Interestingness Elements for Explainable Reinforcement Learning through Introspection

Pedro Sequeira
SRI International
Menlo Park, CA
pedro.sequeira@sri.com

Eric Yeh
SRI International
Menlo Park, CA
eric.yeh@sri.com

Melinda Gervasio
SRI International
Menlo Park, CA
melinda.gervasio@sri.com

ABSTRACT

We propose a framework toward more explainable reinforcement learning (RL) agents. The framework uses introspective analysis of an agent's history of interaction with its environment to extract several interestingness elements regarding its behavior. Introspection operates at three distinct levels, first analyzing characteristics of the task that the agent has to solve, then the behavior of the agent while interacting with the environment, and finally by performing a meta-analysis combining information gathered at the lower levels. The analyses rely on data that is already collected by standard RL algorithms. We propose that additional statistical data can easily be collected by a RL agent while learning that helps extract more meaningful aspects. We provide insights on how an explanation framework can leverage the elements generated through introspection. Namely, they can help convey learned strategies to a human user, justify the agent's decisions in relevant situations, denote its learned preferences and goals, and identify circumstances in which advice from the user might be needed.

CCS CONCEPTS

• **Computing methodologies** → *Artificial intelligence*; **Intelligent agents**; **Sequential decision making**; • **Human-centered computing** → *Human computer interaction (HCI)*; *Collaborative interaction*.

KEYWORDS

Explainable AI; Reinforcement Learning; Autonomy; MDP.

ACM Reference Format:
Pedro Sequeira, Eric Yeh, and Melinda Gervasio. 2019. Interestingness Elements for Explainable Reinforcement Learning through Introspection. In *Joint Proceedings of the ACM IUI 2019 Workshops, Los Angeles, USA, March 20, 2019,* 7 pages.

1 INTRODUCTION

Reinforcement learning (RL) is a very popular computational approach to address problems of autonomous agents facing a sequential decision problem in a dynamic and often times uncertain environment [19]. The goal of any RL algorithm is to learn a *policy*, *i.e.*, a mapping from states to actions, given trial-and-error interactions between the agent and an environment. Typical approaches within RL focus on developing memoryless (reactive) agents, *i.e.*, that select their actions based solely on their current observation [13]. This means that by the end of learning, a RL agent is capable of selecting the most appropriate action in each situation it faces—the learned policy ensures that by doing so, the agent will maximize the reward received during its lifespan, thereby performing according to the underlying task assigned by its designer.

On one hand, RL agents do not need to plan or reason about their future in order to select actions. On the other hand, this means that it is hard for them to come up with explanations for their behavior—all they know is that they *should* perform one action in any given situation, in the case of deterministic policies, or select an action according to a probability distribution over actions, in the case of stochastic policies. Importantly, the "why" behind decision-making is lost during the learning process as the policy *converges* to an optimal action-selection mechanism. At most, agents know that choosing some action is preferable over others, or that some actions have a higher value associated with them. Another key insight of RL that complicates explainability is that it allows an agent to learn from *delayed rewards* [19]—the reward received after executing some action should be "propagated" back to the states and actions leading to that situation, meaning that important actions might not have associated any (positive) reward.

Ultimately, RL agents lack the ability of knowing why some actions are preferable over others, of identifying the goals that they are currently pursuing, of making sense of the dynamics of the environment or recognizing what elements are more desirable, of identifying situations that are "hard to learn", or of summarizing the strategy learned to solve the task. Without such an explanation mechanism, autonomous RL agents may suffer from lack of trust by end users of the system or human collaborators that wish to delegate on them critical tasks.

1.1 Approach Overview

In this paper, we present a framework towards making autonomous RL agents more explainable. The framework is illustrated in Fig. 1 and relies on introspective analysis. In particular, the agent examines its history of interaction with the environment to extract *interestingness elements* denoting meaningful situations, *i.e.*, potentially "interesting" characteristics of the interaction. Each element is generated by applying statistical analysis and machine learning methods to data collected during the interaction. In that regard, we argue that there is a great amount of information that is disregarded by typical RL algorithms that can be of most importance to understand the behavior of the agent and address some the aforementioned problems. Some of such information is already collected by standard algorithms but used only for updates of the policy or

IUI Workshops'19, March 20, 2019, Los Angeles, USA
© 2019 Copyright for the individual papers by the papers' authors. Copying permitted for private and academic purposes. This volume is published and copyrighted by its editors.

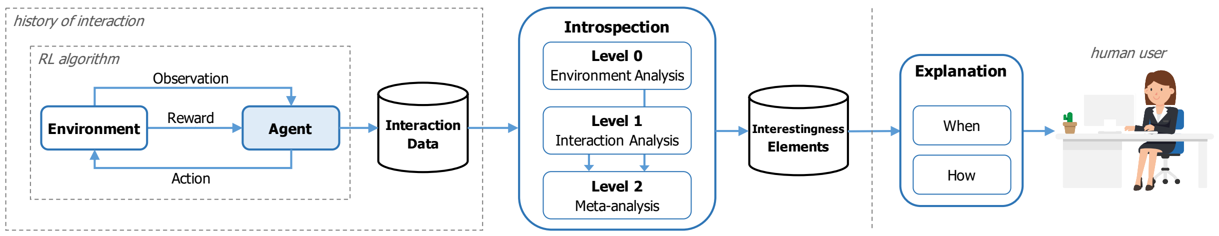

Figure 1: Our framework for explainable autonomous RL agents. During its interaction with the environment, the agent collects relevant statistical information. If the agent is learning, it updates its RL algorithm. Our introspection framework analyzes the collected data and identifies interestingness elements of the history of interaction. These elements can be used by an explanation framework to expose the agent's behavior to a human user.

value function, while other data can easily be collected and updated by the agent during learning. As depicted in Fig. 1, the introspection framework can be the basis of an explanation system operating on top of the interestingness elements, selecting the ones that better help explain the agent's behavior.

There are a few characteristics of the proposed introspection framework that are worth noting. First, the framework is *domain-independent*, meaning that the data that is collected and analyzed is agnostic to the specific learning scenario. Second, the framework is also *algorithm-independent* in the sense that it can be used in conjunction with standard RL tabular methods without having to modify the learning mechanism itself. Furthermore, the framework makes no assumptions with regards to optimality of the observed behavior—it captures important aspects specific to *some* history of interaction, independently of whether the agent was exploring the environment, exploiting its knowledge after learning, or simply acting randomly. Moreover, the framework is suitable to be used at different times and for different explanation modes, namely: *during learning*, by tracking the agent's learning progress and acquired preferences; *after learning*, by summarizing the most relevant aspects of the interaction; *passively*, where a user can query the agent regarding its current goals or ask it to justify its behavior at any given situation; *proactively*, where the agent requests input from the user in situations where its decision-making is more uncertain or unpredictable.

1.2 Reinforcement Learning

RL agents can be modeled using the *partially observable Markov decision process* (POMDP) framework [10], denoted as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{Z}, \mathrm{P}, \mathrm{O}, \mathrm{R}, \gamma)$. At each time step $t = 0, 1, 2, \ldots$, the environment is in some state $S_t = s \in \mathcal{S}$. The agent selects some action $A_t = a \in \mathcal{A}$ and the environment transitions to state $S_{t+1} = s' \in \mathcal{S}$ with probability $\mathrm{P}(s' \mid s, a)$. The agent receives a reward $\mathrm{R}(s, a) = r \in \mathbb{R}$ and makes an observation $Z_{t+1} = z \in \mathcal{Z}$ with probability $\mathrm{O}(z \mid s', a)$, and the process repeats.

In this paper, we deal with the problem of explainability in situations where the agent may have limited sensing capabilities, *i.e.*, the environment may be *partially-observable*. Notwithstanding, as in typical RL scenarios, we assume that the agent's observations are sufficient for it to solve the intended task, in which case $\mathcal{Z}$ is treated *as if it were* $\mathcal{S}$, and O is discarded. The simplified model thus obtained, represented as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathrm{P}, \mathrm{R}, \gamma)$, is referred to as a *Markov decision process* (MDP) [16].

The goal of the agent can be formalized as that of gathering as much reward as possible throughout its lifespan discounted by γ. This corresponds to maximizing the value $v = \mathbb{E}\left[\sum_t \gamma^t r\right]$. To that end, the agent must learn a policy, denoted by $\pi : \mathcal{Z} \to \mathcal{A}$, that maps each observation $z \in \mathcal{Z}$ directly to an action $\pi(z) \in \mathcal{A}$. In the case of MDPs, this corresponds to learning a policy $\pi^* : \mathcal{S} \to \mathcal{A}$ referred to as the *optimal policy* maximizing the value v.

We focus on *value-based* RL, where a function $Q^* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ associated with π^* verifies the recursive relation $Q^*(s, a) = r + \gamma \sum_{s' \in \mathcal{S}} \mathrm{P}(s' \mid s, a) \max_{b \in \mathcal{A}} Q^*(s', b)$. $Q^*(s, a)$ represents the *value* of executing action a in state s and henceforth following the optimal policy. Standard RL algorithms like Q-learning [22] assume that the agent has no knowledge of either P or R. Hence, they typically start by *exploring* the environment—by selecting actions in some exploratory manner—collecting samples in the form (s, a, r, s') which are then used to successively approximate Q^* using the above recursion. After exploring, the agent can *exploit* its knowledge and select the actions that maximize (its estimate of) Q^*.

2 INTROSPECTION FRAMEWORK

2.1 Interaction Data

As depicted in Fig. 1, the introspection framework relies on interaction data that is collected by the agent during its interaction with the environment. Namely, the analyses rely on the following data:

- $n(z)$, $n(z, a)$ and $n(z, a, z')$, which respectively denote the number of times that some observation z, observation-action pair (z, a) and transition (z, a, z') were observed by the agent while interacting with the environment;
- $\hat{\mathrm{P}}(z' \mid z, a)$ corresponds to the *estimated probability* of observing z' when executing action a after observing z. This can be estimated by using $n(z, a, z')/n(z, a)$;
- $\hat{\mathrm{R}}(z, a)$ is the agent's *estimate of the reward* received for performing action a after observing z. This can be estimated by maintaining a running average of the rewards received;
- $Q(z, a)$ is the agent's *estimate of the Q function*. It corresponds to the expected value of executing a having observed z and henceforth following the current policy. It can be estimated by using any value-based RL algorithm;
- $\widehat{\Delta Q}(z, a)$ is the expected *prediction (Bellman) error* associated with $Q(z, a)$. For a transition (z, a, r, z'), the prediction error corresponds to $\Delta Q(z, a) = r + \gamma \max_{b \in \mathcal{A}} Q(z', b) - Q(z, a)$. As such, the agent can maintain a running average of the prediction errors after each visit to (z, a).

As we can see, some of the data is already collected by value-based RL methods, namely the Q function, and by model-based algorithms, namely $\hat{\mathrm{P}}$ and $\hat{\mathrm{R}}$. The remaining data can easily be collected by the agent during its interaction with the environment by updating counters and running averages.

In addition, as is the case with many RL scenarios, let us assume that, at each time-step t, the agent observes its environment through a finite set of *features* $Z_t^i = z^i$, $i = 1, \ldots, N$, each taking values in some feature space $\mathcal{Z}^i$. The observation-space thus corresponds to the cartesian product $\mathcal{Z} = \mathcal{Z}_1 \times \ldots \times \mathcal{Z}_N$. When this is the case, the structure exhibited by such *factored MDPs* can also be exploited to derive interesting aspects related to specific observation elements.

All these data are used by the agent as sources of information for the several introspection analyses. Table 1 lists the analyses implemented so far, how they are group at different levels, and the elements that each generates.[1] The idea is to take the most out of the data collected during the interaction to highlight information that helps explain the agent's behavior.

2.2 Level 0: Analyzing the Environment

In this level we analyze characteristics of the task or problem that the agent has to solve.

2.2.1 Transition Analysis. The estimated transition probability function $\hat{\mathrm{P}}(z' \mid z, a)$ can be used to expose the environment's dynamics. Namely, it allows the identification of (un)certain transitions. Given an observation z and an action a, the transition certainty associated with (z, a) is measured according to how concentrated the observations $z' \in \mathcal{Z}$ following (z, a) are. In particular, we use the *evenness* of the distribution over observations following (z, a) as the normalized true diversity (information entropy) [15] by resorting to the probabilities stored in $\hat{\mathrm{P}}$.

Certain/uncertain transitions: denote situations in which the next state is hard/easy to be predicted by the agent.

Certain/uncertain observation-features: are values of observation features that are active, on average, whenever certain/uncertain observation-action pairs occur.

Transitions leading to many different states—according to a given threshold—have a high evenness and are considered *uncertain*. Likewise, transitions leading only to a few states have a low evenness and are considered *certain*. This analysis thus highlights the (un)certain elements of the agent's transitions, actions and of its observation features. Uncertain elements are especially important as people tend to resort to "abnormal" situations for the explanation of behavior [14]. This information can also be used by the agent in a more proactive manner while interacting with the environment. For example, the agent can express its confidence in the result of its actions when necessary, or request the help of a human user when it faces a very uncertain situation.

2.2.2 Reward Analysis. The idea of this analysis is to identify uncommon situations regarding the reward received by the agent during its interaction with the environment, namely:

Observation-action reward outliers: (z, a) pairs in which, on average among all other states and actions, the agent received significantly more/less reward.[2] This information may be used to identify situations in which the agent is likely to receive relatively low or high rewards.

Feature-action reward outliers: correspond to feature-action pairs that are, on average among all observations taken, significantly more or less rewarding than other pairs. The rationale is that, in typical RL scenarios, the agent designer defines rewards to be provided to the agent when it interacts with elements of the environment in a (in)correct manner, *e.g.*, the agent may have to interact with the appropriate object to achieve some subgoal. However, the reward from executing some action after making some observation is "diluted" among all the features that were active at that time. Therefore, this element may be used to denote significant individual contributions of features to the agent's reward.

2.3 Level 1: Analyzing the Interaction with the Environment

The purpose of this level is to help characterize the environment's dynamics and extract important aspects of the agent's behavior and history of interaction with it.

2.3.1 Observation Frequency Analysis. This analysis identifies interestingness elements that can be found given information stored in the counters n, namely:

Observation coverage: corresponds to how much of the observation-space—regarding all possible combinations between the observation features—were actually observed by the agent. This information may provide an indication of how much of the state-space was covered by the agent's behavior, which is an important quality of its exploration strategy.

Observation evenness: corresponds to how even the distribution of visits to the observation space was. In particular, it analyzes the histogram of observations using the aforementioned distribution evenness metric. It can be used to infer how unbalanced the visits to states were, which in turn may denote how interesting the dynamics of the environment are, *e.g.*, denoting situations that are physically impossible to occur, and how exploratory the agent was during the interaction with it.

Frequent/infrequent observations: these correspond to observations that appeared less/more frequently than others during the interaction. This element can assess the agent's (in)experience with its environment, denoting not only common situations it encounters but also rare interactions. Importantly, the latter may indicate states that were not sufficiently explored by the agent, *e.g.*, locations that are hard to reach in a maze or encounters with elements that are scarce, or situations that had such a negative impact on the agent's performance that its action-selection and learning mechanisms made sure they were rarely visited, such as a death situation in a game.

Strongly/weakly-associated feature-sets: are sets of observation features (feature-sets) that frequently/rarely co-occur. To that end, we resorted to frequent pattern-mining (FPM) [1], a data-mining technique to find patterns of items in a set of transactions. In this case, each observation corresponds to a transaction containing the features that are active in that observation. We then

[1]Due to space restrictions, only some elements are detailed here.

[2]In our analyses, we detect outliers based on whether the distance to the mean is higher than a certain number of standard deviations.

Table 1: Overview of the analyses of the introspection framework and the interestingness elements that each generates.

Level	Purpose	Analysis	Generated Interestingness Elements
0. Task	*analyze characteristics of the task (POMDP)*	Transition	(Un)Certain transitions; (Un)Certain actions; (Un)certain features
		Reward	Mean reward; Obs.-action reward outliers; Actions mean reward; Feature-action reward outliers
1. Interaction	*analyze agent's history of interaction with the environment*	Obs. Freq. and Recency	Obs. coverage; Obs. Dispersion; (In)Freq. obs.; Strongly/weakly-associated feature-sets; Associative feature-rules; Earlier obs. and actions
		Obs.-Action Freq.	Obs.-action coverage; Obs.-action evenness; (Un)Certain obs. and features
		Value	Mean value; Obs.-action value outliers; Feature-action value outliers; Mean pred. error; Obs.-action pred. outliers; Actions mean value and pred. error
2. Meta-analysis	*combine elements generated by the different analyses*	Transition Value	Local minima/maxima; Absolute minima/maxima; Maximal strict-difference obs.; Obs. variance outliers
		Sequence	Uncertain-future obs.; Certain sequences to subgoal
		Contradiction	Contradictory-value obs.; Contradictory-count obs.; Contradictory-goal obs.; Contradictory feature-actions

created a frequent-pattern tree (FP-tree) using the algorithm in [7] that facilitates the systematic discovery of frequent combinations between the items in the data-base.

Typical FPM techniques rely on the *relative frequency* of co-occurrences of items to judge whether a certain item-set is considered a pattern or not. However, other metrics exist that allow the discovery of more meaningful patterns—for example, if two features are always observed together by the agent, the pair should be consider interesting even if their relative frequency (*i.e.*, compared to all other observations) is low. Therefore, we use the *Jaccard index* [9], which can be used to measure the association strength of item-sets [17, 18]. We then use an algorithm based on FP-Growth [7] to retrieve all observation feature-sets that have a Jaccard index above/below a given threshold, in which case they are considered to be strongly-/weakly-associated.

This element may be used to denote both patterns in the agent's perceptions (or regularities in its environment), and also rare on inexistent combinations of features. In turn, these aspects may be important to explain the agent's physical interaction with the environment and expose its perceptual limitations to an external observer.

Associative feature-rules: these are rules generated in the form *antecedent* $\Rightarrow$ *consequent*. The idea is to determine sets of features–the antecedent—that frequently appear conditioned on the appearance of another set of features—the consequent. We used the lift statistical measure [1] to determine the *confidence* of every possible rule given the strongly-associated feature-sets. This element can be used to determine causal relationships in the environment, *e.g.*, the physical rules of the environment or the co-appearance of certain objects, which are important elements of explanation [14].

2.3.2 Observation-Action Frequency Analysis. This analysis produces the following elements:

Observation-action coverage: corresponds to how much of the actions were executed in the observations made. Similarly to the observation coverage, this element reveals how exploratory the interaction with the environment was.

Observation-action dispersion: corresponds to the mean evenness of action executions per observation. It can be used to determine how (un)balanced the selection of actions in certain observations were. In turn, this may denote either how exploratory the agent was during the interaction, or how stochastic the agent's policy is.

Certain/uncertain observations and features: besides transition certainty, we can calculate how certain or uncertain each observation is with regards to action execution. Observations where many different actions have a high count (high evenness) are considered uncertain, while those in which only a few actions were selected are considered certain. This may denote situations in which the agent is (un)certain of what to do, therefore providing good opportunities to ask for a human user for intervention. Similarly, we can identify features denoting situations in which, on average, action selection is very even/uneven. Uneven features may be particularly useful to abstract action-execution rules, *i.e.*, actions that are very likely to be executed whenever some feature is active.

2.3.3 Value Analysis. This analysis uses information stored in Q, V and $\widehat{\Delta Q}$ to generate the following interestingness elements:

Observation-action value outliers: correspond to (z, a) pairs that are significantly more or less valued. This element denotes desirable situations with regards to the agent's goals—high-value pairs indicate situations conducive for the agent to attain its goals while low-valued situations might prevent the agent of fulfilling its task.

Mean prediction error: the mean prediction error among all states and actions. This evaluates the accuracy of the agent's world model, *i.e.*, how well can the agent predict the consequences and future value of its actions in most situations.

Observation-action prediction outliers: correspond to the (z, a) pairs that have associated a significantly higher/lower mean prediction error. These are situations that are very hard (or easy) for the agent to learn. Together with transition uncertainty, this is an important element for explanation as people use social attribution to determine causes and judge others' behavior [14]—*e.g.*, when the next observation and reward received are very stochastic, the agent's future might be very unpredictable and uncertain. Therefore, in such situations the agent should inform the user to avoid misunderstandings.

2.4 Level 2: Meta-Analysis

This level refers to analyses combining information from the different interaction data and the previous levels.

2.4.1 Transition Value Analysis. This analysis combines information from the agent's estimated V function and the transition function $\hat{P}(z' \mid z, a)$. The goal is to analyze how the value attributed to some observation changes with regards to possible observations taken at the next time-step. The following elements are produced:
Local minima/maxima: refers to observations whose values are greater/lower than or equal to the values of all possible next observations. These help explain the desirability attributed by the agent to a given situation. Specifically, local maxima denote subgoals or acquired preferences—*e.g.*, this may help explain situations in which the agent prefers to remain in the same state rather than explore the surrounding environment. In contrast, local minima denote highly-undesirable situations that the agent will want to avoid and in which typically any action leading to a different state is preferable.
Observation variance outliers: correspond to observations where the variance of the difference in value to possible next observations is significantly higher or lower. This element is important to identify highly-unpredictable and especially risky situations, *i.e.*, in which executing actions might lead to either lower- or higher-valued next states.

2.4.2 Sequence Analysis. This analysis combines information from the analyses of observation frequencies, transitions and values. The goal is to extract common and relevant sequences of actions from to the agent's interaction with its environment. In particular, interesting sequences involve starting from important observations identified by the other analyses, then executing the most likely action, and henceforth performing actions until reaching a local maximum. To discover sequences between observations we first used the information stored in $\hat{P}$ to create a graph where nodes are observations and edges are the actions denoting the observed transitions. We then implemented a variant of Dijkstra's algorithm [4] to determine the most likely paths between a given *source* observation and a set of possible *target* observations. Using this procedure, this analysis generates the following interestingness elements:
Uncertain-future observations: correspond to observations, taken from the local minima, maxima, variance outliers, frequent, and transition-uncertain sets of observations, from which a sequence to any local maxima (subgoal) is very unlikely. These enable the identification of very uncertain situations—where the agent is not able to reason about how to reach a better situation—and hence in which help from a human user might be needed.
Certain sequences to subgoal: these denote likely sequences starting from an observation in the same set of sources used for the previous element, and then performing actions until reaching a subgoal. This element determines the agent's typical or most likely actions when in relevant situations. Thus, it can be used to summarize its behavior in the environment, *e.g.*, by distilling a visual representation from the graph or visualizing a sequence of observations. The sequence-finding procedure can also be used by the user to query the agent about its future goals and behavior in *any* possible situation. Notably, this can be used to provide contrastive explanations which help reasoning about why the alternatives to some actions—the foils—are not as desirable as those chosen by the agent [14]. Also, starting points may denote *reasons* for behavior while the combined transition likelihoods denote the agent's *beliefs*—these are two crucial elements commonly used by people to explain intentional events [3].

2.4.3 Contradiction Analysis. This analysis combines information from the value, reward and frequency functions, the value analysis and provided domain knowledge. The goal is to identify *unexpected* situations, where the agent was expected to behave in a certain manner, but the collected data informs us otherwise. Hence, we automatically determine the foils for behavior in specific situations.
Contradictory-value observations: correspond to observations in which the actions' values distribution proportionally diverges from that of their rewards. Specifically, we use the Jensen-Shannon divergence (JSD) [5] that measures how (dis)similar two probability distributions are with regards to the proportion attributed to each element. By using this technique and resorting to the information stored in Q and $\hat{R}$, we can identify situations with a *value-reward* JSD higher than a given threshold. In such situations, the agent may select actions contradicting what an external observer would expect. Further, we analyze the individual components of the JSD to identify which indexes are responsible for the non-alignment or dissimilarity between the distributions. In this manner, the agent can automatically detect the contradictory situations and justify why it chose an unexpected action, *e.g.*, explaining that it leads to a certain subgoal and is thus a better option compared to the expected action (contrastive explanation).
Contradictory-count observations: similarly, we can identify observations in which the actions' selection distribution diverges from that of their values. We use the same technique as above to calculate the *count-value* JSD by using the data stored in n and Q. This element can identify situations where the agent's action-selection mechanism contradicts what it has learned, *e.g.*, by selecting more often actions in situations in which they have lower values. In turn, this could indicate one of several things: an inadequate action-selection mechanism; an unstable and hard-to-learn situation, where the value of an action changed throughout learning; that the agent has acquired a preference for an action but selected other actions with the same frequency during learning, which means that the agent has not started exploiting its learned knowledge.
Contradictory-goal observations: to identify these elements, we assume that the system is provided with domain-knowledge regarding goal states. These correspond to situations that would normally be considered as highly-desirable for the agent to perform the task by an external observer, *e.g.*, collecting relevant items from the environment, reaching a new level in a game, etc. Based on this information, we determine which observations that were found to be subgoals for the agent (local maxima) are not in the known list

of goals. The idea is to identify *surprising* situations in which the agent can justify its behavior by resorting to other interestingness elements.

3 RELATED WORK

Recent works within eXplainable RL (XRL) have recently started to addressed some of the problems identified in this paper. Some frameworks simply develop language templates to translate elements of the problem into human-understandable explanations. Namely, the work in [21] generates explanations for reasoning in POMDPs, *i.e.*, the agent's policy, beliefs over states of the environment, transition probabilities, rewards, etc. All explanations were derived directly from the POMDP elements, therefore corresponding to our Level 0 analysis. In [11], the discounted occupancy frequency is used to form contrastive explanations about the expected return in a given state by executing the optimal action and following some policy. Other works focus on abstracting state representations of the task and creating graph structures denoting the agent's behavior—in [8] and [20], explanations were framed as summaries assembled from outputs of binary classifiers that characterized the agent's trajectories and decisions. Other approaches try to identify key moments of the agent's interaction to summarize its behavior. The approach in [2] used the concept of importance, dictated by the largest difference in the Q-values of a given state, to identify which trajectories are representative of the agent's behavior, which were then shown to a human user.

We note that, overall, these works usually require a great deal of manual adjustments for specific domains while our framework relies on generic data that is already collected by standard RL algorithms and on a factored-state structure. Moreover, while some can summarize the agent's behavior, they do not perform an analysis of the reasons of behavior and thus cannot provide insights about the agent's decision-making. In addition, they lack the capability of automatically detecting situations requiring external human intervention. Finally, current XRL systems operate only *after* learning, making it hard to recover key aspects of the interaction.

4 CONCLUSIONS AND FUTURE WORK

We introduced a framework capable of analyzing a RL agent's history of interaction with its environment. The framework operates at three distinct levels, first analyzing characteristics of the task that the agent has to solve, then the behavior of the agent while interacting with the environment, and finally by performing a meta-analysis combining information gathered at the lower levels. In general, the proposed analyses generate meaningful information from data that is already collected by standard RL algorithms, such as the Q and V functions generated by value-based methods, and state frequencies and expected rewards collected by model-based techniques. In addition, we proposed statistical data that can be easily collected by an RL agent while performing the task that helps further summarizing its history of interaction. We then detailed how, based on this interaction data, several interestingness elements can be generated by the analyses framework.

4.1 Explanation Framework

Explanation for autonomous agents requires determining not just *what* to explain but also *how* and *when*. The interestingness elements described here provide the content for explanation. Throughout the paper, we provided insights on how each element can be used to expose the agent's behavior to a human user, justifications regarding its decisions, situations in which input from the user might be needed, etc. For most of the elements, explanations can easily be converted from the data by coupling them with natural language templates, similarly to what has been done in [21]. For others, visual tools like graphs and images highlighting important situations—including relevant observation features—can be used, in line with [2].

In an autonomy setting—particularly for *learned* autonomy—explanation should be capable of operating with different modes. As mentioned earlier, in a *passive mode* the agent constructs explanations in response to explicit queries from the user. For example, after training the agent in some task, the user may wish to validate the learning by asking the agent to analyze particular situations, its motivations, its foreseeable plans, etc. To that end, we can use the analyses proposed in this paper to summarize the learned policy, *i.e.*, abstract a strategy, or identify the most important situations.

The explanation framework should also be able to operate in a *proactive mode*, where the agent initiates explanation, whether to avert surprise [6], or to request assistance. For example, while the agent is learning, it may use the identified uncertain, unpredictable, or low-valued situations to ask input from a user. This may be particularly useful in situations where the agent alone is not able to perform optimally, *e.g.*, it is learning in a partially-observable domain. Interaction with the user may occur in various forms, by the user indicating which action to perform in some situation, by providing corrective rewards [12], or by providing higher-level guidance [23]. The explanations provided by the agent in this setting provide the user with the context within which to give feedback to better influence the agent's learning process.

Being able to operate at different times is another desirable feature of an explanation framework. In that regard, we can use the proposed framework to perform a *differential analysis* identifying how the interestingness elements change given two histories of interaction with an environment. By analyzing these changes, we can explain transformations of the agent's behavior, changes in the environment, identify novel situations, acquired knowledge, etc. This analysis can be used during training to assess the agent's learning progress. Another possibility is to compare between the agent's behavior *during* and *after* training to identify which challenges were overcome after learning, and which situations remained confusing, uncertain or unpredictable. Finally, one can apply the differential analysis to data captured by a novice/learning agent and an expert in the task. This can be useful to "debug" the agent's behavior and identify its learning difficulties, assess how its acquired goals differ from those of the expert, etc.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB '94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 487–499.

[2] Dan Amir and Ofra Amir. 2018. HIGHLIGHTS: Summarizing Agent Behavior to People. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '18)*. IFAAMAS, Richland, SC, 1168–1176.

[3] Maartje M A De Graaf and Bertram F Malle. 2017. *How People Explain Action (and Autonomous Intelligent Systems Should Too)*. Technical Report FS-17-01. AAAI 2017 Fall Symposium on Artificial Intelligence for Human-Robot Interaction. 19–26 pages.

[4] E. W. Dijkstra. 1959. A note on two problems in connexion with graphs. *Numer. Math.* 1, 1 (01 Dec 1959), 269–271. https://doi.org/10.1007/BF01386390

[5] B. Fuglede and F. Topsoe. 2004. Jensen-Shannon divergence and Hilbert space embedding. In *Proceedings of the International Symposium on Information Theory, 2004. (SIT 2004)*. 31–37. https://doi.org/10.1109/ISIT.2004.1365067

[6] Melinda Gervasio, Karen Myers, Eric Yeh, and Boone Adkins. 2018. Explanation to Avert Surprise. In *Explainable Smart Systems Workshop (ExSS) at ACM IUI 2018*. ACM.

[7] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. 2004. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery* 8, 1 (01 Jan 2004), 53–87. https://doi.org/10.1023/B:DAMI.0000005258.31418.83

[8] Bradley Hayes and Julie 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 '17)*. ACM, New York, NY, USA, 303–312. https://doi.org/10.1145/2909824.3020233

[9] Paul Jaccard. 1912. The Distribution of the Flora in the Alpine Zone. *New Phytologist* 11, 2 (1912), 37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x

[10] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 1 (1998), 99 – 134. https://doi.org/10.1016/S0004-3702(98)00023-X

[11] Omar Zia Khan, Pascal Poupart, and James P. Black. 2009. Minimal Sufficient Explanations for Factored Markov Decision Processes. In *Proceedings of the 19th International Conference on International Conference on Automated Planning and Scheduling (ICAPS'09)*. AAAI Press, 194–200.

[12] W. Bradley Knox and Peter Stone. 2009. Interactively Shaping Agents via Human Reinforcement: The TAMER Framework. In *Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP '09)*. ACM, New York, NY, USA, 9–16. https://doi.org/10.1145/1597735.1597738

[13] Michael L. Littman. 1994. Memoryless Policies: Theoretical Limitations and Practical Results. In *Proceedings of the Third International Conference on Simulation of Adaptive Behavior : From Animals to Animats 3: From Animals to Animats 3 (SAB94)*. MIT Press, Cambridge, MA, USA, 238–245.

[14] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1 – 38. https://doi.org/10.1016/j.artint.2018.07.007

[15] C. P. H. Mulder, E. Bazeley-White, P. G. Dimitrakopoulos, A. Hector, M. Scherer-Lorenzen, and B. Schmid. 2004. Species evenness and productivity in experimental plant communities. *Oikos* 107, 1 (2004), 50–63. https://doi.org/10.1111/j.0030-1299.2004.13110.x

[16] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (1st ed.). John Wiley & Sons, Inc., New York, NY, USA.

[17] Pedro Sequeira and Cláudia Antunes. 2010. Real-Time Sensory Pattern Mining for Autonomous Agents. In *6th International Workshop on Agents and Data Mining Interaction, ADMI 2010 (ADMI 2010)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 71–83. https://doi.org/10.1007/978-3-642-15420-1_7

[18] Pedro Sequeira, Francisco S. Melo, and Ana Paiva. 2013. An Associative State-Space Metric for Learning in Factored MDPs. In *Proceedings of the 16th Portuguese Conference on Artificial Intelligence (EPIA 2013)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 163–174. https://doi.org/10.1007/978-3-642-40669-0_15

[19] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement learning: an introduction*. MIT Press. 322 pages.

[20] Jasper van der Waa, Jurriaan van Diggelen, Karel van den Bosch, and Mark Neerincx. 2018. Contrastive Explanations for Reinforcement Learning in terms of Expected Consequences. In *Proceedings of the 2nd Workshop on Explainable Artificial Intelligence (XAI 2018)*. 165–170.

[21] Ning Wang, David V. Pynadath, and Susan 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 (AAMAS '16)*. IFAAMAS, Richland, SC, 997–1005.

[22] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8, 3 (01 May 1992), 279–292. https://doi.org/10.1007/BF00992698

[23] Eric Yeh, Melinda Gervasio, Daniel Sanchez, Matthew Crossley, and Karen Myers. 2018. Bridging the Gap: Converting Human Advice into Imagined Examples. In *Advances in Cognitive Systems 6 (ACS 2018)*. Cognitive Systems Foundation, 1168–1176.