**Frontiers in Computing and Intelligent Systems**
ISSN: 2832-6024 | Vol. 3, No. 1, 2023

# Research on Reinforcement Learning Explainable Strategies Based on Advantage Saliency

**Dandan Yan***

College of Computer Science and Technology, Qingdao University, Qingdao, 266071, China
* **Corresponding author Email:** 1554011273@qq.com

**Abstract:** Deep reinforcement learning is increasingly being used in difficult environments with sparse rewards and high-dimensional inputs, and it performs well, but its decision-making processes are largely unclear and difficult to explain to end users. Saliency map methods explain an agent's behavior by highlighting state features relevant for the agent to take an action. In this paper, we use the perturbation-based saliency map method, propose the use of advantage function to replace the existing method of calculating state saliency, realize the combination of advantage function and perturbation-based saliency map. A saliency map is generated by noting the saliency of the dependent elements of the agent's chosen action in the Atari game environment. Experimental comparisons show that our method generates more accurate explanatory saliency maps.

**Keywords:** Advantage Function; Explain ability; Perturbation-based Saliency.

## 1. Introduction

Artificial Intelligence technology (AI) has a wide range of applications in various fields such as retail, technology, healthcare, science, and many more. At present, artificial intelligence is more inclined to solve practical problems, complete complex tasks automatically, and perform as little human intervention as possible in the implementation process. Although the efficiency and performance of AI continue to improve, the incomprehensibility reduces the practicality to a certain extent [1]. Therefore, the transparency of the model is identified as the key to improving user trust and user acceptance of the model. Given the importance of interpretability and the performance-transparency trade-off exhibited by AI models, where the more complex the model's internal functionality, the less clear it is about how a prediction or decision is made, a new field of research begins to emerge, referred to as "Explainable Artificial Intelligence (XAI)" [1]. The purpose of an XAI system is to make human behavior more understandable by providing explanations, in the case of an AI model, explaining what it did and what happened next in terms of its capabilities, representing this information in a way that a human (non-expert) user can understand ability, which ultimately enables them to predict the behavior of the model.

Reinforcement learning (RL) is used to describe and solve the problem of agents learning strategies to maximize rewards or achieve specific goals in the process of interacting with the environment. In the process of interacting with the environment, certain states in the environment give positive or negative rewards to the agent, and the agent learns a policy based on these reward signals, a mapping between states and behaviors [2]. However, RL also faces the problem of lack of explain ability, which limits its application in safety-sensitive or high-risk areas, such as unmanned driving technology and intelligent assistance for the elderly and the disabled. In order to overcome this weakness of RL, a large number of studies on “Explainable Reinforcement Learning (XRL)” have emerged. Hence, XRL is a sub-problem of XAI, which is used to enhance human understanding of models and optimize model performance.

Deep learning algorithms have achieved success in many areas, such as image classification [3], machine translation [4], image acquisition [5], etc., which belong to the salient application in the field of computer vision. Visual Saliency in the field of computer vision simulates the perception of the scene by the human visual system and extracts salient areas in the image (that is, areas of interest to humans) for image acquisition or classification. Visual Saliency associates the image content with the detection target, and marks the samples of the corresponding results in the image according to some salient areas in the image. Its most notable feature is that the target in the image is known, which is different from the policy-based saliency mapping in deep reinforcement learning (DRL). The models in DRL support interaction with challenging environments with sparse rewards and noisy high-dimensional inputs, and perform well (e.g., Atari games), but are often criticized as "black boxes". This also leads to the emergence of policy-based saliency mapping -- Policy Saliency. Policy Saliency explains to humans the basis of the agent's choice and highlights the factors that have a great influence on the policy (selected action) in the state characteristics. In RL systems, it is often used to evaluate the internal representation and behaviour of the agent over multiple frames of the environment, rather than evaluating the importance of specific pixels in classifying images. Except for the unknown target, the impact of the policy adopted by the agent on the target is also unknown. Inspired by the saliency method in the visual field, saliency maps are used to explain the agent's behaviour in order to allow people to understand the reason why the agent takes actions.

The saliency map method is divided into four types [6]: First, Jacobian Saliency. The gradient-based saliency mapping is extended to DRL, and the output logits are calculated relative to the Jacobian matrix of a bunch of images to obtain the importance of each pixel of the input image [7]. Second, Perturbation Saliency. It perturbs a certain part of the original input image, and calculates the logit difference generated by the original image and the modified image, and then obtains a significant map [8]. Third, Object Saliency. It performs directional masking for the object f in the image, then forms a perturbed image, detects the object in the input

image, and measures the importance of features by changing the Q value of masked and unmasked objects [10]. Fourth, Attention Saliency. It is more automated, using the attention mechanism to generate a saliency map [11].

In this work, we conduct research under the perturbation-based saliency mapping, and modify the saliency calculation based on previous work, such as Greydanus et al. [8]utilize the difference of value function and policy vector between original state and perturbed state to generate saliency map, Iyer et al. [10] compute a saliency map using the difference in action-value Q(s,a) between the original state and the perturbed state and so on. We propose a new calculation method combined with the advantage function, using the difference of the advantage function value between the original state and the disturbed state to calculate the saliency of the state features. It visualizes the features of an agent taking a selected action by generating a saliency map. We use our method to explain the actions taken by an agent in Atari games (Breakout and Space Invaders), generating saliency maps. And it can be shown that the explanatory saliency maps generated by our method are more accurate compared with Greydanus et al. and Gupta et al., by the experimental comparison figures.

The rest of this paper is organized as follows. The Section 2 provides the relevant basic knowledge definition based on the perturbation saliency method and the advantage function and some related works. The method of calculating the significance and the definition of the formula in our paper will be discussed in the Section 3. We perform experimental simulations of our problem in Section 4, generating experimental comparison plots. Finally, we conclude our paper and provide future work in Section 5.

# 2. Related Work

This paper focuses on a specific saliency mapping method in the field of computer vision—perturbation-based saliency mapping, and applies saliency mapping methods from this field to policy-based applications in RL. Perturbation-based saliency mapping methods study the properties of DNNs by perturbing the model's input and observing changes in the model's output. There are several advantages: first, observing the relationship between input-output is a natural and intuitive way to explore the class of black-box models; second, perturbation allows to analyze the model dynamically, instead of treating the model as an immutable object to study; finally, perturbation-based methods are generally applicable to any DNN, regardless of the model's architecture. Therefore, the perturbation-based saliency mapping method is also one of the important methods to explain the reason why the agent executes the policy (takes action) in the reinforcement learning model.

## 2.1. Perturbation-based saliency methods

The idea of perturbation-based methods is to explore DNNs by modifying some of the input information of the model, such as pixels in images, words in text, etc., and observing how the output of the model changes. Observing changes in the output can indicate whether the corresponding part of the input is important to the interpretation, and a perturbation of an element is considered important if it changes the output significantly. For example, for image classification, the classifier may assign pictures to different categories due to changes that are invisible to the naked eye; for neural network prediction, as the performance improves, the understanding of the prediction process becomes more and more important. A general approach to explaining machine learning systems is called input attribution, so both gradient-based and perturbation-based methods are attribution methods [12]. The intended purpose of attribution methods is mainly to measure global importance, while the main difficulty is the combinatorial explosion. If it were to iterate over all input elements and all possible combinations of them and observe how each one changes the output, this would be impractical in practice due to enormous cost constraints. Gradient-based attribution methods calculate the importance of elements by using model gradients, but tend to generate noise, especially in particularly large networks [13]. In addition, if the backpropagation algorithm is used, it needs to access the internal information of the model to generate the explanation, so it will be limited in use. Perturbation-based attribution methods cover or remove images to measure whether the actual change is in line with the prediction. The main challenge is how to select a reasonable element group and apply the perturbation for testing within a reasonable time. A notable advantage of perturbation-based methods is "dynamicity", that is, while many other methods treat the model as a constant object of study, perturbation allows it to be interrogated repeatedly and hypotheses about it can be developed and tested dynamically. Perturbation of images is performed by applying occlusion masks, blurring, or replacing parts of the image to remove or insert information. For example, if a picture contains a puppy, classify this picture, the prediction is +1 for the puppy, otherwise -1, if the image is disturbed, deleting the information related to the puppy should affect the output of the model, while for Disturbances in other areas should not affect the original prediction results. So it should be noted that it is very difficult to choose a perturbation that does not add any new information but modifies the original information. Therefore, the perturbation we choose in this paper is to partially cover the original image information without adding information.

Some works on the development of perturbation-based methods are as follows. The study of perturbation began with the pioneering work of Zeiler and Fergus [14], where a simple gray square mask was applied to an image, resulting in a change in the output of a classifier. The RISE approach proposed in Petsiuk et al. [15] empirically estimates importance by using a randomly masked version of the input image for model detection and obtaining the corresponding output. Yang et al. [16] proposed the MFPP method, which divides the input image into multi-scale fragments, and then masks some of the fragments as perturbations to generate a saliency map.

## 2.2. Advantage Function

The advantage function is the advantage of an action a relative to the average action in state s. From the perspective of quantitative relationship, it is the deviation of a random variable relative to the mean. In other words, the advantage function measures the plausibility of choosing a certain action in a certain state - directly giving the difference between the chosen action and the mean of all possible actions. Given an agent M, operating on a state space S, with a set of actions As, where $s \in S$, a policy $\pi$ is formally a map from states to the probability of choosing each possible action. The value function of state s under policy $\pi$ is the expected return that can be obtained from state s and following policy $\pi$, denoted as $V\pi(s)$.

And this state value function (namely V function) is just the most suitable form to measure the average performance of all possible actions in state s. Similarly, define the action value function under the policy π, that is, in the current state s, after executing the action a, follow the expected reward that can be obtained by following the policy π, denoted as Qπ(s,a). And this action value function (Q function) is just the most suitable form to measure the performance of taking action a in state s. That is, the state-value function is defined as:

$$V^{\pi}(s) = \mathrm{E}[G_t|S_t = s] = E_{\pi}[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s] \quad (1)$$

The action-value function is defined as:

$$Q^{\pi}(s,a) = E[G_t|S_t = s, A_t = a] = E_{\pi}[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s, A_t = a] \quad (2)$$

where the following equations represent the recursive relationship based on Bellman equation, state-value function and action-value function. In the formula, Eπ represents the expected value of the random variable when the agent follows the policy π, t is the time step, γ is the discount factor, and R is the reward return.

Therefore, the advantage function represents the rationality of selecting an action a under state s . The definition of the advantage function is as follows:

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s) \quad (3)$$

Use the difference between the action value function and the state value function to express the rationality of choosing action a in state s, that is, if the difference is greater than 0, which means that the selected action has an advantage on average and vice versa. Using advantage functions is an extremely important policy for deep reinforcement learning, especially for policy-based learning. The advantage function is naturally compatible with the policy gradient algorithm, just like the followed formula of the policy gradient:

$$g_{\theta} = E_{\tau}\left[\sum_{t=0}^{\infty} f(S_t, A_t)\nabla_{\theta} log\pi_{\theta}(A_t|S_t)\right] \quad (4)$$

where the $\nabla_{\theta} log\pi_{\theta}(A_t|S_t)$ part of the formula is the gradient direction of the logarithmic likelihood of taking action a in state s at time step t. If the parameter θ goes along this direction, the probability of choosing action a will be encouraged; on the contrary, if the parameter goes along the negative direction of the gradient, the probability of choosing action a will be reduced. Therefore, in this paper, we also follow this idea when applying the advantage function to calculate the feature salience, which will be introduced in the Section 3.

Some work in the paper using the advantage function is as follows. Qi et al. [17] introduce the discrete normalized advantage function into DQL, and separates the Q-value function into a state value function and an advantage item, and uses the deterministic policy gradient descent (DPGD) algorithm to avoid the Q-value of the action is calculated unnecessarily. Wang et al. [7] use the advantage function for the dueling network architecture system proposed by himself, and promotes learning between different operations without changing the underlying reinforcement learning algorithm.

# 3. Methodology

In this work, our main research content is based on the agent (policy network) that has been trained, focusing on visually showing which characteristics the agent makes corresponding decisions in the input state, that is, hoping to explain the agent's decision to take a selected basis for the action. In the experiment, Asynchronous Advantage Actor-Critic (A3C) is used as the reinforcement learning framework of the agent, which can put Actor-Critic into multiple threads for synchronous training, and has achieved excellent performance in the Atari environment [18]. Therefore, we used a single DNN to evaluate the policy network and the critic network.

The state is parameterized into a state feature F. For a single feature f ∈ F, the significance score *Score[f]* is used to indicate the importance of the *f*-th feature of the state *S* on the agent's action *a*. In order to understand the decision-making information of the actions taken by the agent, we found that it is necessary to use the $Q^{\pi}(s, a)$ value of each time step *t* and the mean value $V^{\pi}(s)$ of the action to construct and visualize the saliency map, which is inspired by previous work.

## 3.1. Advantage Function

The relevant definition of the advantage function has been introduced in related work. In our paper, we use the advantage function when calculating the state feature salience Score[f]. By observing the positive or negative value of the advantage function value of the state feature, we can judge the advantage change of the agent in the disturbed state and the original state when choosing the same action. If the value changes greatly, it means that the part of the pixel feature of Gaussian blur is an important factor for the agent to select action a, and it should be marked obviously; otherwise, if the value of the advantage function has a small change, it indicates that part is not an important factor for the agent to choose action a, so it is not marked obviously.

## 3.2. Perturbation-based Saliency

The outline of the perturbation-based saliency mapping method is as follows: First, for each feature f, the state s is perturbed to s'. For example, in an Atari game, the input image is perturbed by adding a Gaussian blur centered at the f-th pixel. Second, the expected reward V(s) is obtained by querying the agent M in the original state s. Similarly, the expected reward value V(s′) of agent M in the perturbed state s' is obtained. Finally, calculate Score[f] based on the difference between the original state and the disturbed state.

Below are some saliency measurement methods that work on perturbation-based saliency mapping. For example, Greydanus et al. [8] exploit the difference in the value function or policy vector between the original state and the perturbed state to generate a saliency map. The formula for calculating the saliency measure is:

$$Score_{Greydanus}[f] = \frac{1}{2}|V(s) - V(s')|^2 \quad (5)$$

Iyer et al. [10] compute a saliency map using the difference in action-value (Q(s,a)) between the original state and the perturbed state. The formula is defined as follow:

$$Score_{Iyer}[f] = Q(s, \hat{a}) - Q(s', \hat{a}) \quad (6)$$

Gupta et al. [9] proposed specific and relevant feature attribution (SARFA). In the article, the Score[f] is still calculated using the Q value between the original state and the disturbed state, but two additional attributes are added: Specificity and Relevance.

In our work, the method we propose to calculate the feature saliency Score[f] is different. We start by querying the agent M chooses the expected reward value of action a in state s to get Q(s, a), and obtains the mean value V(s) of all actions selected in state s, and then obtains the advantage function value Aπ (s, a). Similarly, the advantage function value Aπ(s′,a) of the agent M choosing the same action a in the disturbed state s′ is obtained. Finally, calculate the difference

between the advantage function value of the original state and the disturbance state to get Score[f]. The specific calculation method is in Section 3.3.

### 3.3. Calculation of saliency

Combining the advantage function with the perturbation-based saliency mapping method, the input image is perturbed to change the input state to observe the behavior change of the agent. Score[f] is calculated by calculating the difference between the advantage function value of the same action selected in the original state and the disturbed state, and judges whether to mark the disturbed state feature as saliency according to its value, and finally visualizes the important features for agent to take a selected action.

In order to focus on the impact of changes on the agent's choice of action a, we are interested in whether the reward of a will change with the disturbance. As described in Greydanus et al. [8], they directly use V(s) to obtain the reward value in state s, but do not pay attention to the effect of a specific action a, because they aggregate together the changes of all actions in state s. In our work, we focus on the effect of specific actions, Q(s, a) is used to obtain the reward of the agent taking action a in state s. To focus on the specific actions taken, we normalize the values using the softmax over Q-values:

$$q^{\pi}(s,a) = softmax\left(Q^{\pi}(s,a)\right) \tag{7}$$

Introduced in the previous relevant sections, the advantage function is defined as $A\pi(s, a) = Q\pi(s, a) - V\pi(s)$. For the first term, we use the standardized Q-value $q\pi(s,a)$ to represent the expected reward value of agent M taking action a in state s. For the second term, the mean value of the rewards of all actions in the state s is calculated, so the advantage function is as follows:

$$A^{\pi}(s,a) = q^{\pi}(s,a) - \frac{1}{|a|}\sum_{a'} q^{\pi}(s,a') \tag{8}$$

where the $A\pi(s,a)$ obtained by the formula represents the advantage value of following the policy $\pi$ agent M to take action a relative to the average in the state s. the same way. The dominance function value $A\pi(s',a)$ of the perturbed state $s'$ is calculated in the same way. To compute the saliency Score[f], we use a vector $\Delta A$ denoting the relative dominance of the action a to be explained between the original state and the perturbed state. Therefore, the higher the value of Score[f], the more important the feature is for the agent to take action a, and the lower the value, it shows that the effect does not depend on the feature f very much. So $\Delta A$ is calculated as follows:

$$\Delta A = A^{\pi}(s,a) - A^{\pi}(s',a) \tag{9}$$

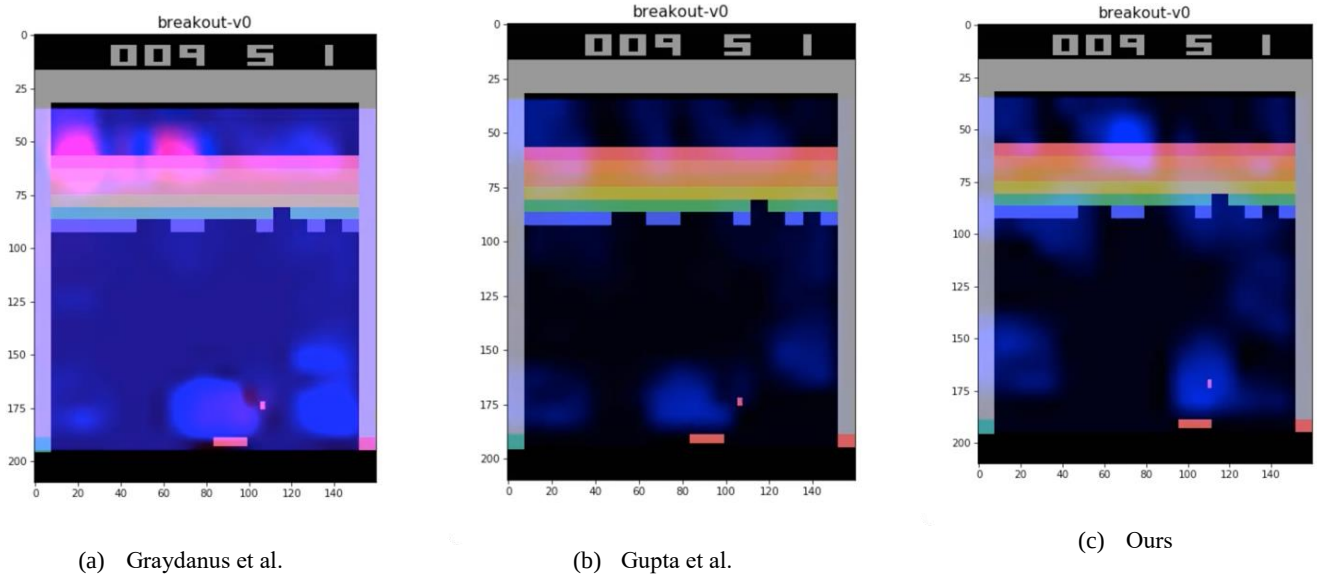
where each item of $\Delta A$ is a saliency score of a feature f.

## 4. Experiment

To show that our method produces more focused saliency maps than existing methods, we use Atari (Breakout and Space Invaders) as the experimental scene. These environments were chosen because each environment presented a different set of challenges, and deep reinforcement learning algorithms have previously been studied to far exceed human performance in these environments.

### 4.1. Settings

The input at each time step in our experiments is a preprocessed version of the current frame, which includes clipping the game space and normalizing the values to [0,1]. All of our Atari agents share the same network architecture, with inputs processed by 4 convolutional layers, an LSTM layer, and a fully connected layer of n+1 unit, where n is the dimensionality of the action space. For all our experiments, we only assume access to the agent's Q(s,a) function. To show that our method generates saliency maps that are more focused than those generated by Greydanus et al. and Gupta et al., we compare two Atari games (Breakout and Space Invaders).




(a) Graydanus et al. (b) Gupta et al. (c) Ours

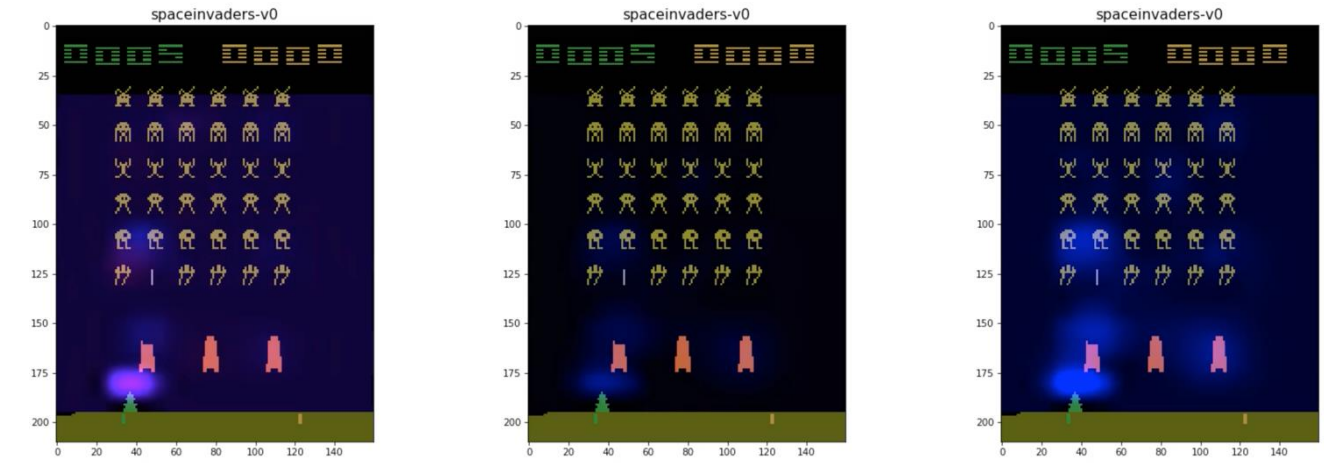
**Figure 1.** Comparison results under Breakout.

### 4.2. Compare results

In the game Breakout, the main task is to bounce the ball by manipulating the baffle and destroy the bricks above it with the ball. Previous work [9] pointed out that a strong breakout agent exploits a tunneling policy. During tunneling, the agent repeatedly points a ball at a certain area of the brick in order to break through it. This policy allows the agent to earn dense rewards by bouncing balls between surrounding walls and bricks. Figure 1 shows the results, our method

accurately highlights the regions of the input image, focusing on the location more accurately. Whereas the method of Greydanus et al. highlights several regions in the input image that are not relevant for interpreting the actions taken by the agent. The region of interest of Gupta et al. is also slightly messier than our method.

In the game SpaceInvaders, the main task is to control the turret to hit the aliens, and at the same time avoid the attacks of the aliens, but there will be three bunkers to provide protection. When there is no saliency map, it is noticeable through the game that the agent learns an aiming-like policy, but we are not sure whether it is shooting at dense clusters of enemies, or aiming at one of them as a single target. Figure 2 shows the results of applying saliency mapping to this agent. We found that in our method, the agent pays more attention to the enemies in the front area directly in front of the muzzle, which is also easy to explain, the enemies in front are better to attack, and the danger is also greater. And the agent in our method also pays attention to the position of the cover, which makes it better protect itself from being killed. The method of Greydanus et al. do not pay attention to the position of the bunker, only the enemy. The area of interest of Gupta et al. is relatively less obvious.




(a) Graydanus et al. (b) Gupta et al. (c) Ours

**Figure 2.** Comparison results under SpaceInvaders.

## 5. Conclusion

In order to explain the factors that cause agents to perform actions in deep reinforcement learning, we combined the perturbation-based saliency method with the advantage function, calculated the advantage function value of the same action before and after the perturbed state, and visualized the results in the form of saliency map. Compared with previous work, the effect of our experiment is more obvious. In future work, we will continue to work on using the advantage-based saliency-based interpretability method in more important applications, generating saliency maps so that non-professionals can better understand the basis for the agent to perform actions. The research can be extended to multi-agent systems, where the method of computing saliency is expected to play a better role in solving multi-agent problems.

## References


[1] Puiutta, Erika, and Eric MSP Veith. "Explainable reinforcement learning: A survey." Machine Learning and Knowledge Extraction: 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25–28, 2020, Proceedings 4. Springer International Publishing, 2020.

[2] Sutton, Richard S., and Andrew G. Barto. Introduction to reinforcement learning. Vol. 135. Cambridge: MIT press, 1998.

[3] Affonso, Carlos, et al. "Deep learning for biological image classification." Expert systems with applications 85 (2017): 114-122.

[4] Singh, Shashi Pal, et al. "Machine translation using deep learning: An overview." 2017 international conference on computer, communications and electronics (comptelix). IEEE, 2017.

[5] You, Quanzeng, et al. "Image captioning with semantic attention." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[6] Atrey, Akanksha, Kaleigh Clary, and David Jensen. "Exploratory not explanatory: Counterfactual analysis of saliency maps for deep reinforcement learning." arXiv preprint arXiv:1912.05743 (2019).

[7] Wang, Ziyu, et al. "Dueling network architectures for deep reinforcement learning." International conference on machine learning. PMLR, 2016.

[8] Greydanus, Samuel, et al. "Visualizing and understanding atari agents." International conference on machine learning. PMLR, 2018.

[9] Gupta, Piyush, et al. "Explain your move: Understanding agent actions using focused feature saliency." arXiv preprint arXiv:1912.12191 (2019).

[10] Iyer, Rahul, et al. "Transparency and explanation in deep reinforcement learning neural networks." Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society. 2018.

[11] Mott, Alexander, et al. "Towards interpretable reinforcement learning using attention augmented agents." Advances in neural information processing systems 32 (2019).

[12] Ivanovs, Maksims, Roberts Kadikis, and Kaspars Ozols. "Perturbation-based methods for explaining deep neural networks: A survey." Pattern Recognition Letters 150 (2021): 228-234.

[13] Khakzar, Ashkan, et al. "Improving feature attribution through input-specific network pruning." arXiv preprint arXiv:1911.11081 (2019).

[14] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13. Springer International Publishing, 2014.

[15] Petsiuk, Vitali, Abir Das, and Kate Saenko. "Rise: Randomized input sampling for explanation of black-box models." arXiv preprint arXiv:1806.07421 (2018).

[16] Yang, Qing, et al. "Mfpp: Morphological fragmental perturbation pyramid for black-box model explanations." 2020 25th International conference on pattern recognition (ICPR). IEEE, 2021.

[17] Qi, Chen, et al. "Deep reinforcement learning with discrete normalized advantage functions for resource management in network slicing." IEEE Communications Letters 23.8 (2019): 1337-1341.

[18] Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." International conference on machine learning. PMLR, 2016.

[19] Zahavy, Tom, Nir Ben-Zrihem, and Shie Mannor. "Graying the black box: Understanding dqns." International conference on machine learning. PMLR, 2016.