Applied Intelligence (2023) 53:6936–6952
https://doi.org/10.1007/s10489-022-03788-7

Interpreting a deep reinforcement learning model with conceptual embedding and performance analysis

Yinglong Dai[1,2] · Haibin Ouyang[3] · Hong Zheng[4] · Han Long[1] · Xiaojun Duan[1]

Accepted: 18 May 2022 / Published online: 12 July 2022

Abstract

The weak interpretability of the deep reinforcement learning (DRL) model becomes a serious impediment to the application of DRL agents in certain areas requiring high reliability. To interpret the behavior of a DRL agent, researchers use saliency maps to discover important parts of the agent's observation that influence its decision. However, the representations of saliency maps still cannot explicitly present the cause and effect between an agent's actions and its observations. In this paper, we analyze the inference procedure with respect to the DRL architecture and propose embedding interpretable intermediate representations for an agent's policy, the intermediate representations that are compressed and abstracted for explanation. We utilize a conceptual embedding technique to regulate the latent representation space of the deep models that can produce interpretable causal factors aligned with human concepts. Furthermore, the information loss of intermediate representation is analyzed to define the model performance upper bound and to measure the model performance degeneration. Experiments validate the effectiveness of the proposed method and the relationship between the observation information and an agent's performance upper bound.

Keywords Deep reinforcement learning · Deep neural networks · Interpretability · Conceptual embedding · Perturbation · Causality

1 Introduction

The great achievements in various image processing and speech processing tasks demonstrate the powerful function approximation ability of deep learning (DL) methods for high-dimensional data processing. Borrowing the techniques of DL, a new subbranch of reinforcement learning (RL), termed deep reinforcement learning (DRL), demonstrates excellent performance in many complex decision-making tasks. Since the deep Q-network (DQN) [1] has achieved human-level performance in a majority of Atari games, researchers have contributed many methods to the DRL approach, e.g., extending DRL to a continuous action space [2], asynchronous training [3], robust exploration [4], and effective sampling [5]. DRL methods have achieved many successes in various fields, such as games [6, 7], robotics [8–10], natural language processing [11, 12], and healthcare [13–15].

DRL algorithms are able to train an effective artificial intelligence (AI) agent, but the agent's behavior becomes uninterpretable due to the introduction of a deep learning model. Human beings cannot feel relieved when AI agents execute important tasks without reason, particularly in fields such as automatic pilot and healthcare. For example, a patient will have no idea how to choose an appropriate therapy if a healthcare agent just recommends to the patient a list of therapies without providing any referred reason. More seriously, the black-box model is susceptible to adversarial attacks [16]. In addition, end-to-end training DRL methods that lack interpretability are very difficult to debug and optimize. DRL agents with good interpretability will be a significant advancement before they are deployed to real-world scenes. Accordingly, the interpretability of DRL methods is becoming increasingly important [17, 18]. Despite its importance, the interpretable property of DRL methods has not received enough attention. The difficulties are mainly derived from the poor interpretability of deep

✉ Yinglong Dai
daiyl@hunnu.edu.cn

✉ Xiaojun Duan
xjduan@nudt.edu.cn

Extended author information available on the last page of the article.

neural networks (DNNs) applied in DRL models, and the interpretability of a model is difficult to measure.

As prior work in terms of interpretable DRL, Greydanus et al. [19] borrowed saliency map techniques from the interpretable DL field for DRL to visualize the important parts of an agent's observation and to interpret the relationship between the observation of an agent and the behavior of an agent. By generating saliency maps by perturbation techniques, they demonstrated the effectiveness of the methods in explaining an agent's attention, the reason for its decision, and the evolving learning process. Puri et al. [20] proposed a specific and relevant feature attribution (SARFA) that improved the precision of the saliency map techniques compared with Greydanus's work. The saliency maps generated by the proposed attribution can focus more on the action-related features, so they can provide more accurate and focused interpretations. Although their methods can provide related information between the observation data and the decision-making processes, the related information is lacks sufficient clarity and needs to be further explained for non-professionals. Specifically, the saliency maps still enable no explicit interpretability of the relationship between the output action and the input observation data.

In this paper, we analyze the relationship between an agent's actions and its observations from the perspective of information theory. A best policy exists in the condition of limited information provided by observation data. An agent's performance upper bound will be determined by the amount of information contained in an observation. To provide explicit interpretability and the causal factors for the deep model, we propose a conceptual embedding technique to enhance the interpretability of a DRL-based agent. We seek the conceptual factors that directly relate to certain actions and need to measure the importance of the conceptual factors that affect the decisions of agents. Consequently, we can directly track the decision-driven factors for the agents' different behaviors. To retain the effective information contained in the observation, we analyze the observation information of the latent features with respect to the agent's performance. Through a simple example, we also demonstrate how the information loss of the observation can degenerate an agent's performance. The key contributions of this paper are concluded as follows:

- Intermediate representations are introduced to address the problem of causality between action and observation.
- A conceptual embedding method is proposed to produce interpretable representation spaces in the deep reinforcement learning model.
- Based on information theory, a relationship between an agent's performance upper bound and its observation information is identified.

In the following part, we divide this paper into five sections. In Section 2, we review research on interpretable DRL methods and summarize the main approaches. In Section 3, we analyze the information processing of DRL-based agents and propose our method to improve the interpretability of DRL agents. Hierarchical conceptual embedding techniques are proposed, and the corresponding analyses are presented based on information theory. Next, certain experiments that can validate the proposed method and corresponding analyses are demonstrated in Section 4. We have a discussion in Section 5 and conclude this paper in Section 6.

2 Related work

In the early stage of the DRL development, Zahavy et al. [21] applied the t-distributed stochastic neighbor embedding (t-SNE) technique to explain the behavior of DQN-based agents in Atari environments and presented the embedding-based approach of aggregating the representations for state space. It is a good starting point to analyze the behavior of DRL-based agents from the perspective of observation space and state space, and the reduced state space is more easily related to an agent's behavior. However, the authors just aggregated the state space and did not trace the influence of different features on an agent's behavior. It is still necessary to identify an explicit relationship between an agent's behavior and its observation.

With the development of interpretable DL, researchers have begun to borrow techniques from the interpretable DL field that can be applied to interpret DRL-based agents. Saliency maps are the main adopted techniques that highlight the important parts of observation data related to an agent's decisions. Generally, the gradient-based approach and perturbation-based approach are the two main approaches. The gradient-based approach observes the sensitive features that greatly influence an agent's decision by calculating the gradients. As an efficient approach, Simonyan et al. [22] calculated the Jacobian of the DNN model weights to extract the class saliency maps. Assume that $S_i[O]$ is the score of class i with respect to observation O. The saliency map of the DNN weights is calculated by the derivative of $S_i[O]$ with respect to a specific observation o, i.e., $\frac{\partial S_i[O]}{\partial o}$. Next, certain improved variants have been proposed to generate more interpretable saliency maps, such as DeepLIFT [23] and Grad-CAM [24]. Although gradient-based methods are mathematically reasonable for computing saliency maps, they are sensitive to noise and often generate meaningless salient points. Specifically, the gradient-based methods cannot be applied to the nondifferentiable models of the agents, and they cannot be carried out when the models of the agents are provided as packaged black-boxes.

The perturbation-based approach discovers the important features that can determine an agent's decision by perturbing certain original features and then observing the variance of an agent's decision. This approach is more intuitive and can provide straightforward explanations. To interpret DL models, Fong and Vedaldi [25] proposed a perturbation framework to discover the class-related parts of an image for any black-box model. Recently, the perturbation-based approach has become popular for interpreting the behavior of a DRL-based agent. Greydanus et al. [19] employed a perturbation-based method based on Gaussian blur that can produce saliency maps with respect to the actor network and critic network in the DRL architecture of A3C [3]. Let $S[f]$ be the importance score or saliency score of the observation feature f with respect to an agent's action a, and let o' be the perturbed observation of the original observation o with respect to feature f. $S[f]$ can be calculated by measuring the agent's policy $\pi(\cdot)$ difference between the two observations, i.e., $S[f] = \|\pi(o) - \pi(o')\|$ or the value function $V_\pi(\cdot)$ difference, i.e., $S[f] = \|V_\pi(o) - V_\pi(o')\|$. Based on the work of Greydanus et al., Puri et al. [20] designed an improved metric, termed SARFA, which can filter many irrelevant features and only highlight the key features with respect to a particular decision. However, it incurs a high computational cost of perturbing high-dimensional observation data, and the saliency maps are not explicitly related to the adopted actions.

The work of Iyer et al. [26] applies an idea similar to the premise of this work. The authors proposed a recognition process in the decision process of a DRL-based agent and produced saliency maps of the conceptual objects rather than raw pixel features to analyze the decision process of a DRL-based agent. Nevertheless, the authors still try to interpret the raw input data, and it is difficult to search the objects and analyze the complicated relationships with high-dimensional data. In addition, certain studies introduced causal models for DRL agents [27, 28]. However, causal models have to be defined previously, and their fatal defect is that they cannot discover unknown causal factors from observations. An overview of the approaches is shown in Table 1. In this work, we borrow ideas from the perturbation-based approach and propose our method based on conceptual embedding techniques. We try to explain the causality between an agent's action and its observation in a reduced representation space. To make the features more explicit and more interpretable for human beings, we propose building the conceptual features in the hidden layers of DNN-based models and analyzing the effective information contained in the features for an agent's policy. Furthermore, we can analyze the relationship between an agent's upper bound performance and its observable information.

3 Conceptual embedding in DRL

For the concept of model interpretability, researchers may have their own definition with respect to their different points of view. Nevertheless, it will not be incorrect to conclude that a model has good interpretability if we can track and detail the decision process of the model. In this work, we aim to track the main decision process of DRL agents and to discover the salient features and interpretable factors that directly impact certain actions of DRL agents. First,we analyze the causality between an agent's observation and its adopted action. The intermediate representation space of DRL model is analyzed based on information theory. Second, we propose using hierarchical conceptual embedding techniques to build DNN-based architectures in DRL agents so that we can embed prior knowledge into the DNN architectures and constrain the representation spaces of DRL agents. Third, we propose to generate saliency values for the conceptual embedding activations in the DNNs so that we can discover the important factors and track the decision process of DRL agents. Because the conceptual embedding techniques can be utilized for any DNN-based model, it is a general framework that can be applied in different DRL algorithms to embed conceptual factors and interpret the causal factors of deep models.

Table 1 Overview of interpretable DRL approaches

References	Approach	Explanation level	Causality	Computational cost	Information loss
[21]	Embedding-based	Reduced state space	Weak	Low	Middle
[23, 24, 26]	Gradient-based	Observation space	Weak	Middle	Low
[19, 20]	Perturbation-based	Observation space	Weak	High	Low
[27, 28]	Causal models	Abstract concepts	Strong	Low	High
Ours	Embedding-based	Reduced representation space	Strong	Low	Low

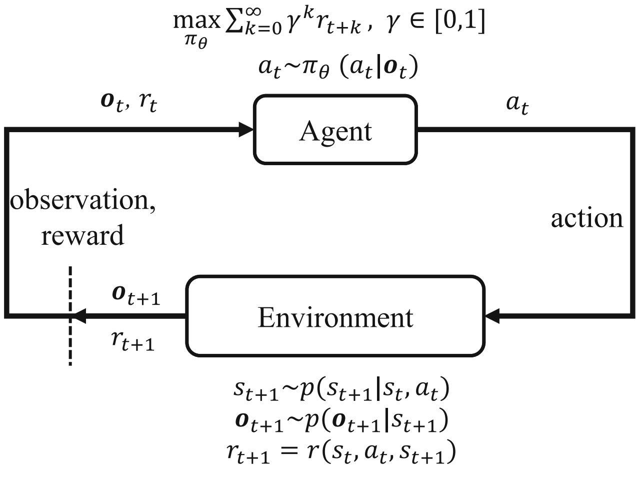
3.1 Preliminaries

RL [29] can be formulated with Markov decision process (MDP) that can be defined by a six-element tuple $< \mathcal{S}, \mathcal{A}, \mathcal{T}, s_0, r, \gamma >$, where $\mathcal{S}$ is a set of states s, $\mathcal{A}$ is a set of actions a, $\mathcal{T}$ is a transition function $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$ used to present the state transition probabilities of the environment from time step t to $t+1$, s_0 is the initial state distribution, r is a reward function to generate the reward at each time step $r_{t+1} = r(s_t, a_t, s_{t+1})$, and $\gamma \in [0, 1]$ is the discount factor.

Generally, DRL refers to the RL methods that use deep neural networks (DNNs) to approximate the value function or/and policy function. DRL usually addresses the partially observable MDP (POMDP) problem [30], in which the agent cannot directly observe the true latent state of the environment. The agent can only receive high-dimensional observable data o generated by the latent state s of the environment and then infer the latent state, expressed as $p(s \mid o)$. The framework formalization can be extended to an expanded tuple $< \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{G}, s_0, r, \gamma >$, where $\mathcal{O}$ is the set of observations o, and $\mathcal{G}$ is the observation generation function that specifies the probability of observation o_t given a certain state s_t, expressed as $o_t \sim p(o_t \mid s_t)$. An extension of the POMDP is a partially observable stochastic game (POSG) [31, 32], which is employed for multiagent systems. This work will focus on an agent, so we will not further discuss the POSG.

The DRL framework of an agent is illustrated in Fig. 1.

The agent, which is a module that receives the observation of the environment o_t with the reward signal r_t and outputs an action a_t, repeatedly interacts with the environment. The POMDP environment is influenced by the agent's action a_t and changes its latent state to s_{t+1} corresponding to observation o_{t+1} with the reward signal r_{t+1}. The objective of the agent is to learn an optimal policy $\pi_\theta(a \mid o)$ to gain maximum future return $\mathcal{R}$, written as



$\max_{\pi_\theta} \sum_{k=0}^{\infty} \gamma^k r_{t+k}$, $\gamma \in [0,1]$
$a_t \sim \pi_\theta\ (a_t|\boldsymbol{o}_t)$
$\boldsymbol{o}_t$, r_t
Agent
a_t
observation, reward
action
$\boldsymbol{o}_{t+1}$
r_{t+1}
Environment
$s_{t+1} \sim p(s_{t+1}|s_t, a_t)$
$\boldsymbol{o}_{t+1} \sim p(\boldsymbol{o}_{t+1}|s_{t+1})$
$r_{t+1} = r(s_t, a_t, s_{t+1})$

Fig. 1 Illustration of the DRL framework of an agent with a POMDP environment. The objective of the agent is to learn a DNN-based policy π_θ to maximize its return, i.e., $\max_{\pi_\theta} \sum_{k=0}^{\infty} \gamma^k r_{t+k}$. The environment has a latent state transition function, which is represented as a conditional probability function $p(s_{t+1} \mid s_t, a_t)$. The latent state s_{t+1} produces an observation o_{t+1} by $p(o_{t+1} \mid s_{t+1})$

$$\mathcal{R} = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \tag{1}$$

where $\gamma \in [0, 1]$ is the discount factor. To judge the expected future return when an agent adopts a given policy in a certain state, we can use the state value function $V_\pi(s)$ with respect to policy π, expressed as

$$V_\pi(s) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s], \tag{2}$$

where $\mathbb{E}_\pi$ denotes the expectation under policy π and $V_\pi(s)$ is used to estimate the expected return when an agent enters a certain state s. To judge the expected future return when an agent adopts a certain action in a certain state and conducts a given policy in the following step, we can use the state-action value function $Q_\pi(s, a)$ with respect to policy π, expressed as

$$Q_\pi(s, a) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s, a_t = a], \tag{3}$$

where $Q_\pi(s, a)$ is used to estimate the expected return if an agent adopts a certain action a in a certain state s. With regard to DRL methods, DNNs are typically deployed to construct and learn the complicated policy function $\pi_\theta(a \mid o)$. For instance, the value-based DRL methods such as DQN [1] employ DNNs to fit the state value function $V_\pi(s)$ or the state-action value function $Q_\pi(s, a)$ to guide the policy, and the policy-based DRL methods such as TRPO [4] directly apply DNNs to learn the policy function $\pi_\theta(a \mid o)$ that directly maps an observation o to an action a. There are also combination methods, such as DDPG [2], A3C [3], and SAC [5]. These methods are termed actor-critic algorithms, whose policy optimization is guided by a learned value function. Although DNNs have great function approximation ability for DRL methods, they also become a main source of poor model interpretability. The end-to-end trained models based on DNNs only consider the results, and the reasons why the results are produced are disregarded.

3.2 Causality between action and observation

For the behavior of an agent, there is a strong causality between the action and the observation. An agent will adopt an action a according to its observation o. Different observations will prodcue different actions. Particularly, assume a well-trained agent with a fixed policy $\pi_\theta(a \mid o)$

that

$$p(A = a \mid O = o) = 1. \tag{4}$$

Equation (4) means that an observation o will determine an action a of the agent. Therefore, we can also state that the observation is the cause and that the action is the effect. The causality can be denoted as

$$o \rightarrow a. \tag{5}$$

The observation spaces in terms of DRL problems, such as images, are usually high-dimensional. Current methods try to discover the relationship between actions and observations. It is difficult to directly obtain the causality between a raw high-dimensional observation and an action. Even though the methods can identify the important observation parts, it is still difficult to provide explicit reasons for an agent adopting a certain action.In the following subsection, we introduce an intermediate representation to construct the causal relationship and analyze the process based on information theory.

3.2.1 Intermediate representation embedding

For human beings, high-dimensional sensory signals will form abstract concepts in the brain's cognitive system, and decisions will be made by reasoning according to the concepts. It is reasonable to map the high-dimensional observation space into a compact representation space for decision-making. The causes in a compact representation space are easier to interpret than in a raw high-dimensional observation space. Therefore, we propose an embedding causality discovery approach for interpreting the behavior of a DRL agent. Instead of directly establishing the causality between an action a and an observation o, we introduce an intermediate representation v, which is a mapping representation from the observation space. We just need to determine the causality between the intermediate representation v and the action a. We obtain v from observation o, denoted as

$$v = \phi(o). \tag{6}$$

The mapping function $\phi(\cdot)$ might be a fixed nonlinear transformation that can map a certain observation or similar observations o into a certain representation v. The mapping function can be a one-to-one mapping or a many-to-one mapping, but not a one-to-many mapping. The representation v is determined by a certain observation o, formulated as

$$p(V = v \mid O = o) = 1. \tag{7}$$

Therefore, the process can be denoted as

$$o \mapsto v \rightarrow a. \tag{8}$$

The observation o maps to (denoted as $\mapsto$) a representation v, and then we use v to deduce action a. We just need to discover the causality between the intermediate representation v and the action a. Therefore, we can transform a high-dimensional observation space to a compressed representation space with good interpretability, in which the main causes of an agent's behavior are easy to obtain. However, this transformation will prompt a question. Will the compressed representation produce information loss and cause an agent's performance decline? To answer this question, we analyze the process based on information theory.

3.2.2 Information loss of observation space reduction

The observation space transformation may cause information loss for action decision-making. For instance, action selection will increase uncertainty when the transformation loses certain observation information. Formally, assume that given an observation o, the original action selection conditional probability distribution is

$$a \sim p(A \mid O = o). \tag{9}$$

The uncertainty of the action selection can be measured by conditional entropy as

$$\begin{aligned} \mathcal{H}(A \mid O) &= \sum_{o} p(o)\mathcal{H}(A \mid O = o) \\ &= -\sum_{o} p(o) \sum_{a} p(a \mid o) log_2 p(a \mid o) \\ &= -\sum_{o,a} p(a, o) log_2 p(a \mid o). \end{aligned} \tag{10}$$

The conditional entropy $\mathcal{H}(A \mid O)$ will indicate how much information the observation can provide for an agent's decision. A larger $\mathcal{H}(A \mid O)$ means a smaller amount of observation information, and vice versa. After we introduce the intermediate representation c for action decision-making, the action selection conditional probability distribution will be

$$a \sim p(A \mid C = c). \tag{11}$$

The uncertainty of the action selection on the basis of the intermediate representation will be

$$\mathcal{H}(A \mid V) = \sum_{h} p(h)\mathcal{H}(A \mid V = v). \tag{12}$$

The conditional entropy $\mathcal{H}(A \mid V)$ will indicate how much information the intermediate representation can provide for an agent's decision. A larger $\mathcal{H}(A \mid V)$ means a smaller amount of information, and vice versa. We assume a rational agent that can make the best decision estimation according to its observation. The more information is observed by the agent, the more certain the behavior of the agent. The information loss of the transformation from observation space O to intermediate representation space V can be defined by the difference between (12) and (10).

Definition 1 (Observation Information Loss) The observation information loss from observation space O to intermediate representation space V can be measured by the increased uncertainty in an agent's behavior according to (10) and (12), as

$$\mathcal{L}_V = \mathcal{H}(A \mid V) - \mathcal{H}(A \mid O). \tag{13}$$

A larger $\mathcal{L}_V$ means a larger amount of information loss when the observation space O maps to intermediate representation space V. A good intermediate representation space V should have low observation information loss. Hence, (13) can be a metric for optimizing the mapping function (6).

Intuitively, the observation information loss will reduce the performance of the agent. It is necessary to minimize $\mathcal{L}_V$ in the agent training phase, such as introducing an additional loss term to the objective function.

There is also a problem of how to estimate the probability distributions $p(a \mid o)$ and $p(o)$. It may be impossible to know the precise latent distributions of the environment. However, one approximation method is to count the historical observation o proportion and observation-action pair (o, a) proportion. The representation v proportion and observation-action pair (o, a) proportion can also be counted by the same experience because each observation o can be mapped to the corresponding representation v.

3.3 Hierarchical conceptual embedding

Due to the increasingly high-dimensional state space and action space, DRL agents have to explore exponentially growth spaces that face the curse of dimensionality. A hierarchical learning architecture is a way to solve this problem by dividing a large problem into small subproblems and by conquering each subproblem to solve the high-level problem [33]. The hierarchical learning architecture partially splits the complicated problem while detailing the multilevel learning process from an easy level to a difficult level, so it can make the model easy to interpret. To explicitly demonstrate the interpretable factors aligned with human concepts, the conceptual embedding technique [34] can be utilized to embed prior knowledge in DNN models. We propose combining a hierarchical learning architecture and conceptual embedding techniques to improve the interpretability for DRL agents, as illustrated in Fig. 2.

The DNN-based policy model of the DRL agent produces actions according to observations from the environment. Generally, the model is a black-box for humans, and we are concerned with only its input and output. The representation spaces of the hidden layers are often meaningless and too complicated to be understood by our concepts. Interpretation of all the variables of the hidden neurons surpass human cognitive ability. Nonetheless, we can catch the important factors and reduce them to a human-interpretable representation space.

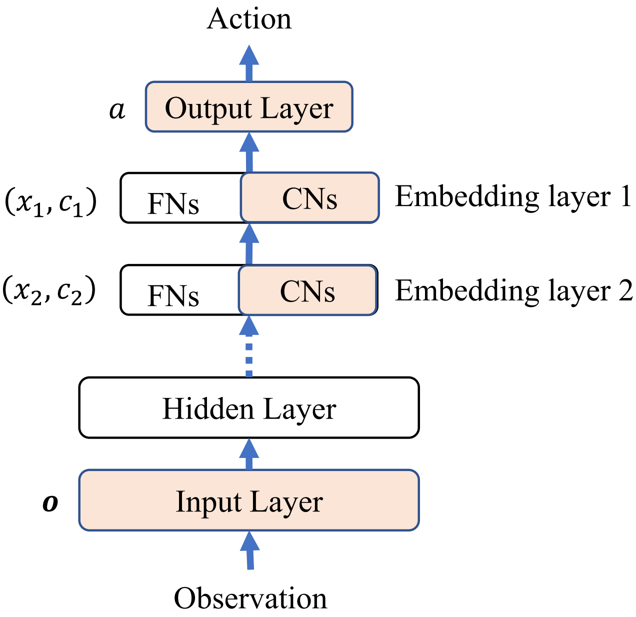


Fig. 2 Illustration of hierarchical conceptual embedding architecture for policy $\pi_\theta(a \mid o)$ approximation. FN means free neuron, and CN means conceptual neurons [34]

The DNN-based model can be regarded as a complex function that maps an observation o to an action a through multilayer nonlinear transformation,

$$a = f(o). \tag{14}$$

When we backtrack the DNN architecture from the output action a layer by layer. The computational result of a is obtained from a previous subfunction that selects the previous computational result (x_1, c_1) as input,

$$a = f_1(x_1, c_1), \tag{15}$$

where x_1 and c_1 denote the activation values of FNs and CNs respectively, in macropolicy representation layer 1, and $f_1(\cdot)$ is a nonlinear transformation function. The activation values of CNs c_1 are meaningful representations related to human concepts. Similarly, we can continue to backtrack to the next layer,

$$(x_1, c_1) = f_2(x_2, c_2), \tag{16}$$

where x_2 and c_2 denote the activation values of FNs and CNs respectively in macro-policy representation layer 2, and $f_2(\cdot)$ is another nonlinear transformation function.

We can also continue to backtrack to the input layer in a similar way. However, the shallow layers near to the input layer are often employed to extract shallow features, such as the edge information in images. We are mainly concerned with the deep layers near the output layer that have abstracted high-level representations, such as macro strategy targets.

When an event is interpretable, it usually means that we can ascertain the reasons and factors related to the event. For an agent, if we can grasp the factors that caused an agent to adopt a certain action, we can determine that the agent's behavior is interpretable. Therefore, to interpret the agents' behavior, we can discover some latent variables related to the adopted actions. The Pearson correlation coefficient (PCC) is used to measure the correlation between two variables. In our case, we can use the PCC to measure the relationship between the activation of conceptual neuron c and the corresponding probability of a specific action $p_a = p(a \mid c)$ as

$$\rho(c, p_a) = \frac{\sum_c (c - \overline{c})(p_a - \overline{p_a})}{\sqrt{\sum_c (c - \overline{c})^2}\sqrt{\sum_c (p_a - \overline{p_a})^2}}, \tag{17}$$

where $\overline{c}$ is the mean of c, $\overline{p_a}$ is the mean of p_a, and $\rho(c, a) \in [-1, 1]$. When $\rho(c, a) > 0$, there is a positive correlation between c and $p(a \mid c)$. The positive correlation becomes stronger as the value increases. When $\rho(c, a) < 0$, it means a negative correlation between c and $p(a \mid c)$. The negative correlation becomes stronger as the absolute value increases. The case $\rho(c, a) = 0$ also exists, which makes it difficult to interpret the relationship between c and $p(a \mid c)$. We hope that the latent embedding c has a strong correlation with the probability p_a of action a. The value of c will influence the probability of adopted action a. Therefore, to the original objective function for training, we can add an extra optimization objective, expressed as

$$\max \mid \sum_c \rho(c, p_a) \mid . \tag{18}$$

3.4 Perturbation technique for conceptual embedding

Saliency maps are usually applied to perturb the input layer and to observe the changes in the output layer. In our case, we apply the techniques of saliency maps with the hidden layers. As we embed certain interpretable prior-knowledge into the hidden layers, we can also generate saliency maps for the conceptual embedding factors using the related methods.

Typically, the saliency of specific features can be measured by policy difference as (19), by the value difference as (20), or by the Q-value difference as (21) between original observation o and perturbed observation o' with respect to feature f.

$$S[f] = \|\pi(o) - \pi(o')\|. \tag{19}$$

$$S[f] = \|V_\pi(o) - V_\pi(o')\|. \tag{20}$$

$$S[f] = \|Q_\pi(o, a) - Q_\pi(o', a)\|, \tag{21}$$

where $S[f]$ is the salient metric value of feature f, $\pi(\cdot)$ is the policy function that outputs an action according to a given observation, $V_\pi(\cdot)$ is the observation value function that directly judges the state value according to an observation, and $Q_\pi(\cdot, \cdot)$ is the observation-action value function that judges the state-action value according to an observation and an action. $\|\cdot\|$ denotes the norm function, typically the ℓ_2 norm, used to calculate the distance between two vectors.

Equations (20) and (21) presented in [19] judge the summarized difference over all actions rather than specific actions or factors. Therefore, they cannot highlight the saliency maps for the specific factor. Iyer et al. [26] adopted the Q-value difference as shown in (21), which is somewhat specific to the actions. However, they still could not provide distinct and sufficient interpretability for the DRL agents and disregared the exclusive effects on certain actions.

We propose to generate saliency values for the interpretable CNs in the hidden layers to provide deeper insight into the behaviors of DRL agents and to track the factors that influence the decisions. For the CN activation c, we can also measure the difference between the original activation and the perturbed activation with respect to feature f as

$$S_c[f] = \|f_c(o) - f_c(o')\|, \tag{22}$$

where $f_c(\cdot)$ denotes a function, which is a part of the feedforward computation of the DNN model, outputting activation c. For more general cases, the measured activation c can also be a CN or a group of CNs.

According to (22), we can generate the saliency values for all the CNs and actions. As a result, we can backtrack the salient decision factors from the action output layer.

On the other hand, we can also regard the CNs as input for the action output. Hence, we can also judge the influence of each CN activation c based on the final action decisions, represented as

$$S_a[c] = \|f_a(c) - f_a(c')\|, \tag{23}$$

where $f_a(\cdot)$ denotes the transformation function to the action output layer, and c' denotes the perturbed c.

3.5 Measurements of model performance

To improve an agent's performance and interpretability, we must quantify the performance and interpretability.

Informally, an agent can achieve good performance when it can receive enough information from the environment, and we can achieve a model with good interpretability when we can identify the key factors that drive the agent adopt a certain action.

3.5.1 Causality completeness

With regard to an agent, its observation will determine its action. To interpret the DRL model, we discover the key causal factors contained in the observation that influence the agent's action. We can measure the model interpretability from the aspect of whether the conceptual neurons can provide all the reasons for a certain action, that is, whether the set of conceptual neurons can provide enough information to determine an agent's action. Hence, our task is to measure the information completeness of the conceptual neurons, i.e., interpretable factors, with respect to the agent's observation.

Without any observation and prior knowledge, we can estimate that all the actions have the same probability according to the principle of maximum entropy. Assume that the number of available actions is m, the uncertainty of an agent's behavior A can be measured by information entropy, represented as

$$\mathcal{H}[p(A)] = -\sum_a p(a)log_2 p(a) = log_2 m. \tag{24}$$

Definition 2 (Complete Causal Factor Set) If a set of causal factors $V = \{V_1, \ldots, V_n\}$ completely determines an agent's action A, the set is a complete causal factor set. Formally, given the specific causal factor $v = \{v_1, \ldots, v_n\}$, the best estimated action of the agent can be determined as

$$p(A = a \mid v) = 1. \tag{25}$$

The behavior uncertainty can be defined by conditional entropy as

$$\mathcal{H}[p(A \mid V)] = -\sum_v p(v) \sum_a p(a \mid v)log_2 p(a \mid v) = 0. \tag{26}$$

The causal factor set V is a complete causal factor set for an agent's behavior A.

For general cases, we can define the degree to which a causal factor set can impact an agent's behavior.

Definition 3 (Causal Factor Set Completeness) Causal factor set completeness can be defined by information entropy as

$$N = \frac{\mathcal{H}[p(A)] - \mathcal{H}[p(A \mid V)]}{\mathcal{H}[p(A)]}, \tag{27}$$

where $N \in [0, 1]$.

Similar to Definition 3, we can define the causal factor set completeness with respect to conceptual causal factor C.

Definition 4 (Conceptual Causal Factor Set Completeness) Conceptual causal factor set completeness can be defined as

$$N_C = \frac{\mathcal{H}[p(A)] - \mathcal{H}[p(A \mid C)]}{\mathcal{H}[p(A)]}. \tag{28}$$

Conceptual causal factor set completeness describes the degree to which the conceptual factor set C impacts the agent's action. We can use N_C to measure the interpretability of an agent's behavior. A larger N_C means better interpretability. $N_C = 0$ means that the agent's behavior is totally unpredictable in terms of conceptual causal factors and that we cannot interpret the agent's behavior. The agent's behavior depends on conceptual causal factors when $N_C = 1$, and we can interpret all the agent's behaviors through the interpretable causal factors.

Definition 5 (Complementary Causal Factor Set) If the conceptual causal factor set C combined with another factor set X can determine the action, i.e.,

$$\mathcal{H}[p(A \mid C, X)] = 0,$$

then the factor set X is a complementary free causal factor set of conceptual factor set C. The causal factor set completeness of the combination (C, X) is

$$N_{C,X} = \frac{\mathcal{H}[p(A)] - \mathcal{H}[p(A \mid C, X)]}{\mathcal{H}[p(A)]} = 1.$$

3.5.2 Observation information and agent performance

Does a causal factor set contain the full observation information for decision-making? How much information will the intermediate causal representations lose, and to what degree will the model performance be reduced? With these questions, we need to analyze the relationship between the observation information and the agent's performance.

Definition 6 (Complete Observation Information) Assume that the best estimation of the action distribution with respect to an observation o is $\hat{p}(a \mid o)$. If a set of causal factors $v = \phi(o)$ has the same optimal estimation $\hat{p}(a \mid v) = \hat{p}(a \mid o)$, v contains the complete observation information.

Lemma 1 *Let $v_i \in V$ be the reduced mapping of a subset of observations $\{o\}_i \in O$, denoted by $v_i = \phi(\{o\}_i)$. For any set $\{o\}_i \in O$, if and only if all the observations $o \in \{o\}_i$ have the same optimal estimation $\hat{p}(a \mid o \in \{o\}_i)$, all v_i in* *the representation space V contains complete observation information.*

Proof For any representation $v_i = \phi(o)$, in which o belongs to any subset $\{o\}_i \in O$, the optimal estimation of the action distribution with respect to v_i will be the mean of $\hat{p}(a \mid o \in \{o\}_i)$, written as

$$\hat{p}(a \mid v_i) = \frac{1}{n_i} \sum_{o \in \{o\}_i} \hat{p}(a \mid o), \tag{29}$$

where n_i is the number of observations in $\{o\}_i$. If all the observations $o \in \{o\}_i$ have the same optimal estimation $\hat{p}(a \mid o \in \{o\}_i)$, it has an optimal estimation

$$\hat{p}(a \mid v_i) = \hat{p}(a \mid o), \tag{30}$$

i.e., the representation v_i has the same optimal estimation as its original observation o. Hence, all v_i in the representation space V contain complete observation information. Otherwise, if two observations $o_a, o_b \in \{o\}_i$ have different optimal estimations than $\hat{p}(a \mid o_a) \neq \hat{p}(a \mid o_b)$, an optimal estimation does not exist for v_i satisfying all observations in subset $\{o\}_i$, e.g.,

$$\hat{p}(a \mid v_i) = \hat{p}(a \mid o_a) = \hat{p}(a \mid o_b). \tag{31}$$

The representation v_i loses certain information that will degrade the agent performance. □

Definition 7 (Optimal Policy Offset) Assume that the best estimation of the action distribution with respect to an observation o is $\hat{p}(a \mid o)$. If a set of intermediate representations $v = \phi(o)$ produces a different best estimation $\hat{p}(a \mid v) \neq \hat{p}(a \mid o)$, the policy based on v is changed. The optimal policy offset $\mathcal{D}_v$ in terms of v can be measured by KL-divergence as

$$\mathcal{D}_{v,o} = \sum_a \hat{p}(a \mid v) log_2 \frac{\hat{p}(a \mid v)}{\hat{p}(a \mid o)}. \tag{32}$$

Intuitively, the observation information can promote the estimated performance of an agent.

Definition 8 (Agent Performance Upper Bound) In a certain task environment, an agent will have a performance upper bound $\sup(\overline{\mathcal{R}})$ given the limited observation space $\mathcal{O}$.

$$\sup(\overline{\mathcal{R}}) = \max_{\pi(a|o)} \mathbb{E}_{\pi(a|o)}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}] \tag{33}$$

After the representation transformation from $\mathcal{O}$ to V, there might be a reduction in the agent performance upper bound, represented as

$$\Delta\overline{\mathcal{R}} = \sup(\overline{\mathcal{R}}_O) - \sup(\overline{\mathcal{R}}_V), \tag{34}$$

where $\sup(\overline{\mathcal{R}}_O)$ denotes the agent performance upper bound in the condition of the original observation space, and $\sup(\overline{\mathcal{R}}_V)$ denotes the agent performance upper bound in the condition of the transformed representation space.

Theorem 1 *For any subset $\{o\}_i \in O$ mapping to a representation v_i in the representation space $V = \phi(O)$, all the observations $o \in \{o\}_i$ should have the same optimal estimation $\hat{p}(a \mid o \in \{o\}_i)$ to guarantee no reduction of the agent performance upper bound, i.e.,*

$$\Delta\overline{\mathcal{R}} = 0.$$

Proof According to Lemma 1, if and only if all the observations $o \in \{o\}_i$ have the same optimal estimation $\hat{p}(a \mid o \in \{o\}_i)$, an agent can achieve the same optimal policy estimation $\hat{p}(a \mid v_i) = \hat{p}(a \mid o \in \{o\}_i)$. According to (32), the optimal policy offset $\mathcal{D}_{v_i, o \in \{o\}_i} = 0$. For any observation o that belongs to any subset $\{o\}_i \in O$, the agent can still achieve optimal policy estimation after mapping to representation space V. The expected returns can remain the same, i.e.,

$$\mathbb{E}_{\pi(a|v)}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}] = \mathbb{E}_{\pi(a|o)}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}].$$

Hence, $\sup(\overline{\mathcal{R}}_V) = \sup(\overline{\mathcal{R}}_O)$. According to (34), we can obtain $\Delta\overline{\mathcal{R}} = 0$. □

Theorem 1 states that improving the interpretability of the DRL model does not mean sacrificing the model performance. In theory, an agent can achieve the performance upper bound if the interpretable representation space contains complete observation information. However, an agent cannot often learn the best policy because of the information loss of the representation space transformations and the instability of the DRL algorithms.

4 Experiments

First, we design a simple environment that can accurately calculate the information loss and an agent's performance upper bound to clearly demonstrate the analysis process. Second, we choose a complex environment, that does not know the latent state transition function to validate the effectiveness of the proposed method and the analyses.

4.1 Computational analyses in a naive environment

4.1.1 Environment setup

Assume a naive experimental environment in which a machine (environment) provides a box to a monkey (agent) at each time. There are three lights of red, green, and blue on the box that the monkey can observe and then choose whether to open the box. When the box turns red and green, there is a banana in it. The monkey can open the box and retrieve the banana (obtain a reward). In the other cases, the monkey will receive an electric shock (obtain a punishment) if it opens the box. The machine is defined as a tuple $< \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{G}, s_0, r, \gamma >$:

- $\mathcal{S} : \{0, 1, 2, 3\}$
- $\mathcal{O} : \{(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0)\}$
- $\mathcal{A} : \{0, 1\}$
- $\mathcal{T} : \begin{cases} p(s_{t+1} = s_t \mid s_t \neq 3, a_t = 1) = 1 \\ p(s_{t+1} = \xi \mid s_t = 3, a_t = 1) = 0.25 \\ p(s_{t+1} = \xi \mid s_t, a_t = 0) = 0.25 \\ \xi \in \{0, 1, 2, 3\} \end{cases}$
- $\mathcal{G} : \begin{cases} p(o = (0, 0, 0) \mid s = 0) = 1 \\ p(o = (0, 1, 1) \mid s = 1) = 1 \\ p(o = (1, 0, 1) \mid s = 2) = 1 \\ p(o = (1, 1, 0) \mid s = 3) = 1 \end{cases}$
- $s_0 : \begin{cases} p(s = 0) = 0.25 \\ p(s = 1) = 0.25 \\ p(s = 2) = 0.25 \\ p(s = 3) = 0.25 \end{cases}$
- $r : \begin{cases} r(s_t \neq 3, a_t = 1, s_{t+1}) = -1 \\ r(s_t = 3, a_t = 1, s_{t+1}) = 1 \\ r(s_t, a_t = 0, s_{t+1}) = 0 \end{cases}$
- $\gamma : 0.9$

Based on the naive example, it is easy to demonstrate the relationship between the observation information and the performance upper bound of an agent.

4.1.2 Analysis of an agent's performance

In the case that the agent can observe the full observation, the upper bound of an agent's performance is

$$\sup(\overline{\mathcal{R}}_{fo}) = \sum_{k=0}^{\infty} 0.9^k \times 0.25 \times 1 = 2.5.$$

We calculate the upper bound of the agent performance after the agent converges to $s = 3$ for computational simplicity. The upper bound performance can be reached when the agent adopts the policy $\pi(a \mid o)$ as

$$\pi(a \mid o) : \begin{cases} p(a = 0 \mid o = (0, 0, 0)) = 1 \\ p(a = 0 \mid o = (0, 1, 1)) = 1 \\ p(a = 0 \mid o = (1, 0, 1)) = 1 \\ p(a = 1 \mid o = (1, 1, 0)) = 1 \end{cases}$$

Assume that the agent's policy model extracts the features of the observation o to an intermediate causal factor $\tilde{s}$ that estimates the latent state of the environment before making a decision, as

$$o \mapsto \tilde{s} \rightarrow a$$

Let the mapping distribution be

$$\phi_1(o) : \begin{cases} p(\tilde{s}_1 = 0, \tilde{s}_2 = 0 \mid o = (0, 0, 0)) = 1 \\ p(\tilde{s}_1 = 0, \tilde{s}_2 = 1 \mid o = (0, 1, 1)) = 1 \\ p(\tilde{s}_1 = 1, \tilde{s}_2 = 0 \mid o = (1, 0, 1)) = 1 \\ p(\tilde{s}_1 = 1, \tilde{s}_2 = 1 \mid o = (1, 1, 0)) = 1 \end{cases}$$

We know that the best policy can be determined by causal factors $\tilde{s}_1, \tilde{s}_2$ as

$$\pi(a \mid \tilde{s}_1, \tilde{s}_2) : \begin{cases} p(a = 0 \mid \tilde{s}_1 = 0, \tilde{s}_2 = 0) = 1 \\ p(a = 0 \mid \tilde{s}_1 = 0, \tilde{s}_2 = 1) = 1 \\ p(a = 0 \mid \tilde{s}_1 = 1, \tilde{s}_2 = 0) = 1 \\ p(a = 1 \mid \tilde{s}_1 = 1, \tilde{s}_2 = 1) = 1 \end{cases}$$

Therefore, $\{\tilde{s}_1, \tilde{s}_2\}$ is a complete casual factor set according to Definition 2. However, for the set $\{\tilde{s}_1\}$, the best action cannot be determined. Assume that the occurrences of the environment states have equal probability. The best policy is

$$\pi(a \mid \tilde{s}_1, \tilde{s}_2) : \begin{cases} p(a = 0 \mid \tilde{s}_1 = 0) = 1 \\ p(a = 0 \mid \tilde{s}_1 = 1) = 0.5 \\ p(a = 1 \mid \tilde{s}_1 = 1) = 0.5 \end{cases}$$

According to (23), we can calculate the saliency values of the intermediate causal factors $\tilde{s}_1$ and $\tilde{s}_2$ as

$$\begin{aligned} S_a[\tilde{s}_1 = 0] &= \| f_a(\tilde{s}_1 = 0) - f_a(\tilde{s}_1' = 1) \| \\ &= p(a = 0 \mid \tilde{s}_1' = 1) \times 0 + p(a = 1 \mid \tilde{s}_1' = 1) \times 1 \\ &= 0.5 \times 0 + 0.5 \times 1 \\ &= 0.5, \end{aligned}$$

$$\begin{aligned} S_a[\tilde{s}_1 = 1] &= \| f_a(\tilde{s}_1 = 1) - f_a(\tilde{s}_1' = 0) \| \\ &= p(a = 0 \mid \tilde{s}_1 = 1) \times 0 + p(a = 1 \mid \tilde{s}_1 = 1) \times 1 \\ &= 0.5 \times 0 + 0.5 \times 1 \\ &= 0.5. \end{aligned}$$

Similarly,

$$S_a[\tilde{s}_2 = 0] = 0.5,$$

$$S_a[\tilde{s}_2 = 1] = 0.5.$$

According to (27), the causal factor set completeness of $\{\tilde{s}_1\}$ is

$$N_{\tilde{s}_1} = 0.5.$$

In the same way, we can calculate the causal factor set completeness of $\{\tilde{s}_2\}$,

$$N_{\tilde{s}_2} = 0.5.$$

According to Definition 5, $\tilde{s}_1$ and $\tilde{s}_2$ are complementary causal factor sets. The action can be determined by considering the two factors of $\tilde{s}_1$ and $\tilde{s}_2$ for the causal factor set completeness of the combination $N_{\tilde{s}_1, \tilde{s}_2} = 1$.

Notably, we can further reduce the dimension of $\tilde{s}$ with respect to another mapping distribution as

$$\phi_2(o):\begin{cases} p(\tilde{s}=0\mid o=(0,0,0))=1\\ p(\tilde{s}=0\mid o=(0,1,1))=1\\ p(\tilde{s}=0\mid o=(1,0,1))=1\\ p(\tilde{s}=1\mid o=(1,1,0))=1\end{cases}$$

We know that the best policy in terms of $\tilde{s}$ is

$$\pi(a\mid\tilde{s}):\begin{cases} p(a=0\mid\tilde{s}=0)=1\\ p(a=1\mid\tilde{s}=1)=1\end{cases}$$

For $\tilde{s}=\phi(o)$, the best estimation of the action distribution has not been changed, i.e., $\hat{p}(a\mid\tilde{s})=\hat{p}(a\mid o)$. Therefore, the causal factor $\tilde{s}$ preserves complete observation information that will not degenerate the model performance. The optimal policy offset is

$$\mathcal{D}_{h,o}=0.$$

In this example, it is also easy to calculate the lower bound of the agent performance when the agent always sets $a=1$ and remains on the states that $s\neq 3$ as

$$\inf(\overline{\mathcal{R}})=-10.$$

To demonstrate the upper bound of an agent's performance in this environment, we use an agent based on Q-learning and an agent based on DQN to fit the environment. The experimental results are illustrated in Fig. 3.

4.1.3 Analysis of the agent's performance reduction

Consider the POMDP case in which the agent can only partially observe the first dimension of the observation and where the occurrence of each state has equal probability. The conditional distribution $p(s\mid o)$ is

$$p(s\mid o):\begin{cases} p(s=0\mid o=(0,))=0.5\\ p(s=1\mid o=(0,))=0.5\\ p(s=2\mid o=(1,))=0.5\\ p(s=3\mid o=(1,))=0.5\end{cases}$$

The best policy of the agent would be to adopt $a=0$ all the time. Because it cannot verify whether $s=3$ or $s=2$ when it observes $o=(1,)$, it might maintain $s=2$ if it adopts $a=1$. The upper bound performance of the agent changes to

$$\sup(\overline{\mathcal{R}}_{po})=0.$$

Due to the information loss, there is a reduction in the agent performance upper bound, which is

$$\Delta\overline{\mathcal{R}}=\sup(\overline{\mathcal{R}}_{fo})-\sup(\overline{\mathcal{R}}_{po})=2.5.$$

To demonstrate the upper bound of an agent's performance in the POMDP environment, we again use the same two agents based on Q-learning and DQN to fit the partially observable case. The experimental results are illustrated in Fig. 4.

4.2 Experiments in a complex environment

4.2.1 Environment setup

To demonstrate the interpretable methods, we perform experiments in Atari game environments with the Open AI Gym API. The observation space of Atari environments is a video screen, which displays frames of color images of 210 × 160 × 3 pixels. The action space is comprised of 18 discrete numbers corresponding to the buttons of the joystick controller, and the different games use different

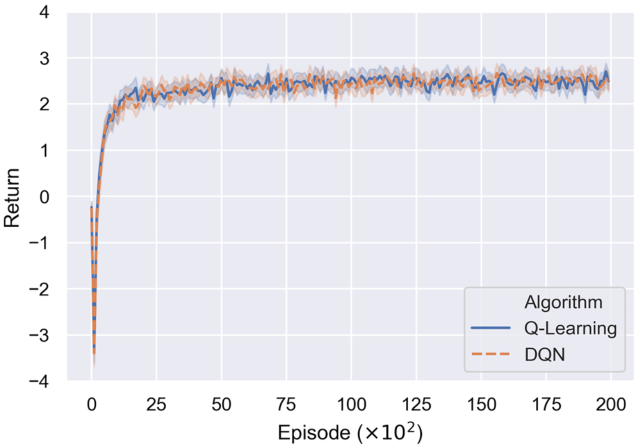


Fig. 3 The training process illustration of the return of two agents based on Q-learning and DQN in terms of $\tilde{s}$

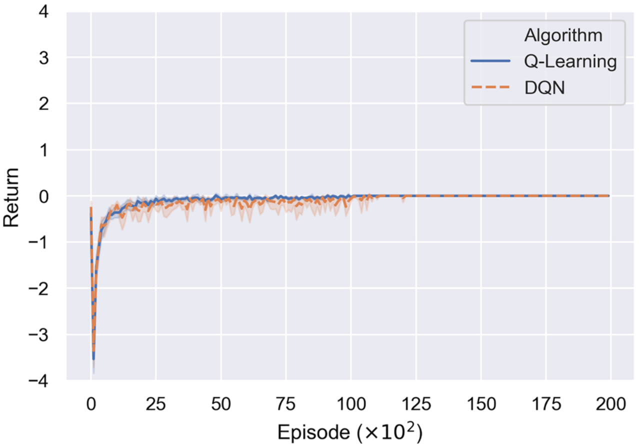


Fig. 4 Training process illustration of the return of two agents based on Q-learning and the DQN that can only observe the first dimension of o

minimal sets of numbers. The reward can generally be defined by the game scores.

We choose the game of Breakout-v4 as our experimental environment and design three observation spaces for the agent to examine the importance of the observable information for an agent's performance upper bound.

- Coordinates of the ball and pad.

  $$\mathcal{O}_1 : (x_1, y_1, x_2, y_2),$$

  where x_1 and y_1 are the horizontal coordinate of the ball and vertical coordinate of the ball respectively, and x_2 and y_2 are the horizontal coordinate of the pad and vertical coordinate of the pad respectively.
- Horizontal and vertical relative positions between the ball and the pad.

  $$\mathcal{O}_2 : (d_1, d_2, 0, 0),$$

  where $d_1 = x_2 - x_1$, and $d_2 = y_2 - y_1$. To keep the same dimensional space with $\mathcal{O}_1$, we use 0 to represent a constant zero dimension. Different states (x_1, y_1, x_2, y_2) and (x_1', y_1', x_2', y_2') will have the same (d_1, d_2) relative positions if $x_2 - x_2' = x_1 - x_1'$ and $y_2 - y_2' = y_1 - y_1'$. The agent will lose the information about the distance from the ball to the sidewall, which will make it difficult for the agent to predict the ball trajectories.
- Horizontal relative position between the ball and the pad.

  $$\mathcal{O}_3 : (d_1, 0, 0, 0).$$

  Compared with the second case, this case will further lose the vertical coordinate information of the ball. In this case, the agent cannot know whether the ball is near or far from the pad. Therefore, the agent's best policy may be to try to reduce the horizontal distance between the ball and the pad at all times.

First, we need to estimate an agent's performance upper bound given a certain observation space. To illustrate the effective information provided by the three different observation spaces $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{O}_3$, we use the DQN to probe the agent performance upper bound with respect to each observation space, as illustrated in Fig. 5. Note that any DRL algorithm can be utilized to probe the agent performance upper bound if only it can achieve good performance because our aim is to discover the best agent in a given observation space rather than to learn how to train the agent. In this environment, the DQN has achieved a state-of-the-art performance, and we use it for simplicity. We also design a rule-based agent that can achieve a baseline performance for comparison. The rule-based agent only utilizes the information of $\mathcal{O}_3$, and it follows simple control rules. For example, the pad moves left if the ball is on the left side of the pad and the pad moves right when the ball is on the right of the pad. We know that the upper bound performance of an agent will not be less than that of the rule-based agent when an agent can only receive the observation information in terms of the horizontal relative distance between the ball and the pad, i.e., observation space $\mathcal{O}_3$.

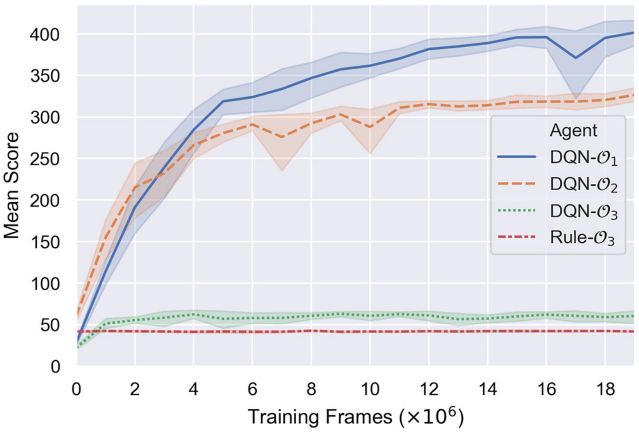


Fig. 5 Illustration of the training processes of the agents with different observation information in the Atari Breakout-v4 games. DQN-$\mathcal{O}_1$ denotes the agent based on the DQN with observation space $\mathcal{O}_1$, DQN-$\mathcal{O}_2$ denotes the agent based on the DQN with observation space $\mathcal{O}_2$, DQN-$\mathcal{O}_3$ denotes the agent based on the DQN with observation space $\mathcal{O}_3$, and Rule-$\mathcal{O}_3$ denotes the agent using defined control rules with the information of observation space $\mathcal{O}_3$. Each performance curve is illustrated by merging 10 training results

According to the experimental results, the agent performance upper bounds of $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{O}_3$ will not be less than 400, 325, and 60, respectively, represented as

$$\sup(\overline{\mathcal{R}}_{\mathcal{O}_1}) \geqslant 400,$$

$$\sup(\overline{\mathcal{R}}_{\mathcal{O}_2}) \geqslant 325,$$

$$\sup(\overline{\mathcal{R}}_{\mathcal{O}_3}) \geqslant 60.$$

The agent, which only receives the observation space of $\mathcal{O}_1$, reaches the performance declared in [1]. We can reasonably speculate that the observation space $\mathcal{O}_1$ contains complete observation information of the raw image observation space for this learning task to reach the performance upper bound. To a certain extent, assuming that the DQN agents have achieved the optimal policies, we can estimate the agent performance upper bounds by the experimental results.

4.2.2 Conceptual embedding in decision-making

In many cases, the prior knowledge of human beings is incomplete, which explains why we need DRL-based agents to explore more useful information and policy rules and can help us improve our knowledge in turn. Therefore, we set FNs and CNs in the conceptual embedding framework to learn latent representations that contain complementary

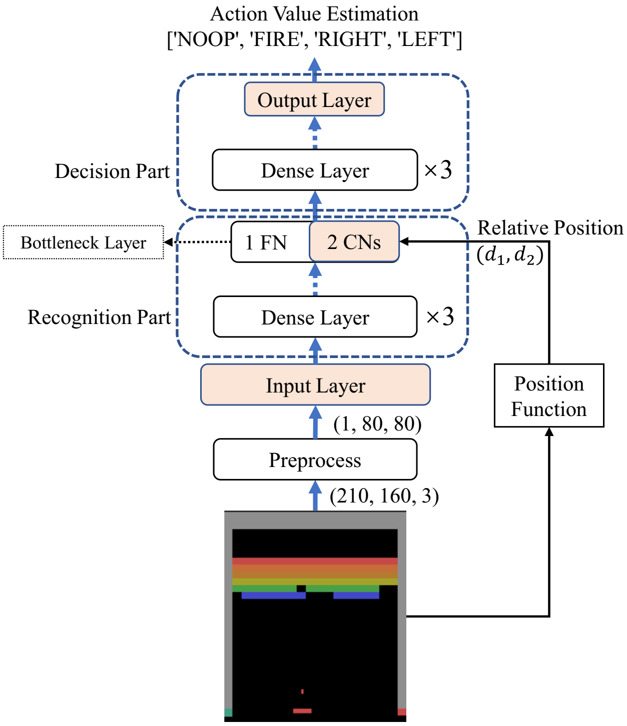


Fig. 6 Illustration of the conceptual embedding architecture for playing Breakout. The architecture can be regarded as two functional parts: the recognition part and decision part. The recognition part, which contains three dense layers, maps the high-dimensional observation data to a low-dimensional representation space. The size of the original color image is 210 × 160 × 3 pixels. We reduce the dimension to a gray image of 1 × 80 × 80 pixels that will not lose effective information for playing the game while it reduces the computational cost of the experiments. The decision part, which also contains three dense layers, estimates the best action according to the extracted information from the compacted representations. The information bottleneck has two conceptual neurons (CNs) and a free neuron (FN). The two CNs are trained to be aligned with the horizontal and vertical relative positions between the ball and the pad (d_1, d_2). The position function is artificially designed to extract the relative position from the image, and the relative position is employed to guide the training of the two CNs

information with respect to the prior knowledge of CNs. To observe the complementary information that can be learned by the DRL model, we set two CNs of the observation space $\mathcal{O}_2$ and an FN in the DRL model, as illustrated in Fig. 6. A CN is related to the relative horizontal position d_1 between the ball and the pad, and another CN is related to the relative vertical position d_2 between the ball and the pad. To embed the concept into the DRL model, we use supervised learning method to align the CNs with the relative horizontal and vertical positions (d_1, d_2). A designed function is employed to extract the positions of the ball and the pad to calculate the true (d_1, d_2). In the training process, we add an extra mean square error (MSE) loss between the activation $(\tilde{d}_1, \tilde{d}_2)$ of the CNs and the true (d_1, d_2).

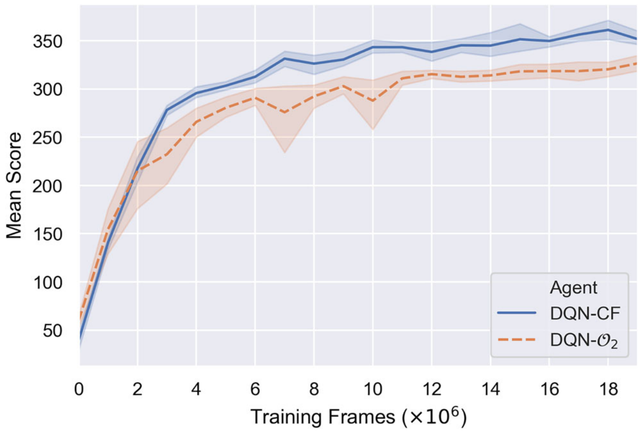


Fig. 7 Illustration of the training process of the conceptual embedding architecture compared with the DQN agent with observation space $\mathcal{O}_2$. DQN-$\mathcal{O}_2$ denotes the DQN agent with observation space $\mathcal{O}_2$. DQN-CF denotes the DQN agent that adopts the conceptual embedding architecture. The DQN-CF performance curve is illustrated by merging 5 training results

From the previous experimental results illustrated in Fig. 5, we know that the agent performance upper bound has a score of approximately the 325 when the observable information is provided by only two CNs of the observation space $\mathcal{O}_2$. The FN is used to extract extra information, which can increase the agent performance upper bound. The training result of the conceptual embedding architecture is illustrated in Fig. 7. We can observe that the agent performance upper bound of the conceptual embedding architecture has an increased score of 350 compared with the DQN agent with observation space $\mathcal{O}_2$. The experimental results validate that the FN could provide extra information to promote the agent performance upper bound.

However, we observe that the FN has not extracted complete complementary information with respect to observation space $\mathcal{O}_1$, in which an agent can reach a score of 400. In the experiments, we also try to compare the convergence of the DRL model both with conceptual embedding and without conceptual embedding. Unfortunately, we discovered that the DRL agent using the same architecture without conceptual embedding usually could not learn a good representation space and converge to an effective policy. The bottleneck layer of the free training DRL agent cannot extract enough effective information to promote the agent performance. Therefore, it is still a significant problem to extract effective information in the representation space transformation process of the free training DRL model. The problem could be a promising research approach to promote DRL algorithms.

To judge the importance of the causal factors, we can use the perturbation technique described in Section 3.4 to estimate the saliency values of the two CNs and FN. Specifically, we add Gaussian noise $\mathcal{N}(0, 0.1)$ to the activation values of neurons and calculate the mean difference value between the perturbed actions and the original actions. We use the well-trained DQN-CF model to run ten groups of tests, and each group runs 1000 steps. The experimental results of the calculated saliency values of the free causal factor and conceptual causal factor are illustrated in Table 2. Generally, a large saliency value of a conceptual causal factor corresponds to a small saliency value of a free causal factor. The saliency values of conceptual causal factor c and free causal factor f, denoted by $S_a[c]$ and $S_a[f]$ respectively, can be estimated by the normalized mean of the experimental results, as

$$S_a[c] \approx 0.7099,$$

$$S_a[f] \approx 0.0013.$$

From the saliency values, we can observe that the perturbed free causal factor has a small influence on the agent's decision. The conceptual causal factor almost dominates the agent's decision and verifies that the two CNs have extracted most of the observation information for the agent's decision.

Because the saliency values weight the influence of the causal factors for an agent's decision, the causal factor set completeness of conceptual causal factor c and free causal factor f, denoted by N_c and N_f respectively, can be estimated by the normalized saliency values as

$$N_c \approx 0.9982,$$

$$N_f \approx 0.0018.$$

Table 2 Saliency values of free causal factor and conceptual causal factor

No.	Free causal factor	Conceptual causal factor
1	0.000000	0.727728
2	0.002001	0.708854
3	0.002001	0.708854
4	0.001500	0.706927
5	0.001200	0.701940
6	0.001500	0.706784
7	0.001286	0.706815
8	0.001125	0.711589
9	0.001000	0.709634
10	0.001200	0.709971
Mean	0.0012813	0.7099096

5 Discussion

In the training process of the experiments, we discovered that the DRL methods were very sensitive to the manually predefined hyperparameters, such as the learning rate, replay memory size, size of training batches, and update frequency of the model. The training results were unstable. The DRL-based agent often cannot explore and converge to an effective strategy, especially in an environment of high-dimensional observation space. Thus, we have to spend substantial amount of time on debugging and searching for many hyperparameters. We suggest that it is indispensable to introduce prior knowledge to increase the interpretability of DRL agents, which can substantially reduce the debugging time and expedite the training process of the agents. Just as people need to teach their generations systematic knowledge and rules, human society can develop common sense. To improve the interpretability of DRL agents, prior knowledge embedding methods will be a potential approach. On the other hand, we determined that the free training DRL methods often could not learn an effective representation space while reserving the maximum observation information. The learned representation space has lost certain key information, which can determine the upper bound performance of an agent. However, in complex unknown environments, we have to estimate the agent performance upper bound. How can the estimated upper bound be guaranteed to be closed to the real upper bound in a certain range? It is still a problem to be researched. Reserving the maximum observation information of the representation space in the DNN-based model could be a promising approach for future work.

The proposed method is a general framework that can be applied in different DRL algorithms to improve model interpretability. In particular, the conceptual embedding techniques can provide interpretable cause factors for certain applications that require good model interpretability and reliability, such as automatic driving and healthcare. For example, an interpretable automatic driving agent should know the conceptual reason that a barrier in front of a car contributes to the braking action. We know that prior concepts are an indispensable component of the conceptual embedding model. However, an agent requires fewer training samples if the concepts can provide sufficient guidance and contain enough information. In many cases, compared with agents that collect training samples by freely exploring, it will cost less to introduce the prior concepts to DRL agents. Nevertheless, it will be interesting to investigate how FNs learn unknown complementary concepts with respect to prior concepts contained in CNs.

6 Conclusion

Interpretability is a key property of DRL agents. For example, why does an agent adopt a certain action? What is the key information of the observation that affect an agent's performance? We discovered that the difficulties in interpreting DRL agents are mainly attributed to the

DNN-based model. We analyzed the decision process of DRL agents based on information theory and identified a relationship between an agent's observable information and its performance upper bound. To make the DRL agents learn a more interpretable representation space, we proposed using a hierarchical conceptual embedding method and introducing prior knowledge to constrain the representation spaces of the DNN-based model. As demonstrated in the experiments, the method can explicitly indicate the action-driven factors, which can render the decision process of the DRL agent tractable and interpretable. In addition, the method has the benefit that the learning process is more efficient and the model converges faster than free training DRL models.

Acknowledgements This work is supported in part by China Postdoctoral Science Foundation under Grant Number 2021M693976, the Hunan Provincial Natural Science Foundation under Grant Number 2020JJ5367, and the Key Project of Teaching Reform in Colleges and Universities of Hunan Province under Grant Number HNJG-2021-0251.

References

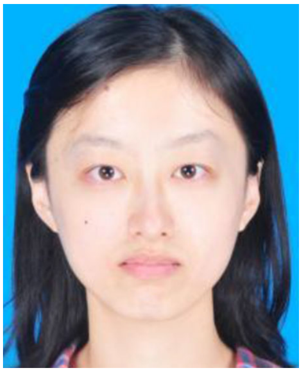
1. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533
2. Lillicrap T, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2016) Continuous control with deep reinforcement learning. In: International conference on learning representations (ICLR), pp 1–10
3. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. In: Proceedings of The 33rd international conference on machine learning, vol 48, PMLR, pp 1928–1937
4. Schulman J, Levine S, Abbeel P, Jordan M, Moritz P (2015) Trust region policy optimization. In: Proceedings of The 32rd international conference on machine learning, PMLR, pp 1889–1897
5. Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor
6. Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A et al (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359
7. Vinyals O, Babuschkin I, Czarnecki WM, Mathieu M, Silver D (2019) Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature 575(7782):350–354
8. Zieliński P, Markowska-kaczmar U (2021) 3D robotic navigation using a vision-based deep reinforcement learning model. Appl Soft Comput 110:107602
9. Saeedvand S, Mandala H, Baltes J (2021) Hierarchical deep reinforcement learning to drag heavy objects by adult-sized humanoid robot. Appl Soft Comput 110:107601
10. Jiang R, Wang Z, He B, Zhou Y, Li G, Zhu Z (2021) A data-efficient goal-directed deep reinforcement learning method for robot visuomotor skill. Neurocomputing 462:389–401
11. Zhang R, Wang Z, Zheng M, Zhao Y, Huang Z (2021) Emotion-sensitive deep dyna-q learning for task-completion dialogue policy learning. Neurocomputing 459:122–130
12. Tiwari A, Saha S, Bhattacharyya P (2022) A knowledge infused context driven dialogue agent for disease diagnosis using hierarchical reinforcement learning. Knowl-Based Syst 242:108292
13. Coronato A, Naeem M, De Pietro G, Paragliola G (2020) Reinforcement learning for intelligent healthcare applications: a survey. Artif Intell Med 109:101964
14. Ebrahimi S, Lim GJ (2021) A reinforcement learning approach for finding optimal policy of adaptive radiation therapy considering uncertain tumor biological response. Artif Intell Med 121:102193
15. Ciampi M, Coronato A, Naeem M, Silvestri S (2022) An intelligent environment for preventing medication errors in home treatment. Expert Systems with Applications 116434
16. Ilahi I, Usama M, Qadir J, Janjua MU, Al-Fuqaha A, Huang DT, Niyato D (2022) Challenges and countermeasures for adversarial attacks on deep reinforcement learning. IEEE Transactions on Artificial Intelligence 3(2):90–109
17. Heuillet A, Couthouis F, Díaz-rodríguez N (2021) Explainability in deep reinforcement learning. Knowledge-Based Systems 214:106685
18. Chen J, Li SE, Tomizuka M (2021) Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. IEEE Trans Intell Transp Syst, pp 1–11. https://doi.org/10.1109/TITS.2020.3046646
19. Greydanus S, Koul A, Dodge J, Fern A (2018) Visualizing and understanding Atari agents. In: Dy J, Krause A (eds) Proceedings of the 35th international conference on machine learning, vol 80, PMLR, pp 1792–1801. http://proceedings.mlr.press/v80/greydanus18a.html
20. Puri N, Verma S, Gupta P, Kayastha D, Deshmukh S, Krishnamurthy B, Singh S (2020) Explain your move: Understanding agent actions using specific and relevant feature attribution. In: International conference on learning representations, pp 1–14
21. Zahavy T, Ben-Zrihem N, Mannor S (2016) Graying the black box: Understanding DQNs. In: Balcan MF, Weinberger KQ (eds) Proceedings of The 33rd international conference on machine learning, vol 48, PMLR, pp 1899–1908
22. Simonyan K, Vedaldi A, Zisserman A (2014) Deep inside convolutional networks: Visualising image classification models and saliency maps. In: International conference on learning representations (ICLR). arXiv:1312.6034
23. Shrikumar A, Greenside P, Kundaje A (2017) Learning important features through propagating activation differences. In: Proceedings of the 34th international conference on machine learning, vol 70, PMLR, pp 3145–3153. http://proceedings.mlr.press/v70/shrikumar17a.html
24. Selvaraju RR, Cogswell M, Das A, Vedantam R, Batra D (2020) Grad-cam: Visual explanations from deep networks via gradient-based localization. Int J Comput Vis 128(8):336–359
25. Fong RC, Vedaldi A (2017) Interpretable explanations of black boxes by meaningful perturbation. In: IEEE International conference on computer vision (ICCV), IEEE Computer Society, pp 3449–3457. https://doi.org/10.1109/ICCV.2017.371
26. Iyer R, Li Y, Li H, Lewis M, Sundar R, Sycara K (2018) Transparency and explanation in deep reinforcement learning neural networks. In: AAAI/ACM Conference on artificial intelligence, ethics, and society, new orleans, LA, pp 144–150
27. Madumal P, Miller T, Sonenberg L, Vetere F (2020) Explainable reinforcement learning through a causal lens. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 2493–2500
28. Duong TD, Li Q, Xu G (2022) Stochastic intervention for causal inference via reinforcement learning. Neurocomputing 482:40–49

29. Sutton RS, Barto AG (2018) Reinforcement learning: An Introduction. MIT press
30. Nguyen DQ, Vien NA, Dang V-H, Chung T (2020) Asynchronous framework with reptile+ algorithm to meta learn partially observable markov decision process. Appl Intell 50(11):4050–4062
31. Zheng W, Jung T, Lin H (2022) The stackelberg equilibrium for one-sided zero-sum partially observable stochastic games. Automatica 140:110231
32. Kovařík V, Schmid M, Burch N, Bowling M, Lisỳ V (2022) Rethinking formal models of partially observable multiagent decision making. Artif Intell 303:103645
33. Pang Z-J, Liu R-Z, Meng Z-Y, Zhang Y, Yu Y, Lu T (2019) On reinforcement learning for full-length game of starcraft. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 4691–4698
34. Dai Y, Wang G, Li K-C (2018) Conceptual alignment deep neural networks. Journal of Intelligent & Fuzzy Systems 34(3):1631–1642

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

Yinglong Dai received B.S. and M.S. degrees in automation and control theory & control engineering from Northeastern University, China, in 2010 and 2012, respectively. He received a Ph.D. degree in computer science from Central South University, China, in 2018. He is a lecturer of College of Information Science and Engineering at Hunan Normal University, China. He is a postdoctoral research fellow of College of Liberal Arts and Sciences at National University of Defense Technology, China. From 2012 to 2013, he was an Electronic Engineer with the Research Institute of Intelligent Engineering, Sany Heavy Industry, Changsha, China. His research interests include multimodal deep learning, deep reinforcement learning, multiagent system, and healthcare process.

Haibin Ouyang received the M.S. and Ph.D. degree in control theory and control engineering from Northeastern University (NEU), Shenyang, China, in 2012 and 2016. Currently He is an Associate Professor at School of Mechanical and Electric Engineering, Guangzhou University, Guangzhou, china. He had published over 50 journal papers like Information Sciences, Applied Soft Computing, Soft Computing and Applied Mathematics and Computation. His current research interests are intelligent optimization algorithm, robotics path planning, artificial intelligence, and optimal control. Currently he is the editorial board member of Applied soft computing journal.



Hong Zheng received B.S. degree in Automation from Central South University of Forestry and Technology, China, in 2016. majoring in Electronic Information of School of Physics and Electronics at Hunan Normal University, China. His research interests include deep learning, deep reinforcement learning, and multimodal deep learning.

Han Long received the Ph.D. degree in Construction Engineering from Tongji University, in 2014. He was a Postdoctoral Researcher in Applied Mathematics. He was a visiting scholar at Warwick University. He is currently an Associate Professor of College of Liberal Arts and Sciences at National University of Defense Technology. He has published two books and eight SCI/EI articles. He has been a director or a member in 10 national grants and projects. He has received the First Prize of Military Science and Technology Progress Award in 2013. His current research interests include reinforcement learning, multi-agent system, and game theory.

Xiaojun Duan received B.S. and M.S. degrees in Applied Mathematics, and Ph.D. degree in System Engineering at the National University of Defense Technology, China, in 1997, 2000 and 2003, respectively. She is a professor of College of Liberal Arts and Sciences at the National University of Defense Technology, China. Her research interests include system modeling and evaluation, experimental design and data processing.

Affiliations

Yinglong Dai[1,2] · **Haibin Ouyang**[3] · **Hong Zheng**[4] · **Han Long**[1] · **Xiaojun Duan**[1]

Haibin Ouyang
oyhb1987@gzhu.edu.cn

Hong Zheng
459402067@qq.com

Han Long
lonyhan@163.com

[1] College of Liberal Arts and Sciences, National University of Defense Technology, Changsha, 410073, Hunan, China

[2] Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing, Hunan Normal University, Changsha, 410081, Hunan, China

[3] School of Mechanical and Electric Engineering, Guangzhou University, Guangzhou, 510006, Guangdong, China

[4] School of Physics and Electronics, Hunan Normal University, Changsha, 410081, Hunan, China