Learn. Syst.*, 34, 4, 1972–1987. doi:[10.1109/TNNLS.2021.3105725](https://doi.org/10.1109/TNNLS.2021.3105725).

- Y. Tang, D. Nguyen, and D. Ha. 2020. "Neuroevolution of self-interpretable agents." In: *GECCO '20: Genetic and Evolutionary Computation Conference, Cancún Mexico, July 8-12, 2020*. Ed. by C. A. C. Coello. ACM, 414–424. doi:[10.1145/3377930.3389847](https://doi.org/10.1145/3377930.3389847).
- G. N. Tasse, S. D. James, and B. Rosman. 2020. "A Boolean Task Algebra for Reinforcement Learning." In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. <https://proceedings.neurips.cc/paper/2020/hash/6ba3af5d7b2790e73f0de32e5c8c1798-Abstract.html>.
- P. Theumer, F. Edenhofner, R. Zimmermann, and A. Zipfel. 2022. "Explainable deep reinforcement learning for production control." In: *Proceedings of the Conference on Production Systems and Logistics: CPSL 2022*. Hannover: publish-Ing., 809–818. <https://doi.org/10.15488/12158>.
- A. L. Thomaz, G. Hoffman, and C. Breazeal. 2005. "Real-time interactive reinforcement learning for robots." In: *AAAI 2005 workshop on human comprehensible machine learning*. Vol. 3, 1. <https://cdn.aaai.org/Workshops/2005/WS-05-04/WS05-04-002.pdf>.
- E. Todorov. 2009. "Compositionality of optimal control laws." In: *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*. Ed. by Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta. Curran Associates, Inc., 1856–1864. <https://proceedings.nips.cc/paper/2009/hash/3eb71f6293a2a31f3569e10af6552658-Abstract.html>.
- E. Todorov, T. Erez, and Y. Tassa. 2012. "MuJoCo: A physics engine for model-based control." In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*. IEEE, 5026–5033. doi:[10.1109/IROS.2012.6386109](https://doi.org/10.1109/IROS.2012.6386109).
- N. Topin, S. Milani, F. Fang, and M. Veloso. 2021. "Iterative Bounding MDPs: Learning Interpretable Policies via Non-Interpretable Methods." In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 9923–9931. doi:[10.1609/AAAI.V35I11.17192](https://doi.org/10.1609/AAAI.V35I11.17192).
- N. Topin and M. Veloso. 2019. "Generation of Policy-Level Explanations for Reinforcement Learning." In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2514–2521. doi:[10.1609/AAAI.V33I01.33012514](https://doi.org/10.1609/AAAI.V33I01.33012514).
- M. Towers, Y. Du, C. T. Freeman, and T. J. Norman. 2024. "Explaining an Agent's Future Beliefs Through Temporally Decomposing Future Reward Estimators." In: *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)* (Frontiers in Artificial Intelligence and Applications). Ed. by U. Endriss, F. S. Melo, K. Bach, A. J. B. Diz, J. M. Alonso-Moral, S. Barro, and F. Heintz. Vol. 392. IOS Press, 2790–2797. doi:[10.3233/FAIA240814](https://doi.org/10.3233/FAIA240814).
- D. Trivedi, J. Zhang, S. Sun, and J. J. Lim. 2021. "Learning to Synthesize Programs as Interpretable and Generalizable Policies." In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 25146–25163. <https://proceedings.neurips.cc/paper/2021/hash/d37124c4c79f357cb02c655671a432fa-Abstract.html>.
- S. Tsirtsis, A. De, and M. Rodriguez. 2021. "Counterfactual Explanations in Sequential Decision Making Under Uncertainty." In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, 30127–30139. <https://proceedings.neurips.cc/paper/2021/hash/fd0a5a5e367a0955d81278062ef37429-Abstract.html>.
- S. Tsirtsis and M. Rodriguez. 2023. "Finding Counterfactually Optimal Action Sequences in Continuous State Spaces." In: *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. http://papers.nips.cc/paper%5C_files/paper/2023/hash/09ae6beae5f1ff38f05c05979097ea0f-Abstract-Conference.html.
- Y. Tsuchiya, Y. Mori, and M. Egi. 2023. "Explainable Reinforcement Learning Based on Q-value Decomposition by Expected State Transitions." In: *Proceedings of the AAAI 2023 Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering (AAAI-MAKE 2023)*, Hyatt Regency, San Francisco Airport, California, USA, March 27-29, 2023 (CEUR Workshop Proceedings). Ed. by A. Martin, H. Fill, A. Gerber, K. Hinkelmann, D. Lenat, R. Stolle, and F. van Harmelen. Vol. 3433. CEUR-WS.org. <https://ceur-ws.org/Vol-3433/paper5.pdf>.
- J. M. P. Ullauri, A. García-Domínguez, N. Bencomo, C. Zheng, C. Zhen, J. Boubeta-Puig, G. Ortiz, and S. Yang. 2022. "Event-driven temporal models for explanations - ETemoX: explaining reinforcement learning." *Softw. Syst. Model.*, 21, 3, 1091–1113. doi:[10.1007/S10270-021-00952-4](https://doi.org/10.1007/S10270-021-00952-4).
- P. E. Utgoff, N. C. Berkman, and J. A. Clouse. 1997. "Decision Tree Induction Based on Efficient Tree Restructuring." *Mach. Learn.*, 29, 1, 5–44. doi:[10.1023/A:1007413323501](https://doi.org/10.1023/A:1007413323501).
- W. T. B. Uther and M. M. Veloso. 1998. "Tree Based Discretization for Continuous State Space Reinforcement Learning." In: *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA*. Ed. by J. Mostow and C. Rich. AAAI Press / The MIT Press, 769–774. <http://www.aaai.org/Library/AAAI/1998/aaai98-109.php>.

- K. Valmeekam, S. Sreedharan, S. Sengupta, and S. Kambhampati. 2021. "RADAR-X: An Interactive Interface Pairing Contrastive Explanations with Revised Plan Suggestions." In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 16051–16053. doi:10.1609/AAAI.V35I18.18009.
- L. Van der Maaten and G. Hinton. 2008. "Visualizing data using t-SNE." *Journal of machine learning research*, 9, 11. <https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf?fbclid=...>
- V. R. Varma. 2021. *Interpretable Reinforcement Learning with the Regression Tsetlin Machine*. (2021). <https://fse.studenttheses.ub.rug.nl/id/eprint/26483>.
- M. Vasic, A. Petrovic, K. Wang, M. Nikolic, R. Singh, and S. Khurshid. 2022. "MoET: Mixture of Expert Trees and its application to verifiable reinforcement learning." *Neural Networks*, 151, 34–47. doi:10.1016/J.NEUNET.2022.03.022.
- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. "Attention is All you Need." In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, 5998–6008. <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>.
- R. Veerapaneni, J. D. Co-Reyes, M. Chang, M. Janner, C. Finn, J. Wu, J. B. Tenenbaum, and S. Levine. 2019. "Entity Abstraction in Visual Model-Based Reinforcement Learning." In: *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings* (Proceedings of Machine Learning Research). Ed. by L. P. Kaelbling, D. Kragic, and K. Sugiura. Vol. 100. PMLR, 1439–1456. <http://proceedings.mlr.press/v100/veerapaneni20a.html>.
- A. Verma, H. M. Le, Y. Yue, and S. Chaudhuri. 2019. "Imitation-Projected Programmatic Reinforcement Learning." In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, 15726–15737. <https://proceedings.neurips.cc/paper/2019/hash/5a44a53b7d26bb1e54c0522f186dcfb-Abstract.html>.
- A. Verma, V. Murali, R. Singh, P. Kohli, and S. Chaudhuri. 2018. "Programmatically Interpretable Reinforcement Learning." In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10-15, 2018* (Proceedings of Machine Learning Research). Ed. by J. G. Dy and A. Krause. Vol. 80. PMLR, 5052–5061. <http://proceedings.mlr.press/v80/verma18a.html>.
- O. Vinyals et al. 2017. "StarCraft II: A New Challenge for Reinforcement Learning." *CoRR*, abs/1708.04782. <http://arxiv.org/abs/1708.04782> arXiv: 1708.04782.
- M. Vinzent, M. Steinmetz, and J. Hoffmann. 2022. "Neural Network Action Policy Verification via Predicate Abstraction." In: *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*. Ed. by A. Kumar, S. Thiébaux, P. Varakantham, and W. Yeoh. AAAI Press, 371–379. <https://ojs.aaai.org/index.php/ICAPS/article/view/19822>.
- S. Volodin. 2021. "CauseOccam: Learning interpretable abstract representations in reinforcement learning environments via model sparsity." Master's thesis. École Polytechnique Fédérale de Lausanne. <https://infoscience.epfl.ch/handle/20.500.14299/180168>.
- G. A. Vouras. 2023. "Explicable Deep Reinforcement Learning: State of the Art and Challenges." *ACM Comput. Surv.*, 55, 5, 92:1–92:39. doi:10.1145/3527448.
- J. van der Waa, J. van Diggelen, K. van den Bosch, and M. A. Neerincx. 2018. "Contrastive Explanations for Reinforcement Learning in terms of Expected Consequences." *CoRR*, abs/1807.08706. <http://arxiv.org/abs/1807.08706> arXiv: 1807.08706.
- S. Wäldchen, S. Pokutta, and F. Huber. 2022. "Training Characteristic Functions with Reinforcement Learning: XAI-methods play Connect Four." In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA* (Proceedings of Machine Learning Research). Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato. Vol. 162. PMLR, 22457–22474. <https://proceedings.mlr.press/v162/waldchen22a.html>.
- T. Walker, L. Torrey, J. W. Shavlik, and R. Maclin. 2007. "Building Relational World Models for Reinforcement Learning." In: *Inductive Logic Programming, 17th International Conference, ILP 2007, Corvallis, OR, USA, June 19-21, 2007, Revised Selected Papers* (Lecture Notes in Computer Science). Ed. by H. Blockeel, J. Ramon, J. W. Shavlik, and P. Tadepalli. Vol. 4894. Springer, 280–291. doi:10.1007/978-3-540-78469-2_27.
- T. J. Walsh. 2010. *Efficient learning of relational models for sequential decision making*. Rutgers The State University of New Jersey, School of Graduate Studies.
- J. Wang, L. Gou, H. Shen, and H. Yang. 2019. "DQNViz: A Visual Analytics Approach to Understand Deep Q-networks." *IEEE Trans. Vis. Comput. Graph.*, 25, 1, 288–298. doi:10.1109/TVCG.2018.2864504.
- N. Wang, D. V. Pynadath, and S. G. Hill. 2016. "The Impact of POMDP-Generated Explanations on Trust and Performance in Human-Robot Teams." In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*. Ed. by C. M. Jonker, S. Marsella, J. Thangarajah, and K. Tuyls. ACM, 997–1005. <http://dl.acm.org/citation.cfm?id=2937071>.
- T. Wang, R. Liao, J. Ba, and S. Fidler. 2018. "NerveNet: Learning Structured Policy with Graph Neural Networks." In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=S1sqHMZCb>.

- X. Wang, S. Yuan, H. Zhang, M. Lewis, and K. P. Sycara. 2019. "Verbal Explanations for Deep Reinforcement Learning Neural Networks with Attention on Extracted Features." In: *28th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2019, New Delhi, India, October 14-18, 2019*. IEEE, 1–7. doi:[10.1109/RO-MAN46459.2019.8956301](https://doi.org/10.1109/RO-MAN46459.2019.8956301).
- X. Wang, Y. Chen, J. Yang, L. Wu, Z. Wu, and X. Xie. 2018. "A Reinforcement Learning Framework for Explainable Recommendation." In: *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*. IEEE Computer Society, 587–596. doi:[10.1109/ICDM.2018.00074](https://doi.org/10.1109/ICDM.2018.00074).
- Y. Wang, M. Mase, and M. Egi. 2020. "Attribution-based Saliency Method towards Interpretable Reinforcement Learning." In: *Proceedings of the AAAI 2020 Spring Symposium on Combining Machine Learning and Knowledge Engineering in Practice, AAAI-MAKE 2020, Palo Alto, CA, USA, March 23-25, 2020, Volume I* (CEUR Workshop Proceedings). Ed. by A. Martin, K. Hinkelmann, H. Fill, A. Gerber, D. Lenat, R. Stolle, and F. van Harmelen. Vol. 2600. CEUR-WS.org. <https://ceur-ws.org/Vol-2600/short4.pdf>.
- Z. Wang and S. Wang. 2022. "XRouting: Explainable Vehicle Rerouting for Urban Road Congestion Avoidance using Deep Reinforcement Learning." In: *IEEE International Smart Cities Conference, ISC2 2022, Pafos, Cyprus, September 26-29, 2022*. IEEE, 1–7. doi:[10.1109/ISC255366.2022.9922404](https://doi.org/10.1109/ISC255366.2022.9922404).
- L. Weitkamp, E. van der Pol, and Z. Akata. 2019. "Visual Rationalizations in Deep Reinforcement Learning for AtariGames." CoRR, abs/1902.00566. <https://arxiv.org/abs/1902.00566> arXiv: 1902.00566.
- L. Wells and T. Bednarz. 2021. "Explainable AI and Reinforcement Learning - A Systematic Review of Current Approaches and Trends." *Frontiers Artif. Intell.*, 4, 550030. doi:[10.3389/FRAI.2021.550030](https://doi.org/10.3389/FRAI.2021.550030).
- D. G. Wilson, S. Cussat-Blanc, H. Luga, and J. F. Miller. 2018. "Evolving simple programs for playing Atari games." In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018, Kyoto, Japan, July 15-19, 2018*. Ed. by H. E. Aguirre and K. Takadama. ACM, 229–236. doi:[10.1145/3205455.3205578](https://doi.org/10.1145/3205455.3205578).
- S. Wollenstein-Betech, C. Muise, C. G. Cassandra, I. C. Paschalidis, and Y. Khazaeni. 2020. "Explainability of Intelligent Transportation Systems using Knowledge Compilation: a Traffic Light Controller Case." In: *23rd IEEE International Conference on Intelligent Transportation Systems, ITSC 2020, Rhodes, Greece, September 20-23, 2020*. IEEE, 1–6. doi:[10.1109/ITSC45102.2020.9294213](https://doi.org/10.1109/ITSC45102.2020.9294213).
- B. Wu, J. K. Gupta, and M. J. Kochenderfer. 2019. "Model Primitive Hierarchical Lifelong Reinforcement Learning." In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*. Ed. by E. Elkind, M. Veloso, N. Agmon, and M. E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems, 34–42. <http://dl.acm.org/citation.cfm?id=3331671>.
- Y. Xiong et al.. 2024. "XRL-Bench: A Benchmark for Evaluating and Comparing Explainable Reinforcement Learning Techniques." CoRR, abs/2402.12685. arXiv: [2402.12685](https://arxiv.org/abs/2402.12685). doi:[10.48550/ARXIV.2402.12685](https://doi.org/10.48550/ARXIV.2402.12685).
- D. Xu and F. Fekri. 2021. "Interpretable Model-based Hierarchical Reinforcement Learning using Inductive Logic Programming." CoRR, abs/2106.11417. <https://arxiv.org/abs/2106.11417> arXiv: 2106.11417.
- Z. Xu, I. Gavran, Y. Ahmad, R. Majumdar, D. Neider, U. Topcu, and B. Wu. 2020. "Joint Inference of Reward Machines and Policies for Reinforcement Learning." In: *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020*. Ed. by J. C. Beck, O. Buffet, J. Hoffmann, E. Karpas, and S. Sohrabi. AAAI Press, 590–598. <https://ojs.aaai.org/index.php/ICAPS/article/view/6756>.
- D. Yan. 2023. "Research on Reinforcement Learning Explainable Strategies Based on Advantage Saliency." *Frontiers in Computing and Intelligent Systems*, 3, 1, 124–129. <http://dx.doi.org/10.54097/fcis.v3i1.6348>.
- B. Yang, S. Shen, and W. Gao. 2019. "Weighted Oblique Decision Trees." In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 5621–5627. doi:[10.1609/AAAI.V33I01.33015621](https://doi.org/10.1609/AAAI.V33I01.33015621).
- F. Yang, M. Du, and X. Hu. 2019. "Evaluating Explanation Without Ground Truth in Interpretable Machine Learning." CoRR, abs/1907.06831. <https://arxiv.org/abs/1907.06831> arXiv: 1907.06831.
- F. Yang, D. Lyu, B. Liu, and S. Gustafson. 2018. "PEORL: Integrating Symbolic Planning and Hierarchical Reinforcement Learning for Robust Decision-Making." In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. Ed. by J. Lang. ijcai.org, 4860–4866. doi:[10.24963/IJCAI.2018/675](https://doi.org/10.24963/IJCAI.2018/675).
- Q. Yang, H. Wang, M. Tong, W. Shi, G. Huang, and S. Song. 2024. "Leveraging Reward Consistency for Interpretable Feature Discovery in Reinforcement Learning." *IEEE Trans. Syst. Man Cybern. Syst.*, 54, 2, 1014–1025. doi:[10.1109/TSMC.2023.3312411](https://doi.org/10.1109/TSMC.2023.3312411).
- Y. Yang, I. G. Morillo, and T. M. Hospedales. 2018. "Deep Neural Decision Trees." CoRR, abs/1806.06988. <https://arxiv.org/abs/1806.06988> arXiv: 1806.06988.
- Z. Yang, S. Bai, L. Zhang, and P. H. S. Torr. 2018. "Learn to Interpret Atari Agents." CoRR, abs/1812.11276. <https://arxiv.org/abs/1812.11276> arXiv: 1812.11276.
- H. Yau, C. Russell, and S. Hadfield. 2020. "What Did You Think Would Happen? Explaining Agent Behaviour through Intended Outcomes." In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*,

- December 6-12, 2020, virtual. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. <https://proceedings.neurips.cc/paper/2020/hash/d5ab8dc7ef67ca92e41d730982c5c602-Abstract.html>.
- Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. 2019. “GNNExplainer: Generating Explanations for Graph Neural Networks.” In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, 9240–9251. <https://proceedings.neurips.cc/paper/2019/hash/d80b7040b773199015de6d3b4293c8ff-Abstract.html>.
- S. W. Yoon, A. Fern, and R. Givan. 2007. “FF-Replan: A Baseline for Probabilistic Planning.” In: *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling, ICAPS 2007, Providence, Rhode Island, USA, September 22-26, 2007*. Ed. by M. S. Boddy, M. Fox, and S. Thiébaux. AAAI, 352. <https://engineering.purdue.edu/~givan/papers/icaps07-ffreplan.pdf>.
- H. L. Younes. 2005. *Exploding Blocksworld*. (2005).
- Z. Yu, J. Ruan, and D. Xing. 2023. “Explainable Reinforcement Learning via a Causal World Model.” In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*. ijcai.org, 4540–4548. doi:10.24963/IJCAI.2023/505.
- T. Zahavy, N. Ben-Zrihem, and S. Mannor. 2016. “Graying the black box: Understanding DQNs.” In: *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (JMLR Workshop and Conference Proceedings). Ed. by M. Balcan and K. Q. Weinberger. Vol. 48. JMLR.org, 1899–1908. <http://proceedings.mlr.press/v48/zahavy16.html>.
- V. F. Zambaldi et al.. 2018. “Relational Deep Reinforcement Learning.” *CoRR*, abs/1806.01830. <http://arxiv.org/abs/1806.01830> arXiv: 1806.01830.
- A. E. Zelvelde, M. Westberg, and K. Främling. 2021. “Assessing Explainability in Reinforcement Learning.” In: *Explainable and Transparent AI and Multi-Agent Systems - Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3-7, 2021, Revised Selected Papers* (Lecture Notes in Computer Science). Ed. by D. Calvaresi, A. Najjar, M. Winikoff, and K. Främling. Vol. 12688. Springer, 223–240. doi:10.1007/978-3-030-82017-6_14.
- Y. Zeng, R. Cai, F. Sun, L. Huang, and Z. Hao. 2023. “A Survey on Causal Reinforcement Learning.” *CoRR*, abs/2302.05209. arXiv: 2302.05209. doi:10.48550/ARXIV.2302.05209.
- A. Zhang, S. Sukhbaatar, A. Lerer, A. Szlam, and R. Fergus. 2018. “Composable Planning with Attributes.” In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018* (Proceedings of Machine Learning Research). Ed. by J. G. Dy and A. Krause. Vol. 80. PMLR, 5837–5846. <http://proceedings.mlr.press/v80/zhang18k.html>.
- H. Zhang, A. Zhou, and X. Lin. 2020. “Interpretable policy derivation for reinforcement learning based on evolutionary feature synthesis.” *Complex & Intelligent Systems*, 6, 741–753. <https://doi.org/10.1007/s40747-020-00175-y>.
- K. Zhang, J. J. Zhang, P. Xu, T. Gao, and D. W. Gao. 2022. “Explainable AI in Deep Reinforcement Learning Models for Power System Emergency Control.” *IEEE Trans. Comput. Soc. Syst.*, 9, 2, 419–427. doi:10.1109/TCSS.2021.3096824.
- L. Zhang, X. Li, M. Wang, and A. Tian. 2021. “Off-Policy Differentiable Logic Reinforcement Learning.” In: *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2021, Bilbao, Spain, September 13-17, 2021, Proceedings, Part II* (Lecture Notes in Computer Science). Ed. by N. Oliver, F. Pérez-Cruz, S. Kramer, J. Read, and J. A. Lozano. Vol. 12976. Springer, 617–632. doi:10.1007/978-3-030-86520-7_38.
- Q. Zhang, X. Ma, Y. Yang, C. Li, J. Yang, Y. Liu, and B. Liang. 2021. “Learning to Discover Task-Relevant Features for Interpretable Reinforcement Learning.” *IEEE Robotics Autom. Lett.*, 6, 4, 6601–6607. doi:10.1109/LRA.2021.3091885.
- R. Zhang, Z. Liu, L. Zhang, J. A. Whritner, K. S. Muller, M. M. Hayhoe, and D. H. Ballard. 2018. “AGIL: Learning Attention from Human for Visuomotor Tasks.” In: *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI* (Lecture Notes in Computer Science). Ed. by V. Ferrari, M. Hertz, C. Sminchisescu, and Y. Weiss. Vol. 11215. Springer, 692–707. doi:10.1007/978-3-030-01252-6_41.
- R. Zhang, C. Walshe, Z. Liu, L. Guan, K. S. Muller, J. A. Whritner, L. Zhang, M. M. Hayhoe, and D. H. Ballard. 2020. “Atari-HEAD: Atari Human Eye-Tracking and Demonstration Dataset.” In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 6811–6820. doi:10.1609/AAAI.V34I04.6161.
- B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba. 2016. “Learning Deep Features for Discriminative Localization.” In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2921–2929. doi:10.1109/CVPR.2016.319.
- J. Zhou, A. H. Gandomi, F. Chen, and A. Holzinger. 2021. “Evaluating the quality of machine learning explanations: A survey on methods and metrics.” *Electronics*, 10, 5, 593. <http://dx.doi.org/10.3390/electronics10050593>.
- G. Zhu, Z. Huang, and C. Zhang. 2018. “Object-Oriented Dynamics Predictor.” In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, 9826–9837. <https://proceedings.neurips.cc/paper/2018/hash/713fd63d76c8a57b16fc433fb4ae718a-Abstract.html>.
- G. Zhu, J. Wang, Z. Ren, Z. Lin, and C. Zhang. 2020. “Object-Oriented Dynamics Learning through Multi-Level Abstraction.” In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference,*

- IAAI 2020, *The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence*, EAAI 2020, New York, NY, USA, February 7-12, 2020. AAAI Press, 6989–6998. doi:[10.1609/AAAI.V34I04.6183](https://doi.org/10.1609/AAAI.V34I04.6183).
- H. Zhu, Z. Xiong, S. Magill, and S. Jagannathan. 2019. “An inductive synthesis framework for verifiable reinforcement learning.” In: *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, Phoenix, AZ, USA, June 22-26, 2019*. Ed. by K. S. McKinley and K. Fisher. ACM, 686–701. doi:[10.1145/3314221.3314638](https://doi.org/10.1145/3314221.3314638).
- A. Zien, N. Krämer, S. Sonnenburg, and G. Rätsch. 2009. “The Feature Importance Ranking Measure.” In: *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part II* (Lecture Notes in Computer Science). Ed. by W. L. Buntine, M. Grobelnik, D. Mladenic, and J. Shawe-Taylor. Vol. 5782. Springer, 694–709. doi:[10.1007/978-3-642-04174-7_45](https://doi.org/10.1007/978-3-642-04174-7_45).
- M. Zimmer, X. Feng, C. Glanois, Z. Jiang, J. Zhang, P. Weng, J. Hao, D. Li, and W. Liu. 2021. “Differentiable Logic Machines.” *CoRR*, abs/2102.11529. <https://arxiv.org/abs/2102.11529> arXiv: 2102.11529.

A Summaries of Policy-level Methods

This appendix provides a summary of each policy-level method mentioned in the paper. These methods explain the agent’s policy. As a reminder, a total of four ways of explaining the policy have been identified. Constructing *Interpretable Policies* makes the agent’s reasoning transparent. *Summarising Policies* provides an overview of the agent’s capabilities. The use of *Human-readable MDP* allows the user to understand the information available to the agent, its actions and the dynamics of the environment. *Visual analysis* helps the user to study the agent’s policy visually, using various tools and metrics.

A.1 Interpretable Policy

A.1.1 Inherently Understandable.

Hierarchical Policy. As a reminder, Hierarchical RL (HRL) is a sub-domain of RL consisting in learning a policy which is structured in a hierarchical manner, generally comprising 2 levels, so as to break down the task to be performed into a set of sub-tasks. For each sub-task, a *low-level* policy is learned. A *high-level* policy is learned using the various *low-level* policies as an action.

The low-level policies are represented using a human language description in the following works. For Minecraft games, agents learn to determine when it is necessary to learn a new policy (Shu et al. 2018). If not, the agent chooses one of the policies it has already learned. These policies are interpretable due to their description, such as ‘*Get object o*’. To learn the dependencies between different tasks/policies (for example, it is necessary to use ‘*Get object o*’ before ‘*Put object o*’), a stochastic temporal grammar model is learned. In the same vein, Jiang et al. (2019) present Hierarchical Abstraction with Language using high-level actions, sub-goals, represented by a human language description. The high-level policy and the low-level policy are trained separately, where the low-level policy is learned based on a reward function that depends on an (assumed known) function determining whether a sub-goal is respected or not. The results show that this framework produces interpretable policies that perform well on long-horizon problems with sparse reward.

Model-based RL methods use the dynamics of the environment (i.e. the transition function and the reward function) to obtain an agent’s policy. Note that the dynamics are not necessarily known beforehand. In a model-based HRL context, Xu and Fekri (2021) propose to learn a transition function for high-level symbolic states (i.e. sub-goals) by modifying DILP (Evans and Grefenstette 2018) so as not to generate meaningless clauses. The transitions generated are logic rules expressing the preconditions and effects of the various sub-goals. In the same context, the Model Primitives HRL framework (B. Wu et al. 2019) allows a set of modular interpretable policies to be learned and combined based on a set of imperfect models of the environment. The idea is that approximate models of the environment are specialised in different regions of the environment. In addition to the policies, a gating controller is learned to compose the policies and solve the basic task.

Three works propose to mix Planning and HRL. The Symbolic DRL framework (Lyu et al. 2019) combines a high-level planner that returns a plan composed of symbolic actions with sub-policies learned by DRL for each

symbolic action. The framework is made up of three elements: a planner that builds a plan of symbolic actions, a controller that learns a sub-policy and a meta-controller that obtains the quality of the proposed plan and thus helps the planner to propose a better plan. An intrinsic and environment reward are respectively used to learn the sub-goals, i.e. symbolic actions, and to measure the plan quality. The combination of Planning and HRL is also proposed in Chang et al. (2020) for a human-UAVs cooperation problem. The plan represented by a tree and the sequential order of the actions to be performed is displayed to the user in such a way as to make the plan interpretable. Jin et al. (2022) use the option framework (Sutton, Precup, et al. 1999). An option consists of pre-conditions, a policy and effects. Option policies are learned in RL and a planner is used to generate a plan. To use this approach, the user requires access to a function defined upstream which extracts a symbolic state from an image state.

For a robotic manipulation task, the Dot-to-Dot method (Beyret et al. 2019) uses HRL to decompose the problem into simpler tasks, assimilated to sub-goals, which the low-level agent learns to perform. These sub-goals are iteratively generated by the high-level agent to guide the low-level agent. The low-level agent's input is its observation and the sub-goal to be achieved. This approach makes it possible to interpret the impact of the sub-goals on the agent's learned Q-values, and to observe that the high-level agent has learned a notion of distance for the given problem.

In Rossum et al. (2021), a reward function is based on the achievement of a sub-goal for learning. In addition, a curiosity reward is generated using a Generative Adversarial Network (GAN) (Goodfellow et al. 2014) that generates the state resulting from an action a from a state s . The sub-goal is generated by a neural network that selects a feature of the agent's current state as the sub-goal to be achieved. This method is specific to a problem where the state represents a grid in which the agent can move. The sub-goal is transcribed manually using a natural language description. In this work, a single policy is learned and is not technically an HRL method, although the task is decomposed into sub-objectives.

Rietz et al. (2023) combine HRL with reward decomposition to provide explanations of the agent's decisions. A context is associated with the reward decomposition, providing the user with the sub-goal that the agent is currently aiming to achieve. This method explains the policy with the different sub-goals but also the agent's local decision by providing a reward decomposition.

Using policy sketches, the agent learns to solve multi-task RL problems (Andreas et al. 2017). These policy sketches are sequences of symbolic labels. Each label is a comprehensible task that is learned by a sub-policy in the form of a neural network.

Composition (Todorov 2009), a slightly different approach to HRL, has also been used in Tasse et al. (2020). It consists of learning a new skill based on old skills that have already been learned. The interpretable aspect presented in this paper is that the new skill is composed without new learning, simply by using a boolean algebra on the tasks (modelled by MDP's) and on the value functions already learned.

In this paragraph, we give a brief presentation of some HRL methods or methods close to the domain which are not focused on explainability, but which are nonetheless worth mentioning. Two works (Leonetti et al. 2016; Sridharan et al. 2019) use Answer Set Programming (Lifschitz 2008) to constrain the agent's policy (Leonetti et al.) or define a high-level plan made up of abstract actions (Sridharan et al.). Furelos-Blanco et al. (2021) mix Inductive Logic Programming (ILP) and RL by learning an automaton composed of edges which represent a sub-goal by a logic formula. The framework proposed in F. Yang, Lyu, et al. (2018) combines planning and RL to provide robust symbolic plans. In A. Zhang et al. (2018), the agent learns a graph, the environment model based on a high-level state representation and uses it to learn a policy. In the same vein, Eysenbach et al. (2019) build a graph from the sub-goals obtained using the replay buffer and then use Dijkstra's algorithm (Dijkstra 1959) to plan. HRL is combined with the RRL in Driessens and Blockeel (2001) to learn an efficient policy where each sub-policy is learned by a Q-tree. The hierarchical DQN (Kulkarni et al. 2016) represents a set of Q-functions in a

hierarchical manner, so that the low-level ones learn a sub-policy for a given objective and the high-level ones solve the base problem.

Rules. A model-based approach is presented in Hein, Hentschel, et al. (2017), entitled Fuzzy Particle Swarm RL. The aim is to learn a set of fuzzy rules offline with particle swarm optimization (Kennedy and Eberhart 1995), using NN's to represent world models. The produced rules take the form: 'if s is m then o ' where s is a state, m a Gaussian membership function and o a real number. The number of rules to be produced must be provided prior to learning. Note that only 2 rules are needed to learn a good policy for the Mountain Car and CartPole problem. The rules are visualised by presenting the membership functions and an example state. The learning of world models has an impact on the production of efficient policies: models that do not approximate the real dynamics correctly can cause the agent to learn the wrong policy.

In the same vein, Huang et al. (2020) propose the interpretable fuzzy RL method to produce an interpretable policy in the form of fuzzy rules. This method is model-free and uses an actor-critic structure to learn the rules. To use this approach, the state-action space is discretised.

As a preliminary work (Akrour et al. 2019), the interpretable policy takes the form of a succession of blocks 'if $close(s, center[k])$ do $action[k]$ ' where the state space is split into clusters with associated actions. The *close* function, which is assumed to be known, determines whether a state s belongs to a cluster. The number of clusters is limited upstream and the cluster centres are not optimised. Learning such a policy is based on the approximate iteration framework (Bertsekas 2011).

With a state represented by an image, the architecture described in Ou et al. (2024) breaks down into three parts: a Compact Convolutional Transformer (CCT) (Hassani et al. 2021) extractor which uses an attention mechanism to extract relevant features, a fuzzy-based decision network which learns fuzzy if-then rules from the output of the CCT, and a Convolutional NN (CNN) latent decoder which reconstructs the agent's state. The policy is analysed by visualising the attention map extracted from the CCT and the most influential rules in the agent's choice of action.

In the following works, the rules are expressed in first order logic, each taking the form of a clause $a \leftarrow a_1, a_2, \dots, a_n$ where a is the head atom (i.e. the action to be performed) and $a_1, a_2, \dots, a_n$ the body atoms (i.e. the preconditions to be met). Most of the works (Hazra and Raedt 2023; Z. Jiang and Luo 2019; Payani and Fekri 2020; L. Zhang et al. 2021) is based on DILP (Evans and Grefenstette 2018) to produce interpretable policies. This neuro-symbolic approach makes it possible to learn a set of symbolic rules weighted by a confidence score, while still having the advantages of using NN's.

Z. Jiang and Luo present Neural Logic RL (NLRL) for learning a policy in the form of first-order logic rules. For this purpose, a DILP architecture is proposed as well as an MDP with logic interpretation. For the experiments, the authors provided only minimal atoms as input to describe the agent's states and background. The NLRL policies are interpretable and achieve near-optimal performance on the problems tested.

Also based on DILP, L. Zhang et al. propose an off-policy algorithm for learning a set of rules. This method is designed to use approximate inference to reduce the number of rules and has the same or better performance than (Jiang and Luo).

Deep Explainable Relational RL (Hazra and Raedt) represents an interpretable policy by a set of if-then rules. The objective is to find a policy such that given a state and background knowledge, an action is predicted. An NN is learned to produce rules and semantic constraints are proposed to avoid redundancy between rules. The approach outperforms NLRL (Jiang and Luo) in terms of generalisation and computation time.

In Payani and Fekri, the approach is a joint use of RRL (Dzeroski et al. 2001) and DILP. RRL is used to transpose the state of an agent in image form into a set of auxiliary predicates. To do this, a CNN is used to take a low-level representation as input and convert it, using auxiliary predicates, into a high-level representation. The dNL-ILP engine (Payani and Fekri 2019) is then used to express the policy in the form of rules.

In comparison with DILP-based methods, Neural Symbolic RL (Ma et al. 2021) avoids storing all the rules by using an attention module. This framework is divided into three parts: the attention module, the reasoning module and the policy module. This approach provides near-optimal policies composed of chain-like rules. However, the rules generated are not always true and must be selected by an expert.

Neural Logic Machines (NLM) (Dong et al. 2019) is a neuro-symbolic architecture for learning the underlying logical rules of a problem. To do this, starting from a set of basic predicates applied to objects, a succession of first-order rules are applied using Multi-Layer Perceptron's (MLP), to obtain a set of conclusions about the objects. NLM is able to solve simple reinforcement learning tasks. The problem with this approach is that the rules learned are not interpretable.

Based on the NLM architecture (Dong et al.), Zimmer et al. (2021) introduce the Differentiable Logic Machine (DLM), a high-performance neuro-logic architecture which outperforms dNL-ILP (Payani and Fekri 2019) and NLM in terms of computation time and memory used. Two operations on predicates are added compared to NLM, namely negation and preservation, and MLP's are replaced by logic modules, which correspond to fuzzy AND and OR operations on predicates, to improve the interpretability of the model. A post-process approach is used to extract logical formulas from the model, making its reasoning interpretable. In Song et al. (2022), this competitive method is tested on a set of autonomous driving scenarios represented in ILP problems. DLM achieves good performance (similar to or better than NLM) while extracting an interpretable set of rules.

Decision Tree. Conservative Q improvement (Roth et al. 2019) is an algorithm that manages the trade-off between interpretability and performance during learning. The policy takes the form of a DT where the internal nodes are conditions based on features of a state and the leaves contain the Q-values of possible actions. Using a dynamic threshold, the tree is extended only if this extension results in a sufficient expected discounted future reward.

Differentiable DT (DDT) is used in Silva et al. (2019). The DDT can be used for Q-learning, where each leaf contains the Q-values of the actions, or for a gradient-based approach, where each leaf contains the distribution of the actions. The tree is then discretised to obtain an interpretable DT (an example for the CartPole environment (Brockman et al. 2016) is shown in Figure 3). In addition, a decision list can be extracted. The results show that these trees have a good performance compared to a MLP, while being more interpretable, according to the user study carried out.

Topin et al. (2021) propose a specific representation of an MDP, called Iterative Bounding MDP (IBMDP), to provide a policy in the form of DT's for the original MDP. This policy is learned using modified versions of either policy-gradient or Q-learning algorithms. The aim is to create a policy that also works for the base MDP. The CUSTARD algorithm presented for solving IBMDPs obtains better results than VIPER (Bastani, Pu, et al. 2018) concerning tree size and policy performance.

An evolutionary approach coupled with Q-learning is presented in Custode and Iacca (2023). The evolutionary approach, inspired by the Grammatical Evolution algorithm (Ryan et al. 1998), is used to find the tree structure and Q-learning to determine the action to be taken for each leaf of the tree. Once the policy has been obtained, a test phase is carried out on 100 episodes to compress the tree. Nodes that are not visited are deleted, as well as internal nodes whose leaves lead to the same action.

Paleja et al. (2022) present the Interpretable Continuous Control Trees (ICCTs) which represent the policy in a sparse way. The interpretable representation consists of internal nodes whose condition is set only by one feature and leaves which are sparse linear controllers (i.e. based on few features). On a set of control tasks tested, ICCTs achieved a good compromise between performance and interpretability.

The Policy Tree algorithm (Gupta et al. 2015) represents the policy as a tree. This method is applied with agent states described by binary features and the policy is learned by gradient-based methods. The tree returned by the algorithm is a binary DT where each internal node corresponds to a feature and a leaf to a distribution of

actions. Starting from any policy and a binary DT containing only one node, two steps are repeated: Parameter Optimisation, which optimises the leaves, and Tree Growth, which extends the tree starting from a leaf.

For a set of autonomous driving scenarios, Likmeta et al. (2020) propose parametric rule-based policies, visualised in the form of trees. Such a policy is a parametric set of rules which is learned using policy gradients with parameter-based exploration (Sehnke et al. 2008).

Although the following works do not aim at interpretability, we think it is interesting to mention them, because of their representation of the Q-table as a tree. Only a few works are presented, to give an overview. Ersnt et al. (2005) present two methods: extra trees and totally randomized trees. The RL batch mode consists of approximating the Q-function based on a set of agent's interactions within the environment. In this context, the two methods are compared with a set of classical supervised learning methods. In Relational MDP, states, actions and background knowledge are represented by a set of predicates and constants forming ground atoms. Das et al. (2020) describe a sample efficient approach to approximate the Q-function in the form of a regression tree in a Relational MDP. By providing a set of knowledge about the problem, the Q function and policy are approximated in Cole et al. (2003). The ALKEMY algorithm (K. S. Ng 2004) is used to learn a set of DT's for each agent task. In addition, a P function is deduced from the Q function, allowing a compact Boolean representation of the policy.

Graph. Two works use a neural network based on the structure of a graph.

Using a total of 20 expert demonstrations, a single Graph Neural Network (GNN) is learned via IL to solve robot manipulation tasks (Y. Lin et al. 2022). The GNN is used to model the underlying structure of the task. Based on GNNExplainer (Ying et al. 2019), the explanation consists of determining the neighbouring nodes and edges of the graph that contributed most to the agent's choice.

In T. Wang et al. (2018), the NerveNet model is proposed to solve continuous control problems. It has been tested on a set of MuJoCo (Todorov et al. 2012) environments. The graph structure used for a given robot is based on its skeleton. Thus, two types of nodes are part of the graph: bodies and joints that connect two bodies. Learned representations are interpreted using different plots, making it possible to analyse the learned policy.

The following three works use abstract machines (or automata), assimilated to graphs, to represent the agent's policy.

The approach proposed by Inala et al. (2020) consists of learning, in a teacher-student setting, a policy represented in the form of a state machine. This policy can be easily interpreted and modified. The proposed approach is called adaptive teaching. Alternatively, the student learns to imitate the teacher's behavior, and the teacher learns to provide behavior similar to that of the student. The student's policy takes the form of a state machine, which consists of a set of modes (an internal memory state) to which is associated an action function (which is akin to a sub-policy). Transitions between two modes are modelled by switching conditions, which reflect the probability of an agent observation switching from one mode to another. Action functions and switching conditions are represented by programs.

DeepSynth (Hasanbeig et al. 2021) synthesises a set of sequences of objects detected from agent states into an automaton. Using an image segmentation method, a set of objects is extracted from a state. With the sequences collected, a deterministic finite automaton (DFA) is produced. This DFA is used to guide the agent in learning its policy. Using both the DFA and the MDP, a Product MDP is generated, where each transition corresponds to a sub-task of the agent.

In Varma (2021), a Tsetlin Machine (Granmo 2018) is used to learn a policy which is an alternative to the classical NN which uses Tsetlin Automaton to form clauses. Thus, the Regression Tsetlin Machine method (Abeyrathna et al. 2019) is combined with Q-learning. The results show that too many clauses are needed to have a good policy, making the method uninterpretable.

Equation. The work of Hein et al. (2019) proposes to use a genetic programming (GP) approach to find interpretable policies in a model-based RL setting. To do this, a dataset of interactions is collected (using any

policy) to learn a world model $\hat{g}$. This is used to learn a policy via GP. This method is compared with two baselines: a NN model-based policy π_m using $\hat{g}$, and a policy learned via GP based on a dataset of interactions generated by π_m . To measure the complexity of a GP-generated policy represented by a function tree, it is simply necessary to sum the different nodes by weighting them according to pre-defined weights (e.g. an addition costs 1 and the function $\tanh$ 4). The policies returned by the algorithm have a similar/better performance to the baselines, while being less complex and therefore more interpretable.

Deep Symbolic Policy learns interpretable policies for environments where the action space is continuous (Landajuela et al. 2021). The framework is split into two parts: the Policy Generator, which uses an RNN to iteratively generate an expression tree representing the policy, and the Policy Evaluator, which evaluates the policy and returns the average episode reward. This is used to train the Policy Generator. Experiments show that the policies of this approach outperform the 7 DRL baselines, such as *Deep Deterministic Policy Gradient* (Lillicrap et al. 2016) and *Trust Region Policy Optimization* (Schulman et al. 2015). To handle the multiple action dimensions of certain environments, the authors propose to use a pre-trained policy to ‘anchor’ the symbolic policy during training.

Finding an efficient and interpretable policy, represented by a small formula, is modelled by a Multi-arm bandit problem in Maes et al. (2012). The space of policies represented by a formula F , such that its size does not exceed a certain threshold (i.e. $|F| \leq K$), is approximated by a clustering method. A formula of minimum size is extracted from each cluster, resulting in N potential policies, each of them represented via an arm in the Multi-arm bandit setting. The best performing policy is then extracted by solving this problem. Compared to previous work described in this section, this approach does not take into account a continuous action space.

For a traffic signal control problem, a regulatable precedence function is learned jointly with the DQN (Ault et al. 2020). A precedence function is regulatable if it is monotonic on state features. In this work, such a function is assumed to be interpretable. 3 variants of DQN are proposed to approximate such a function in addition to learning Q-values.

Program. The Learning Embeddings for lAtent Program Synthetis (LEAPS) framework (Trivedi et al. 2021) synthesises a policy in the form of a program based on a domain-specific language. This framework consists of two stages: firstly, LEAPS learns an encoder-decoder that allows a program to be represented in a latent space, and secondly, an iterative search for a latent program to solve a given task. The latent space is learned in an unsupervised fashion, using randomly generated programs. The aim is to ensure that two programs with similar policies are close in the latent space. Finding a good latent program according to the reward obtained is performed based on the Cross Entropy Method (CEM) (Rubinstein 1997). To demonstrate the interpretability of the programs generated, a user study shows that users have succeeded in analysing and modifying the program so as to obtain a better reward.

In Wilson et al. (2018), simple programs are proposed for playing games in the arcade learning environment (Bellemare et al. 2013). Cartesian Genetic Programming (Miller and P. Thomson 2000) represents the programs by a directed graph indexed by Cartesian coordinates. The input is an image, and the output is a set of scalars used to select the action to be performed. This evolutionary approach relies on a set of functions to build an interpretable policy where each internal node corresponds to a function from this set.

For a traffic signal control task, Gu et al. (2024) use Monte Carlo tree search to find a program, and Bayesian optimisation (Snoek et al. 2012) to fine tune its parameters. Beforehand, a domain-specific language is defined including control flows, conditions and instructions. In addition, four transformation rules are defined to iteratively build a program. The best programs found by the algorithm are small in terms of number of instructions and are easily interpretable.

A.1.2 Surrogate Model.

Decision Tree. A Deep RL (DRL) method is a method that combines RL and Neural Networks (NN) to obtain an agent's policy. A DRL agent is one that has learned using a DRL method. To extract an interpretable policy from a DRL agent, Dai et al. (2022) propose an *information gain rate weighted oblique DT* (IGR-WODT). Based on three defined metrics, IGR-WODT provides a policy that requires a much smaller number of parameters (only 1.1% of the number of DRL model parameters). An *oblique DT* is composed of internal nodes that separate data based on linear combinations of features (i.e. of the form $w_1f_1 + w_2f_2 + \dots + w_nf_n > 0$ where w_i are weights, f_i are state features and n is the number of features). The WODT (B. Yang et al. 2019), on which IGR-WODT is based, uses a logistic regression model to determine the probability of an instance belonging to the left or right child node. The extension provided by IGR-WODT consists of using a different objective function inspired by the information gain (Quinlan 1986).

In the same vein, a *soft DT* (SDT) is used to approximate the policy of a DRL agent trained on the Mario AI benchmark (Karakovskiy and Togelius 2012) in Coppens et al. (2019). An SDT (Frosst and G. E. Hinton 2017) has a pre-defined depth, where each internal node corresponds to a weighted vector to which a bias is added (i.e. perceptron form) and the leaves correspond to a softmax distribution (which in this case represents the actions distribution). The advantage of this approach is that the weights learned from each perceptron can be displayed as heatmaps of the agent's state (which takes the form of an image). In this way, the user can successively visualise the pixels that do or do not have an impact on the agent's decision-making. A compromise between interpretability and performance is observed: as the depth of the tree increases, the SDT performance gets closer to the performance of the DRL agent, at the expense of interpretability.

The same approach is proposed in W. Guo and Wei (2022) for an aircraft separation problem. A visualisation module allows the user to see both the tree plot, where each node is represented by a heatmap, and the trajectory plot, which provides the features that have the greatest impact on the agent's choice of action.

In order to outperform the SDT-based approaches, Ding et al. (2020) present a *cascading DT*, an architecture comprising two kinds of trees: a forest of learning trees F used to provide a compact representation of features, and a decision-making tree D which uses the features learned by F to learn a policy. This architecture reduces the number of parameters and improves performance compared with other SDT-based methods, e.g. Coppens et al.

Another type of DT, the *non linear DT* (NLDT), initially proposed for classification problems (Dhebar and Deb 2021), allows a DRL agent to be represented by a more interpretable model (Dhebar, Deb, et al. 2020). Such a tree has internal nodes that take the form of conditions on non-linear functions. Based on a dataset of DRL agent interactions within the environment, the NLDT is learned using an evolutionary approach. This tree is then pruned to give an interpretable NLDT and re-optimised for performance.

With a concern for reliability, the VIPER algorithm is used to learn a "robust" DT that has the same performance as the DRL agent (Bastani, Pu, et al. 2018). Given a set of state-action situations, imitation learning (IL) (Halbert 1984) aims to learn a policy that mimics the given decisions and generalizes over unseen situations. Based on IL and model compression methods, VIPER produces a DT used to verify the policy according to certain properties such as *stability*, *robustness* and *correctness*. Stability seeks to determine whether the agent asymptotically achieves its objective. The robustness degree of a policy is defined by comparing the action returned for states close to a state s with the one returned for s . Correctness varies according to the environment: for the Atari Pong game, the objective is to prove that the policy never loses, and for CartPole, the objective is to prove that the pole never falls below a certain height. An extension of this work focuses on multi-agent settings. IVIPER (Milani, Z. Zhang, et al. 2022) produces a good quality DT for each agent, but does not take into account the coordinated behavior of the agents. To overcome this, MAVIPER (Milani, Z. Zhang, et al.) was introduced. Experiments have shown that MAVIPER generates better and more robust policies than IVIPER and the other baselines tested.

Also, in a sort of multi-agent view, the MoËT method (Vasic et al. 2022) is based on *mixture of experts* (Jacobs et al. 1991). It consists of a set of expert DT's and a gating function that uses weights to define the extent to which an expert's decision is referred to in the agent's choice of action. The DT's are learned by imitating a DRL agent.

The softmax function is used as MoET's gating function. A variant called MoET_h consists of selecting only one DT using the gating function. This variant is used to transform the learned policy into an Satisfiability Modulo Theories (SMT) formula and thus verify properties using the SMT solver Z3 (Moura and Bjørner 2008). MoET_h outperforms VIPER (Bastani, Pu, et al.) in terms of policy fidelity and reward obtained.

The method proposed by Sieusahai et al. (2021) involves learning a DT based not on the image but on a set of *sprites* associated with their position in the image. A sprite is an image representing a character or object within a game. To learn the DT, a function transformation is used to extract a set of sprites with their coordinates from the image representing the state of the DRL agent. The DT learns to approach the DRL agent based on this more compact representation of the agent's state, which makes the internal nodes of the tree interpretable.

In Cichosz and Pawelczak (2014), DT's and Random Forests (RF) are learned in The Open Racing Car Simulator (TORCS) environment¹ based on IL. The continuous actions are vectors composed of the steering, acceleration, brake and gear parameters in TORCS. RF generalise better and perform better, at the expense of interpretability, than DT's. In order to learn such models, a number of modifications had to be made to the problem: the decomposition of the multidimensional actions problem into a set of one-dimensional actions and the discretization of the continuous actions.

A more user friendly approach uses a tree of objects (G. Liu, Sun, et al. 2021). The RAMi framework is used to learn a tree that mimics a DRL agent based on a representation of states by objects, where an object is a sub-part of an image. This representation is learned using the Identifiable Multi-Object Network and then, the tree is learned using a Monte Carlo Regression Tree Search algorithm. It has a good compromise between fidelity to the agent and simplicity in terms of the number of nodes in it. In addition, a method is proposed for calculating the importance of features and obtaining causal relationships from the mimic tree.

In the similar aim to produce interpretable nodes, Acharya et al. (2020) propose an algorithm to generate a set of strategy labels and interpretable experiential features representations, i.e. predicates, from a set of sequences of interactions of the DRL agent within the environment. Using these elements, a DT is constructed where an internal node uses a generated predicate. The tree is used to represent the agent's different strategies. The condition for following a certain strategy is thus represented by a branch (which is a conjunction of interpretable features).

The algorithm based on IL and program induction proposed in Skirzynski et al. (2021) is used to cluster agent interaction data into a set of rules expressed by a set of predicates. The output of the algorithm returns a set of DT's, which are displayed to the user in the form of flowcharts. These flowcharts represent the predicates of the internal nodes of the DT's in the form of natural language questions.

Linear Model Trees (LMT) are used to approximate the policy directly. The leaves are linear models representing an action to be taken. LMT's have been compared in (Carbone 2020; Løver et al. 2021) with the model-agnostic methods SHAP (Lundberg and S. Lee 2017) and LIME (Ribeiro et al. 2016). The main advantage of LMT's is the lower computational cost, as the model is learned in advance. To explain an action choice, LMT's allow the influence of features to be extracted directly, whereas other approaches require the influences to be calculated. Also, representing the policy by an LMT makes it directly interpretable, to the extent of the size of the tree. As with most other work, a trade-off between simplicity and fidelity is necessary.

For a state s , the value function represents the expected return starting from s . For a state-action pair (s, a) , the Q function represents the quality, or expected return, of executing a from s . Q -learning aims to learn the optimal Q -function by iteratively updating the Q -values based on the Bellman equation. Deep- Q -Network (DQN) (Mnih, Kavukcuoglu, et al. 2015) is a method that learns to approximate Q -values using NNs. G. Liu, Schulte, et al. (2018) introduce Linear Model U-trees (LMUTs) to approximate the Q -values of a DQN agent. This method differs from the other approaches previously presented in this section, where the aim is not to mimic the policy directly.

¹<https://sourceforge.net/projects/torcs/>

LMUTs is an extension of CUTs (Uther and M. M. Veloso 1998), which are regression trees for value functions. Compared to CUTs, LMUTs has leaf nodes that contain linear models of state features. The model is interpreted in three ways: by identifying the importance of the splitting features in the LMUT, by extracting the rule that led to the Q-values obtained and by displaying the pixels that have an above-average influence on the Q-values.

With the same idea of approximating Q-values, Jhunjunwala et al. (2020) proposes a new structure, called Q-BSP Tree, which models the Q-value in an interpretable way and an Ordered Sequential Monte Carlo algorithm which learns to approximate the Q-value of the DQN. For a problem comprising n actions, the Q-BSP forest is composed of n Q-BSP trees, each learning the Q-value of one action. This approach outperforms the baselines in terms of performance and fidelity with respect to the DQN agent policy, including LMUT (G. Liu, Schulte, et al.).

Graph. Topin and Veloso (2019) present *abstracted policy graphs* (APG), which are Markov chains of abstract states. In an APG, edges are transitions labelled by their probability of occurring, nodes are abstract states labelled by the agent action performed from that state. An abstract state represents a grouping of similar states that is constructed with respect to the actions and feature importance of the states, calculated via FIRM (Zien et al. 2009). The transition probability from node S to node S' is an average of the transition probabilities on the states represented by S and S' . To build the APG, a policy, a value function and a set of data on the agent's interactions within the environment are required. An example of APG is described in Figure 4.

In the same vein, McCalmon et al. (2022) propose CAPS, a method that constructs a directed graph composed of abstract states. The state space is abstracted using a decision tree-based clustering algorithm called CLTree (B. Liu et al. 2005). This algorithm returns a hierarchy of cluster configurations, the best of which is extracted by taking into account the interpretability of the clusters and the accuracy of the state transitions. To improve user comprehension, the authors use an approach to highlight the most important abstract states according to the agent's stochastic policy (S. H. Huang, Bhatia, et al. 2018) and a method to describe each abstract state by a natural language sentence.

Based on a set of agent's interactions within the environment, the Policy Graph proposed in Domenech i Vila et al. (2024) is a directed graph that represents the policy with states expressed via predicates (a state is a formula in propositional logic). These handcrafted predicates are defined upstream according to the environment. In the graph, the nodes represent the states, the edges represent the transitions with their probability and the agent's action. With this graph, the authors propose to answer three questions (inspired by Hayes and Shah 2017) using custom algorithms: 'What will you do when you are in state region X ?', 'When do you perform action a ?' and 'Why did you not perform action a in state s ?' This approach was tested on a single-agent and a multi-agent environment.

MARLeME (Kazhdan et al. 2020) is a library that fits into a Multi-Agent RL (MARL) setting to create different directed graphs from agent policies, each representing a Value-based Argumentation Framework (VAF). A VAF is used to model a set of action arguments and the attack relations between them. These arguments are valued in such a way as to order the arguments according to their usefulness. MARLeME can be used to represent an agent, or a group of agents, in the form of a VAF. To use this library, it is necessary to enter the arguments, their valuations and the attack relations.

In order to verify safety properties of a DRL agent, Vinzent et al. (2022) construct an abstract graph based on predicates. Abstraction is only performed for observable states, taking as input the agent's policy and the predicates. Abstract states are constructed on the basis of satisfying the set of predicates (so that states satisfying the same predicates belong to the same abstract state). An SMT solver, Z3 (Moura and Bjørner 2008), is used to define the transitions in the graph.

The following work focuses on transforming a policy that takes the form of a Recurrent NN (RNN) into a Moore Machine (MM) (Moore 1956), which is a finite state machine whose output depends only on the current state and where each state is labelled by an output value (an action in this context). To do this, Koul et al. (2019)

first propose an auto-encoder called Quantized Bottleneck Network (QBN), which learns to quantize the agent's observations and the RNN's memory states for a MM representation. Next, the QBN is combined with the RNN to create a Moore Machine Network (MMN). Finally, a MM is extracted from the MMN and minimised to be interpretable. In the MM, the nodes represent the states of the memory, and the edges represent the agent's observations. Each node is labelled with the action to be performed from a memory state. In some environments, it has been observed that the number of memory states and observations to represent the agent's policy is small. An extension of this work, Danesh et al. (2021), criticises the interpretability of the MM provided and proposes four ways of reducing the size of the MM in an interpretable way. Indeed, in Koul et al. the minimisation techniques used do not take into account the user's comprehension, for example certain states judged to be different for a user can be merged into a single state. The proposed minimisation methods improve the visualisation of the MM, while preserving the key decision points of the agent's behavior. A tool based on the saliency map principle (presented later in this survey) is proposed as a complement to understand the differences in the agent's choice of action between 2 observations.

Based on a set of sequences of interactions of the agent within the environment, Hüyük et al. (2023) present a Bayesian algorithm for learning decision dynamics and decision boundaries. Decision dynamics aggregate sequences as beliefs, which correspond to the probability that a state s exists at a time t . Decision boundaries represent policy by partitioning decision dynamics into regions where the same action is performed.

The quasi-symbolic (QS) agent (J. H. Lee 2019) is composed of two interpretable networks made up of a single layer: the matching network, which memorises transitions, and the value network, which evaluates them. This agent uses the RL agent and the environment function to predict a plan of actions that leads to one of the most valuable transitions (in terms of reward). The structure of the QS agent makes its behavior interpretable.

Program. In order to design interpretable and verifiable policies, the *programmatically interpretable reinforcement learning* (PIRL) framework is proposed in Verma, Murali, et al. (2018). Policies are generated using an environment-specific programming language. To compensate for the size of the policy search-space, the specification of the policy form is used in PIRL through a sketch, which is a restriction of the grammar proposed by the language. The Neurally Directed Program Sketch (NDPS) algorithm is used to locally search for a policy that takes the form of a program based on the NN agent policy, a Partially Observable MDP (POMDP) and a sketch. POMDP (Åström 1965) is a generalisation of MDP where the agent does not have full visibility of the state it is in. NDPS returns a policy close to that of the DRL agent and which obtains the highest expected aggregated reward among the programs evaluated during the search.

In the same vein, Zhu et al. (2019) synthesise a program close to the DRL agent policy which avoids reaching unsafe states of the environment. To this end, the *counterexample-guided inductive synthesis* (CEGIS) framework is presented to search for a program parameterised by a sketch (i.e. a high-level description of the sought program). In this approach, a counterfactual example is an initial state in which the program under study has not yet been proven safe. In addition to the interpretability aspect, the program produced is used as a safety shield: it takes the relay when the agent chooses an action that may lead to an unsafe region.

Based on a DRL agent policy, INTERPRETER (Kohler, Delfosse, et al. 2024) is used to obtain an interpretable and editable python program. To do this, a regularized oblique tree (Murthy et al. 1994) is learned from the DRL agent policy via IL, then converted into a python program, notably by translating the nodes of the tree using *if-else* statements.

For robot manipulation tasks, a program is extracted from a set of demonstrations in Burke et al. (2019). To do this, a probabilistic generative task model is used to infer from the demonstrations a sequence expressed by interpretable low-level motion primitives. Program induction is then used on this sequence to simplify it into a program.

The PROPEL approach (Verma, H. M. Le, et al. 2019) makes it possible to discover policies that are interpretable, verifiable and generalisable. The idea is to iteratively search for a policy represented by a NN using policy-gradient methods and to project the policy into the space of programmatic policies (i.e. policies representable by programs). Two PROPELs searching for policies in different programmatic policy classes have been proposed: PROPELPROG, which uses an environment-specific language, like NDPS (Verma, Murali, et al.) and PROPELTREE, which uses tree regression like VIPER (Bastani, Pu, et al. 2018).

For a detailed overview of program synthesis for interpreting DRL agents, we recommend Bastani, Inala, et al. (2020) to the interested reader.

Rules. In Nageshrao et al. (2019), a set of fuzzy rules is learned to approximate an agent's policy. Based on a dataset representing the agent's interactions with the environment, the rules are learned using the Evolving Takagi-Sugeno method (Angelov and Filev 2004). To obtain these rules, it is necessary to learn the rule antecedents, which decompose the state space into relevant regions, and to estimate the feature coefficients of the rule consequents, which are linear model of each rule used for determining the action to be taken. This approach allows the policy to be approximated by a set of conditioned interpretable linear models.

Soares et al. (2021) propose prototype-based fuzzy rules which are then visualised by displaying either features or prototypes. To reduce the number of fuzzy rules generated, the authors propose a hierarchical mechanism to group adjacent prototypes in the data space that have the same consequent (i.e. action).

Equation. The substitute policy takes the form of equations in the following two works. It is in a context where the agent's action is determined as a function of continuous features of the agent's state. Genetic algorithms are used to find such policies.

Zhang et al. (2020) use the evolution feature synthesis (Arnaldo et al. 2015) to generate control policies that mimic the DRL agent policy. A generated policy is a combination of operations (defined on a set of operators O) using reals and features of a state. In order to make the policy interpretable, a method based on the complexity-performance trade-off is proposed.

In the same line, the method presented in Kubalik et al. (2017) is a variant of Single Node Genetic Programming (Jackson 2012), which represents the individuals in a population by a node in a graph. For a robotic task, this approach approximates a policy that produces smooth control by an equation, based on a set of interactions of the agent within the environment.

A.2 Policy Summary

A.2.1 Sequences. The HIGHLIGHTS algorithm (D. Amir and O. Amir 2018) provides a set of sequences that summarise the agent's behavior. To select sequences, Amir and Amir focus on important states using the notion of *state importance* (Clouse 1996) (which uses the agent's Q-values). From the important states, sequences are created to provide the user with more context. Thus, a sequence includes states that precede the important state, the state itself, and states that follow it. In the same work, the HIGHLIGHTS-DIV algorithm is proposed to provide more varied sequences as a summary. Several studies have used HIGHLIGHTS to provide explanations: Huber, Weitz, et al. 2021; Septon et al 2023. Huber et al. use HIGHLIGHTS-DIV coupled with saliency maps to provide additional information. A saliency map makes it possible to determine which part of the state (which takes the form of an image) is responsible for the agent's choice of action. In the same vein, Septon et al. combine HIGHLIGHTS-DIV with the reward decomposition approach. This method decomposes the reward function into several interpretable objectives, making it possible to determine which goal(s) the agent is seeking to maximise by performing a certain action. Saliency maps and reward decomposition will be described in more detail in Section 6.1.1 and Section 6.2.2 respectively. Amitai et al. (2024) propose the CoVIZ algorithm to compare the outcomes between the action a taken by the agent from a state s and a counterfactual action a' . A variant of this

algorithm is described for extracting a summary of sequences. In this case, the agent's sequences are proposed at the same time as the counterfactuals. These counterfactual sequences start with a counterfactual action a' , then follow the agent's policy. The pairs of sequences that contrast the most in terms of outcomes (based on the value function) are displayed to the user.

Sequeira and Gervasio (2020) propose a different approach to providing a summary. During the agent's interaction with the environment, a set of data is collected to extract information about, for example, the frequency of situations encountered or the agent's confidence in these decisions. These Interestingness Elements are used to provide the user with a visual summary. In Feit et al. (2022), this approach was used in conjunction with the reward decomposition method. Interestingness Elements has been extended by the IxDRL framework (Sequeira and Gervasio 2023) which contains new interestingness elements to analyse, a method for clustering trajectories based on interestingness elements and the use of SHAP (Lundberg and S. Lee 2017). This model-agnostic method is used to determine the impact of each feature of a state on the agent's decision. SHAP will be described in more detail in Section 4.2.3.

Another approach (Lage et al. 2019a,b) shows the impact of the user's mental model and the model used to create the summary on the user's reconstruction of the agent's policy. Lage et al. compare IL with Inverse RL (IRL) (A. Y. Ng and Russell 2000). As a reminder, given a set of state-actions situations, IL aims to learn a policy that mimics the given decisions and generalizes over unseen situations. IRL consists of extracting a reward function based on a set of sequences that matches the observed behavior of the agent. In order to provide informative sequences to communicate the agent's goal, S. H. Huang, Held, et al. (2019) use an algorithmic teaching approach to model users' mental models.

Frost et al. (2022) use an exploratory policy to reach a set of various states not seen during the training phase. Sequences are created from each of these states using the agent's policy. The aim is to provide an informative summary of the agent's behavior in the case of a state distribution shift that may occur during the test phase. In the same idea, Altmann et al. (2024) use an evolutionary algorithm to perturb the initial state and obtain a varied set of sequences.

In order to learn a policy based on a dataset, Gottesman et al. (2020) propose an efficient approach to identify the transitions in the dataset that are most important in the agent's learning. The idea is that removing an influential transition has a big impact on learning. To determine the influence of a transition t , a disparity of predictions is performed between a critical function that has learned with the entire dataset and a critical function that has learned without t .

A.2.2 Critical States. An agent's policy can be summarised simply as a set of so-called critical states in the agent's interactions with the environment.

In this sense, Huang et al. (2018) provide the user with a set of critical states of a policy as an explanation. This provides a restricted view of the agent's capacities on critical states, i.e. states where the Q-value varies greatly between different possible actions.

Based on a generative model producing an agent's state and a user-defined objective function, the method proposed by Rupprecht et al. (2020) can be used to identify and visualise agent weaknesses. It has been tested on Atari games, where the state is an image. With this approach, it is possible, for example, to look at a given action a by generating states in which the agent assigns a large Q-value to a , then seeing whether this action is coherent from the states generated.

In the context of MDP, Sreedharan et al. (2020) propose a policy summary in the form of milestones, called landmarks, that the agent seeks to reach. The summary is a partial order of landmarks. For the exploding blocks-world problem (Younes 2005), an ordering of landmarks is described in Figure 5. The identification of landmarks is done using compilation procedures and the ordering between these landmarks using methods for reasoning about task hierarchies in MDP's.

Reward redistribution is the process of creating a dense reward function for a given problem (Arjona-Medina et al. 2019) (as opposed to a sparse reward function). Using this, Dinu et al. (2020) present a method for extracting key events from an agent's behavior. An event corresponds to a cluster of state differences. The agent's policy is then expressed as a sequence of key events.

Other works are particularly interested in summarising the agent's training: Dao, Huff, et al. 2021; Dao, I. Mishra, et al. 2018; I. Mishra, et al. 2018. The idea is to save relevant tuples of snapshot images (i.e. states), actions, and weight distributions in memory to keep the agent's important learning experiences. The weight distribution associated with each snapshot-action pair corresponds to the impact that snapshot has on the development of the policy. Important snapshots can then be viewed to understand the agent's decisions. In Dao, Huff, et al., saliency maps are generated to reduce the number of important snapshots using Grad-CAM (Selvaraju et al. 2017).

As a verification framework for DRL, Verily (Kazak et al. 2019) allows the user to check whether a policy satisfies a certain requirement. This framework combines scalable model checking and formal NN verification methodologies. This verification is carried out at the level of the states of an agent. If the requirement is not met, a counter-example is proposed (in the form of a state).

A.2.3 SHAP. This work is based on the model agnostic method, called SHAP (Lundberg and S. Lee 2017). It is interesting to note that this method is mainly used to explain classifiers. As already mentioned, it consists of determining the impact of each feature of an input on the output of the model. In our context, the input is a state, the output an action, and the model the agent's policy. This method is an approximation to the computation of Shapley values (Shapley et al. 1953). Originally from the field of game theory, Shapley values are used to determine the extent to which a member of a coalition contributes to the final value obtained. To determine Shapley values, it is necessary to calculate for each player m of the game, the set of possible coalitions c containing m , and to see the impact of its removal from c on the final value. SHAP proposes different approximations to Shapley values, which consist in limiting the number of coalition samples. Two approximations have been used in the work presented in this survey: Kernel SHAP, a model-agnostic estimation, and DeepSHAP, an estimation specific to neural networks. Kernel SHAP is based on Linear LIME (Ribeiro et al. 2016) (this method will be described in Section A.4.3) and DeepSHAP on DeepLIFT (Shrikumar et al. 2016). As this section is concerned with the explanation of policies, SHAP is used globally: it identifies the impact of each feature on the agent's policy.

There are two works that use the method as is: Zhang et al. (2022) use the DeepSHAP method in the context of power system emergency control and Wang et al. (2020) use Kernel SHAP for a machine control use case (an example of SHAP is shown in Figure 6).

In the ixDRL framework (Sequeira and Gervasio 2023), SHAP values are calculated using XGBoost machines (T. Chen and Guestrin 2016) to simplify the analysis. Beechey et al. (2023) distinguish their approach from previous work by proposing a general framework for the use of shapley values in the context of RL. They also propose a method for providing global explanations of the agent's performance, i.e. the expected return.

In the context of cooperative MARL, Heuillet et al. (2022) compute SHAP values using Monte Carlo sampling. This use of SHAP is closer to the basic principle of Shapley values. In fact, this method makes it possible to determine the contribution of each agent. In the same context, Angelotti and Diaz-Rodriguez (2023) compare the calculation of Shapley values with that of Myerson values (Myerson 1977), preceded by the construction of a Hierarchical Knowledge Graph (HKG) representing the agents and their features. Myerson values are an equivalent of Shapley values specific to cooperative games constrained by a graph. They are therefore calculated on the basis of the HKG. In this work, the contribution of agents' policies and their features are studied jointly. The computation of Myerson values leads to a significant time saving compared to Shapley values.

A.2.4 Policy Comparison. The DISAGREEMENTS algorithm (Amitai and O. Amir 2022) shows a visual summary of the most important disagreements between the two agents. The agents' Q-values are used to determine their disagreement. The summary takes the form of a set of various trajectories. This disagreement is measured in

terms of ability whereas Gajcin, Nair, et al. (2021) focuses on a disagreement based on preferences. Indeed, in a problem where different winning strategies may exist, an agent may prefer one strategy over another. Based on a set of disagreements obtained between two policies, the method determines the type of states (in terms of features) that the different agents seek to achieve.

A.3 Human-readable MDP

A.3.1 *Inherently Understandable.*

MDP Representation. This section groups works that represents the MDP, especially the dynamics of the environment (i.e. the reward function and the transition function) in an interpretable way.

In the following work, interpretability is based solely on the reward function.

In a close air combat context, the reward decomposition (Erwig et al. 2018) is used globally to determine in each tactical region the type of reward that is dominant (Saldiran et al. 2024). A tactical region is defined beforehand and represents a set of states. In addition, a visualisation method is used to analyse the impact of the different types of reward on the different tactical regions. This representation takes the form of heatmaps of normalised Q-values.

The reward function learned in Bica et al. (2021) is a weighted sum over potential outcomes, i.e. changes in features of a state. The approach is in an IRL batch setting, where the objective is to learn the reward function based on a dataset of expert policy trajectories. The weights of the different outcomes are used to determine their influence on the agent's choice of action. The main limitation of this approach is that it assumes that the reward function is a linear function over the features.

In Jenner and Gleave (2022), the idea is to transform a reward function r into a more interpretable function r' . To use this framework, it is necessary to instantiate an equivalence relation between two reward functions and a cost function measuring the interpretability of a function. For example, the cost function can take into account the sparsity or smoothness of a reward function. In a gridworld-type environment, additional pre-processing is carried out to make the generated function more interpretable, which is then displayed in the form of a heatmap.

EXPRD (Devidze et al. 2021) designs interpretable reward functions by making a compromise between informativeness and sparseness. This function is learned from a dense MDP reward function. The interpretable reward function must satisfy an invariance requirement, i.e. the optimal policy obtained with this function must belong to the set of optimal policies of the base MDP. To construct such functions in a large state space, a state abstraction method is proposed.

The Adversarial IRL algorithm (AIRL) (Fu et al. 2017) is extended by Srinivasan and Doshi-Velez (2020) to build more interpretable reward functions. AIRL consists of using a discriminator to help learn a reward function from which the learnt policies are close to expert demonstrations. The interpretability aspect of the reward function is added using the Deep Neural DT architecture (Y. Yang et al. 2018), which represents in tree structure the different feature valuations (called binning layer in NN), before using a decision layer to provide a reward. This method obtains smooth and sparse reward functions.

Preference-based RL (PbRL) consists of solving a problem modelled by an MDP where the reward function is not provided. A fitness function reflecting the user's preference on pairs of trajectories is used to learn a reward function upstream of the agent's policy. In Bewley and Lécué (2022), the proposed PbRL algorithm represents the interpretable reward function as a binary tree structure where each internal node is a test on a state-action pair and each leaf returns a reward. An example is shown in Figure 7.

The following works do not speak of explainability or interpretability but are mentioned because of the non-opaque form of the reward functions presented. Icarte et al. (2019) propose to express the reward function in the form of a Finite State Machine as well as an algorithm, Q-learning for Reward Machines, to learn policies efficiently. Based on specifications given in a formal language such as Linear Temporal Logic (LTL), Camacho

et al. (2019) construct a reward machine to represent the different goals and temporal properties. For partially observable environments (i.e. the POMDP setting), Icarte, Klassen, et al. (2018) learns with the help of experiences a reward machine and Z. Xu et al. (2020) jointly learns high level knowledge modelled by a reward machine and the agent's policy. In Gaon and Brafman (2020), a model comprising non-Markovian Rewards is transformed into a Markovian model by learning a reward deterministic finite automaton from a set of sequences. The robustness degree (X. Li et al. 2017) is used to extract a reward function from a logical formula expressed in Truncated LTL (TLTL). Such a formula corresponds to a specification of a task to be performed. Based on a set of demonstrations, Kasenberg and Scheutz (2017) represent the task specification in the form of an LTL formula, instead of inferring a reward function that cannot be interpreted using IRL techniques.

Only the transition function is interpretable in the three following works.

Using expert knowledge about the environment, Kaiser et al. (2019) describe a model-based RL method that learns an interpretable transition function. In the illustrative problem, the expert knowledge gives the information that the state s_{t+1} obtained from the state s_t by performing the action a_t depends on three different sources. The transition function is learnt using a Bayesian approach, taking into account a structure including the three sources. The resulting probabilistic model shows the sources impact on the transition for each agent's state-action pair.

Motion-Oriented REinforcement Learning (MOREL) (Goel et al. 2018) improves the interpretability and, depending on the problem, the sample complexity of the RL algorithm used, by tracking moving objects in states. This approach is tested for a set of 59 Atari Games, where the state of an agent takes the form of an image. First, an unsupervised method is applied to learn to predict object and camera motions as well as object masks using a random policy to generate the data. Then, the agent uses and refines the previously learned model to learn its policy. Users can view the object masks to understand the agent's policy.

Although not presented as XAI papers, we find it interesting to mention the following works. Metzen (2013) proposes an efficient heuristic called FIGE to represent the transition function by a graph. The SPITI algorithm (Degris et al. 2006) learns a decision tree for the reward function and one for the transition function using the decision tree induction algorithm ITI (Utgoff et al. 1997).

Finkelstein et al. (2022) set out an approach based on a partial contrastive policy on a subset of states described by the user. The explainer looks for a meaningful transformation sequence to modify the current MDP into a contrastive MDP in which the agent would act according to the user's expected policy. A set of such user-interpretable transformations is provided as input to the problem. An example of such a transformation is the single-outcome determinization transform (Yoon et al. 2007), which makes the transition function deterministic by considering only the most likely transitions. To find a sequence of transformations, a Dijkstra-like search is performed.

Lazy-MDP's (Jacq et al. 2022) are MDP's that include a default policy. The agent must then learn to estimate when it is necessary to replace the default policy and what action to take in replacement. Such an approach makes the policy sparse, making it possible to understand the states in which it is critical to take over from the default policy. In this work, the modelling problem and the policy generated make the approach interpretable.

In a human-robot cooperation setting, the framework proposed by Tabrez and Hayes (2019) consists of detecting differences between the human and robot task models and proposing an interpretable modification for the human model. To do this, the human's reward function is inferred based on its behavior during cooperation using Hidden Markov Models. A particular POMDP is presented, so that the agent can choose between an action that informs the human or an action related to the task to be performed. This communicative action tells the human about a missing piece of information in its reward function.

Relational RL and MDP. The following works provide an interpretable representation of states and transitions and are based on works presenting relational MDPs (Guestrin et al. 2003) and Relational RL (Dzeroski et al. 2001).

As a reminder, the objective is to be able to learn a generalizable and efficient policy based on a representation of the environment by a set of objects and relations between them. In this section, we present only an overview of work in these domains. For more details, we recommend reading the state of the art Otterlo et al. (2012) dating from 2012, which does not therefore include recent work.

The works presented in this paragraph are described briefly because they are not proposed as explainability methods, although the approaches do make the MDP interpretable. Diuk et al. (2008) introduce the Oriented-Object MDP (OO-MDP) to model the problem based on objects and their interaction and describe an algorithm for solving deterministic OO-MDPs. This algorithm is modified to solve Deictic OO-MDPs (Marom and Rosman 2018), which represent relationships between an object and an object class (instead of a simple object-object relationship), enabling better abstraction. Based on this idea, Scholz et al. (2014) focus on physical domains, and propose two approaches: an extension of OO-MDP and a more efficient approach using a physics engine. In a model-based RL context, Veerapaneni et al. (2019) approximate the observation model which represents an image (i.e. agent state) by a set of semantic masks representing objects and the dynamics model which describes the temporal evolution of the different objects for planning. Schema networks (Kansky et al. 2017) is a generative model of an MDP that learns the dynamics of the environment in a setting where the state is represented by a set of objects in an image. An interpretable MDP is built in Walker et al. (2007) where abstract states are learned via ILP techniques and transitions and rewards are estimated. The policy is simply learned by value iteration in this MDP. The relational actions schemas class (Walsh 2010) groups together a set of languages that compactly represent the transition function using relational conditions and effects.

An RRL algorithm is proposed in Martínez et al. (2017) in the context of robotic tasks where the agent can request demonstrations from the expert in order to learn efficiently. This allows the agent to learn in a sample-efficient way and to request a demonstration when an action is unknown to it. In this work, each action is defined by a set of preconditions and effects weighted by probabilities. To guide the expert in generating demonstrations, the excuse principle (Göbelbecker et al. 2010) is used to explain why the plan failed. An excuse returns the important predicates that caused the plan to fail.

The framework described in G. Zhu, Z. Huang, et al. (2018) is used to extract objects from an image and learn the transition function. The objective of this framework is to generate the next image (i.e. state) using the information extracted from the image (its background and the various objects) and the action of the agent. To do this, the background extractor learns to extract the background, the object detector learns to produce masks of different types of static or dynamic objects and the dynamics net learns to make the transition. The object masks show that the method correctly learns to distinguish objects in the game tested, even on different levels of the game (i.e. unseen environments). An extension of this work, G. Zhu, J. Wang, et al. (2020) proposes a three-level learning architecture for learning model dynamics from the most to the least abstract. To aid learning, the output of an abstract level is provided to the level below. In order from most to least abstract, motion detection detects regions containing dynamic objects from an image sequence, instance segmentation produces coarse masks of dynamic objects and dynamics learning learns the transition function based on agent action and object relationships (as in G. Zhu, Z. Huang, et al.).

The Object-Level RL (Agnew and Domingos 2018) is a framework that produces a high performance policy by learning a representation of states using relative and absolute distances, velocities, acceleration and contacts between the agent and objects in an image. This method was tested on a set of Atari games. During sample-efficient learning, the agent's exploration is based on a prior knowledge, which states that interesting experiences occur when two objects come into contact.

Zambaldi et al. (2018) describe an architecture where the agent learns to extract objects from the image, determine relations between them and produce the policy and value for a state. For extraction, a CNN is used and a self attention mechanism (Vaswani et al. 2017) is used to compute relations between objects. These representations are then given as input to an RL algorithm for learning the policy and value function. The

visualization of attention weights on different objects allows to analyze the learned relations between objects and to understand the decision making of the agent.

Symbolic Relation Network (Adjodah et al. 2018) takes as input a set of objects which describes the agent's state, represents them as a unary or binary relation and then concatenates them into a relational state used to learn Q-values with a DQN. In the environment used, the learned relations are interpretable: they represent the type of object and the relative location of the objects with respect to the agent. In this work, the object representation of states is not learned.

A.3.2 Surrogate Model.

States Clustering. Bekkemoen and Langseth (2023) propose the ASAP method, which groups states into clusters representing abstract states. The explanation consists of two parts: a display of a representative state and its associated action for each abstract state identified and a modelling of a policy graph, which is a Markov chain showing the agent's behavior in the abstract state space. Abstract states are identified using attention maps (which can be likened to saliency maps). ASAP is illustrated with the Mountain Car environment (Brockman et al. 2016) in Figure 8.

In the next two paragraphs, the explanation describes a specific cluster of states in natural language that answers the user's question.

Given a set of communicable predicates for describing states, Hayes and Shah (2017) describe a framework for answering different questions about policy, such as 'When do you do action a?'. To do this, the user's question is first identified, then a region of states is extracted based on the question. Using the optimised Quine-McCluskey algorithm (Jain et al. 2008), this region represented by a logical formula is minimised. Since the predicates are communicable, the resulting compact formula is transposed into a human language description for the user. A complementary study of this approach is carried out in Climent et al. (2021) on the CartPole problem. This approach is combined with Reward Decomposition (Erwig et al. 2018) to provide both local and global explanations of agent behavior in Iucci et al. (2021). Based on this method, Booth et al. (2019) carried out a user study to analyse the interpretability of different propositional theories.

Using knowledge compilation techniques, the framework of Wollenstein-Betech et al. (2020) answers a set of questions relating to the choice of a given agent action. For example, this method can provide the probability of doing an action knowing that we are in a state where one of its features is true. To do this, given a set of state-action tuples and a question, a deterministic Decomposable Negation Normal Form (d-DNNF) is constructed, model counting and probabilistic inference are performed on the d-DNNF and the result is translated into natural language.

By collecting all the transitions during agent training, Bewley et al. (2022) propose a state abstraction as well as a temporal abstraction to analyse agent learning. The state space is decomposed into regions, called abstract states. The characterisation of these states is provided in the form of a tree diagram. The learning phase is divided into several windows, where each window is represented by a transition graph using the abstract states. This reflects the agent's interactions with the environment over a learning period. In addition, two graphs are provided to represent the rate of visits to abstract states as the training progresses, as well as the abstract states in which an agent's episode ends.

TRIPLETREE (Bewley and Lawry 2021) is an algorithm that constructs a decision tree where the branches represent the action, the value and the state derivative estimate. This method allows the state space to be discretised according to three criteria: agent's action, value function and temporal dynamics. In addition, the transition probabilities between the different regions obtained are computed. A set of visualisation tools is used to analyse the environment and the agent. In addition, with these different regions, a set of explanations can be provided on the different criteria: factual explanations describing the limits of the region or counterfactual explanations based on a principle of minimal change to reach a certain region.

State Transformation. The field of State Representation Learning (SRL) consists of learning a low-dimensional representation of the state, which evolves over time and reflects the change induced by the agent's actions. Although the main objective of SRL is the performance of the learned policy, proposing a state representation composed of a small number of comprehensible features makes it possible to provide interpretability, as stated in the survey Lesort et al. (2018). The objective of SRL is to learn a mapping function which, for a history of observations, returns the state in a low-dimensional representation. To do this, supervised approaches use the true high-dimensional state, while unsupervised approaches (the ones described in the survey) do not. Raffin et al. (2018) propose a toolbox for SRL methods to compare approaches in different environments, using visualisation and metrics.

In Garnelo et al. (2016), the agent module decomposes into two sequential parts: a neural network that learns to compress a state into a symbolic form, and a Q-learning algorithm that learns to reason based on the symbolic representation. In the tested environment, the neural network represents an image symbolically by a set of objects, their characteristics and interactions. This approach outperforms the compared DQN, and allows the agent's behavior to be interpreted through symbolic states.

D'Avila Garcez et al. (2018) present a simplified version of the above approach as well as an extension. The symbolic representation of a state is composed of sub-states reflecting the relative position of the agent with respect to an object. Learning and action choice are modified to take account of this representation in the proposed extension. This addition of common sense results in a high-performance policy.

By combining a hierarchical learning architecture with conceptual embedding techniques to embed prior knowledge, the agent learns a more interpretable representation of the state (Dai, Ouyang, et al. 2023). A saliency map is generated on the hidden layers including the embedding of prior knowledge, to determine the impact of this knowledge on the agent's decision-making.

François-Lavet et al. (2019) propose to combine the model-based and model-free approaches of RL algorithms to learn an abstract state representation. A specific loss is added for learning the representation of an abstract state so that features of the abstract state are impacted by the agent's actions. This makes the representation interpretable. The method is applied in a deterministic environment, but can be adapted for stochastic environments.

The HRL method described in A. Zhang et al. (2018) proposes to learn a representation of states in the form of a set of attributes, as well as a transition function between these attributes. In this work, the binary attributes are learned in a supervised manner and correspond to the relative positions of objects in a 3D environment.

In J. Chen et al. (2022), a latent model of the environment is learned at the same time as the agent's policy in order to reduce the sample learning complexity and generate semantic bird-eye masks. For an urban autonomous driving problem, the input consists of a front view image and a lidar image. The generated masks (learned from ground truth) are based on the input to display the map, the route that will be taken, the surrounding objects and the car position.

ReCCoVER (Gajcin and Dusparic 2022) detects causal confusion in the agent decision-making on a set of critical states. From each critical state, the policy is tested on a set of alternative environments. These environments consist of imposing a certain value on a feature of the state. To detect causal confusion, it is necessary to learn a set of feature-parameterised policies for each subset of features. Causal confusion occurs when the agent's policy does not perform well in an alternative environment compared with a feature-parameterised policy. ReCCoVER then proposes a correction for these problematic states by providing the user (in this context, a developer/expert) with the feature subset on which the agent should base itself when it encounters this state.

A.4 Visual Analysis

A.4.1 Visual Toolkit. Mishra et al. (2021) propose a total of 8 tools to be used to understand the agent's policy. This interface, which is described in Figure 9, requires the features describing a state to be interpretable. PolicyExplainer

provides a global view of the frequency of actions and rewards, the policy, the value function (cf. panels A-C in Figure 9) and can be used to look at particular states and their associated Q-value, or a particular trajectory (cf. panels D-G in Figure 9). In addition, the ‘Why?’, ‘Why not?’ and ‘When?’ questions are answered by providing the state features that have an impact on the choice of action (cf. panel H in Figure 9).

The other visualisation interfaces are designed for experts, to help them with debugging. DRLViz (Jaunet et al. 2020) analyses the latent internal memory of a DRL agent in the context of a video-based environment. DQNViz (J. Wang et al. 2019) focuses on the interpretation of DQN training and illustrates it using the Atari Breakout game. DynamicsExplorer (W. He et al. 2020) focuses on a DRL agent that includes a Long-Short-Time-Memory (Hochreiter and Schmidhuber 1997) layer for encoding environment dynamics. MDPvis (McGregor et al. 2017) is used to analyse and modify an MDP and optimise the policy.

A.4.2 Latent Space Visualization. Zahavy et al. (2016) propose to visualise the activations of the DQN by applying t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and G. Hinton 2008) to reduce the dimension. Before visualisation, a set of data is collected, including hand-crafted features such as the agent’s position, so that clusters can be identified and analysed. In addition, saliency maps are also generated on the different states collected. By analysing the dynamics between clusters, the authors identified that the DQN has learned a hierarchical aggregation of the state space.

Following on from this, Zrihem et al. (2016) present a method for modelling the t-SNE representation of the DQN by a Semi Aggregated MDP, a human-interpretable approximation of the MDP. Also, clustering algorithms are used to identify the structure of the t-SNE maps, instead of having to construct features by hand. These clusters are states of the Semi-MDP and the probability of moving from one cluster to another defines the transition matrix of the Semi-MDP. An example of Semi-MDP obtained based on the policy of an agent that has learned to play Atari 2600 Breakout (Bellemare et al. 2013) is shown in Figure 10.

A specific architecture is presented by Annasamy et al. (2019) with the aim of interpreting the DQN when the agent’s states are images. The agent learns to focus on keys of a store composed of an action and a Q-value. These keys can be assimilated to cluster centres. Using deconvolutions, these cluster centres (a, q) are then visualised in order to understand the important elements of the reconstructed image that lead to the choice of a and a Q-value q . This image represents an aggregate of states. For example, in Mrs Pacman, the visualisation of an action-return pair displays the different positions of the yellow blob in which the agent chooses the pair (a, q) .

A.4.3 Inspection. ETemoX (Ullauri et al. 2022) is a framework for keeping track of the evolution of certain metrics as the agent is trained. This framework consists of three parts. The Translator, which transforms the data collected from the agent’s interaction with the environment. The Complex Event Processing (Luckham and Frasca 1998), which filters the data to select only the relevant ones. The Temporal Graph Database, which represents relationships between multiple evolving metrics with time dimension. In this work, three RL algorithms were compared using ETemoX by monitoring the overall reward, the exploration/exploitation trade-off and a feature specific to the environment used. Dethise et al. (2019) study the *Pensieve* agent by looking at, for example, the probability of action choice or confidence in the choice of actions. In addition, a study of the global importance of the agent’s state features is carried out using LIME (Ribeiro et al. 2016).

This method estimates the influence that each feature of input x has on the model’s output. In our context, the input is a state of the agent, the model the agent’s policy and the output an action. To calculate the influence of features, a transparent surrogate model is learned locally around x . This surrogate model is a linear model. To train it, a set of points is generated by perturbing x and weighted as a function of distance from x . After training, the weights of the linear model are used to explain the influence of each feature of x on the output of the model. In Dethise et al., LIME is used on a set of states to provide the average contribution of each feature to the agent’s choice of actions.

B Summaries of Sequence-level Methods

This appendix provides a summary of each sequence-level method mentioned in the paper. These methods explain sequences of interaction between the agent and the environment. As a reminder, a total of three ways of explaining sequences have been identified. *Counterfactual Sequences* allow a comparison of the agent's behavior with a sequence that uses an alternative policy, and thus determine the agent's strengths and weaknesses. The *Important Elements* of a sequence are those that have the greatest impact on the agent's ability to achieve a given objective. The use of *Human-readable MDP* makes it possible to understand, in the context of a sequence, the information available to the agent, the actions and the dynamics of the environment.

B.1 Counterfactual Sequence

Van der Waa et al. (2018) propose to create a counterfactual sequence based on the user's query. An alternative policy is obtained by modifying the agent's Q-value so that the policy locally follows the user's contrastive query. The sequences are generated using the most probable transition at each time step (note that the transition function was obtained by training). The explanation consists of describing the sequences with the occurrence of the actions performed and states encountered, as well as the positive and negative outcomes.

In a context where the environment includes different reward classes, Alabdulkarim and Riedl (2022) train a set of influence predictor models for each reward type during agent training. The counterfactual sequence is generated by considering the action (or sequence of actions) proposed by the user, then using the agent's policy. The sequences are compared using influence predictors (i.e. a calculation of the average influence of each type of reward in each sequence), the cost of the actions and the final reward obtained. An example showing the interest of this approach is presented in Figure 12.

The evolutionary algorithm, named ACTER (Gajcin and Dusparic 2024a), generates a diverse set of counterfactual sequences to propose to the user sequences that could have prevented the agent from reaching a state of failure. The algorithm performs multi-criteria optimisation, in accordance with the 5 properties defined upstream. Among these, validity ensures that the counterfactual avoids failure and proximity ensures that the sequence of actions resulting from the agent and the counterfactual one are as similar as possible.

The objective of Tsirtsis et al. (2021) is to provide a counterfactual sequence that leads to a better outcome under the constraint that at most k actions are modified with respect to the agent's sequence. This work is carried out in a MDP context where the transition function is modelled by the Gumbel-Max structural model (Oberst and Sontag 2019). The algorithm proposes a polynomial-time solution based on dynamic programming. In the case where the state space is continuous, this problem is NP-hard (Tsirtsis and Rodriguez 2023). Tsirtsis and Rodriguez then proposed a search method based on the A* algorithm.

For a specific drone parcel delivery environment, Stefik et al (2021) use MSX (Erwig et al. 2018) (which is described later) to explain in natural language an action sequence in terms of risk factors. The risk is decomposed into different factors, so as to be able to understand what types of risk the agent is trying to avoid by performing a certain action. In this paper, the authors answer two questions related to the agent's action sequence: 'Why?' by comparing two sequences on the different risk factors and 'What?' by identifying in a sequence the action that leads to the greatest difference in risk.

B.2 Important Elements

The EDGE algorithm is a self-explainable model that predicts, for a given episode, the final agent's reward (W. Guo, X. Wu, et al. 2021). The provided explanation takes the form of a set of important time-steps within the agent's interaction episode with the environment. An explanation of a game of Atari 2600 Pong (Bellemare et al. 2013) is shown in Figure 13.

In a video-based environment, Liu et al. (2023) identify the critical states of a sequence sufficient to predict the final reward obtained by the agent. Two models are learned for this method: the return predictor, which predicts the agent's final reward based on a (partial) video (i.e. a sequence of frames), and the critical state detector, which identifies critical frames by learning to mask non-critical ones.

The History eXplanation based on Predicates (HXP) method (Saulières et al. 2023a) and its variant, named Backward-HXP (Saulières et al. 2024) are part of this line of work. The aim of these methods is to provide, within a sequence, the most important actions for the realisation of a predicate. This predicate is respected in the final state of the sequence studied, and may represent the success or failure of an agent, or other characteristics specific to the environment studied.

B.3 Human-readable MDP

B.3.1 Surrogate Model. Relying on a user-defined vocabulary, Sreedharan et al. (2022) present the idea of learning propositional concepts to explain the agent's action choices in a contrastive manner. With these propositional concepts, actions are represented in a symbolic way by preconditions, effects and their cost. The user proposes an alternative sequence of actions to the agent. The explanation shows that the user's sequence does not lead the agent to achieve the objective, or leads it into an invalid state, or describes that the proposed sequence is more costly than the agent's sequence.

Soni (2024) proposes to explain a set of transitions (or sequence) using a sequence explanatory message. Firstly, by collecting a set of data by interacting with various users, the method groups users into user-types and learns their associated labelling function. These functions allow to determine whether an explanation works for a user-type, and is therefore useful for providing personalised explanatory messages. Once this has been done, explanations can be generated for an unknown user. However, the user's type is not given. To provide personalised explanation, the problem consists in solving a POMDP where the hidden part of the state corresponds to the user's type.

B.3.2 Inherently Understandable. In a context of sequential reconstruction of magnetic resonance images, Li et al. (2020) use RL to successively apply interpretable pixel-wise operations. In this environment, the action space is directly interpretable because each action corresponds to common Computer Vision filters, such as the Sobel filter or the Gaussian filter.

C Summaries of Action-level Methods

This appendix provides a summary of each action-level method mentioned in the paper. These methods explain the agent's choice of action. As a reminder, the various methods have been grouped into two types of explanation: *Feature Importance*, which determines the features of the agent's state that led it to choose an action, and *Expected Outcomes*, which justifies the agent's decision by providing its future potential impact.

C.1 Feature Importance

C.1.1 Saliency Maps.

Attention Mechanism. In Mott et al. (2019), the current state of a Long Short Term Memory is used to make a set of requests, sent to an MLP. Its output is used, together with the agent state, to generate saliency maps. This system is called *attention head*. Experiments have shown that the agent has learned to focus its attention on regions/objects present in the image.

Based on A3C (Mnih, Badia, et al. 2016), Itaya et al. (2021) propose two attention masks: one for the policy and one for the state value. These attention masks are generated from the feature map extracted by the first part of

the neural network, called the feature extractor, and then used to predict the agent's action and estimate the value of the agent's state.

Two feature extractors are presented in Nikulin et al. (2019): Sparse FLS and Dense FLS (where FLS stands for Free Lunch Saliency). The main objective of this work is to provide an attention mechanism that makes both visualisations interoperable, without impacting the agent's performance. Both approaches and baselines were tested using Atari-head (R. Zhang, Walshe, et al. 2020), a dataset containing human gameplay and visual tracking. None of the approaches studied stood out from the others.

The Action Region Scoring (ARS) module proposed in Y. Liu et al. (2022) is used both to explain the choice of action and to improve learning. In the CNN, at the output of each convolutional layer to which a ReLU activation has been applied, an ARS module is used to identify the important regions and then combined with the current image representation. This helps learning by indicating the regions to focus on. For the explanation part, the saliency map is produced by retrieving the outputs of the ARS modules and combining them.

In an approach based on fuzzy rule learning, Ou et al. (2024) propose to use the Compact Convolutional Transformer (Hassani et al. 2021) to provide attention masks. In addition to this visualisation, the choice of agent is explained by providing the influence of each fuzzy rule.

Based on the U-net architecture (Ronneberger et al. 2015), SSINet takes the form of an encoder-decoder to obtain an attention mask of the input image (Shi et al. 2022). SSINet is built in such a way as to respect 2 properties: maximum behavior resemblance and minimum region retaining. The first describes that the agent's prediction must be consistent between the image and the image overlapped with the attention mask and the second describes that the attention mask must focus on as little information as possible, so as to provide sparse explanations.

Tang et al. (2020) only provide the agent with a subset of patches from the original image for its decision making. Indeed, the image is first cut into a set of patches, which are then evaluated using a self-attention module. Finally, the k most important patches are extracted from this evaluation and given to the agent. This technique makes it possible to understand on which patches of the original image the agent bases its decision.

In Josef and Degani (2020), different types of input are tested for an unmanned ground vehicle navigation problem. Several attention mechanisms are implemented depending on the features (visual or not). Thus, the self-attention mechanism used to explain the problem is applied to an image or a vector of features.

The DRIVE model (Bao et al. 2021) is proposed for a traffic accident anticipation problem using videos. Two attention maps are generated for the same frame f_t : a bottom-up approach which generates a map based directly on f_t and a top-down approach which applies a transformation to f_t before generating the map. This second approach focuses the attention mechanism on a risky region of f_t , using a foveal vision module based on a fixation point p predicted by the agent with the previous frame f_{t-1} .

Region-sensitive Rainbow (Z. Yang et al. 2018) is an enhancement to Rainbow DQN (Hessel et al. 2018) which uses a region-sensitive-module to determine the important regions in which the agent focuses to choose an action. This module is similar to an attention mechanism, and returns importance scores for k regions of an image. Of the three ways of visualising salient regions, the most user-friendly approach has been retained. It simply consists of a binary saliency map which displays just the salient region to the user (replacing the rest of the image with black pixels).

Focusing on the immediate reward, Yang et al. (2024) propose to generate attention masks that reflect as much as possible the agent's reward. The reward obtained by using the policy from state s is compared with the reward obtained by using it from state s' constructed by combining s and the attention mask. To minimise the absolute difference of these rewards, an RL approach is proposed. An extension of this method is provided to handle multi-step rewards.

The approach proposed by Wang et al. (2019) is different from the rest of the work. In addition to an attention map, an explanation in the form of natural language is produced. The architecture of the algorithm presented breaks down into three parts: an encoder, which encodes the image into a set of features, an attention mechanism,

which returns the salient features and the decoder, which generates a verbal explanation from the salient features. A total of four attention mechanisms have been proposed and compared with the BLEU metric (Papineni et al. 2002) for the textual explanation. The best of these is called *adaptive attention*, which consists of assigning a weight to the features dynamically using the last word generated by the verbal explanation.

Wang and Wang (2022) use an attention mechanism on non-visual inputs to understand the agent's decision choices. In the same idea, Zhang et al. (2021) present the Temporal-Adaptive Feature Attention algorithm. This identifies the most relevant features in the agent's choice of action. In MuJoCo (Todorov et al. 2012) environments, the weights of attention show that the features specific to the agent's position are more important than those linked to its velocity in the choice of action throughout the duration of an episode.

The approach proposed by Huber et al. (2019) does not require any additional module or specific architecture, but generates, as with the attention mechanisms, a saliency map. To do this, Layer-wise Relevance Propagation (Bach et al. 2015) is used: this is a general concept that identifies relevant pixels during the forward-pass by looking at the activations of each neuron. To limit the information to the most relevant, the authors use Mopuri et al. (2019): during the forward pass, neurons are monitored layer by layer, and only the neuron with the highest activation in each convolutional layer is selected. Relevance propagation is then restricted to the paths that connect these selected neurons across layers, allowing to filter out less relevant contributions and produce compact saliency maps.

Perturbation-based Approach. The perturbation method proposed by Greydanus et al. (2018) involves adding spatial uncertainty around the pixel at coordinates (i, j) . This is done by interpolating the original image with a Gaussian blur applied around (i, j) . In practice, instead of applying this perturbation for each pixel, it was applied per group of 5 pixels, which reduces the computational cost and produces good saliency maps. Using the A3C learning algorithm, saliency maps are generated and displayed jointly for the agent's policy and for its value function V (i.e. this captures the regions that are important for the agent's choice of action and evaluation of the image).

We have identified three works based on this method. Built on top of the agent implementation, Persiani and Hellström (2022) propose a new architecture, called the Mirror Agent Model, to make the agent's behavior interpretable by the user. Both the agent model and the user model are represented by Bayesian networks. A metric is presented to measure the distance between the two models. The explanatory layer added to the agent (which adds a node to the Bayesian network) consists of using saliency maps based on the method of Greydanus et al. This method is also used in S. Guo et al. (2021) to compare the saliency maps explaining the agent with the visual attention of humans. To do this, this study uses the Atari-head dataset (R. Zhang, Walshe, et al. 2020) and AGIL (R. Zhang, Z. Liu, et al. 2018), a model for predicting human visual attention. The authors found that, as learning progressed, the agent's attention maps became closer to human visual attention, which can also be used to understand agent failures. By modifying the method of Greydanus et al. to apply it for a 2D grid-world environment, Douglas et al. (2019) propose a display called Towers of Saliency, to interpret the agent's behavior over an entire episode.

Puri et al. (2020) construct saliency maps based on two properties: specificity and relevance. This method is called Specific and Relevant Feature Attribution (SARFA). The Specificity property defines that the salience of a feature must focus on the perturbation of the action to be explained, a . Thus, with s the base state and s' the perturbed state, the salience must be high if $Q(s, a) - Q(s', a)$ is substantially greater than $Q(s, a') - Q(s', a')$, $\forall a' \neq a$. The Relevance property defines that the salience of a feature should only have an impact on the Q value of a . In other words, the salience of a feature must be low if the perturbation also affects the Q values of the other actions.

Rather than directly using the Q function (Puri et al.) or the V function (Greydanus et al.) to measure the impact of a perturbation, Yan et al. (2023) propose using the Advantage function. Given a state s , an action a and

a policy π , the advantage of performing a from s and then following π is: $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. To locally perturb the image, a Gaussian blur is applied (as in Greydanus et al.).

Huber et al. (2022) evaluate a total of 5 perturbation-based approaches by inspecting the dependency of the methods on the NN parameters learned by the agent using the *sanity checks* metric (Adebayo et al. 2018) as well as their fidelity on the agent's reasoning using the *insertion* metric (Petsiuk et al. 2018). Among these approaches, there are the two aforementioned works: SARFA (Puri et al.) and Noise Sensitivity (Greydanus et al.). *Sanity checks* consist of successively randomising the layers of the neural network and calculating a saliency map each time. These saliency maps are then compared with the original, with the expected positive result being that the maps differ significantly from the original. All the methods depend on the learned parameters, but Noise Sensitivity shows little dependency. The *insertion* metric starts from a noisy image (using black occlusion and uniform random perturbation) and iteratively reconstructs the base image by resetting the correct pixel values, starting with the most salient pixels indicated by the saliency map. The expected positive result is that the agent's action prediction should quickly be similar to that of the agent on the noiseless image. This would mean that the saliency map has put forward the 'critical' pixels in the agent's choice of action. With this metric, SARFA is one of the two best approaches. The authors suggest further research into the generation of perturbation-based saliency maps, by first determining which type of perturbation might be appropriate for the task in hand.

The following papers use saliency maps to focus on objects rather than pixels. The work by Iyer et al. (2018; 2017) uses a computer vision technique called template matching to recognise objects in the image and use them as a basis for saliency maps. To provide the agent with more information for choosing an action, object channels are added to the base image. These object channels are extracted using the recognizer object and are used to determine the positions of the various objects detected in the image. The perturbation in the image s consists of masking an object, resulting in s_o and determining its impact on the choice of action by comparing the Q values. The maps generated highlight the objects that are salient in the agent's decision-making. A comparison between a 'classic' pixel saliency map and an object saliency map, based on a state-action pair from a game of Atari 2600 Ms PacMan (Bellemare et al. 2013), is shown in Figure 15. In Anderson et al. (2020), a user study was carried out comparing object saliency maps with reward decomposition (which is described later), a combination of the two and the control strategy, which simply displays the impact of the action on the state and the score obtained. In summary, participants understood the agent's behavior better when the explanation contained reward decomposition, although this may have led to cognitive overload for some participants.

Gradient-based Approach. Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al. 2017) is a method that can be used on CNN's without having to modify the architecture or re-train the model, in order to create saliency maps. To find out the influence of pixels on a class c for a given image, a forward pass is performed, the gradient of class c is calculated, then the signal is propagated with a backward pass. Importance weights of the feature maps (where a feature map is the output of a convolutional layer of a CNN) are calculated and then the saliency map is generated by applying a ReLU operation on the weighted linear combination of the feature maps. This method has been used in several works to explain RL agents (Dao, Huff et al. 2021; Joo and K. Kim 2019; Nie et al. 2019; Weitkamp et al. 2019).

Joo and K. Kim propose an architecture combining A3C (Mnih, Badia, et al. 2016) with Grad-CAM in order to visualise the salient parts of the image for the agent's choice of action. To describe the behavior of a swarm robotic system, Nie et al. combined it with a deconvolutive network to produce saliency maps. Weitkamp et al. use it by replacing ReLU activation with ELU activation. To reduce the number of important snapshots from the agent's learning phase, Dao, Huff, et al. use Grad-CAM. Similar states with common attentions are thus grouped together. The SmoothGrad method (Smilkov et al. 2017), which adds noise to the image so that the Grad-CAM saliency map is less noisy, was used. Note that in this paper, the method is not used to explain, but to make another explainable approach more sparse.

An in-depth analysis of an agent that has learned in the CoinRun environment (Cobbe et al. 2019) is proposed by Hilton et al. (2020) by analysing a hidden layer of the agent that has learned to recognise objects in CoinRun. The integrated gradient method (Sundararajan et al. 2017) is used to generate saliency maps.

In order to provide additional information for the visual analysis of a DQN, Zahavy et al. (2016) calculate the Jacobian of the DQN as a function of the input image.

He et al. (2021) propose an approach that does not use gradients, but works in a similar way to Grad-CAM. In addition to the saliency maps, textual explanations are provided. The agent uses an image and a feature vector to choose an action a . The explanation of a based on the image is produced by a new method using both CAM (B. Zhou et al. 2016) (on which Grad-CAM is based) and SHAP (Lundberg and S. Lee 2017). The textual explanation of a based on the vector is provided using SHAP. Note that both approaches are also used to provide global explanations of agent behavior.

C.1.2 Model-agnostic Approach.

SHAP. Most of the works focus on SHAP, using either Kernel SHAP (Carbone 2020; Løver et al. 2021; Remman, Strümke, et al. 2022; Rizzo et al. 2019; Y. Wang et al. 2020) or DeepSHAP (Liessner et al. 2021; Remman and Lekkas 2021; Schreiber et al. 2021; Theumer et al. 2022; K. Zhang et al. 2022).

For example, Rizzo et al. use Kernel SHAP for a traffic light control problem and Wang et al. for an automatic crane control problem. Two works (Carbone; Løver et al.) propose to compare SHAP with linear model trees (LMT), already presented in the policy explanation section. In Carbone, the author suggests that LMT and SHAP ‘capture similar relationships between input features’. Løver et al. also compare LIME and use a K-means summarizer to reduce the number of samples to be used for the calculation of SHAP values. Both studies show that LMT is the fastest at providing an explanation, and Løver et al. state that LIME is too slow to be used in real time. Kernel SHAP and Causal SHAP (Heskes et al. 2020) are compared in Remman, Strümke, et al. Causal SHAP is a variant of Kernel SHAP which computes SHAP values by taking into account the dependencies between the features of a state. The results highlight the benefits of using Causal SHAP.

DeepSHAP is used in K. Zhang et al. for a power system emergency control problem and in Theumer et al. where the explanation in a MARL context consists of an explanation of the decision of a single agent or the decisions of all agents. Remman and Lekkas use DeepSHAP to understand the impact of different variables in a robotic lever manipulation problem. Values are calculated and displayed for each agent’s decision over an entire episode. Based on the same use, Liessner et al. propose to calculate values with DeepSHAP in a longitudinal control task and Shreiber et al. in a traffic signal control task.

Within the IxDRL framework (Sequeira and Gervasio 2023), SHAP is used both to study the impact of features on the agent’s decision-making in a single state and also globally, by displaying the average impact of features on a set of states. In the same vein, Beechey et al. (2023) propose two methods for explaining locally (i.e. from a given state) and globally the agent’s performance, i.e. its expected return.

In addition to proposing a benchmark specific to the XRL domain, Xiong et al. (2024) present an algorithm entitled TabularSHAP. It is limited to explaining tabular states. To use it, the user first needs a set of interactions of the agent within the environment, which will be used to learn an ensemble tree model. Finally, TreeSHAP (Lundberg, Erion, et al. 2018), which is specific to tree-based models, is used on the basis of the ensemble tree model learned.

LIME. In order to analyse the behavior of an agent on a heating ventilation and air-conditioning control task, Kotevska et al. (2020) propose three modules: Model assessment, Model local view and Model global view. The first consists of a probabilistic analysis of the agent’s behavior and a statistical analysis of the impact of features on its behavior. The second involves using LIME for an agent decision and the third consists of using visualisation tools, e.g. a partial independence plot, to understand the impact of features on agent decisions. As already mentioned,

Dethise et al. (2019) uses LIME to explain a single agent's decision, or the agent's global behavior. LIME is also used in H. Li et al. (2022) to add an explanatory layer to an advanced persistent threats defense mechanism.

C.1.3 Counterfactual State. A body of work aims to provide an answer to the question ‘*Why does the agent perform action a from state s rather than action a' ?*’ by proposing a counterfactual state s' , close to s , in which the agent would have chosen a' .

A number of studies (Huber, Demmler, et al. 2023; Olson et al. 2021; Samadi et al. 2024) have used GAN (Goodfellow et al. 2014) for this purpose. As a reminder, this architecture consists of jointly learning a generative model G that creates data and a discriminative model D that determines whether the input data is generated or comes from the training dataset. After learning, only the generative model is used.

In order to provide a counterfactual explanation where the state is an image, Olson *et al.* propose to learn, in addition to these two models, an encoder which encodes the state s in order to omit information from s related to the action choice. D is trained to determine the action distribution based on the encoded state. G is trained to reconstruct s based on the action distribution and the encoded state. For training, it is necessary to get an initial dataset of state-action pairs using the agent to be explained. This method, tested on Atari2600 games, generates a state s' , close to s , in which the agent performs a different action.

In the same line, GANterfactual-RL (Huber, Demmler, et al.) is based on the StarGAN architecture (Choi et al. 2018). This helps G to construct states in which the agent chooses the desired counterfactual action. Examples of explanations are shown in Figure 16. With the help of a user study and metrics, GANterfactual-RL is more efficient and has better results in terms of metrics than the previous approach mentioned above (Olson et al.). Among these metrics, proximity and sparsity measure the distance of the generated counterfactual state s' from the base state s , and validity verifies that from s' , the action predicts the desired action a' .

Another competitive approach is proposed by Samadi et al. to generate counterfactual states using saliency maps. The proposed method, called SAFE-RL, is based on attentionGAN (H. Tang et al. 2023). G learns to generate a counterfactual state by taking as input the state, the associated saliency map and a counterfactual action. D learns to differentiate true states from counterfactuals.

With states also corresponding to images, Druce et al. (2021) present counterfactual states in their interface to help user comprehension. In this work, counterfactual states are obtained by pre-defined interventions on state s , such as the player's position in the image. For the same state, several counterfactual states are displayed to see how the agent would have acted according to certain modifications.

The RACCER algorithm (Gajcin and Dusparic 2024b) is used to identify a sequence of actions that leads to a counterfactual state in which the agent has a high probability of performing the desired action. This method is not image-specific, unlike the previous methods, and provides a recourse for the user to perform a certain action (with a certain probability). However, RACCER requires access to the environment dynamics.

C.1.4 Others. To use the method proposed by Davoodi and Komeili (2021), the features of the agent's state must be actionable. A feature is said to be actionable if the user knows which action or sequence of actions will increase/decrease the value of the feature. The aim of the method is to explain, for a given state s , the impact of features in reaching risky states. To do this, a transition graph T limited by a number of actions is constructed from s , then a linear model is learned on all the states of T to estimate the risk. The model weights describe the importance of the different features in reaching a risky state.

In Elizalde et al. (2009), the aim is to explain a recommended action in an MDP by returning the most relevant feature of the current state s , with respect to the value function V^π (which represents the expected reward of following π from s). To determine this feature, we need to measure the impact of the change in valuation of each feature of s on V^π . Each feature is tested separately, leaving the other features fixed. In addition to the relevant feature, a verbal explanation is proposed based on a knowledge-base dataset.

C.2 Expected Outcomes

C.2.1 State.

Causal Lens. In order to focus on causal relationship between action and state variables, Madumal et al. (2020b) build an Action Influence Model (AIM) which is a structural causal model (Halpern and Pearl 2001) with the addition of actions. This model takes the form of a graph which is used to answer ‘Why?’, ‘Why not?’ questions by providing a (partial) causal chain describing the impact of the action on the features of the model. An extension of this work (Madumal et al. 2020a) introduces a distal explanation model to provide causal chain explanations using a decision tree representing the agent’s policy. The shortcoming of this work is that the causal model is hand-crafted.

CauseOccam (Volodin 2021) is a model-based RL framework where the model learned is a sparse causal graph.

Herlau and Larsen (2022) also propose learning a sparse causal graph, which the agent uses during training. The search for relevant causal variables is carried out by maximising the ‘*natural indirect effect*’. These two approaches could serve as a basis for the construction of AIMs.

With the same idea, Yu et al. (2023) extend the use of AIM to problems with a continuous action space, without the need of prior knowledge of the environment causal structure. Indeed, to represent the dynamics of the environment, the causal model is learned based on the agent’s interactions with the environment. An example of a learned causal model for the LunarLander environment (Brockman et al. 2016) is shown in Figure 17.

Probability. The Memory-based eXplainable Reinforcement Learning approach (Cruz, Dazeley, and Vamplew 2019) provides the probability of success (i.e. reaching a goal state) and the number of transitions to reach the goal state. When the agent is trained, the success probabilities P and the number T of transitions leading to a goal state are calculated and updated for each state. The agent can then provide answers to the questions ‘Why does the agent choose action a ?’, ‘Why does the agent choose action a and not b ?’ or ‘What is the probability of reaching a goal state in 8 steps from s ?’. The problem with this approach is that it is necessary to memorise P and T in order to explain. To overcome this problem, Cruz, Dazeley, Vamplew, and Moreira (2023) propose 2 additional methods for calculating or approximating P : learning-based and introspection-based approaches. The first approach learns a P -table of the probability of success in parallel with learning the agent’s Q -table. The second requires no additional memory, as the probability of success is calculated directly from the Q -values.

In a Factored MDP context, Khan et al. (2009) introduce Minimal Sufficient Explanations (MSE) to provide a minimal explanation in terms of the number of templates sufficient to explain the agent’s action. Templates are phrases used to describe the probability of reaching a certain state (e.g. ‘[action] is likely to take you to [state description] about [λ] times’).

Dodson et al. (2011) present a system that provides an explanation of the optimal action via a natural language paragraph in a MDP context. One of the two explanation modules proposed is called Case-based explanation (CBE). CBE explains the usefulness of an action by using a database of previous transitions, called *cases*, to calculate statistics on the *cases* relevant to the recommended action. In terms of probabilities, this explanation makes it possible to explain the influence of an action on future feature values.

Contrastive. Stein (2021) provides counterfactual explanations based on the abstraction of sub-objectives. These sub-goals must be completed in order to achieve the agent’s goal. Thus, the user can ask why the agent prefers to perform one sub-goal rather than another. The answer contrasts the two sub-goals with the probability of achieving the final goal (i.e. a state) and the cost of achieving the sub-goal.

Tsuchiya et al. (2023) present a method similar to reward decomposition (which will be described in the next section), but focused on states. Additional neural networks are learned to estimate the Q -value of each state type. The different state types are provided in advance by the user. A contrastive explanation between the agent’s

choice of action and that of the user is performed by displaying and then comparing the different Q-values decomposed by state type.

Contrastive explanations are proposed using action-values based on given human-readable features (Z. Lin et al. 2021). To this end, deep generalized value functions (Sutton, Modayil, et al. 2011) are learned to predict future feature accumulation. To avoid information overload, the MSX approach (Juozapaitis et al. 2019) (described in the next section) is applied to obtain a clear comparison of the features impacted by the actions. Note that features can represent the state of the agent, but also components of the reward function.

The Moody framework (Barros et al. 2020) allows the agent to self-assess these decisions. Based on the agent's Q-values, its confidence of reaching the final state is calculated (Cruz, Dazeley, Vamplew, and Moreira), then converted into a pleasure/arousal scale (Costa Jr and McCrae 1996) consisting of two dimensions: pleasing/unpleasing and excited/calm. Finally, a Growing-When-Required Network (Marsland et al. 2002) is used to define the agent's current mood. Note that this method can also be used by an agent to evaluate the mood of other agents.

Others. Interactive RL (Thomaz et al. 2005) is a framework in which the agent's learning is accelerated by instructions given by an expert. In this context, the works by Fukuchi et al. (2017a,b) develop the Instruction-based Behavior Explanation (IBE) framework, which reuses expert instructions to explain the agent's choice of action a_t from state s_t . To do this, IBE estimates $\delta(s_t)$ which represents the change in value of the features between state s_t and the state reached n steps later s_{t+n} . Then, an expert expression, which is a user interpretable signal, is associated to $\delta(s_t)$, either by a clustering method (Fukuchi et al. 2017a), or by classification using a NN (Fukuchi et al. 2017b).

With a state represented by an image, Pan et al. (2019) propose to predict the semantics of pixels in future states. Image semantic segmentation consists of labelling each pixel of an image. It is used here in a vehicle driving context, where each pixel can be labelled as *road*, *grass*, *sky*, ... The agent learns to predict this visual representation of the future state, which acts as an explanation.

C.2.2 Reward. The vast majority of works uses the principle of reward decomposition: Amitai, Septon et al. 2024; Anderson et al. 2020; Erwig et al. 2018; Feit et al. 2022; Iucci et al. 2021; Juozapaitis et al. 2019; Rietz et al. 2023; Saldiran et al. 2024.

As a reminder, reward decomposition is used to represent the reward function in an interpretable way: it is decomposed into a set of comprehensible reward types. Thus, to understand the agent's choice of action, it is sufficient to look at the type of reward it seeks to maximise. Based on this method, (Erwig et al.; Juozapaitis et al.) propose Reward Difference eXplanations (RDX) and Minimal Sufficient eXplanation (MSX) to explain deep adaptive programs learned via RL (Erwig et al.) and agents from the CliffWorld and LunarLander domains (Juozapaitis et al.). RDX simply compares 2 actions from a state by comparing their Q-values for each type of reward. An example of RDX, from Iucci et al., is shown in Figure 18. In accordance with RDX, MSX provides a more compact explanation of an agent's preference for action a over another action a' : only the reward types that have the greatest impact (positively and negatively) on the preference of a over a' are displayed to the user.

Several works combine reward decomposition (by simply displaying the different Q-values or by using RDX, MSX) with other methods. Of these, most combine it with approaches that explain the agent's policy. Saldiran et al. use a global reward decomposition and a heatmap visualisation of Q-values. Iucci et al. use autonomous policy explanation (Hayes and Shah 2017), Feit et al. the Interestingness Elements framework (Sequeira and Gervasio 2020) and Rietz et al. the HRL setting. Anderson et al. employ object perturbation-based saliency maps, a complementary explanation of the agent's choice of action. Amitai, Septon, et al. present a counterfactual explanation using reward decomposition.

A new estimator proposed in Towers et al. (2024) allows the agent to predict the n future expected rewards from a given state. This new estimator makes it possible to explain the agent's choice of action from a given state through three questions: 'What rewards to expect and When?', 'What observation features are important?' and

‘What is the impact of an action choice?’. The answers to these questions from a state s are respectively a plot of the n future expected rewards, a use of Grad-CAM (Selvaraju et al. 2017) to understand the important features for the different future expected rewards and a contrastive explanation by displaying the n future expected rewards for two different actions performed from s .

In a human-machine cooperation context, Kampik et al. (2019) describe different types of goals that the agent could communicate to explain a sympathetic action, i.e. an action that helps the user at the agent’s expense. Although not for RL, this type of explanation could be useful in a cooperative MARL context, where agents would explain their sympathetic action by describing the expected outcome in terms of reward.

C.2.3 Sequence. A belief map is learned together with the agent’s Q-values, to give the user the states that the agent is trying to reach by doing a from s (Yau et al. 2020). This belief map provides a local representation of the agent’s intentions. In the Taxi problem, these expected states form a path, as shown in Figure 19.

The CoViz algorithm (Amitai, Septon, et al. 2024), already presented in this survey with its policy summary variant, proposes to display to the user two sequences from a state s , one in which the action a chosen by the agent is performed, and one in which a different action a' is performed. This enables the user to compare the sequences. In addition, reward decomposition is used.

In Rouveirol et al. (2023), the set of possible trajectories from a pair (s,a) is divided into different groups that can be explained in the same way, by training a rule-based classifier. Subsequently, a minimal common explanation is built for each group, which takes the form of an existentially quantified conjunction of literals. In a MDP context, these explanations describe a partial plan made up of actions and/or state features to reach a winning state.

The Scenario eXplanation (SXp) method (Saulières et al. 2023b) falls into this category of work. Its objective is to summarise all possible sequences starting with action a from state s , by simply providing 3 representative sequences: the best-case, worst-case and most probable scenario.

Luss et al. (2023) present a method for clustering the state space locally and extracting a set of critical states. Starting from a state s , the states to cluster are obtained using the stochastic policy for x steps. Reachable states are grouped into meta-states and strategic states are identified. These states are identified as sub-goals. With the help of visualisation, these sub-goals and clusters are used to understand the agent’s future behavior.

C.2.4 Action. Only one work explains the choice of an action by indicating future actions that could be taken. The work (Dodson et al. 2011) is closely related to the problem studied and so future work could draw on it to explain RL agents (provided that the MDP is known or at least approximated).

Dodson et al. provide a natural language explanation for advising students on their choice of courses. This problem, represented by a MDP, is explained using two modules, one of which is called Model-Based Explanation. This module extracts information from the MDP about the usefulness of performing the recommended action in terms of future actions. As an example, if the student takes course x , this will enable him to take courses y, z in the future.

The following four works use both types of explanation of action (i.e. *Feature Importance* and *Expected Outcomes*).

The PeCoX framework (Neerincx et al. 2018) proposes two types of explanation based on the perceptual and cognitive aspects of the agent. PeCoX models the explanation problem in three parts: explanation generation, communication to the user and reception. The perceptual explanation comprises the calculation of a confidence index allowing the agent to express its confidence in its decision and a contrastive method that identifies an alternative output for comparison. The cognitive explanation consists of providing a restricted set of goals and beliefs for the agent. The authors also suggest modelling emotions to enrich the explanation. Note that the way in which the agent has learned and the explanation methods are not specified in this work, the idea being to provide a general framework for explaining agent behavior.

An approach is presented in N. Wang et al. (2016) for directly extracting explanations based on the POMDP. The idea is to retrieve the various types of information from the POMDP model and translate them into natural language sentences using predefined templates. For example, the agent can use its expected reward outcome to justify its choice of action, the transition function to describe the probability of achieving different outcomes, or its observation to clarify it to the user.

The method used in the work by Ehsan et al. (2018; 2019) consists of collecting a training corpus with the help of users so that the agent can explain its choice of action ‘as a human would have done’. So the first step is to collect explanations associated with state-action pairs. Users have to ‘think aloud’ while performing the agent’s task. The second step is to train an encoder-decoder to return, based on a state-action pair, an explanation of the choice of action from the state, in a natural language form.

RL as an Explanation Framework. A body of work uses RL to generate counterfactual inputs to a model that are close to the original input, to answer the question ‘*Why predict a rather than b from instance x ?*’ In Z. Chen, et al. (2022), the problem of generating a counterfactual instance of a classification or regression model is described as a decision-making problem, which is solved using a DRL algorithm. Based on the same idea, Samoilescu et al. (2021) generates batches of counterfactuals from the instance to be explained. For the Drug Target prediction problem, a multi-agent framework is proposed in Nguyen et al. (2021) to generate counterfactuals by taking into account two distinct inputs, the drug and the target, which are assimilated to two agents with distinct actions. To explain the classification of a point, Lash (2022) describes an approach using DRL to find the closest point to the classified point which is located in the decision boundary, based on the user’s set of preferred features. The use of two RL agents allows the question ‘*Why an item is recommended?*’ to be answered by a personalised explanation (X. Wang, Y. Chen, et al. 2018). To explain the choice of the recommendation system f , agent 1 learns to provide an explanation consisting of a subset of items and agent 2 learns to use this to predict the output ratings: the explanation is a subset of items sufficient to provide the same output ratings as f . Kohler et al. (2023) use Iterative Bounding MDP (IBMDP) (Topin, Milani, et al. 2021) to build compact and efficient DT’s for classification tasks. In addition to a classical MDP, IBMDP includes feature bounds, information gathering actions and a reward function that defines the interpretability-performance trade-off. The SXp method (Saulières et al. 2023b) learns favorable and hostile agents representing the response of the environment with RL, in order to explain an RL agent.

Received 25 September 2025; accepted 02 August 2026