Journal of Artificial Intelligence Research 72 (2021) 1083-1102 Submitted 07/2021; published 11/2021

Research Note

Teaching People by Justifying Tree Search Decisions: An Empirical Study in Curling

Cleyton R. Silva CLEYTON.SILVA@UFV.BR
Departamento de Informática, Universidade Federal de Viçosa
Viçosa, Brazil

Michael Bowling MBOWLING@UALBERTA.CA
Levi H. S. Lelis LEVI.LELIS@UALBERTA.CA
Department of Computing Science, University of Alberta
Alberta Machine Intelligence Institute
Edmonton, Canada

Abstract

In this research note we show that a simple justification system can be used to teach humans non-trivial strategies of the Olympic sport of curling. This is achieved by justifying the decisions of Kernel Regression UCT (KR-UCT), a tree search algorithm that derives curling strategies by playing the game with itself. Given an action returned by KR-UCT and the expected outcome of that action, we use a decision tree to produce a counterfactual justification of KR-UCT's decision. The system samples other possible outcomes and selects for presentation the outcomes that are most similar to the expected outcome in terms of visual features and most different in terms of expected end-game value. A user study with 122 people shows that the participants who had access to the justifications produced by our system achieved much higher scores in a curling test than those who only observed the decision made by KR-UCT and those with access to the justifications of a baseline system. This is, to the best of our knowledge, the first work showing that a justification system is able to teach humans non-trivial strategies learned by an algorithm operating in self play.

1. Introduction

We have recently witnessed tremendous achievements of learning systems in complex sequential decision problems. Artificial intelligence (AI) algorithms achieved superhuman performance on Atari (Mnih et al., 2015), Go, shogi, chess (Silver et al., 2018a), and strong performance in StarCraft (Vinyals et al., 2019). Yoshiharu Habu, a 9-dan Shogi player stated the following about AlphaZero, a system that mastered Shogi through self play: *"Some of its moves, such as moving the King to the centre of the board, go against shogi theory and—from a human perspective—seem to put AlphaZero in a perilous position. But incredibly it remains in control of the board. Its unique playing style shows us that there are new possibilities for the game"* (Silver, Hubert, Schrittwieser, & Hassabis, 2018b). While AlphaZero and other systems can learn creative and novel strategies for solving complex sequential decision problems, we are still unable to effectively transfer these systems' machine-generated knowledge to humans as they are represented in opaque models.

In this research note we show that a simple justification system, which we name Counterfactual Feature-Based Justifications (CFJ), can be used to transfer to humans some of the

knowledge generated by Kernel Regression UCT (KR-UCT) (Yee, Lisy, & Bowling, 2016) in the context of the Olympic sport of curling. Given an action returned by KR-UCT, CFJ produces a counterfactual justification of the algorithm's decision by sampling other possible outcome states and selecting for presentation the states that are most similar to the state KR-UCT intends to achieve in terms of visual features and most different in terms of expected end-game value. The rationale of this procedure is to show the user how each feature might affect the end-game value.

We hypothesize that humans can learn non-trivial curling strategies by observing how different features affect the end-game value. We test our hypothesis with a user study with 122 participants where we employ CFJ to generate justifications of a single decision made by KR-UCT in a simulation of curling. In order to test our hypothesis, we apply KR-UCT and CFJ on a manually selected state of curling. The state we use in our study requires the player to reason about important curling concepts that, once learned, can be helpful in deciding which actions to take in other states of the game. In our study, one group of people had access to CFJ's justifications in addition to KR-UCT's decision. A second group had access to KR-UCT's decision and to the justifications generated by a baseline justification system. Finally, a third group had access only to the decision made by KR-UCT.

The results of our user study show that the participants who had access to CFJ's justifications for a single KR-UCT decision achieved higher scores in our curling test than the participants of the other groups. An analysis of the participants' optional justifications of their answers suggests that most of the people who had access to CFJ's justifications were able to learn non-trivial curling concepts that most of the participants of the other two groups either failed to learn or failed to apply to other states of the game.

Example 1 *Figure 1 shows an example of a justification CFJ produces. The image at the top-left corner shows KR-UCT's intended action (i.e., the action the agent tries to deliver but might fail to do so due to uncertainties in the environment), where the blue line shows the intended trajectory of the curling rock. The other three figures show actions that are slightly different than the intended action, but that can significantly affect the chances of the player winning the game. CFJ selects the other three trajectories to display to the user as a means of justifying KR-UCT's decision. The CFJ justifications highlight the visual differences between the intended action and the other actions. The justifications also inform the user how each highlighted difference affects the player's chances of winning the game. Firstly, CFJ informs the user that the player has a 63% probability of winning the game with the intended action. Then, CFJ presents the image at the top-right corner and informs the user that if the rock crosses the vertical line at the center of the red circle, as depicted in the image, the chances of the player winning the game are reduced by 5%. This procedure is repeated for the two images at bottom, which show that if the rock does not touch the vertical line the chances of winning the game are reduced by 12%; and that if the rock is not aligned with the red rock outside the blue circle the chances of winning the game are reduced by 5%.*

Contributions. The main contribution of this note is our study showing that a simple justification system can be used to transfer to humans knowledge generated by a tree search algorithm. This is, to the best of our knowledge, the first work showing that a justification system is able to teach humans non-trivial strategies learned by an algorithm operating

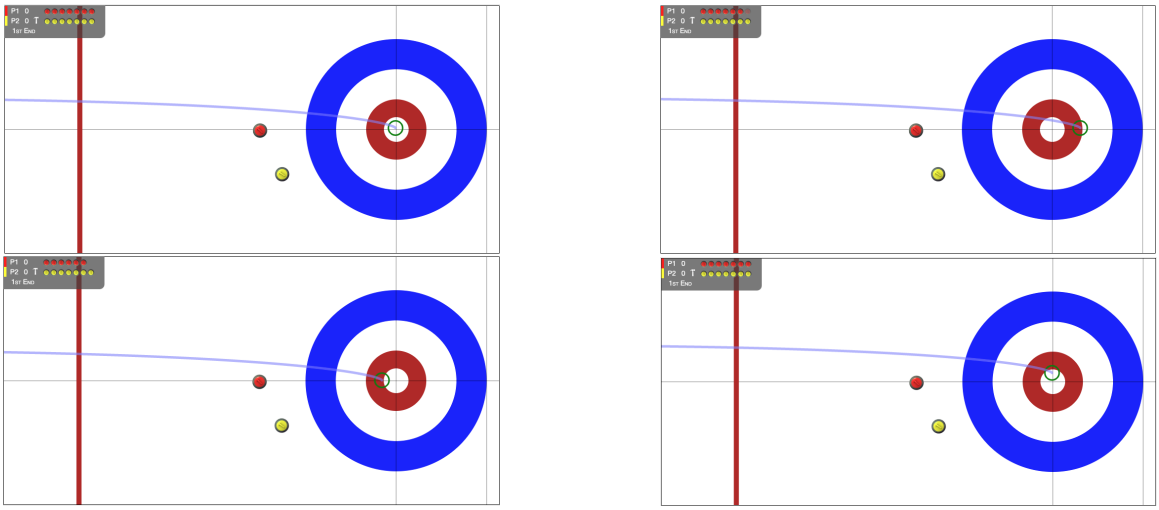

Figure 1: Example of a justification CFJ generates.

in self play. Another contribution is CFJ, which is a simple extension to adversarial and sequential decision-making problems of systems that use decision trees to justify decisions of black box models induced by supervised learning algorithms (Guidotti et al., 2018).

2. Related Work

The system we present in this paper is related to explainable intelligent systems. We review a few representative works in this section. See Biran and Cotton (2017), Miller (2017), and Guidotti et al. (2018) for surveys on the topic.

Supervised Learning Much of the work in explainability has focused on explaining the decisions of supervised learning algorithms (Andrews, Diederich, & Tickle, 1995; Ribeiro, Singh, & Guestrin, 2016). Guidotti et al. (2018) presented LORE, an agnostic approach for explaining the classifications of black-box methods. LORE induces a decision tree with an artificially constructed set of instances. The rules of the decision tree are then used to explain the classification and also to provide a set of counterfactual rules, which show how the instance being explained would need to be modified to have its classification changed. CFJ is similar to LORE because it also uses a decision tree to select a set of features from which counterfactual states can be derived. CFJ differs from LORE and other explanation systems for supervised learning because it considers the sequential and adversarial nature of curling. CFJ generates justifications by looking ahead in the game tree and accounting for how the opponent might react to the player's action.

Explainable Planning and RL Previous works on explainable planning (Fox, Long, & Magazzeni, 2017) introduced methods for providing provable correct explanations of plans (Seegebarth et al., 2012), systems for allowing the user to query decisions made by robots (Lomas et al., 2012), systems to visualize the decisions made by robots (Brooks, Shultz, Desai, Kovac, & Yanco, 2010), algorithms for generating explanations based on the difference between the algorithm's and the humans' models of the world (Chakraborti et al.,

2017), and algorithms that change the reward function of an agent with explanations as a means of obtaining a desirable behavior (Tabrez, Agrawal, & Hayes, 2019). For a recent survey see the work by Chakraborti, Sreedharan, and Kambhampati (2020). Similarly to the planning literature, previous works on explainable reinforcement learning (RL) have focused on the single-agent setting (Cruz, Dazeley, & Vamplew, 2019). One key distinction made in the RL literature is regarding local and global explainability. In the former one is interested in explaining single actions of an agent, while in the latter one is interested in explaining the general behavior of an agent (Amir & Amir, 2018; Verma et al., 2018, 2019). Our work differs from these papers because we deal with an adversarial scenario where agents act strategically with respect to the opponent's actions. The justifications generated by our system show how different features affect the decision making process while accounting for the actions of the opponent. Moreover, we are interested in generating justifications that can be used to teach humans.

Mixed-Initiative Systems Our work is also related to mixed-initiative systems, where a computer system collaborates with a person. Chattopadhyay et al. (2017) studied human-system collaboration in the GuessWhich game, where one player describes an image and the other has to guess the image described. In the context of planning, Kulkarni et al. (2019) developed algorithms that attempt to minimize the gap between the plan executed by the artificial agent with that expected by the human partner. Pérez-D'Arpino and Shah (2015) developed methods that try to predict the human's intent as a means to maximize the productivity of the human-robot team. Sreedharan et al. (2020) described how explanations of an agent's actions can help in the human-agent performance in cooperative settings. In all these works a system collaborates "online" with humans. By contrast, we use justifications of decisions produced by a high-performing algorithm to teach humans in an "offline" process. People trained by our system are evaluated without the system's online help.

Tutoring Systems AI algorithms have been used for decades in the context of computer-aided instruction and tutoring systems. Carbonell (1970) introduced SCHOLAR, a knowledge-based system able to interact with students through a dialogue system. Students were able to ask questions and learn from the knowledge stored in the system. Brown and Burton (1974) presented SOPHIE, a system that leveraged the knowledge encoded in an expert system to instruct students how to troubleshoot circuits. Clancey (1979) introduced GUIDON, a tutoring system built on top of the expert system MYCIN (Shortliffe & Buchanan, 1975). GUIDON was used to tutor students in both medical and engineering domains. These systems attempt to transfer knowledge generated with supervised learning, which were trained with human-labelled data. By contrast, we show how justifications can be used to transfer to humans machine-generated knowledge, i.e., knowledge generated by having an algorithm play the game with itself.

3. Stochastic Games with Continuous Spaces

Let $\mathcal{G} = (\mathcal{N}, \mathcal{S}, \mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{K}, \mathcal{R})$ be a stochastic game with continuous action and state spaces and a known model of uncertainty. Here, $\mathcal{N}$ is a pair of players, $\mathcal{S}$ is an infinite set of states with each state $s \in \mathcal{S}$ being described by a set of n human-interpretable and manually selected binary features $\mathcal{F}$. For $f \in \mathcal{F}$, $f(s)$ denotes the value of feature f for state s. $\mathcal{P}$ is

a function mapping a state s to the player who is to act at s, $\mathcal{A}$ is the set of actions player $\mathcal{P}(s)$ can perform, $\mathcal{T}$ is a deterministic transition function mapping a state s and an action a to a state s'. Function $\mathcal{K}(a, a')$ is a known uncertainty model that returns the probability action a' being executed when a is the intended action. For example, a curling player might intend to deliver a specific action a, but due to factors such as skill level and ice conditions, they deliver a different action a' (see Section 7.2 for a description of the uncertainty model $\mathcal{K}$ we use in our experiments). $\mathcal{R}$ is a utility function mapping a state to a pairs of utility values, one for each player. A stochastic game can be represented by a *game tree* with each node in the tree representing a state, where the root represents the game's current state. In the tree there is a directed edge from a node representing state s to another representing s' if there exists an action $a \in \mathcal{A}$ such that $\mathcal{T}(s, a) = s'$. We call the *children* of s all states s' with an incoming edge from s. Since the state and action spaces are continuous, in practice the game tree is sampled.

4. Kernel Regression UCT

We use Kernel Regression UCT (KR-UCT) in our experiment because it is able to find strong strategies in curling (Yee et al., 2016). Similarly to UCT, KR-UCT implements the steps *selection*, *expansion*, *simulation*, and *backpropagation* while growing its search tree. KR-UCT grows its tree while it has not exhausted a predefined computational budget, which is specified in terms of a number of samples in our experiments (each sample is comprised of the completion of the four steps mentioned above). Once KR-UCT exhausts its budget, it performs a *final selection* step, which we also describe below, to decide which action the agent will perform in the environment. The set of actions available at each node in the tree in the selection step is initially given by a domain-dependent action generator that receives as input a state s and returns a finite set of actions available at s. This set of actions can grow as KR-UCT performs more iterations, as explained in the expansion step below. We use the action generator employed in the original KR-UCT work (Yee et al., 2016).

Selection In the selection step, starting at the root of the tree, KR-UCT iteratively selects the action with largest UCB value. KR-UCT initially considers the set of actions given by the action generator and it adds new actions to this set as we explain in the expansion step below. The UCB formula KR-UCT uses is adapted to stochastic games by using an information sharing scheme as Yee et al. described. Namely, KR-UCT uses a kernel function to approximate the expected utility the agent receives if the agent chooses an action a at a state s based on all actions KR-UCT had previously evaluated for s. The expected utility is denoted as $Q(s, a)$. The kernel function, which is denoted as $K(a, a')$, gives the probability of action a' being executed when the agent intends to perform action a. Similarly to Yee et al., in our experiments, we use the uncertainty model $\mathcal{K}$ of our curling simulator as the kernel function K. For a given data set of action and utility value pairs $(a_i, r_i)_{i=1,\ldots,n}$ for state s, KR-UCT approximates $Q(s, a)$ by performing a sum of the utilities weighted by their kernel values.

$$\hat{Q}(s, a) = \frac{\sum_{i=0}^{n} K(a, a_i) r_i}{\sum_{i=0}^{n} K(a, a_i)}. \tag{1}$$

The value $\sum_{i=0}^{n} K(a, a_i)$, known as the kernel density and denoted as $W(a)$, measures the amount of information available to action a from the actions in the data set for state s.

The adapted UCB formula is defined as

$$\arg\max_{a \in \mathcal{A}} \left(\hat{Q}(s,a) + C \cdot \sqrt{\frac{\log \sum_{b \in A} W(b)}{W(a)}} \right) . \quad (2)$$

The second term is UCB's exploration term adapted to kernel regression, with C being the exploration constant.

Expansion A node n' is added to the UCT tree as a child of n whenever one of the two conditions is satisfied: (i) n is a leaf node in the UCT tree or (ii) the square root of the number of times n was visited in a UCT's selection step is greater than the number of n's child nodes. Condition (i) is the usual UCT condition for expanding the tree, while condition (ii) is an implementation of progressive widening (Coulom, 2007). The intuition behind progressive widening is to increase the branching factor of a node in the UCT tree once the current children are "evaluated well enough." For both conditions (i) and (ii), given the action a returned in the selection step, the node that is added to the UCT tree is determined by an action a' that is sampled with probability $\mathcal{K}(a, a')$. The action a' used to expand the tree is added to the list of actions of node n. That way a' can be selected in the selection step of KR-UCT despite it not being among the actions returned by the action generator.

Simulation and Backpropagation Once a node n' is added to the UCT tree, it is evaluated with a play-out policy by simulating the game from n' for both players to the end of the game. The utility obtained with the simulation is then used to update the data sets of action and utility values pairs of all nodes on the path from the root of the tree to n'. We use the same domain-specific play-out policy used in the original implementation of KR-UCT (Yee et al., 2016).

Final Selection Once KR-UCT exhausts its computational budget, it returns the action with the greatest lower confidence bound given by the kernel approximation (Yee et al., 2016), which is given by the following equation

$$\arg\max_{a \in \mathcal{A}} \left(\hat{Q}(s,a) - C \cdot \sqrt{\frac{\log \sum_{b \in A} W(b)}{W(a)}} \right) .$$

For two-player games, KR-UCT plans for both players, trying to find strategies that are best responses to each other. Although in our experiments KR-UCT does not learn a policy or a value function, KR-UCT still derives strategies by playing the game against itself. Our goal is to transfer the knowledge encoded in such strategies to human learners.

5. Counterfactual States

Let A be an algorithm that receives a state s as input and returns an action a to be performed at s that maximizes the player's expected utility. Let s^* be the state the deterministic transition model $\mathcal{T}(s,a)$ returns. We call s^* the *intended state*. Our system justifies A's decisions by presenting a set of *counterfactual states* of s^* with respect to a feature f. The set of counterfactual states of s^* with respect to feature f is the set of states most similar to s^* that differ from s^* with respect to f. Counterfactual states are defined as follows.

Definition 1 (Counterfactual States) *Let $\mathcal{S}_{f(s)\neq f(s^*)}$ be the set of states $s \in \mathcal{S}$ with $f(s) \neq f(s^*)$. The* counterfactual states *of state s^* and feature $f \in \mathcal{F}$ are those that optimize*

$$\max_{s\in\mathcal{S}_{f(s)\neq f(s^*)}} \left|\{f'|f'(s) = f'(s^*), f' \in \mathcal{F}\}\right|.$$

We present counterfactual states to the user with the goal of showing how individual features affect the player's expected utility. Our system justifies an intended state by presenting a counterfactual state altogether with a text of the form *"given that feature f is present (or absent), the player's expected utility is decreased by x."* This form of presentation allows the user to relate the textual description with the states presented by the system. Alternatively, one could highlight in the text how pairs, triples, or more generally, tuples with m features affect the player's utility, i.e., *"given that features $f_1, f_2, \cdots, f_m$ are present (or absent), the player's chances of winning the game is reduced by x."* By definition, counterfactual states minimize the number of features m because counterfactual states are those that maximize the number of features matching the intended state. We chose to minimize the value of m to ease the task of understanding the effects of individual features in the game. Depending on the state of the game, it could happen that the intended state and the counterfactual state for feature f differ on multiple features. In that case, aiming at reducing the user's cognitive load, the text presented by our system mentions only f.

Also related to the idea of allowing users to focus on one feature at a time, CFJ presents a sequence of *consistent counterfactual states*, instead of a sequence of counterfactual states. Let $\{f_1, f_2, \cdots, f_n\}$ be a total ordering of the features in $\mathcal{F}$ defined according to a metric of importance (defined below), with any f_j being more important than f_k if $j < k$. This ordering of features is used to define the ordering in which the counterfactual states are presented to the user: the first counterfactual state presented is related to the most important feature, the second state presented is related to the second most important feature and so on. We constrain the k-th counterfactual state s presented to the user, which is related to feature f_k, to be *consistent* with all counterfactual states already presented to the user. A state s is consistent for iteration k if it satisfies $f_j(s) = f_j(s^*)$ for all $j < k$.

Definition 2 (Consistent Counterfactual States) *The set of* consistent counterfactual states *of state s^* with respect to feature f_k and an ordered set of features $\{f_1, f_2, \cdots, f_n\}$ are the states s that optimize*

$$\max_{s\in\mathcal{S}^k_{f_k(s)\neq f_k(s^*)}} \left|\{f'|f'(s) = f'(s^*), f' \in \mathcal{F}\}\right|.$$

Where $\mathcal{S}^k_{f(s)\neq f(s^*)}$ is the set of consistent states, defined as

$$\mathcal{S}^k_{f_k(s)\neq f_k(s^*)} = \{s|f_k(s) \neq f_k(s^*) \wedge f_j(s) = f_j(s^*), \forall j < k\}.$$

By presenting consistent counterfactual states we avoid scenarios where the system presents in iteration j a counterfactual state s showing how the presence (or absence) of feature f_j can decrease the player's expected utility and then, in a later iteration $k > j$, presents a counterfactual state s' that differs from s^* in terms of both f_j and f_k. This scenario can increase the user's cognitive load because the user will be aware of the importance

Algorithm 1 CFJ

Require: Game $\mathcal{G}$, state s, algorithm A, number of states n.
Ensure: Set of counterfactual states C of s

1: $a \leftarrow A(s)$
2: $s^* \leftarrow \mathcal{T}(s, a)$
3: $S \leftarrow \{s^*\}$
4: **for** $k = 1$ to n **do**
5: $\quad a' \sim \mathcal{K}(a, \cdot)$
6: $\quad S \leftarrow S \cup \mathcal{T}(s, a')$
7: **return** Select-Counterfactuals(S, s^*, $\mathcal{F}$, $\emptyset$)

Algorithm 2 Select-Counterfactuals

Require: States S, intended state s^*, a set of features F, set of counterfactual states C of s^*.
Ensure: Set of counterfactual states C of s^*

1: **if** $f(s_1) = f(s_2), \forall s_1, s_2 \in S$ and $\forall f \in F$ **then**
2: $\quad$ return C
3: $f \leftarrow \arg\min_{f \in F} Var(S_{f=0}) + Var(S_{f=1})$
4: $F \leftarrow F \setminus f$
5: **if** $f(i) = 0$ **then**
6: $\quad s \leftarrow \arg\max_{s \in S_{f=1}} \left|\{f' | \big(f'(s) = f'(s^*), f' \in F\}\right|$
7: $\quad C \leftarrow C \cup s$
8: $\quad$ **return** Select-Counterfactuals($S_{f=0}$, s^*, F, C)
9: **else**
10: $\quad s \leftarrow \arg\max_{s \in S_{f=0}} \left|\{f' | \big(f'(s) = f'(s^*), f' \in F\}\right|$
11: $\quad C \leftarrow C \cup s$
12: $\quad$ **return** Select-Counterfactuals($S_{f=1}$, s^*, F, C)

of feature f_j when presented s', and will then have to reason about the pair of features f_j and f_k. If $f_j(s') = f_j(s^*)$, then the user will be able to reason about f_k separately, i.e., how f_k affects the player's expected utility given that f_j matches the intended state.

6. Counterfactual Feature-Based Justifications

Our system for justifying the decisions of algorithm A, Counterfactual Feature-Based Justifier (CFJ), is shown in Algorithm 1. CFJ receives a game $\mathcal{G}$, the state s in which algorithm A is to make a decision, and a number of states n to be collected to justify A's decision (we use $n = 10,000$ in our experiments). CFJ returns a set of consistent counterfactual states for A's intended state, one counterfactual state for each "important feature."

CFJ starts by invoking algorithm A for state s (line 1) and computing the intended state s^* with the deterministic transition function $\mathcal{T}$ and the action a to be taken at s (line 2). The intended state s^* is stored in the set S (line 3). Then, CFJ collects n states by sampling n actions from the uncertainty model given that a is the intended action (lines 4–6). Due to the uncertainty model $\mathcal{K}$, CFJ collects a set of states that are different than s^*, but that

might have many features in common with s^*; all n states are also stored in S. Once S is collected, the procedure Select-Counterfactuals (Algorithm 2) returns a set of consistent counterfactual states for s^* (line 7 of Algorithm 1), which is a subset of S.

Algorithm 2 receives as input the set S, the intended state s^*, the set of features F, initially equal to $\mathcal{F}$, and a set of counterfactual states, which is initially empty. Algorithm 2 works similarly to a procedure for inducing a decision tree for regression problems. It recursively splits S into disjoint subsets $S_{f=0}$ and $S_{f=1}$ according to the feature f that minimizes the sum of the variance of the expected utility of the states s in each subset. Here, states s in $S_{f=0}$ have $f(s) = 0$ and states s in $S_{f=1}$ have $f(s) = 1$ (see line 3 for choosing the feature f used to split S and lines 8 and 12 for the recursive calls).

The variance of sets $S_{f=0}$ and $S_{f=1}$ is computed based on the expected utility of the state-action pair (s, a') used to obtain each s' in S ($\mathcal{T}(s, a') = s'$). We evaluate (s, a') instead of s' to capture the uncertainty associated with each action a'. For example, the intended state s' of the action a' represented by the trajectory in the image at the bottom-left corner of Figure 1 is not worse than the KR-UCT's intended state (shown at the top-left corner of Figure 1). However, due to the uncertainty model, a' is more likely to hit another rock in its trajectory than the action $A(s)$ KR-UCT returns. That is why the player increases their chances of winning the game by executing $A(s)$ and not a' in Example 1. We use the play-out policy used in the original implementation of KR-UCT (Yee et al., 2016) to evaluate the pairs (s, a'); the evaluation of the pairs is not shown in the pseudocode.

The counterfactual state returned in lines 6 and 10 are consistent counterfactual states because, if $f(s^*) = 0$ (resp. $f(s^*) = 1$) for the selected feature f, then the recursive call is performed for $S_{f=0}$ (resp. $S_{f=1}$), thus ensuring that the set S received as input by the algorithm is consistent with s^*. The consistent counterfactual states are added to the ordered set C (lines 7 and 11). F is the set of features that were not selected in previous calls of the algorithm. In the first call to Select-Counterfactuals all features $\mathcal{F}$ are assigned to F, and the f selected in each call is removed from F before the recursive call (line 4).

Select-Counterfactuals stops when no more consistent counterfactual states can be extracted from S. This happens if all states in S agree on all features F (lines 1 and 2) and are thus identical to s^* with respect to F. CFJ then returns the ordered set C of consistent counterfactuals, with the ordering defined by how much each feature related to each state reduces the variance of the player's utility, which is CFJ's metric of importance. The set of counterfactual states CFJ returns contains states that are similar to the intended state (counterfactual states and intended state differ by one feature) and that can differ substantially in terms of end-game utility value as the feature selected in each iteration of the algorithm minimizes the variance of the end-game outcomes.

Our system does not present to the user the images of the counterfactual states Algorithm 1 returns, but it presents the images of the state-action pairs (s, a') used to deterministically generate the counterfactual states CFJ returns. For example, the image at the top-left corner of Figure 1 shows the state-action pair (s, a), where s is the state passed as input to CFJ and a is the action KR-UCT chooses to play at s. The other images in Figure 1 show the state-action pairs (s, a') that can be used to deterministically obtain the counterfactual states CFJ returns. We present the state-action pairs (s, a') instead of the intended states $\mathcal{T}(s, a')$ to highlight the differences between the actions a' and the action a.

Although we apply CFJ to the game of curling, the algorithm is general and can potentially be applied to other domains. The requirement for applying CFJ to another domain is the existence of a set of human-interpretable features $\mathcal{F}$ and the ability of sampling the set of states S used to produce the consistent counterfactual states. In curling, the sampling procedure is conveniently done by sampling the same action multiple times with the uncertainty model. In deterministic domains one could use search procedures (e.g., breadth-first search) to collect a set of states that CFJ can use.

7. Empirical Methodology

We hypothesize that humans are able to learn strategies derived by search algorithms once they learn how changes in different features affect the end-game value. We test our hypothesis by evaluating CFJ with a user study where people read the justifications provided by CFJ for a single decision KR-UCT makes. Next, we describe our empirical methodology.

7.1 Curling

Curling is a two-player stochastic game played on a sheet of ice. In each round of the game a player slides a rock from one end of the sheet of ice to the other. The goal is to place rocks near the center of "the house," a scoring area (see the blue, white, and red circles in Figure 2). A player scores one point for each rock placed closer to the center of the house than any of the other player's rocks. The score of the game is tallied after players slide 8 rocks each. The red circle at the center of the house marks the 4-foot ring, the white circle the 8-foot ring, and the blue circle the 12-foot ring. The thick vertical line is the hog line; rocks that do not cross the hog line are removed from play. Rocks that stop between the hog line and the house are known as guards. The horizontal and vertical gray lines going across the center of the house are the "center" and "tee" lines, respectively.

The player controls for the rock's velocity, angle, and rotation while sliding it across the sheet of ice. The rotation of the rock influences the direction in which the rock curls. The curl of the rock allows it to reach spaces that would not be reachable if it followed a straight line. The challenges of curling can be divided into two categories: skill-based and strategy-based. The first category encompasses the challenges related to the skill of the player delivering the shot, i.e., delivering the correct angle, velocity, and rotation. The second category of the challenges relates to the strategy of the game, i.e., deciding the intended states. In addition to the plays of the opponent team, the strategy followed by a team should account for uncertainties of the game, e.g., players might perform suboptimal plays. In our user study we use CFJ to teach strategies to curling rookies.

We use a set of binary features that we manually extracted from curling manuals. The features we selected are often used by curlers while talking to each other in the matches. Table 1 summarizes the features, where the first column shows the name of the features and the right column the condition for the feature being true.

7.2 Curling Simulator

We use the curling simulator of Yee et al. (2016), which was designed to mimic the uncertainty of plays of Olympic-level male players. In an actual match of Curling, players can

Feature	Condition to be true: If the delivered rock...
R-foot	stops in the R-foot ring
R-foot Biter	touches the boundary of the R-foot ring when it stops
Tee Line Biter	touches the tee line when it stops
Center Line Biter	touches the center line when it stops
Behind Tee Line	crosses the tee line
Behind Guard	is at least 2/3 covered by a guard
Partially Behind Guard	is at least 1/3 covered by a guard
Is Guard	is a guard
Is Touching	is in contact with another rock
Hit	hit another rock in its trajectory
Moved	moved another rock in its trajectory
Above Center Line	is above the center line
Removed	is removed from play

Table 1: Set of binary features used in our experiment. The conditions for a feature being true considers the rock the player delivers. R can assume the values of $4, 8$, or 12.

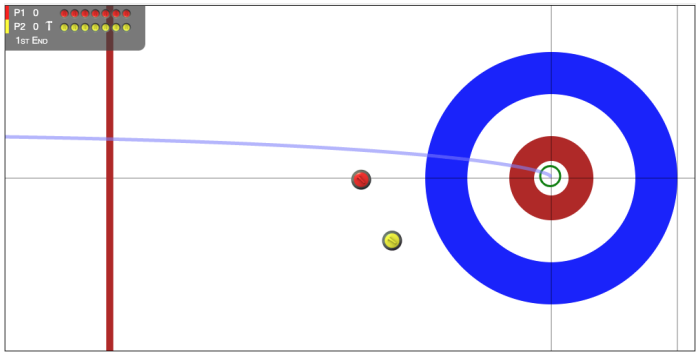

Figure 2: State used in our user study. CFJ justifies the intended state of KR-UCT, depicted by the trajectory of the next red rock (see blue line).

sweep the ice during the rock's trajectory to correct for an imperfect play. By sweeping the players reduce the friction between the rock and the ice, allowing the rock to move faster and to have a straighter trajectory. Yee et al.'s simulator does not model sweeping directly, its uncertainty model mimics the error of players when sweeping is applied. The uncertainty model $\mathcal{K}$ the simulator implements uses a heavy-tailed Student-t distribution, which models the intuitive notion that actions that are similar to the intended action are more likely to occur. Also, the probability of observing an action decreases as the action is more dissimilar to the intended action. The parameters of the Student-t distribution were fit to match the statistics of men from the 2010 and 2014 Olympic games (Yee et al., 2016).

7.3 Justifications of KR-UCT Intended State

In our experiment we use CFJ to produce a justification for the decision KR-UCT makes for the state s shown in Figure 2. The state shows a typical opening in a match of curling, where both players place a guard, which are used to prevent rocks inside the house from being easily removed from the game. In curling, guards can only be removed after five rocks have been played. Thus, by placing guards early in the game the player creates protected areas inside the house. We ran KR-UCT with 1,000 samples and it returned, for the red player, the action a depicted by the blue trajectory in the figure, where the rock should be placed at the center of house. We then estimated the probability of the player winning the match by simulating the game forward from the state-action pair (s, a) to the end of the game with a domain-specific policy for both players. We use the play-out policy from the original KR-UCT implementation (Yee et al., 2016). We performed 1,000 simulations and discovered that the red player has a 63.56% chance of winning the match. We also performed 1,000 simulations to estimate the probability of the player winning the game from each state-action pair (s, a') that generates a counterfactual state CFJ returns.

We use the state s because there are important curling concepts supporting the decision chosen by KR-UCT. We hypothesize that, by learning these concepts through CFJ's justifications, novice curling players will be able to apply the learned concepts in other states of the game. Namely, in this play the red player is trying to "hide" their rock behind the red guard, thus increasing the difficulty for the yellow player to remove the red rock from the house (in the intended state the rock is placed on the same line of the red's guard). Another concept related to this decision is that the rock does not cross the tee line. If the rock crosses the tee line, then it is easier for the opponent to place a rock closer to the center of house as there will not be a rock occupying that space.

7.4 Baseline Approaches to Justifications

We also experimented with two baselines (we refer to different approaches as treatments) in a between-subject study, where each participant is exposed to only one treatment. In one baseline, denoted as "Plain", the participants only observe the decision made by KR-UCT in the state shown in Figure 2. In addition to the image of the intended state, the participants also read the sentence *"In the figure we present a state from the beginning of the game, where each team plays a rock. The scenario shown is common in curling matches. The figure also shows a good play for the red player, the next player to act. With this move, the red team has a probability of winning the game of 63.56%."*

The other baseline replaces the $\arg\min$ operator in line 3 of CFJ (see Algorithm 2) by an $\arg\max$ operator. This baseline presents exactly the same number of features as the justification provided by CFJ for the state we use in our study. Our goal with this baseline is to show that it is not the extra information provided by CFJ's justification that allows people to learn strategies of curling, but the quality of the information CFJ provides. We refer to this baseline as "Max."

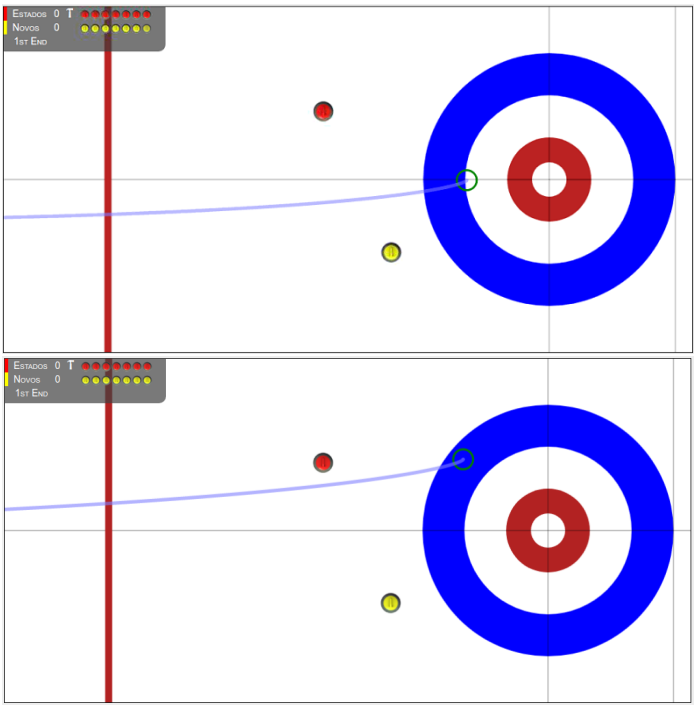

Figure 3: Representative question used in our user study. Participants were asked which action maximized the yellow player's chances of winning the game. The play shown in the image at the bottom maximizes the yellow player's chances of winning the game because the played rock will be "hidden" behind the red rock.

7.5 Structure of the Experiment

The experiment was advertised in mailing lists and performed online by volunteers. We used a Randomized Controlled Trial (RCT) design, where each participant is randomly assigned to one of the three treatments. All participants, independently of their treatment, read a curling tutorial we have prepared. In our tutorial we explain the name of components of the game (e.g., tee and center lines, R-foot rings), as some components were later used in the explanations of CFJ and of the Max baseline. We also explained how the game of curling is scored. Each participant also watched a video of players playing all their eight rocks.

Once the participants finished the tutorial, they answered four questions about curling. In these questions we asked the name of components taught in the tutorial. We also asked two questions related to the scoring scheme of curling, to ensure that the participants understood the rules of the game.

Only after answering the curling-related questions that the participants read the content of each treatment. Then, they were asked to decide on an action for four different states of the game. For each state, the participant was given two options of plays and had to choose one of the following options, where Y is the player to act in the state (either red or yellow): *(1) the first play is better than the second for the Y player; (2) the second play is better than the first for the Y player; (3) both plays are equally good for the Y player; (4) both plays are equally bad for the Y player.* Figure 3 shows one of the four states used, with the

Treatment	Curling Score	Strategy Score	Time Curling	Time Strategy
Plain	3.56 ± 0.73^a	1.68 ± 1.26^a	60.07 ± 27.48^a	142.57 ± 51.48^a
Max	3.71 ± 0.56^a	1.61 ± 1.01^a	79.55 ± 54.56^a	102.95 ± 48.90^b
CFJ	3.68 ± 0.60^a	2.83 ± 1.36^b	61.56 ± 41.08^a	114.15 ± 53.70^b

Table 2: Average results and standard deviation for the curling and strategy parts of the study for each treatment; maximum score is 4.0. The "Time" columns show the average time in seconds spent by each participant on the multiple-choice questions. Numbers with different superscript letters within a column are significantly different (see text for details).

two options of play. The four states used for testing were states where the player had to play either the third or the fourth rock of the game, which is similar to the state used in training (Figure 2), where the player had to decide the play of the third rock of the game.

In all four test states the player had the chance to choose an option where the rock would be either "hidden" behind another rock or it would "hide" another rock. We chose test states for which the property of a rock being hidden did not make the play an obvious choice, as shown in Figure 3. While the play shown at the bottom maximizes the chances of the yellow player winning the game, the intended play depicted at the top figure places the rock closer to the center of the house, which is often played by inexperienced players.

The participants answered a demographics questionnaire after answering all four strategy-related question for the test states. They were asked to report their age, gender (female, male, or other), education level (tertiary or not), and familiarity with curling in a 5-Likert scale ranging from "never heard of the sport" (1) to "has curled before" (5). In total we had 122 participants, 41 for treatments Plain and CFJ, and 40 for Max. We had 77 male and 43 female participants, and 2 participants identified themselves as 'other'. The average age was 26.74 years. The average 'experience in curling' was 1.18, indicating that most of the participants had little to no experience in curling. Finally, 77 participants had finished their tertiary studies. As expected in an RCP design, there is little difference in the average value of the participants across different treatment groups in terms of age, gender, curling experience, and education level.

The user study was approved by research ethics board of Universidade Federal de Viçosa.

8. Empirical Results

In this section we present the score and timing results of the participants of each treatment. Then, we also present a quantitative analysis of the justification written by the participants for their choices in the strategy test.

8.1 Score and Timing Results

Table 2 presents the score and timing results of our user study. The first column ("Treatment") presents the three treatments used, while the second and third columns present the average scores for participants of each treatment for the curling questions, used to verify

the participants understanding of the game rules, and for the strategy-related questions, used to measure the strategy knowledge of the participants after reading the content of each treatment. The maximum score in these two columns is 4.0. The last two columns present the average time in seconds each participant spent answering the questions.

The time results measure the time in seconds the participants spent answering the multiple-choice questions. There was an optional text field where the participants could justify their answers. The time spent writing in the justification form was not accounted for in the results. Since the participants did not have a time limit to complete the experiments, a few participants spent much more time answering the questions than others, suggesting that a break was taken during the experiment. We removed the data from participants that spent more (resp. less) time than the average plus (resp. minus) two standard deviations. This data was disregarded only from the timing results and not from the score results as we believe that a break taken by a participant could not affect their score at the strategy test.

The superscript letters in Table 2 present the statistical differences among different treatments (rows) for a given metric (column). A metric value is statistically different for two treatments if they have different superscript letters. As examples, the strategy score of CFJ is statistically different from the strategy score of Plain and Max, and there is no statistical difference among the curling scores of the three treatments. Superscripts are determined as follows. We perform the non-parametric Kruskal-Wallis statistical test to verify if there are statistical differences between the treatments in terms of score. For the curling questions we have $p = 0.708$, which does not allow us to reject the null hypothesis that the average scores are the same. This is expected in our RCT experimental design as there are no differences in the treatments for the curling questions. For the strategy questions we obtained $p = 2.888\text{e}{-5}$, which allows us to confidently reject the null hypothesis. Using the non-parametric Mann-Whitney test we obtained $p = 0.827$ for the average scores of Plain and Max in the strategy questions. The p values for the pairwise test comparing CFJ with Plain and Max were $1.347\text{e}{-5}$ and $2.134\text{e}{-5}$, respectively. The effect sizes of the CFJ treatment compared to Plain and Max are large, 0.416 and 0.459, respectively.

We also perform the non-parametric Kruskal-Wallis statistical test to verify for differences in terms of time spent by the participants answering the questions. For the curling questions we obtained $p = 0.206$. For the strategy questions $p = 4.125\text{e}{-3}$, which allows us to perform the pairwise tests. While there is no difference between the time spent by participants of the CFJ and Max treatments ($p = 0.369$), Max and Plain as well as CFJ and Plain differ significantly, with p-values of $1.119\text{e}{-3}$ and $2.086\text{e}{-2}$ and effect sizes are medium to small: 0.384 and 0.277, respectively.

8.2 Analysis of the Participants' Justifications

The participants had the option to write a justification for their answers to the strategy questions. We annotated the participants' justifications with respect to four binary features: whether the participant considered how the opponent would respond to their choice of action ("Considers Opponent"), whether the participant chose an action where the played rock would either protect or be protected by another rock ("Protects Rocks"), chose the suboptimal strategy of placing rocks near the center of the house ("Plays Center"), or chose another suboptimal strategy ("Other Strategies"). Our objective with this analysis is to

Treatment	Considers Opponent	Protects Rocks	Plays Center	Other Strategies	Total Responses
Plain	63.9	37.7	27.9	23.0	61
Max	42.9	25.0	32.1	35.7	28
CFJ	75.0	73.1	7.7	25.0	52

Table 3: Percentage of justifications that had each of the four features for each treatment. The rightmost column presents the total number of justifications written for each treatment.

verify if the treatments might have helped the participants think strategically about the game, e.g., by considering how the opponent could react to their play.

Two annotators, who are knowledgeable on the rules of curling, independently annotated the features of all justifications. The inter-agreement κ-value of the annotations was 0.87, indicating a high-level of agreement of the annotations. The annotators discussed the justifications for which they disagreed on the feature value until a consensus was reached. Table 3 shows the results of our analysis. The last column shows the total number of responses for each treatment. Plain had a participant justification rate of $61/(41 \times 4) = 0.37$, Max and CFJ had a participant justification rate of 0.18 and 0.31, respectively (each participant could justify the answer of all four questions). The other numbers in the table show the percentage of participant justifications that had a given feature. For example, 75.0% of the justifications written by the CFJ participants considered how the opponent could react to their play. The values do not sum to 100.0 for each row because each justification could have multiple features (e.g., all justifications that mentions the "Protect Rocks" strategy is also considering how the opponent could react to the player's action).

8.3 Discussion

The results of the user study support our hypothesis that the CFJ justifications can help humans learn strategies of curling. While the Plain and Max participants achieved an average score of approximately 1.6, the CFJ participants achieved the significantly larger average score of 2.83. The statistical tests also point to a large effect size, indicating that CFJ's justifications played an important role in preparing the participants for the test. Participants of the Plain and Max group scored similarly in the test, suggesting that simply providing the participants with more information about a single decision of KR-UCT, as is done by Max, is not enough to teach non-trivial strategies to inexperienced players.

The statistically significant difference between the time spent by the Plain participants and participants of the other two groups suggests that people who were exposed to some form of justification were either confident in their decisions and quickly marked an option they believed to be correct or found the task to be difficult and quickly guessed an answer. While the CFJ participants were able to generalize the lesson learned with the system's justification for the training state, the participants of the Max group were unable to learn from the decision made by KR-UCT. It is also possible that the participants of the Plain treatment spent more time answering the questions because they were not exposed to any

justification during training and had to come up with their own strategy for playing the game. The increased time spent by the participants in the Plain group taken together with their average score suggest that the task of directly generalizing KR-UCT's decision from the training state to the test states is not trivial. The scores of the CFJ participants suggest that the justifications eased the learning task.

Our justification analysis suggest that people from the CFJ treatment learned the concept of protecting rocks, while most of the participants of the other treatments failed to learn the concept. This is because 73.1% of the justifications of the CFJ participants correctly noted the concept, while only 37.7% and 25.0% of the Plain and Max participants mentioned it. Another difference regards the "Plays Center" feature, while a significant number of participants of the Plain and Max treatments justified their decisions based on this suboptimal strategy, only 7.7% of the justifications of CFJ participants mentioned it.

Participants of the Max treatment had a much lower response rate for the written justification, approximately half of the rate of the other two treatments. The low response rate might be another evidence that some of the Max participants guessed their answers or did not feel confident enough to justify their choice. Among the participants who wrote a justification, only 42.9% considered how the opponent could react to their action, which is a much smaller percentage than CFJ's 75% and Plain's 63.9%. The justifications generated by the Max approach seem to have confused the participants, who did not even consider curling as an adversarial game. Finally, Max participants justified their answers with suboptimal strategies more often than the participants of Plain and CFJ. These results suggest that humans might learn more effectively by simply observing the decisions of KR-UCT than by reading low-quality justifications such as those generated by Max.

8.3.1 Limitations of Study

In our experiment we manually selected a training state s that requires the player to reason about important curling concepts such as protecting rocks. Our manual selection of s allowed us to test our hypothesis that CFJ is able to teach inexperienced curling players non-trivial concepts through justifications of the decision made by KR-UCT at s. Naturally, one should not expect CFJ to teach important curling concepts if the state used in training does not require the player to reason about such concepts. An interesting research direction is to use Highlights (Amir & Amir, 2018) to automatically select informative training states.

Although our results provide evidence that simple justification systems can be used to teach people non-trivial strategies of curling, it is unclear from our study if CFJ will also be effective in other sequential games or if CFJ is able to teach complex strategies to experienced curlers. Our results suggest that CFJ is able to teach concepts and strategies that can be explained by a single binary feature. CFJ isolates features and shows to the user how each feature individually affects the player's likelihood of winning the game. More complex strategies might require a complex combination of features or might require features CFJ does not even consider. More research is required to allow for automated discovery of features and for an automated selection of training states that can be used to teach specific concepts and strategies. Our study also does not shed light on the long-term effects of the justifications. For example, how long does it take until users who only observe the decisions

of KR-UCT can learn the strategies that users of CFJ are able to learn with a single decision of the algorithm?

9. Conclusions

In this paper we showed that a simple justification algorithm can be used to teach humans non-trivial concepts of curling that were learned in a self-play scheme. Our justification algorithm, CFJ, is an adaptation of existing decision tree-based approaches for explaining decisions of models induced with supervised learning algorithms. CFJ selects a set of consistent counterfactual states to illustrate how different features affect the player's chances of winning the game. We performed a user study with 122 participants where we evaluated three approaches in the domain of curling with KR-UCT. In one approach people had access to CFJ's justification, in another they had access to a modified version of CFJ, and in a last approach people only witnessed a decision of KR-UCT, with no justifications. The results of our study show a strong effect of the justification system on the capability of people generalizing the strategy used by KR-UCT in a training state to a set of test states.

Acknowledgements

This research was partially funded by Brazil's CAPES and by Canada's CIFAR AI Chairs program. We thank Rob Holte for his feedback on an earlier draft of this paper and the anonymous reviewers for their suggestions.

References

Amir, D., & Amir, O. (2018). Highlights: Summarizing agent behavior to people. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1168–1176. International Foundation for Autonomous Agents and Multiagent Systems.

Andrews, R., Diederich, J., & Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, *8*(6), 373?389.

Biran, O., & Cotton, C. (2017). Explanation and justification in machine learning: A survey. In *International Joint Conference on Artificial Intelligence, Workshop on Explainable AI (XAI)*.

Brooks, D. J., Shultz, A., Desai, M., Kovac, P., & Yanco, H. A. (2010). Towards state summarization for autonomous robots. In *Dialog with Robots, Papers from the 2010 AAAI Fall Symposium*, AAAI Technical Report. AAAI.

Brown, J. S., & Burton, R. R. (1974). Sophie: A pragmatic use of artificial intelligence in cai. In *Proceedings of the Annual ACM Conference*, p. 571?579. Association for Computing Machinery.

Carbonell, J. R. (1970). Ai in cai: An artificial-intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems*, *11*(4), 190–202.

Chakraborti, T., Sreedharan, S., & Kambhampati, S. (2020). The emerging landscape of explainable automated planning and decision making. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 4803–4811. International Joint Conferences on Artificial Intelligence Organization.

Chakraborti, T., Sreedharan, S., Zhang, Y., & Kambhampati, S. (2017). Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 156–163.

Chattopadhyay, P., Yadav, D., Prabhu, V., Chandrasekaran, A., Das, A., Lee, S., Batra, D., & Parikh, D. (2017). Evaluating visual conversational agents via cooperative human-ai games. In *Fifth AAAI Conference on Human Computation and Crowdsourcing*.

Clancey, W. J. (1979). Dialogue management for rulebased tutorials. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 155–161, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Coulom, R. (2007). Computing "elo ratings" of move patterns in the game of go. *ICGA Journal*, *30*(4), 198–208.

Cruz, F., Dazeley, R., & Vamplew, P. (2019). Memory-based explainable reinforcement learning. In Liu, J., & Bailey, J. (Eds.), *Australasian Joint Conference on Artificial Intelligence*, pp. 66–77. Springer International Publishing.

Fox, M., Long, D., & Magazzeni, D. (2017). Explainable planning..

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Comput. Surv.*, *51*(5), 93:1–93:42.

Kulkarni, A., Zha, Y., Chakraborti, T., Vadlamudi, S. G., Zhang, Y., & Kambhampati, S. (2019). Explicable planning as minimizing distance from expected behavior. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2075–2077. International Foundation for Autonomous Agents and Multiagent Systems.

Lomas, M., Chevalier, R., Cross, E. V., Garrett, R. C., Hoare, J., & Kopack, M. (2012). Explaining robot actions. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, p. 187?188, New York, NY, USA. Association for Computing Machinery.

Miller, T. (2017). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Pérez-D'Arpino, C., & Shah, J. A. (2015). Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification. In *IEEE international conference on robotics and automation*, pp. 6175–6182. IEEE.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, New York, NY, USA. Association for Computing Machinery.

Seegebarth, B., Müller, F., Schattenberg, B., & Biundo-Stephan, S. (2012). Making hybrid plans more clear to human users - a formal approach for generating sound explanations. In *International Conference on Automated Planning and Scheduling*.

Shortliffe, E. H., & Buchanan, B. G. (1975). A model of inexact reasoning in medicine. *Mathematical Biosciences*, *23*(3), 351 – 379.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2018a). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, *362*(6419), 1140–1144.

Silver, D., Hubert, T., Schrittwieser, J., & Hassabis, D. (2018b). Alphazero: Shedding new light on chess, shogi, and go..

Sreedharan, S., Soni, U., Verma, M., Srivastava, S., & Kambhampati, S. (2020). Bridging the gap: Providing post-hoc symbolic explanations for sequential decision-making problems with black box simulators. *arXiv preprint arXiv:2002.01080*.

Tabrez, A., Agrawal, S., & Hayes, B. (2019). Explanation-based reward coaching to improve human performance via reinforcement learning. In *Proceedings of the 14th ACM/IEEE International Conference on Human-Robot Interaction*, p. 249?257. IEEE Press.

Verma, A., Le, H., Yue, Y., & Chaudhuri, S. (2019). Imitation-projected programmatic reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 15752–15763.

Verma, A., Murali, V., Singh, R., Kohli, P., & Chaudhuri, S. (2018). Programmatically interpretable reinforcement learning. In *International Conference on Machine Learning*, pp. 5052–5061.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., ..., & Silver, D. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, *575*(7782), 350–354.

Yee, T., Lisy, V., & Bowling, M. (2016). Monte carlo tree search in continuous action spaces with execution uncertainty. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 690–696. AAAI Press.