Causal explanation for reinforcement learning: quantifying state and temporal importance

Xiaoxiao Wang · Fanyu Meng · Xin Liu · Zhaodan Kong · Xin Chen

Abstract
Explainability plays an increasingly important role in machine learning. Because reinforcement learning (RL) involves interactions between states and actions over time, it’s more challenging to explain an RL policy than supervised learning. Furthermore, humans view the world through a causal lens and thus prefer causal explanations over associational ones. Therefore, in this paper, we develop a causal explanation mechanism that quantifies the causal importance of states on actions and such importance over time. We also demonstrate the advantages of our mechanism over state-of-the-art associational methods in terms of RL policy explanation through a series of simulation studies, including crop irrigation, Blackjack, collision avoidance, and lunar lander.

Keywords
Explainability · Reinforcement learning · Causal · Temporal importance

1 Introduction
Reinforcement learning (RL) is increasingly being considered in domains with significant social and safety implications such as healthcare, transportation, and finance. This growing societal-scale impact has raised a set of concerns including trust, bias, and explainability. For example, can we explain how an RL agent arrives at a certain decision? When a policy performs well, can we explain why? These concerns mainly arise from two factors. First, many popular RL algorithms, particularly deep RL, utilize neural networks, which are essentially black boxes with their inner workings being opaque not only to lay persons but also to data scientists. Second, RL is a trial-and-error learning algorithm in which an agent tries to find a policy that minimizes a long-term reward by repeatedly interacting with its environment. Temporal information such as relationships between states at different time instances plays a key role in RL and subsequently adds another layer of complexity compared to supervised learning.

The field of explainable RL (XRL), a sub-field of explainable AI (XAI), aims to partially address these concerns by providing explanations as to why an RL agent arrives at a particular conclusion or action. While still in its infancy, XRL has made good progress over the past few years, particularly by taking advantage of existing XAI methods [9, 24, 36]. For instance, inspired by the saliency map method [30] in supervised learning which explains image classifiers by highlighting “important” pixels in terms of classifying images, some XRL methods attempt to explain the decisions made by an RL agent by generating maps that highlight “important” state features [8, 12, 20]. However, there exist at least two major limitations in state-of-the-art XRL methods. First, the majority of them take an associational perspective. For instance, the aforementioned studies quantify the “importance” of a feature by calculating the correlation between the state feature and an action. Since it is well known that “correlation doesn’t imply causation” [22], it is possible that features with a high correlation may not necessarily be the...
real “cause” of the action, resulting in a misleading explanation that can lead to user skepticism and possibly even rejection of the RL system. Second, temporal information is not generally considered. Temporal effects, such as the interaction between states and actions over time, which as mentioned previously is essential in RL, are not taken into account.

In this paper, we propose a causal XRL mechanism. Specifically, we explain an RL policy by incorporating a causal model that we have about the relationship between states and actions. To best illustrate the key features of our XRL mechanism, we use a concrete crop irrigation problem as an example, as shown in Fig. 1 (more details can be found in the Evaluation section). In this problem, an RL policy $\pi$ controls the amount of irrigation water ($I_t$) based on the following endogenous (observed) state variables: humidity ($H_t$), crop weight ($C_t$), and radiation ($D_t$). Its goal is to maximize the crop yield during harvest. Crop growth is also affected by some other features, including the observed precipitation ($P_t$) and other exogenous (unobserved) variables $U_t$. To explain why policy $\pi$ arrives at a particular action $I_t$ at the current state, our XRL method quantifies the causal importance of each state feature, such as $H_t$, in the context of this action $I_t$ via counterfactual reasoning [3, 19], i.e., by calculating how the action would have changed if the feature had been different.

Our proposed XRL mechanism addresses the aforementioned limitations as follows. First, our method can generate inherently causal explanations. To be more specific, in essence, importance measures used in associational methods can only capture direct effects while our causal importance measures capture total causal effects. For example, for the state feature $H_t$, our method can account for two causal chains: the direct effect chain $H_t \rightarrow I_t$ and the indirect effect chain $H_t \rightarrow C_t \rightarrow I_t$, while associational methods only consider the former. Second, our method can quantify the temporal effect between actions and states, such as the effect of today’s humidity $H_t$ on tomorrow’s irrigation $I_{t+1}$. In contrast, associational methods, such as saliency map [8], cannot measure how previous state features can affect the current action because their models only formulate the relationship between state and action in one time step and ignore temporal relations. To the best of our knowledge, our XRL mechanism is the first work that explains RL policies by causally explaining their actions based on causal state and temporal importance. It has been studied that humans are more receptive to a contrastive explanation, i.e., humans answer a “Why X?” question through the answer to the often only implied counterfactual “Why not Y instead?” [10, 19]. Because our causal explanations are based on contrastive samples, users may find our explanations more intuitive.

2 Related work

Explainable RL (XRL) Based on how an XRL algorithm generates its explanation, we can categorize existing XRL methods into state-based, reward-based, and global surrogate explanations [9, 24, 36]. State-based methods explain an action by highlighting state features that are important in terms of generating the action [8, 25]. Reward-based methods generally apply reward decomposition and identify the sub-rewards that contribute the most to decision making [14]. Global surrogate methods generally approximate the original RL policy with a simpler and transparent (also called intrinsically explainable) surrogate model, such as decision trees, and then generate explanations with the surrogate model [35]. In the context of state-based methods, there are generally two ways to quantify feature importance: (i) gradient-based methods, such as simple gradient [29] and integrated gradients [32], and (ii) sensitivity-based methods, such as LIME [26] and SHAP [17]. Our work belongs to the category of state-based methods. However, instead of using associations to calculate importance, a method generally used in existing state-based methods, our method adopts a causal perspective. The benefits of such a causal approach have been discussed in the Introduction section.

Causal explanation Causality has already been utilized in XAI, mainly in supervised learning settings. Most existing
studies quantify feature importance by either using Granger causality [27] and average or individual causal effect metric [4] or by applying random valued interventions [5]. Two recent studies [18] and [21] are both focused on causal explanations in an RL setting. Compared with [18], the main difference is that we provide a different type of explanation. Our method involves finding an importance vector that quantifies the impact of each state feature, while [18] provides a causal chain starting from the action. We also demonstrate the ability of our approach to provide temporal importance explanations that can capture the impact of a state feature or action on the future state or action. This aspect has been discussed in the crop irrigation experiment in Section 6.1. Additionally, we construct structural causal models (SCM) differently. While the action is modeled as an edge in the SCM in the paper [18], our method formulates the action as a vertex in the SCM model, allowing us to quantify the state feature impact on action. As for [21], our approach is unique in that it can calculate the temporal importance of a state, which is not achievable by their method. Furthermore, we have provided a value-based importance definition of Q-value that differs from their method. Another significant difference between our approach and [21] is the underlying assumption. Our method takes into account intra-state relations, which are ignored in Olson’s work. Neglecting intra-state causality is more likely to result in an invalid state after the intervention, leading to inaccurate estimates of importance. Therefore, our approach considers the causal relationships between state features to provide a more accurate and comprehensive explanation of the problem.

3 Preliminaries

We introduce the notations used throughout the paper. We use capital letters such as $X$ to denote a random variable and small letters such as $x$ for its value. Bold letters such as $\mathbf{X}$ denote a vector of random variables and superscripts such as $X^{(i)}$ denote its $i$-th element. Calligraphic letters such as $\mathcal{X}$ denote sets. For a given natural number $n$, $[n]$ denotes the set $\{1, 2, \ldots, n\}$.

Causal graph and skeleton

Causal graphs are probabilistic graphical models that define data-generating processes [22]. Each vertex of the graph represents a variable. Given a set of variables $\mathcal{V} = \{V_i, i \in [n]\}$, a directed edge from a variable $V_j$ to $V_i$ denotes that $V_i$ responds to changes in $V_j$ when all other variables are held constant. Variables connected to $V_i$ through directed edges are defined as the parents of $V_i$, or “direct causes of $V_i,” and the set of all such variables is denoted by $\mathcal{Pa}_i$. The skeleton of a causal graph is defined as the topology of the graph. The skeleton can be obtained using background knowledge or learned using causal discovery algorithms, such as the classical constraint-based PC algorithm [31] and those based on linear non-Gaussian models [28]. In this work, we assume the skeleton is given.

SCM

In a causal graph, we can define the value of each variable $V_i$ as a function of its parents and exogenous variables. Formally, we have the following definition of SCM: let $\mathcal{V} = \{V_i, i \in [n]\}$ be a set of endogenous (observed) variables and $\mathcal{U} = \{U_i, i \in [n]\}$ be a set of exogenous (unobserved) variables. A SCM [22] is defined as a set of structural equations in the form of

$$V_i = f_i(\mathcal{Pa}_i, U_i), \mathcal{Pa}_i \subset \mathcal{V}, U_i \subset \mathcal{U}, i \in [n],$$  \hspace{1cm} (1)

where function $f_i$ represents a causal mechanism that determines the value of $V_i$ using its parents and the exogenous variables.

Intervention and do-operation

SCM can be used for causal interventions, denoted by the do(·) operator. do($V_i = v$) means setting the value of $V_i$ to a constant $v$ regardless of its structural equation in the SCM, i.e., ignoring the edges into the vertex $V_i$. Note that the do-operation differs from the conditioning operation in statistics. Conditioning on a variable implies information about its parent variables due to correlation.

Counterfactual reasoning

Counterfactual reasoning allows us to answer “what if?” questions. For example, assume that the state is $X_t = x$ and the action is $A_t = a$. We are interested in knowing what would have happened if the state had been at a different value $x’$. This implies a counterfactual question [22]. The counterfactual outcome of $A_t$ can be represented as $A_t|X_t = x’ | X_t = x, A_t = a$. Given an SCM, we can perform counterfactual reasoning based on intervention through the following two steps:

1. Recover the value of exogenous variable $U$ as $u$ through the structural function $f$ and the values $X_t = x$, $A_t = a$;
2. Calculate the counterfactual outcome as $A_t|do(X_t = x’), U = u$. More specifically, in SCM, we set up the value of $X_t$ to $x’$. Then we substitute all exogenous variable values to the right side of the functions and get the counterfactual outcome $A_t$.

MDP and RL

An infinite-horizon Markov Decision Process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S} \in \mathbb{R}^m$ and $\mathcal{A} \in \mathbb{R}$ are finite sets of states and actions, $P(s, a, s’)$ is the probability of transitioning from state $s$ to state $s’$ after taking action $a$, and $R(s, a)$ is the reward for taking $a$ in $s$. An RL policy $\pi$ returns an action to take at state $s$, and its associated Q-function, $Q_\pi(s, a)$, provides the expected infinite-horizon $\gamma$-discounted cumulative reward for taking action $a$ at state $s$ and following $\pi$ thereafter.
4 Problem formulation

Our focus is on policy explainability, and we assume that the policy \( \pi \) and its associated Q-function, \( Q_\pi(s, a) \), are given. Note that the policy may or may not be optimal. We require a dataset containing trajectories of the agent interacting with the MDP using the policy \( \pi \). A single trajectory consists of a sequence of \((s, a, r, s')\) tuples. Additionally, We assume that the skeleton of the causal graph, such as the one shown in Fig. 1 for the crop irrigation problem, is known. We do not assume that the SCM, more specifically its structural functions, is given. We assume the additive noise for the SCM but not its linearity (discussed in Eq. (2) in Section 5.1).

The goal is to answer the question “why does the policy \( \pi \) select the current action \( a_t \) at the current state \( s_t \)?” We provide causal explanations for this question from two perspectives: state importance and temporal importance.

**Importance vector for state** The first aspect of our explanation is to use the important state feature to provide an explanation. Specifically, we seek to construct an importance vector for the state, where each dimension measures the impact of the corresponding state feature on the action. For instance, in the crop irrigation problem, we can answer the question “why does the RL agent irrigate more water today?” by stating that “the impact of humidity, crop weight, and radiation on the current irrigation decision is quantified as \([0.8, 0.1, 0.1]\) respectively. Formally, we have the following definition of the importance vector for state explanation.

Given state \( s_t \) and policy \( \pi \), the importance of each feature of \( s_t \) for the current action \( a_t \) is quantified as \( w_{s_t}^\pi \). The explanation is that the features in state \( s_t \) have causal importance \( w_{s_t} \) on policy \( \pi \) to select action \( a_t \) at state \( s_t \).

**Temporal importance of action/state** The second aspect of our explanation considers the temporal aspect of RL. Here, we measure how the actions and states in the past impact the current action. We can generalize the importance vector above to past states and actions. Formally, given state \( s_t \), policy \( \pi \) and the history trajectory of the agent \( H_t := \{(s_{\tau}, a_{\tau}), \tau \leq t\} \), we define the effect of a past action \( a_\tau \) on the current action \( a_t \) as \( w_{s_t}^{\pi, a_t} \). Similarly, for a past state \( s_\tau \), we define the temporal importance vector \( w_{s_\tau}^{\pi, a_t} \), in which each dimension measures the impact of the corresponding state feature at time step \( \tau \) on current action \( a_t \). Then we use \( w_{s_t}^{\pi, a_t} \) and \( w_{s_\tau}^{\pi, a_t} \) to quantify the impact of past states and action.

5 Explanation

5.1 Importance vector for state

Our mechanism implements the following two steps to obtain the importance vector \( w_{s_t}^{\pi, a_t} \).

1. Train SCM structural functions between the states and actions using the data of historical trajectories of the RL agent;
2. Compute the important vector by intervening in the SCM.

First, we notice that there are three types of causal relations between the states and actions: intra-state, policy-defined, and transition-defined relations. As shown in Fig. 2, the green directed edges represent the intra-state relations, which are defined by the underlying causal mechanism. The orange edges describe the policy and represent how the state variables affect the action. The third type of relation shown as blue edges is the causal relationship between the states across different times. They represent the dynamics of the environment and depend on the transition probability \( P(s_\tau, a_\tau, s_{\tau+1}) \) in the MDP.

We assume that the intra-state and transition-defined causal relations are captured by the causal graph skeleton. For the policy-defined relations, we assume a general case where all state features are the causal parents of the action. In the causal graph, each edge defines a causal relation, and the vertex defines a variable \( V \) with a causal structural function \( f \). Then we only need to learn the causal structural functions between the vertices. To achieve this, we can learn each vertex’s function separately. For a vertex \( V_i \) and its parents \( P(a_i) \),
based on Eq. (1), we make an additive noise assumption to simplify the problem and formulate the function mapping between $V_i$ and $P\alpha_i$ as

$$V_i = f_i(P\alpha_i) + U_i,$$  \hspace{1cm} (2)

where $U_i$ is an exogenous variable. We note that the additive noise assumption is widely used in the causal discovery literature [11, 23]. We then use supervised learning to learn the function mapping among the vertices. Specifically, $f_a$ for action $a_t$ is defined as

$$A_t = f_a(S_t^{(1)}, \ldots, S_t^{(m)}, U_a),$$

where $m$ is the dimension of the state, and $U_a$ is the exogenous variable for the actions.

For the state variables, we denote all exogenous variables as a vector $U_S := [U_1, \ldots, U_m]$ and learn the structural functions. Intuitively, the exogenous variables $U_a$ and $U_S$ represent not only random noise but also hidden features or the stochasticity of the policy for the intra-state and policy-defined causal relations. For transition-defined relations, the exogenous variables can be regarded as the stochasticity in the environment.

### 5.2 Action-based importance

Given a state $s_t$ and an action $a_t$, the importance vector $w_t$ is calculated by applying intervention on the learned SCM. Based on the additive noise assumption, we recover the values of the exogenous variables $U_s$ and $U_a$ according to the value of $a_t$. $s_t$ and the learned causal structural functions. Then we define $w_t$ using the intervention operation (counterfactual reasoning). Specifically, we define the importance vector $w_t = [w_t^{(1)}, \ldots, w_t^{(m)}]$ as

$$w_t^{(i)} = \left| \left( A_{t,S_t^{(i)}=s_t^{(i)}+\delta} | S_t = s_t, A_t = a_t \right) - a_t \right|,$$  \hspace{1cm} (3)

where $| \cdot |$ is a vector norm (e.g., absolute-value norm) and $\delta$ is a small perturbation value chosen according to the problem setting. The term $A_{t,S_t^{(i)}=s_t^{(i)}+\delta} | S_t = s_t, A_t = a_t$ represents the counterfactual outcome of $A_t$ if we set $S_t^{(i)} = s_t^{(i)} + \delta$. In our case, the value of the exogenous variables can be recovered using the additive noise assumption, so the value of $A_{t,S_t^{(i)}=s_t^{(i)}+\delta} | S_t = s_t, A_t = a_t$ can be determined. We interpret the result as that the features with a larger $w_t^{(i)}$ have a more significant causal impact on the agent’s action $a_t$. Note that in the simulation, we average the importance from both positive and negative $\delta$ and return the average as the final score. The perturbation amount $\delta$ is a hyperparameter and should be selected according to each problem setting.

### 5.3 Q-value-based importance

While action-based importance can capture the causal impact of states on the change of the action, it may not capture the more subtle causal importance when the selected action does not change, especially when the action space is discrete. Specifically, $A_{t,S_t^{(i)}=s_t^{(i)}+\delta}|S_t = s_t, A_t = a_t$ may not change after a perturbation of $\delta$, which will result in $w_t^{(i)} = 0$. However, this is different from when there are no causal paths from feature $S_t^{(i)}$ to the action $A_t$, also resulting in $w_t^{(i)} = 0$. Therefore, we also define Q-value-based importance as follows:

$$Q_{\pi}^{perturb} = \left| Q_{\pi}^{perturb} - Q_{\pi}(s_t, a_t) \right|,$$  \hspace{1cm} (4)

where $Q_{\pi}^{perturb} = Q_{\pi}(S_t, s_t^{(i)}+\delta, A_t, s_t^{(i)}+\delta, A_t = a_t)$. In detail, we use counterfactual reasoning to compute the counterfactual outcome of $A_t$ and $S_t$ after setting $S_t^{(i)} = s_t^{(i)} + \delta$ and then substituting them into $Q_{\pi}$ to evaluate the corresponding Q-value. Similar to the action-based importance, we account for both positive and negative importance in practice. See the Blackjack Section 6.3 in evaluation for a comparison between Eq. (3) and Eq. (4) on an example with a discrete action space.

In most RL algorithms, Q-value critically impacts which actions to choose. Therefore, we consider Q-valued-based importance as explanations on the action through the Q-value. However, we note that the Q-value-based importance method sometimes cannot reflect which features the policy really depends on. Some features may contribute largely to the Q-value of all state-action pairs $\{Q(s_t, a_t), a_t \in A\}$, but not to the decision making process - the action with the largest Q-value (arg max$_{a_t \in A} Q(s_t, a_t)$). In such cases, these features may have an equal impact on the Q-value regardless of the action. For example, in the crop irrigation problem, crop pests have an impact on the crop yield (Q-value) but don’t impact the amount of irrigation water (the action). Some related simulations are shown in Appendix C. In summary, we suggest using the action-based importance method by default and the Q-value-based method as a supplement.

### 5.4 Temporal importance and cascading SCM

Temporal importance allows us to quantify the impact of past states and actions on the current action. In RL, estimating of temporal effect is important because policies are generally non-myopic, and actions should affect all future states and actions. To measure the importance beyond the previous step, we define an extended causal model that includes state features and actions in the previous time step, as shown in Fig. 1. In this model, the vertices in the graph are $\{S_t, A_t\}_{t=1}^T$. For
simplicity, we assume the system is stationary, so the causal relations are stationary and do not change over time. Therefore, the structural functions are the same as those defined in Fig. 2, i.e., the mechanism of an edge \((S^{(i)}_t, S^{(j)}_{t+1})\) will be the same as the edge \((S^{(i)}_t, S^{(j)}_{t+1})\). The extended causal model can be regarded as a cascade of multiple copies of the same module, where each module is similar to that in Fig. 2. We can estimate the effect of perturbing any features or actions at any step through intervention, and the effect will propagate through the modules to the final time step. We illustrate the temporal importance in the Blackjack experiment in Section 6.3.

5.5 Comparison with associational methods

In Eq. (3), we define importance by applying intervention. If we have the following definition, which is the same as the association-based saliency map method:

\[
sal^{(i)} = \frac{|A_t|S_t = [s_t^{(1)}, \ldots, s_t^{(m)}] - A_t|}{\delta}
\]  

(5)

Associational models cannot perform individual-level counterfactual reasoning and hence cannot infer the counterfactual outcome after changing the value of one feature of the current state. As pointed out by [22], counterfactual reasoning can infer the specific property of the considered individual that is related to the exogenous variables, and then derives what would have happened if the agent had been in an alternative state. In our method, we use counterfactual reasoning to recover the environment at the current state and estimate how the action responds to the change in one of the state features. So our causal importance can capture more insights compared to the associational methods.

In Fig. 3, we use a one-step MDP toy example to demonstrate the difference. Omitting the time step subscript in the notation, we assume the policy is defined on the state space \(S = [S^{(1)}, S^{(2)}, S^{(3)}]\. An observed variable \(V_p\) is a causal parent of \(S^{(3)}\) but is not defined in the state space.

We define the ground truth of the state and policy as Eq. (3), where \(c_1, c_2, c_3, c_12, c_p\) are constant parameters and \(U_1, U_2, U_3, U_p\) are exogenous variables. We use a linear SCM to show the difference between the two methods. We do not assume the SCM to have linear dependencies.

We assume that both the associational method saliency map and our causal method can learn the ground truth functions. Given a state \(s\), the importance vectors using the two methods are compared in Table 1. We notice that, for \(s^{(1)}\), our method can capture the effect of \(s^{(1)}\) through two causal chains \(S^{(1)} \rightarrow A\) and \(S^{(1)} \rightarrow S^{(2)} \rightarrow A\), while the saliency map method captures only \(S^{(1)} \rightarrow A\). Our causal method considers the fact that a change in \(s^{(1)}\) will result in a change of \(S^{(2)}\) and thus additionally influence the action \(A\). The non-direct paths are also meaningful in explanation and should be considered in measuring the importance of \(S^{(1)}\). However, they are ignored in the saliency map method. The causal importance vector for \(s^{(1)}\) also considers the effect of \(u_2\), which is recovered through counterfactual reasoning. This makes the causal-based importance specific to the current state. Additionally, our method can calculate the effect of \(V_p\) on the action \(A\), which can not be achieved by the associational method saliency map.

We also note that for features \(s^{(2)}\) and \(s^{(3)}\), the two methods obtain the same result. In cases where a state feature is (1) not a causal parent of other features, (2) the policy is deterministic, and (3) there are no exogenous variables, our method is equivalent to the saliency-style approach. However, these conditions may not be common in RL. In general, there are causal relations among state features, such as the chess positions in the game of chess, the state features [position, velocity, acceleration] in a self-driving problem, and the state features [radiation, temperature, humidity] in a greenhouse control problem.

6 Evaluation

We test our causal explanation framework in three toy environments: crop irrigation (Section 6.1), collision avoidance (Section 6.2), and Blackjack (Section 6.3). We also conduct experiments on Lunar Lander, which is a more sophisticated
RL environment (Appendix A.4). For each experiment, the system dynamics, policy, training details, and perturbation values used can be found in Appendix A.

6.1 Crop irrigation problem

We show the results of our explanation algorithm for the crop irrigation problem. We assume a simplified environment dynamic based on agriculture models [37]. The growth of the plant at each step is determined by the state features humidity ($H_t$), crop weight ($C_t$), and radiation ($D_t$). The policy controls the amount of water to irrigate each day. Intuitively, it irrigates more when the crop weight is high, and less when the crop weight is low. Details about the environment dynamics and policy are described in Appendix A.1. We use Fig. 1 as the causal skeleton and apply a neural network to learn the structural equations. Figure 4 shows the importance vector of the state for a given environment [$P_t = 0.07, H_t = 0.12, C_t = 0.44, D_t = 0.70$] and its corresponding action $I_t = 0.67$. First, we notice that our method can estimate the importance of the feature precipitation($P_t$), which is not defined in the state space of the policy. Second, in estimating the causal importance of $H_t$, our method can estimate the effect of $H_t \rightarrow C_t \rightarrow I_t$, which results in higher importance compared to the saliency map method. Since an intervention on $H_t$ can induce a change in $C_t$, causing the action to change more drastically. This effect cannot be measured without a causal model. The same applies to the feature $D_t$. The full trajectory and the importance vector at each time step can be found in Fig. 10 in Appendix A.1.

The causality-based action influence model [18] can find a causal chain $I_t \rightarrow C_t \rightarrow CropYield$ and provide the explanation as “the agent takes current action to increase $C_t$ at this step, which aims to increase the eventual crop yield.” This explanation only provides the information that $C_t$ is an important factor in the decision-making for the current action but can’t quantify it. Moreover, this explanation can’t provide information for other state features, such as $H_t$ and $D_t$ which are also measured in our importance vector.

6.2 Collision avoidance problem

We use a collision avoidance problem to further illustrate that our causal method can find a more meaningful importance vector than saliency map, i.e., which state feature is more impactful to decision-making.

Figure 5a shows the state definition for this problem. A car with zero initial velocity travels from the start point to an endpoint over a distance of $X_{goal}$. The system is controlled in a discrete-time-slot manner and we assume acceleration of the car is constant within each time step. The state $S_t$ includes the distance from the start $X_t$, the distance to the end $D_t$, and the velocity $V_t$ of the car, i.e., $S_t = [V_t, X_t, D_t]$, where $V_t \leq v_{max}$ and $v_{max}$ is the maximum speed of the car. The action $A_t$ is the car’s acceleration, which is bounded $|A_t| \leq e_{max}$. We assume the acceleration of the car is constant within each time step. More detailed settings are described in the simulation section in the supplementary materials. The object is to find a policy $\pi$ to minimize the traveling time under the condition that the final velocity is zero at the endpoint (collision avoidance).

An RL agent learns the following optimal control policy for this avoidance problem, which is also known as the bang-bang control (optimal under certain technical conditions) [2]:

$$A_t = \begin{cases} e_{max} & \text{if } D_t \leq v^2_{max}/(2e_{max}) \\ -e_{max} & \text{otherwise} \end{cases}$$

(7)

Intuitively, this policy accelerates as much as possible until reaching the critical point defined above. Then it will decelerate until reaching the goal.

We use Fig. 5b as the SCM skeleton and use linear regression to learn the structural equations as the entire dynamics are linear. The detail about the system dynamics is described in the appendix.

Figure 6a shows a trajectory under the policy bang-bang control and Fig. 6b shows its corresponding causal importance results. The importance of $V_t, A_{t-1}, D_{t-1}, V_{t-1}$ are zero throughout the time history, and those of $X_t, D_t, X_{t-1}$ have peak importance of $[0.502, 0.502, 0.502]$, respectively, between time step 303-322, during which the car changes the direction of acceleration to avoid hitting the obstacle. The importance values of $X_t, D_t, X_{t-1}$ have the same shape, but that of $X_{t-1}$ is off by one time step, corresponding to their time step subscript. If we were to use the associational saliency method [8] $X_t$ would have a constant zero importance since the action is solely determined by the feature $D_t$. In comparison, our method can find non-zero importance.
through the edge $X_t \rightarrow D_t$. It is reasonable that $X_t$ causally affects $A_t$, because, in the physical world, the path length $X_t$ is the cause of the measurement of the distance to the end $D_t$. Although in Eq. (7) the action $A_t$ is only decided by $D_t$, the source cause of the change in $D_t$ is $X_t$. We can only obtain such information through a causal model, not an associational one.

6.3 Blackjack

We test our explanation mechanism on a simplified game of Blackjack. The state is defined as $[\text{hand}, \text{ace}, \text{dealer}]$, where $\text{hand}$ represents the sum of current cards in hand, $\text{ace}$ represents if the player has a usable ace (an ace that can either be a 1 or an 11), and $\text{dealer}$ is the value of the dealer’s shown card. There are two possible actions: to $\text{draw}$ a new card or to $\text{stick}$ and end the game. We use an on-policy Monte-Carlo control [33] agent to test our mechanism. Since the problem dynamic is non-linear, we use a neural network to learn each structural equation. Figure 7 shows the skeleton of the SCM. More details about the rules of the game are explained in Appendix A.2. Note that in Blackjack, the exogenous variable $U_t$ of some features can be interpreted as the stochasticity or the “luck” during the input trajectory. e.g., $U_{\text{hand},t}$ corresponds to the value of the card drawn at step $t$ if the previous action is $\text{draw}$.

**Using Q-values as metric** The solid bars in Fig. 8 on the next page show the result of Q-value-based importance based on Eq. (4). We interpret the result as follows: (1) The importance of all features are highest at step 1. This is because state 1 is closest to the decision boundary of the policy, and thus applying a perturbation at this step is easier to change the Q-value distribution; (2) The importance of $\text{dealer}$ and $\text{dealer}_\text{prev}$ are the same throughout the trajectory. This is due to the fact that $\text{dealer}$ and $\text{dealer}_\text{prev}$ are always the same. Thus, applying a perturbation on $\text{dealer}_\text{prev}$ will have the same effect as applying a perturbation on $\text{dealer}$ assuming changing $\text{dealer}_\text{prev}$ won’t incur a change in the previous action; (3) A similar phenomenon can be observed between $\text{hand}$ and $\text{hand}_\text{prev}$. Increasing the hand at step $t-1$ by one will have the same outcome as drawing a card with one higher value at $t$. The occasional difference comes from the change in $\text{hand}_\text{prev}$ causing $a_{\text{prev}}$ to change; (4) The importance of $\text{ace}$ is highest at steps 2 and 5. In both of these two states, changing if the player has an ace or not while keeping other features the same will change the best action and a larger difference in the Q-values, which causes the importance to be higher.
Using action as metric  The hatched bars in Fig. 8 show the result of action-based importance based on Eq. (3). The importance is more “bursty”, and features, such as hand, have an importance of zero in the majority of the steps since a perturbation of size one could not trigger a change in the action. However, intuitively, hand is crucial to the agent’s decision-making. Therefore, in this case, we note that the Q-value-based method produces a more reasonable explanation in this example.

Multi-step temporal importance  We cascade the causal graph of blackjack in Fig. 7 to estimate the impact of the past states and actions on the current action, and the full SCM is shown in Fig. 12 in Appendix A.2. Figure 9 shows the results of Q-value-based importance. The importance of \( A_4 \) on itself is omitted since it will always be one regardless of any other part of the graph. We interpret the results as follows: (1) The importance of hand, dealer, is flat over time. As discussed above, perturbing these two features at any given step will mostly change the last state in the same way, resulting in constant importance; (2) The importance of the action \( a_\tau \) increases as \( \tau \) gets closer to the last step \( t = 4 \). An action taken far in the past should generally have a smaller impact on the current action, which corresponds to the increasing importance for \( a_\tau \) in our explanation.

6.4 Additional evaluation

We also evaluate our scheme in a more complex RL environment, Lunar Lander, in Appendix A.4. Lunar Lander
Causal explanation for reinforcement learning: quantifying state and temporal importance 22555

Fig. 9 The Q-value-based temporal importance on $A_4$ for all state features and actions at past time steps in the Blackjack experiment is a simulation testing environment developed by OpenAI Gym [1]. The simulation shows that our scheme can explain some specific phases (state) of the spaceship in the landing process.

7 Discussions

Our causal importance explanation mechanism is a post-hoc explanation method that uses data collected by an already learned policy. We focus on providing local explanations based on a particular state and action. Counterfactual reasoning is required to recover the exogenous variables and estimate the effect on the given state and action. In this case, the intervention operation is not enough to achieve this goal, as it can only evaluate the average results (population) over the exogenous variables, which is not a local explanation for the given state.

Intra-state relations One crucial characteristic of our method is that we consider intra-state relations when computing the importance, which is essential in accurately quantifying the impact of a state feature on the action. Although the MDP defines that a state feature at a certain time step cannot affect another state feature at the same step, it is essential to consider causal relationships within state features when measuring their impact if we use causal intervention or associational perturbation. Since these types of methods require modifying the value of a specific state feature, it should subsequently affect the value of other state features based on real-world causality. For instance, in the collision avoidance problem (Section 6.2), the distance to the end ($D_t$) will change in response to the distance from the start ($X_t$), and in the crop irrigation problem (Section 6.1), the crop weight ($C_t$) will vary based on the humidity level ($H_t$). Ignoring the intra-state causality can lead to an invalid state after the intervention, resulting in inaccurate importance estimates for the given state feature. Hence, we formulate the intra-state relations in the SCM to provide more accurate and comprehensive explanations of the problem.

Additive noise assumption With the additive noise assumption in Eq. (2), the exogenous variable (noise) can be fully recovered and used for counterfactual reasoning. We note that the full recovery noise assumption can be relaxed for our mechanism. In the case where the exogenous variables have multiple values (not deterministic), we can generalize our definition of importance vector in Eq. (3) by replacing the first term with the expectation over different values of exogenous variables using probabilistic counterfactual reasoning [7]. Furthermore, the additive noise assumption is not mandatory. We can use bidirectional conditional GAN [13] to model the structure function and use its noise to conduct counterfactual reasoning and obtain the importance vector.

Known SCM skeleton assumption Our approach is based on the assumption that the SCM skeleton is known, which can be obtained either through background knowledge of the problem or learned using causal discovery algorithms. Causal discovery aims to identify causal relations by analyzing the statistical properties of purely observational data. There are several causal discovery algorithms available, including the classical constraint-based PC algorithm [31], algorithms based on linear non-Gaussian models [28], and algorithms that use the additive noise assumption [11, 23]. These algorithms can be used to learn the SCM skeleton from observational data, which can then be used in our method to quantify the impact of state features and actions on the outcome. There are also existing toolboxes such as [15] and [39] that can be easily applied directly to data to identify the SCM structure.

Perturbation In addition to the method we employed in the simulation, which averages the importance derived from both positive and negative $\delta$, maximizing them is also a viable option. To compute the causal importance vector defined in Eq. (3), we need to choose a perturbation value $\delta$. As shown in Table 1, the importance may depend on $\delta$. Therefore, it is not meaningful to compare importance vectors calculated with different $\delta$. This is a common issue of perturbation-based algorithms, including the saliency map method. In our case, $\delta$ should be as small as possible but still be computationally feasible. More detailed sensitivity analysis and normalization on the perturbation value $\delta$ can be found in Appendix B.
Limitations Our study has limitations when the state space has high dimensions, for example, in visual RL, where state features are represented as images. Image data is inherently high-dimensional, with multiple features that can interact in complex ways. The SCMs we used may struggle to fully capture the complexity of these interactions, especially when a large number of variables are involved. To address this issue, we suggest utilizing the algorithm of causal discovery in images [16] and representation learning [38]. Further work is needed to explore this direction.

Another question that might be raised is what will happen if the trained SCM is not perfect. An imperfect SCM will cause the counterfactual reasoning result to be biased, and thus affecting the final importance. One potential solution is quantifying the uncertainty of the explanation. If the explainer can output its confidence on top of the importance score, users can identify potential out-of-distribution samples where our explanation framework might fail. To achieve this, we need to separate aleatoric uncertainty (which comes from the inherent variability in the environment) and epistemic uncertainty (which represents the imperfection of the model) [6]. Our use of SCM may help us to differentiate the two, and this is one of the directions we are currently exploring.

8 Conclusion

In this paper, we have developed a causal explanation mechanism that quantifies the causal importance of states on actions and their temporal importance. Our quantitative and qualitative comparisons show that our explanation can capture important factors that affect actions and their temporal importance. This is the first step towards causally explaining RL policies. In future work, it will be necessary to explore different mechanisms to quantify causal importance, relax existing assumptions, build benchmarks, develop human evaluations, and use the explanation to improve evaluation and RL policy training.

Funding The work was partially supported by NSF through grants USDA-020-67021-32855, IIS-1838207, CNS 1901218, and OIA-2134901.

Code availability The code is available in the supplementary file.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Conflict of interest The authors declare that there is no conflict of interest.

Appendix A: Additional experiments and details

In this section, we provide additional details regarding the crop irrigation problem, the collision avoidance problem, and the Blackjack experiments. Furthermore, we describe our results on an additional testing environment, Lunar Lander.

All experiments were conducted on a machine with 8 NVIDIA RTX A5000 GPU, an dual AMD EPYC 7662 CPU, and 256 GB RAM.

A.1 Crop irrigation

This section contains details of the crop irrigation experiment.
System dynamics

Precipitation \(= U(0, 1)\)

SolarRadiation \(= U(0, 1)\)

Humidity \(= 0.3 \cdot \text{Humidity}_{\text{prev}} + 0.7 \cdot \text{Precipitation}\)

CropWeight \(= \text{CropWeight}_{\text{prev}} + 0.07 \cdot (1 - (0.4 \cdot \text{Humidity} + 0.6 \cdot \text{Irrigation} - \text{Radiation}^2)^2) + 0.03 \cdot U(0, 1)\)

The change in CropWeight at each step is determined by humidity, irrigation and radiation, and maximum growth is achieved when \(0.4 \cdot \text{Humidity} + 0.6 \cdot \text{Irrigation} = \text{Radiation}^2\). An additional exogenous variable is also included in the change of CropWeight. This can be regarded as some unobserved confounders that affect the growth that are not included in the system dynamics, such as \(\text{CO}_2\) Concentration or the temperature.

Policy

Irrigation \(= (\text{Radiation}^2 - 0.4 \cdot \text{Humidity}) \cdot (1.6 \cdot \text{CropWeight} + 0.2)/0.6\)

The policy we used is a suboptimal policy that multiplies an additional coefficient \(1.6 \cdot \text{CropWeight} + 0.2\) on the optimal policy. This will cause the irrigation value to be less than optimal when CropWeight is less than 0.5, and more than optimal and vice versa.

Training

We use a neural network to learn the causal functions in the SCM. The network has three fully-connected layers, each with a hidden size of four. We use Adam with a learning rate of \(3 \times 10^{-5}\) as the optimizer. The training dataset consists of 1000 trajectories (10000 samples) and the network is trained for 50 epochs.

Perturbation

The perturbation value \(\delta\) used in the intervention is 0.1 w.r.t. the range of each value.

A.2 Blackjack

This section contains details and additional figures for the blackjack simulation.

System dynamics

This simulation is done in the blackjack environment in OpenAI Gym [1]. The goal is to draw cards such that the sum is close to 21 but never exceeds it. Jack, queen and king have a value of 10, and an ace can be either a 1 or an 11, and an ace is called “usable” when it can be used at an 11 without exceeding 21. We assume the deck is infinite, or equivalently each card is drawn with replacement.

In each game, the dealer starts with a shown card and a face-down card, while the player starts with two shown cards.
The game ends if the player’s hand exceeds 21, at which the player loses, or if the player chooses to stick, the dealer will reveal the face-down card and draw cards until his sum is 17 or higher. The player wins if the player’s sum is closer to 21 or the dealer goes bust.

**Policy** We trained the agent using on-policy Monte-Carlo control. Figure 11 shows the policy and the decision boundary.

**SCM structure** We assume the blackjack game has a causal structure as shown in Fig. 7. Additionally, Fig. 12 shows the 5-step cascading SCM we used to test the temporal importance.

**Training** We use a neural network to learn the causal functions in the SCM. The network has three fully-connected layers and each layer has a hidden size of four. We use Adam with a learning rate of $3 \times 10^{-5}$ as the optimizer. The training dataset consists of 50000 trajectories ($\sim 76000$ samples) and the network is trained for 50 epochs.

**Perturbation** Since blackjack has a discrete state space, for numerical features “hand” and “dealer”, we use a perturbation value $\delta = 1$. For the boolean feature “ace”, we flip its value as the perturbation.

### A.3 Collision avoidance problem

We use the collision avoidance problem to further illustrate that our causal method can find a more meaningful importance vector than saliency map, i.e., which state feature is more impactful to decision-making.

**System dynamics** The state $S_t$ includes the distance from the start $X_t$, the distance to the end $D_t$, and the velocity $V_t$ of the car, i.e., $S_t := [V_t, X_t, D_t]$, where $V_t \leq v_{\text{max}}$ and $v_{\text{max}}$ is the maximum speed of the car. The action $A_t$ is the car’s acceleration, which is bounded $|A_t| \leq e_{\text{max}}$. The state transition is defined as follows:

$$V_{t+1} := V_t + A_t \Delta t$$

$$X_{t+1} := X_t + V_t \Delta t + \frac{1}{2} A_t \Delta t^2$$

$$D_{t+1} := X_{\text{goal}} - X_{t+1}$$

The objective of the RL problem is to find a policy $\pi$ to minimize the traveling time under the condition that the final velocity is zero at the endpoint (collision avoidance).

**Policy** An RL agent learns the following optimal control policy also known as the bang-bang control (optimal under certain technical conditions) defined as Eq. (7)

**SCM structure** We use Fig. 5b as the SCM skeleton and use linear regression to learn the structural equations as the entire dynamics are linear.

**Perturbation** The perturbation value $\delta$ used in the intervention is 0.1 after normalization.

### A.4 Lunar lander

**System dynamics** Lunar lander problem is a simulation testing environment developed by OpenAI Gym [1]. The goal is to control a rocket to land on the pad at the center of the surface while conserving fuel. The state space is an 8-dimensional vector containing the horizontal and vertical coordinates, the horizontal and vertical speed, the angle, the angular speed, and if the left/right leg has contacted or not.

The four possible actions are to fire one of its three engines: the main, the left, or the right engine, or to do nothing.

The landing pad location is always at $(0, 0)$. The rocket always starts upright at the same height and position but has a random initial acceleration. The shape of the ground is also randomly generated, but the area around the landing pad is guaranteed to be flat.

**Policy** We train our RL policy using DQN [34].

**SCM structure** We use the Fig. 13 as the skeleton of SCM. The structural functions are learned with linear regression using 100 trajectories ($\sim 25000$ samples).

**Evaluation** Figure 14 shows a trajectory of the agent interacting with the lunar lander environment and the corresponding causal importance using our mechanism. We notice that our mechanism discovers three importance peaks, and we explain this as the agent’s decision-making during the landing process consisting of three phases: a “free fall phase”, in which the agent mainly falls straight and slightly adjusts its angle to negate the initial momentum; an “adjusting phase”, in which the agent mostly fires the main engine to reduce the Y-velocity; and a “touchdown phase”, during which the lander is touching the ground and the agent is performing final adjustments to stabilize its angle and speed. Figures 15a, 15b and 15c show our causal importance vector during each of the three phases. We notice that during the “free fall phase”, features such as angle, angular velocity and x-velocity are more important since the agent needs to rotate to negate the initial x-velocity. However, as the rocket approaches the ground during the “adjusting phase”, we find an increase in importance for y-velocity since a high vertical velocity is more dangerous to control when the rocket is closer to the ground. In the last “touchdown phase”, a large x-position and x-velocity importance can be observed as a change in those features is highly likely to cause the lander to fail to land inside the designated landing zone. Since the lander is already touching the ground, it will take much more effort for the agent to adjust compared to when the lander is still high in the air.
Fig. 13 The causal structure of lunar lander that includes previous state and actions. There should also be edges from each feature to the action at its time step, e.g. edges from \( x_{\text{pos\_prev}} \) to \( a_{\text{prev}} \), or from \( x_{\text{pos}} \) to \( a \). These edges are not shown in this graph for simplicity.

Fig. 14 (a) A lunar lander trajectory instance we used to evaluate our algorithm and the corresponding causal importance vector. The “freefall phase” is roughly between steps 0-70, “adjusting phase” is between steps 70-170, and “touchdown phase” is from about step 170 to the end.

The results are similar to those of saliency-based algorithms [8], and Fig. 16 shows the difference in importance vector between our algorithm and saliency-based algorithm. Note that differences only occur for the positions and the angle. This is because other features don’t have any additional causal paths to the action besides the direct connection. Therefore, the intervention operation is equivalent to the conditioning operation for these features. The features position and angle have an additional causal path through the legs, which causes the difference. Notably, our method captures higher importance for angle, which we interpret as that the landing angle is crucial and is actively managed by the agent.

We are also able to compute the importance of the features in the previous steps, and Fig. 14c and the shaded bars in Fig. 15 represent such importance vectors. The previous-step importances are rather similar to those of the current-step features since the size of the time step is comparatively small. However, our algorithm captures that during the “adjusting phase”, the previous-step importance for the angle is in general higher than the current-step importance, as changing the...
previous angle may have a cascading effect on the trajectory and is especially important to the agent when it is actively adjusting the angle.

Appendix B: Sensitivity analysis

This section performs a sensitivity analysis on how the perturbation amount affects the result of our explanation.

For action-based importance, too small of a perturbation may not yield a meaningful result. This is due to the fact that, depending on the environment and the policy, a too small perturbation may fail to trigger a noticeable change in the action, resulting in a zero importance. This differs from the zero importance case where the policy disregards the feature when making decisions. In our experiments, we use 0.01 with respect to the range of the features for continuous features and the smallest unit for discrete features.

In general, using different perturbation amounts $\delta$ on the same state in the same SCM may result in different importance vectors, and vectors calculated using different $\delta$ cannot be meaningfully compared. However, if we desire the importance of using different $\delta$ to be more on the same level, we suggest finding the highest importance across all features and all time steps and normalizing all results by said number. Section 2 contains an example comparing the importance score with and without the aforementioned normalization.

B1. One-step MDP

As we demonstrated in the example of one-step MDP in Fig. 3 and Table 1, our importance vector will sometimes...
be affected by the perturbation amount. For this experiment, we use Fig. 3 as the skeleton and the following settings. The constants are

\[ c_1 = 1, c_2 = -2, c_3 = 3, c_{12} = 2, c_p = -1 \]

We use unit Gaussian distributions as the exogenous variables and the values are

\[ u_1 = 0.50, u_2 = -0.14, u_3 = 0.65, u_p = 1.52, u_a = -0.23 \]

The state value and the corresponding action are then

\[ s^{(1)} = 0.50, s^{(2)} = 0.86, s^{(3)} = -0.88, v_p = 1.52, a = 3.83 \]

The result of running our method and the saliency map method on the feature \( S^{(1)} \) is shown in Fig. 17. Same as in Table 1. Our algorithm is linear w.r.t. \( \delta \) while the saliency map result is constant. The increased importance comes from the causal link \( S^{(1)} \rightarrow S^{(2)} \rightarrow A \), which also introduces the linear relationship.

**B.2 Collision avoidance**

Figure 18 shows the importance vector of \( X_t \) in the collision avoidance problem and different color lines correspond to different perturbation amounts. Note that similar to the result shown in Fig. 6b, the importance of \( D_t \) is the same as \( X_t \), and \( X_{t-1} \) is the same but off by one time step. Other features have negligible importance.
There are two effects of using different perturbation amounts: 1) The number of steps with non-zero importance is increasing as $\delta$ increases since a larger $\delta$ will cause states further away from the decision boundary to cross the boundary after the perturbation; 2) The value of peak importance is lower. Since we use the action-based importance and the action is essentially binary, the difference in importance solely comes from the normalization we applied on $\delta$ (the denominator in Eq. (3)). If this is undesirable, one way to combat this is to normalize the result using the highest importance across all features and time steps. The normalized result is shown in Fig. 18b, in which the peak value will be one regardless of $\delta$.

### B.3 Lunar lander

Figure 19 shows the sensitivity analysis on lunar lander and the different color lines correspond to different perturbation amounts. Binary features including left and right leg are not included. The general trend of the result is the same while the value and the exact shape of the curve vary slightly when different $\delta$ is used and our result is robust w.r.t. $\delta$.

### B.4 Blackjack

Figure 20 shows the sensitivity analysis for blackjack, with different color lines representing different perturbation amounts. The binary feature ace is not included. In blackjack, since the smallest legal perturbation amount is one and the range of the value is at most 21, increasing $\delta$ has a much larger effect on the result. However, we can observe that the general shape of the curves is similar, indicating the robustness of our method.

Appendix C: Action-based importance versus Q-value-based importance

This section discusses the comparison between the action-based importance method and the Q-value-based importance method. It demonstrates that the Q-value-based method sometimes fails to reflect the features in the state that the policy relies on.

Consider a one-step MDP with the SCM shown in Fig. 21, where the state $S = [S_1, S_2], S_i \in [-1, 1], i = 1, 2,$ and the action $a \in [-1, 1]$. The reward is defined as $R(S, a) = 100 \times S_2 + a \times S_1$. Under this setting, the optimal policy is:

\[
A = \begin{cases} 
-1 & S_1 < 0 \\
1 & \text{otherwise}
\end{cases}
\]

Intuitively, the policy selects the minimum value in the action space when $S_1$ is negative, and the maximum value otherwise.

The action-based importance method correctly identifies $S_1$ as more important, as the policy only depends on $S_1$. However, the Q-value-based method produces a different result. In a one-step MDP, the Q-function is the same as the reward function. As the coefficient in the Q(reward) function is larger for $S_2$, the Q-value-based method finds $S_2$ more important, which is different from the features that the policy relies on.

References

1. Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, Zaremba W (2016) Openai gym. arXiv preprint arXiv:1606.01540
2. Bryson AE (1975) Applied optimal control: optimization, estimation and control. CRC Press, Boca Raton
3. Byrne RM (2019) Counterfactuals in explainable artificial intelligence (xai): Evidence from human reasoning. In: IJCAI, pp 6276–6282
4. Chattopadhyay A, Manupriya P, Sarkar A, Balasubramanian VN (2019) Neural network attributions: A causal perspective. In: International Conference on Machine Learning, PMLR, pp 981–990
5. Datta A, Sen S, Zick Y (2016) Algorithmic transparency via quantitative input influence: Theory and experiments with learning
systems. In: 2016 IEEE symposium on security and privacy (SP), IEEE, pp 598–617
6. Gawlikowski J, Tassi CRN, Ali M, Lee J, Humt M, Feng J, Kruspe A, Triebel R, Jung P, Roscher R, et al. (2021) A survey of uncertainty in deep neural networks. arXiv preprint arXiv:2107.03342
7. Glymour M, Pearl J, Jewell NP (2016) Causal inference in statistics: A primer. John Wiley & Sons, Hoboken
8. Greydanus S, Koul A, Dodge J, Fern A (2018) Visualizing and understanding atari agents. In: International Conference on Machine Learning, PMLR, pp 1792–1801
9. Heuillet A, Couthouis F, Díaz-Rodríguez N (2021) Explainability in deep reinforcement learning. Knowledge-Based Systems 214:106685
10. Hilton D (2007) Causal explanation: From social perception to knowledge-based causal attribution
11. Iyer R, Li Y, Li H, Lewis M, Sundar R, Sycara K (2018) Transpareny and explanation in deep reinforcement learning neural networks. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, pp 144–150
12. Juozapaitis Z, Koul A, Fern A, Erwig M, Doshi-Velez F (2019) Explainable reinforcement learning via reward decomposition. In: IJCAI/ECAI Workshop on Explainable Artificial Intelligence
13. Kalainathan D, Goudet O (2019) Causal discovery toolbox: Uncover causal relationships in python. arXiv preprint arXiv:1903.02278
14. Lopez-Paz D, Nishihara R, Díaz-Rodríguez N (2021) Explainability in deep reinforcement learning: Quantifying state and temporal importance 22563
15. Mott A, Zoran D, Chrzanowski M, Ali M, Lee J, Humt M, Feng J, Kruspe A, Triebel R, Jung P, Roscher R, et al. (2021) A survey of uncertainty in deep neural networks. arXiv preprint arXiv:2107.03342
16. Pearl J (2009) Causality. Causality: Models, Reasoning, and Inference, Cambridge University Press, Cambridge, https://books.google.com/books?id=f4nuexsNVZIC
17. Peters J, Mooij JM, Janzing D, Schölkopf B (2014) Causal discovery with continuous additive noise models
18. Peters J, Mooij JM, Janzing D, Schölkopf B (2014) Causal discovery with continuous additive noise models
19. Ribeiro MT, Singh S, Guestrin C (2016) “why should i trust you?” explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1135–1144
20. Schaub P, Karlen W (2019) Cxplain: Causal explanations for model interpretation under uncertainty. arXiv preprint arXiv:1910.12336
21. Shimizu S, Hoyer PO, Hyvärinen A, Kerminen A, Jordan M (2006) A linear non-gaussian acyclic model for causal discovery. Journal of Machine Learning Research 7(10)
22. Simonyan K, Vedaldi A, Zisserman A (2013) Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034
23. Simonyan K, Vedaldi A, Zisserman A (2014) Deep inside convolutional networks: Visualising image classification models and saliency maps
24. Sprites P, Glymour CN, Scheines R, Heckerman D (2000) Causation, prediction, and search. MIT press, Cambridge
25. Sutton RS, Barto AG (2018) Reinforcement learning: An introduction. MIT press, Cambridge
26. Van Hasselt H, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 30
27. Verma A, Murali V, Singh R, Kohli P, Chaudhuri S (2018) Programmatical interpretable reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 5045–5054
28. Wells L, Bednarz T (2021) Explainable ai and reinforcement learning—a systematic review of current approaches and trends. Frontiers in artificial intelligence 4:550030
29. Williams J, Jones C, Kiniry J, Spanel DA (1989) The epic crop growth model. Transactions of the ASAE 32(2):497–501
30. Yang M, Liu F, Chen Z, Shen X, Hao J, Wang J (2021) Causalvae: Disentangled representation learning via neural structural causal models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9593–9602
31. Zhang K, Zhu S, Kalander M, Ng I, Ye J, Chen Z, Pan L (2021) gcastle: A python toolbox for causal discovery. arXiv preprint arXiv:2111.15155
32. Williams J, Jones C, Kiniry J, Spanel DA (1989) The epic crop growth model. Transactions of the ASAE 32(2):497–501
33. Yang M, Liu F, Chen Z, Shen X, Hao J, Wang J (2021) Causalvae: Disentangled representation learning via neural structural causal models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9593–9602
34. Zhang K, Zhu S, Kalander M, Ng I, Ye J, Chen Z, Pan L (2021) gcastle: A python toolbox for causal discovery. arXiv preprint arXiv:2111.15155
35. Williams J, Jones C, Kiniry J, Spanel DA (1989) The epic crop growth model. Transactions of the ASAE 32(2):497–501
36. Yang M, Liu F, Chen Z, Shen X, Hao J, Wang J (2021) Causalvae: Disentangled representation learning via neural structural causal models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9593–9602
37. Williams J, Jones C, Kiniry J, Spanel DA (1989) The epic crop growth model. Transactions of the ASAE 32(2):497–501
38. Yang M, Liu F, Chen Z, Shen X, Hao J, Wang J (2021) Causalvae: Disentangled representation learning via neural structural causal models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9593–9602
39. Zhang K, Zhu S, Kalander M, Ng I, Ye J, Chen Z, Pan L (2021) gcastle: A python toolbox for causal discovery. arXiv preprint arXiv:2111.15155

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.
Xiaoxiao Wang is a Ph.D. candidate in the computer science department at the University of California, Davis. He received his B.S. in Computer Science from the same institution in 2018. In 2016, he obtained his B.E. in Electrical Engineering from the University of Science and Technology of China. His current research primarily focuses on reinforcement learning, explainable machine learning.

Fanyu Meng is a third-year Ph.D. student in the computer science department at UC Davis. He obtained his B.A. in computer science at UC Berkeley. His current research focus is on using a causal model to explain deep reinforcement learning agents.

Xin Liu received her Ph.D. degree in electrical engineering from Purdue University in 2002. She is currently a Professor in Computer Science at the University of California, Davis. Before joining UC Davis, she was a postdoctoral research associate in the Coordinated Science Laboratory at UIUC. During 2012-2014, she took a leave of absence and was with Microsoft Research Asia. Her current research interests fall in the general areas of machine learning algorithm development and machine learning applications in human and animal healthcare, food systems, and communication networks. Her research on networking includes cellular networks, cognitive radio networks, wireless sensor networks, network information theory, network security, and IoT systems. Prof. Liu is an IEEE Fellow. She received the Computer Networks Journal Best Paper of Year award in 2003 for her work on opportunistic scheduling. In 2005, she received the NSF CAREER award for her research on Smart-Radio-Technology-Enabled Opportunistic Spectrum Utilization. She received the Outstanding Engineering Junior Faculty Award from the College of Engineering, University of California, Davis in 2005. She became a Chancellor’s Fellow in 2011. She is a co-PI and AI cluster co-lead for the $20M AI Institute for Next Generation Food Systems.

Zhao dan Kong is an Associate Professor in Mechanical and Aerospace Engineering at UC Davis. He received his Bachelor’s and Master’s degrees in Astronautics and Mechanics from Harbin Institute of Technology, Harbin, China, in 2004 and 2006, respectively, and his Ph.D. degree in Aerospace Engineering with a minor in Cognitive Science from the University of Minnesota, Minneapolis, MN, USA, in 2011. Before joining UC Davis in 2015, he was a Postdoctoral researcher at the Laboratory for Intelligent Mechatronic Systems and the Hybrid and Networked Systems Lab at Boston University. His research focuses on control theory, machine learning, formal methods, and their applications to human-machine systems, cyber-physical systems, and neural engineering.

Xin Chen is the James C. Edenfield Chair and Professor in H. Milton School of Industrial and Systems Engineering at Georgia Institute of Technology. He obtained his PhD in Operations Research from MIT (2003), MS in Computational Mathematics from Chinese Academy of Sciences (1998) and BS in Computational Mathematics from Xiangtan University (1995). His research interest lies in optimization, revenue management and supply chain management. He received the Informs revenue management and pricing section prize in 2009. He is the coauthor of the book “The Logic of Logistics: Theory, Algorithms, and Applications for Logistics and Supply Chain Management (Second Edition 2005 & Third Edition 2014)”. His research interests are in data analytics; revenue management and dynamic pricing; operations research; operations management; optimization; optimal stochastic control; computational mathematics; and production, inventory and supply chain management.