Attribution-based Salience Method towards Interpretable Reinforcement Learning

Yuyao Wang, Masayoshi Mase, and Masashi Egi
Research & Development Group
Hitachi, Ltd.
{yuyao.wang.fe@hitachi.com, masayoshi.mase.mh@hitachi.com, masashi.egi.zj@hitachi.com}

Abstract

Reinforcement Learning (RL), a general learning, predicting and decision-making paradigm, has achieved great success in a wide range of games and robotics. Recently, RL has also proven its worth in real world scenarios, such as adaptive decision control and recommendation. It is promising to deploy RL in the real world to gain real benefits. However, RL is criticized for its being black-box. The real systems are owned and operated by humans, who need to be reassured about the controller's intentions and insights regarding failure cases. Therefore, policy explanation is important. Existing methods towards interpretable RL include Jacobian saliency map and perturbation-based saliency map, which are limited to visual input problems. To model the complicated real-world use cases, numerical data are widely employed. In this paper, we propose an attribution-based salience method that is applicable on visual and numerical input. We aim to understand RL agents in terms of the information they attend to for decision making. We verify our method with a machine control use case. Explanations we provided are understandable to both AI experts and non-experts alike. (short paper)

Introduction

Reinforcement learning (RL) is a general learning, predicting and decision-making paradigm. It provides solution methods for decision making problems. RL has achieved remarkable success in a broad range of game-playing, continuous control and robotics. Deep Reinforcement Learning (Deep RL) exceeded human baseline in Atari games (Mnih et al. 2015) and beat professional human player in GO (Silver et al. 2016). Recently, RL has also proven its worth in real world scenarios, such as production system and recommendation. Growing numbers of real-world use cases show that it is promising to deploy RL in the real world to gain real benefits. However, there are many issues for RL to be widely deployed in the real world. One of them is about RL being black box. The real systems are owned and operated by humans, who need to be reassured about the controller's intentions and insights regarding failure cases. For this reason, policy explanation is important.

Research on Explainable Artificial Intelligence (XAI) is becoming increasingly popular these years. One trend of research in providing post-hoc explanations focuses on how to explain individual predictions by learning local approximation of a model. SHAP (Lundberg and Lee 2017) is one of the state-of-art techniques. SHAP decomposes the AI prediction into the sum of the contribution degree of each input feature. SHAP works well for regression and classification problems, while it does not work well for RL. We will discuss this issue in latter sections.

Existing methods for explaining deep RL include Jacobian saliency map (Zahavy, Ben-Zrihem, and Mannor 2016) and perturbation-based saliency map (Greydanus et al. 2017). These tools use visual inputs test beds and are not applicable to problems with numerical feature values. There is a need for an explainable method for numerical inputs which are widely employed to model complicated real-world use cases. For example, in our machine control use case, RL rely on sensor data to control the machine.

One of the challenges that arise in reinforcement learning, and not in other kinds of learning, is the trade-off between exploration and exploitation (Sutton and Barto 2018). Another key feature of reinforcement learning is that it explicitly considers the whole problems of a goal-directed agent interacting with an uncertain environment (Sutton and Barto 2018). These features make the explanation requested in RL different from other approaches.In this paper, we want to find out how RL agents make decisions. We aim to understand RL agents in terms of the information they attend to for decision making.

The contribution of the paper is as follows:

- Clarify the problem on application of attribution methods for RL
- Generate attribution by background data selection with domain knowledge for interpretable RL
- Evaluate on machine control use case

Prerequisite

Attribution Method

The concept of attribution is studied in various papers, such as integrated gradient (Sundararajan, Taly, and Yan 2017)

Copyright © 2020 held by the author(s). In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2020 Spring Symposium on Combining Machine Learning and Knowledge Engineering in Practice (AAAI-MAKE 2020). Stanford University, Palo Alto, California, USA, March 23-25, 2020. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

and SHAP (Lundberg and Lee 2017). We give the definition of attribution following the statement in paper above.

Definition (Attribution):

Suppose we have a function $f : R^n \rightarrow R^m$ that represents a model, and an input $x = (x_1, ..., x_n) \in R^n$. An attribution of the prediction at input x relative to a baseline input $x\ '$ is a vector $\phi(x, x') = (\phi_1, ..., \phi_n) \in R^n$ where ϕ_i is the contribution of x_i to the prediction $f(x)$.

Shapley Value

Let f be the original prediction model and g the explanation model. The explanation model uses simplified inputs x' that map to the original inputs through a mapping function $x = h_x(x')$. Assuming $g(z') \approx f(h_x(z'))$ whenever $z' \approx x'$, the attribution method is defined as

$$g(z') = \phi_0 + \sum_{i=1}^{N} \phi_i z_i' \tag{1}$$

where $z' \subset \{0, 1\}^N$, N is the number of simplified input features, and $\phi_i \subset R$.

Assume four axioms such as efficiency, symmetry, dummy and additivity, the attribution is proved to have a single unique solution known as Shapley value (Shapley 1953) in cooperative game theory:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(N - |z'| - 1)!}{N!} [f_x(z') - f_x(\ z' \backslash i)] \tag{2}$$

where $|z'|$ is the number of non-zero entries in z' and $z' \subseteq x'$ represents all z' vectors where the non-zero entries are a subset of the non-zero entries in x'.

SHAP (SHapley Additive exPlanation) (Lundberg and Lee 2017) is a state-of-art explanation framework using Shapley value. The SHAP value is defined as an approximation to equation 2:

$$f_x(z') = f(h_x(z')) = E[f(z)|z_S] \tag{3}$$

where S is the set of non-zero indexes in z'.

Thus, SHAP value attributes to each feature the change in the expected model prediction when the feature is toggled on. They explain how to get from the base value $E[f(z)]$ that would be predicted if we did not know any features to the model f(x).

Problem of Attribution Methods on RL

The effect of each feature on a prediction is calculated based on a baseline prediction. The input features of the baseline prediction (or base value) are called background data (or reference data). Usually, the background data is set to zero or the average value of the training dataset in prediction tasks. In image recognition tasks, the background data can be a black image, i.e., all pixel intensities are zero for example. However, reinforcement learning proceeds by making training data by exploitation and exploration in uncertain environment. The dynamic learning process of a deep RL agent makes some problems to use SHAP. According to our experiment results, different selection of the background data will lead to different explanation results. We want to solve this problem in our work. Also, we want to understand deep RL agents in terms of what information of the environment they take to make decisions. This match the intuition of post-hoc explanations. Among the group of attribution methods, we use SHAP to analyze RL. We focus on the agent trained on Deep Q-Network (DQN) (Mnih et al. 2015). Figure 1 shows the intuition of our problem setting.

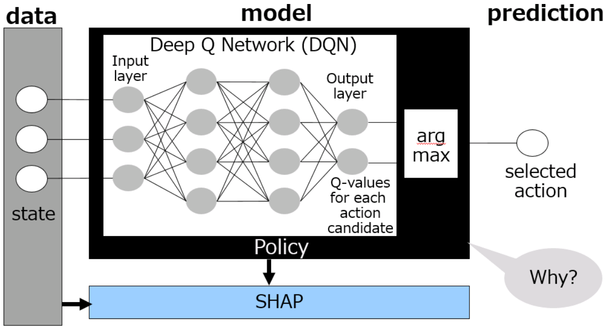

Figure 1: Problem Setting

Attribution-based Salience Method towards interpretable RL

Attribution generation

Deep RL agents learn what to do so as to maximize the cumulative reward or the value. In DQN, the value is approximated by Q-function. The output of the DQN model is the Q-value for each action candidate. We adjust the original DQN model with $argmax$ operator in order to bridge the gap between the outputs and the action selection (decision-making). We load the trained DQN model f_{model} from deep RL agents and adjust the output by adding an activation layer. Note that this is done after the training process of our deep RL agent. In this way, the output of the modified model $f_{modified}$ is the selected action with higher Q-value.

Next, we deal with the issue of background data. Instead of using one fixed set of background data, we embed domain knowledge to select the background data according to the environment RL interacts with.

In RL environment, we make a transition from one state s to the next state s' by performing some action a and receive a reward r. We load the learnt policy trajectory of our deep RL agent along the learning process and regard it as the dataset of our approach. Let $P_{1:t}$ denote the trajectory of learnt policies from time step 1 to time step t, the trajectory file contains the state s and action a pair at each time step t. Therefore, we have $P_t = P_t(s_t, a_t)$. Our background data is selected according to the trajectory $P_{1:t} = P_{1:t}(s_{1:t}, a_{1:t})$.

Then we calculated the attribution of each input, which is the SHAP value with our trained model and selected background data.

Salience Method

The higher value of attribution means bigger impact of the input on the output of the model. The impact of the input is

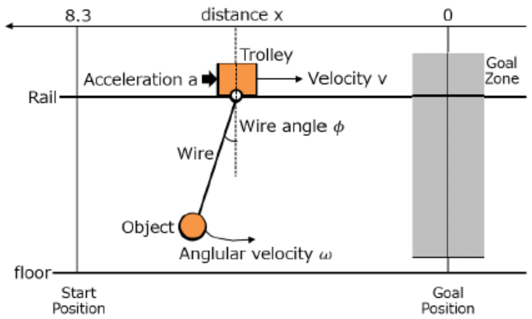

Figure 2: Image of Automatic Crane Control Use Case

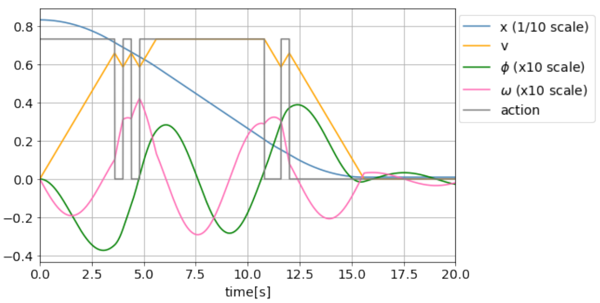

Figure 3: Image of State/Action Pair

changing along the time. This means that the information RL attend to for decision-making changes. We select the higher attributions at each time step and visualize it to demonstrate the attention change of RL agent.

Experiment

We evaluated the proposed method on the automatic crane control use case.

Automatic Crane Control

A crane is a type of machine, generally equipped with a hoist rope, wire ropes or chains and sheaves, that can be used to lift and lower materials and to move them horizontally. We want to realize automatic control of crane with deep RL agent and explain the policies of the agent. In Figure 2, we model the crane control problem.

The object is connected to a trolley with a piece of wire. The object is supposed to be delivered by the trolley from the start position to the goal position. Operators could add acceleration and deceleration signal to the trolley to accomplish the delivery. Note that the trolley can only travel horizontally on the rail. The trolley would either be accelerated by a specific constant value until the velocity of travelling reaches the maximum, or de-accelerated by the same value until the velocity reaches zero. As the trolley starts moving, the object starts swinging like a pendulum. The objective is to deliver the object to the goal position as soon as possible and at the mean time with neglectful swinging at the goal position.

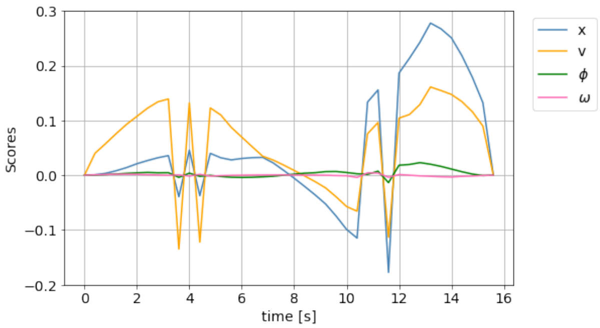

Figure 4: SHAP Values (Background Data: Start Position)

Figure 3 is a scaled version of the trajectory - the state and action pair at each episode. In Automatic Crane Control, there are four states (inputs of our DQN model); the traveling distance of the trolley x; the velocity of the travelling trolley v; the angular of the wire ϕ; the angular velocity of swing ω. For the intuitive understanding, we scaled the states in the figure. The grey line represents the action selected at each time step, which is the acceleration (targets $0.73m/s$) or de-acceleration (targets $0m/s$) signal our agent conducted at each time step. The blue line represents the distance to the goal of the travelling trolley x. The orange line represents the velocity of the traveling trolley v. The green line represents the swing angle for the moving direction ϕ. And, the pink line represents the angular velocity of the swing ω.

We applied our attribution-based salience method on the automatic crane control trajectory. We used KernelSHAP (Lundberg and Lee 2017) for the attribution method. We selected the start position as the background data. Figure 4 shows the SHAP values scores for the four states. The blue, orange, green and pink lines in the figure correspond to x, v, ϕ, and ω, respectively. The horizontal axis represents the attribution value score for each state.

The result shows that at the beginning, the RL agent cares more about the velocity of the trolley. Gradually, it pays attention to the angle of the wire, or swing, during travelling at high speed. It takes the traveling distance as the most important state near the goal.

The strategy above is different from the one usually conducted by a human operator. A human operator firstly looks at the traveling distance and velocity to travel the trolley and stops near to the goal as fast as possible. But in there, the wire is swinging. Then, the operator looks at the wire angle and accelerate and brake the trolley a little at an appropriate wire angle to stabilize the swing at the goal position.

The RL agent conveys faster than a human operator because the RL agent does not wait for the appropriate angle of the swing by once stopping near the goal position. The adjustment of the swing phase is realized by paying attention to the swing angle and putting a little acceleration and brake while travelling at high speed as described above. This result might be surprising for human operators but would be intuitive after understanding the attention sequence of the RL agent.

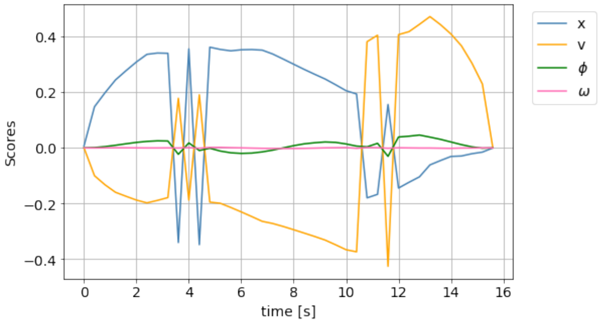

Figure 5: SHAP Values (Background Data: Goal Position)

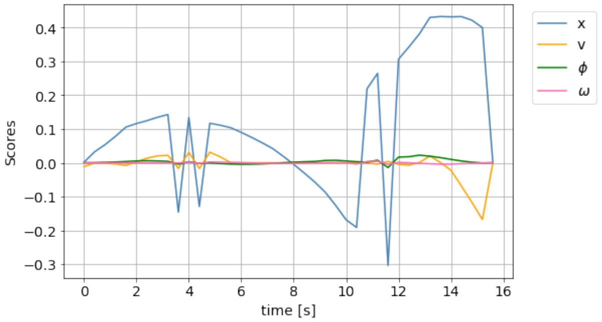

Figure 6: SHAP Values (Background Data: Middle Position)

Discussion

In this section, we discuss about the background data selection problem. We take automatic crane control as an example.

We also tried other candidate background data as comparative experiments. We selected the middle position and the goal position as the background data. Figure 5 shows the SHAP values results for the problem with the goal position selected as the background data. As shown in the figure, the traveling distance and traveling velocity are still the main features that contributes to the decision making. In this case, SHAP values of the traveling distance of the trolley and the traveling velocity are approximately similar but in different directions. At the beginning, the traveling distance contributes most, while near the goal direction, the traveling velocity contributes most. This is in contrast to what we observed in the experiment that used the start position as the background data.

Figure 6 shows the SHAP values result for the problem where we selected the middle position as background data. From $0s$ to around $5s$, the traveling distance has much contribution. However, their contributions decrease from $5s$ to $10s$, and other states becomes greater around $8s$. At the end of the trajectory, the traveling distance contributed most.

According to our investigation, when domain experts operate the crane, they will firstly accelerate the crane. Then, when crane reaches the maximum velocity, they operate to remain the crane at the maximum velocity. When the crane comes close to the goal position, they deaccelerate the crane. Apparently, there are three phases in the operation of domain experts. According to the experiment result, it makes sense when we select start position for these three phases of crane. However, in more complicated use cases, there will be more phases. Different background data should be selected for comparing with different patterns of data,

Conclusion

Our experiments show that different selection of background data generates different explanation. And some of the explanations match human intuition, while others are not straightforward enough for humans to understand. Since the calculation of attribution methods includes the selection of background data, we claim that this is a key issue for implementing attribution methods and reaching human-understandable explanations. Therefore, we select the background data and the generated explanation considering the domain knowledge and human intuition. Our proposed method explains the policies in regarding to the contribution of each input state. We will verify our method with more use cases as the future work. How to embed in domain knowledge and human intuition in the explanation that make them understandable to both expert and non-expert alike is also an open question.

References

Greydanus, S.; Koul, A.; Dodge, J.; and Fern, A. 2017. Visualizing and understanding atari agents. *arXiv preprint arXiv:1711.00138*.

Lundberg, S. M., and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 4765–4774.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529.

Shapley, L. S. 1953. A value for n-person games. *Contributions to the Theory of Games* 2(28):307–317.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L. a.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Pãnneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature* 529(7587):484–489.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*.

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement learning: An introduction, Second edition*, volume 1. MIT press Cambridge.

Zahavy, T.; Ben-Zrihem, N.; and Mannor, S. 2016. Graying the black box: Understanding dqns. In *International Conference on Machine Learning*, 1899–1908.