Enhancing Explainability of Deep Reinforcement Learning through Selective Layer-Wise Relevance Propagation

Tobias Huber[1][0000−0002−5010−4006], Dominik Schiller[1], and Elisabeth André[1][0000−0002−2367−162X]

Universität Augsburg, Universitätsstraße 6a, 86159 Augsburg, Germany
{huber,schiller,andre}@hcm-lab.de
http://www.hcm-lab.de

Abstract. Modern deep reinforcement learning agents are capable of achieving super-human performance in tasks like playing Atari games, solely based on visual input. However, due to their use of neural networks the trained models are lacking transparency which makes their inner workings incomprehensible for humans. A promising approach to gain insights into the opaque reasoning process of neural networks is the layer-wise relevance propagation (LRP) concept. This visualization technique creates saliency maps that highlight the areas in the input which were relevant for the agents' decision-making process. Since such saliency maps cover every possible cause for a prediction, they are often accentuating very diverse parts of the input. This makes the results difficult to understand for people without a machine-learning background. In this work, we introduce an adjustment to the LRP concept that utilizes only the most relevant neurons of each convolutional layer and thus generates more selective saliency maps. We test our approach with a dueling Deep Q-Network (DQN) agent which we trained on three different Atari games of varying complexity. Since the dueling DQN approach considerably alters the neural network architecture of the original DQN algorithm, it requires its own LRP variant which will be presented in this paper.

1 Introduction

Reinforcement learning addresses the problem of optimizing a long-term reward that an agent receives while interacting with an environment. Deep reinforcement learning (DRL) describes the combination of those methods with deep neural networks (DNN) by using a DNN as decision function of the agent.

One of the first successful applications of DRL was the deep Q-Network (DQN) developed by Mnih et al. [12]. This approach was able to achieve high-level performance across a set of 49 games for the Atari 2600 console, using the same hyperparameters and network architecture for all games, while only receiving the pixels of the game screen and the game score as input.

For a long time, DRL research only focused on optimizing the performance of DRL agents, but recent years saw an increasing interest in making the decision

process of DRL agents more explainable [23, 9, 19, 7, 21]. One problem with explaining the actions of a DRL agent is that the inner workings of the underlying DNNs are incomprehensible to humans, making it difficult to identify the parts of the input on which the agent bases its decision. A common approach to tackle this challenge is the generation of saliency maps that visualize the relevance of each input pixel for the output of the network [1].

While such saliency map algorithms are already well established and were even used to improve classification models [15], they are usually developed with experienced machine learning practitioners in mind. This can make the generated explanations difficult to interpret for beginners or users who are unrelated to the field of machine learning. Weitz et al. [22], for example, found that traditional saliency maps are too fine-granular for humans to easily detect relevant features for the classification. In a recent meta-study, Miller [11] explored the explanation process between humans to derive new design paradigms for explainable artificial intelligence algorithms that can help to make such methods more accessible to non-expert users. One major finding of this study was that people usually prefer selected explanations that focus on specific evidence instead of showing every possible cause of a decision. Based on this insight, we aim to adjust an existing saliency map approach to be more focused on the parts of the input that are most relevant for the decision-making process of a system.

We base our approach on layer-wise relevance propagation (LRP): A promising concept for generating saliency maps, which visualizes how much each pixel of the input picture contributed to the output based on the activations of each neuron during the forward pass. In contrast to most other approaches, LRP offers the benefit of conserving the certainty of the prediction throughout its process, which provides the user with additional useful information. Furthermore, LRP concepts do not contain contradictory evidence, because they do not generate negative relevance values [13]. Our adjustment uses an $argmax$ function to follow only the most contributing neurons of each convolutional layer, which enables us to filter out the most relevant information. Therefore we can create selective and more focused saliency maps while maintaining the advantageous properties of LRP mentioned above.

Modern DQN variants, like the rainbow algorithm [8], are employing dueling DQN systems which use two separate estimators to measure the value of the current state and the advantage of each action the agent can take in that state. To test our approach with state-of-the-art dueling DQN algorithms, we introduce a slightly adapted version of LRP that can handle the dueling DQN architecture without losing its advantageous properties. Since no other improvement of the DQN algorithm considerably changes the underlying neural network architecture, this extension allows us to use LRP on any DQN based DRL algorithm without any further adjustments.

We test our approach on three Atari 2600 games of varying complexity using the OpenAi gym and baselines libraries [4, 5]. The Atari game domain is well suited for testing and introducing new RL algorithms because it offers a wide array of different tasks in similar environments.

2 Related work

In this section, we look at successful applications of saliency maps for DNNs and DRL. Because saliency maps work best on visual input data, those methods focus on increasing the explainability of CNNs which are most often used on visual input data.

One of the first methods used to measure the relevance of pixels of visual input data is to see how much a change in that pixel impacts the prediction of the CNN. If a pixel is relevant for the decision of the model, then even small changes of the pixel will greatly impact the output of the model. This local rate of change with respect to certain inputs of the CNN can be calculated by using partial derivatives. Simonyan et al. [17] for example use the derivative of the neural network with respect to an input pixel to determine the relevance of that input pixel. To get this derivative they use the backpropagtion algorithm also used during the training of the neural network. The deconvolution [24] and guided backpropagation [18] approaches are based on the same theory but use modified versions of the backpropagation algorithm to get relevance values for the input pixels. Another similar approach is Grad-CAM [16], which uses partial derivatives of the fully connected part of a CNN with respect to the output of the last convolutional layer to identify regions inside the input, which were relevant for the specific prediction of the CNN. Guided backpropagation and Grad-CAM can be combined by computing the component-wise product of the attention maps created by the different approaches. The result is called guided Grad-CAM and creates a fine granular but class specific saliency map [16].

In contrast to those gradient-based saliency maps Bach et al. [3] proposed a method that directly uses the activations of the neurons during the forward pass to calculate the relevance of the input pixels. This is computationally efficient compared to gradient-based methods because they can reuse the values of the forward pass. Instead of calculating how much a change in an input pixel would impact the prediction, Bach et al. investigate the contribution of the input pixels to prediction. For this purpose, they do not only describe a single specific algorithm but introduce a general concept which they call layer-wise relevance propagation (LRP). This concept has two advantageous properties which gradient-based saliency maps lack. The first is the conservation property which says that the sum of all relevance values, generated by LRP, is equal to the value of the prediction. This ascertains that the relevance values reflect the certainty of the prediction. The second property is positivity which states that all relevance values are non-negative. This ascertains that the generated saliency maps do not contain contradictory evidence [13]. Some gradient-based approaches achieve positivity by squaring the partial derivatives, but this only masks the negativity for the viewer.

Another approach that uses the activations of each neuron during the forward pass was proposed by Mopuri et al. [14]. Instead of measuring the relevance of each input pixel, they search for the position of all input pixels that contributed positively to the prediction. While doing so they only track the most contributing neuron in each convolutional layer. We aim to combine this idea with the LRP

concept to get a more focused version of LRP which still contains relevance values.

So far we only covered methods to generate saliency maps for deep neural networks in general. From this point on we look at implementations of saliency maps which focused on DRL. Because many DRL algorithms utilize CNNs it is possible to directly use the methods we just covered on DRL agents. Zahavy et al. [23] and Wang et al. [20] for example used gradient-based saliency maps similar to [17] on traditional and Dueling DQN algorithms. Weitkamp et al. [21] tested Grad-CAM on an Actor-Critic DRL algorithm. LRP has been used to visualize DRL in [10] but, to our knowledge, it has not been used to visualize the Dueling DQN architecture yet.

Iyer et al. [9] proposed a completely new visualization algorithm for DRL. They use template matching to identify objects in each input image and use this information as additional channels of the input to retrain the DRL agent. Given an agent trained in this way, they can measure the relevance of an identified object by comparing the prediction of the input image containing that object with the prediction for the same input image without this specific object.

Greydanus et al. [7] also propose a new algorithm, where they selectively blur regions of the input image and measure how much this changes the output of the DRL agent. The idea behind this is to introduce uncertainty to the blurred area and to see how much the DRL agent is influenced by the loss of information in that area.

The approaches of Iyer et al. and Greydanus et al. both lack the conservation property of LRP.

3 Saliency Maps

In this section, we revisit the foundations of LRP and show how to use it on the original DQN. Then we propose an adjustment to this algorithm which generates more focused saliency maps. In the last subsection, we introduce a way to apply those LRP algorithms to the Dueling DQN architecture.

3.1 Foundations

LRP does not describe a specific algorithm but a concept which can be applied to any classifier f that fulfills the following two requirements. First, f has to be decomposable into several layers of computation where each layer can be modeled as a vector of real-valued functions. Secondly, the first layer has to be the input x of the classifier containing, for example, the input pixels of an image and the last layer has to be the real-valued prediction of the classifier $f(x)$. Any DRL agent fulfills those requirements if we only consider the output value that corresponds to the action we want to analyze.

For a given input x, the goal of any method following the LRP concept is to assign relevance values R_j^l to each computational unit j of each layer of computation l in such a way that R_j^l measures the local contribution of the unit

j to the prediction $f(x)$. A method of calculating those relevance values R_j^l is said to follow the LRP concept if it sets the relevance value of the output unit to be the prediction $f(x)$ and calculates all other relevance values by defining

$$R_j^l := \sum_{k \in \{j \text{ is input for neuron } k\}} R_{j \leftarrow k}^{l,l+1}, \tag{1}$$

for **messages** $R_{j \leftarrow k}^{l,l+1}$, such that

$$R_k^{l+1} = \sum_{j \in \{j \text{ is input for neuron } k\}} R_{j \leftarrow k}^{l,l+1}. \tag{2}$$

In this way a LRP variant is determined by choosing messages $R_{j \leftarrow k}^{l,l+1}$. Through definition 1 it is then possible to calculate all relevance values R_j^l in a backward pass, starting from the prediction $f(x)$ and going towards the input layer. Furthermore equation 2 gives rise to

$$\begin{aligned} \sum_k R_k^{l+1} &= \sum_k \sum_{j \in \{j \text{ is input for neuron } k\}} R_{j \leftarrow k}^{l,l+1} \\ &= \sum_j \sum_{k \in \{j \text{ is input for neuron } k\}} R_{j \leftarrow k}^{l,l+1} = \sum_j R_j^l. \end{aligned} \tag{3}$$

This ensures that the relevance values of each layer l are a linear decomposition of the prediction

$$f(x) = \cdots = \sum_{j=1}^{dim(l)} R_j^l = \cdots = \sum_{j=1}^{dim(input)} R_j^{input}. \tag{4}$$

Such a linear decomposition is easier to interpret than the original classifier because we can think of positive values R_j^l to contribute evidence in favor of the decision of the classifier and of negative relevance values to contribute evidence against the decision.

To use LRP on a DQN agent we first have to look at its network architecture. The DQN f, as introduced by Mnih et al. [12], consists of three convolutional layers $conv_1, ..., conv_3$ followed by two fully connected layers fc_1 and fc_2. For an input x we write $fc_i(x)$ and $conv_i(x)$ for the output of the layers fc_i and $conv_i$ respectively during the forward pass that calculates $f(x)$. In this notation, the Q-Values (i. e. the output of the whole DQN) are $fc_2(x)$.

Following the LRP notation, we denote the relevance value of the j-th neuron in the layer l with R_j^l. As seen before we have to define messages $R_{j \leftarrow k}^{l,l+1}$ for any two consecutive Layers $l, l+1$ to determine a LRP variant. For now we assume that $l+1$ is one of the fully connected layers fc_i. The convolutional case works analogously and will be covered in more detail in the next chapter. $R_{j \leftarrow k}^{l,l+1}$ should measure the contribution of the j-th neuron of fc_{i-1} to the k-th neuron of fc_i, therefore we have to look at the calculation of $fc_i(x)_k$. The fully connected layer

fc_i uses a weight matrix W_i, a bias vector b_i and an activation function σ_i as parameters for its output. Let W_i^k be the k-th row of W_i and b_i^k the k-th entry of b_i. Then the activation of the k-th neuron in $fc_i(x)$ is

$$\sigma_i(W_i^k \cdot fc_{i-1}(x) + b_i^k), \tag{5}$$

where $\cdot$ denotes the dot product and fc_0 is the flattened output of $conv_3$.

Usually the ReLU function $\sigma(x) = max(0, x)$ is used as activation function σ_i in the DQN architecture. Bach et al. [3] argue that any monotonous increasing function σ with $\sigma(0) = 0$, like the ReLU function, conserves the relevance of the dot product $W_i^k \cdot fc_{i-1}(x)$. Newer LRP variants, like the one used by Montavon et al. [13], also omit the bias when defining $R_{j\leftarrow k}^{l,l+1}$. With those two assumptions the relevance of each neuron of fc_{i-1} to $fc_i(x)_k$ is the same as their contribution to the dot product $W_i^k \cdot fc_{i-1}(x) = \sum_j w_{jk}\, fc_{i-1}(x)_j$. This is a linear decomposition, so we can use $w_{jk}\, fc_{i-1}(x)_j$ to measure the contribution of the j-th neuron of fc_{i-1}.

Since we want to find the parts of the input that contributed evidence in favor of the decision of the DQN agent, we restrict ourself to the positive parts of that sum. That is we set

$$z_{jk}^+ := \begin{cases} w_{jk}\, fc_{i-1}(x)_j & \text{if } w_{jk}\, fc_{i-1}(x)_j > 0 \\ 0 & \text{if } w_{jk}\, fc_{i-1}(x)_j \leq 0 \end{cases}. \tag{6}$$

With this, we define the messages as $R_{j\leftarrow k}^{l,l+1} := \frac{z_{jk}^+}{\sum_j z_{jk}^+} R_k^{l+1}$. This method is called z^+-rule (without bias) and satisfies the LRP equation 2.

3.2 An argmax approach to LRP

In this subsection, we introduce our adjustment to an LRP variant called z^+-rule which we revisited in the last subsection. Recent work [9, 6] indicates that DRL agents mainly focus on whole objects, for example cars or balls, within the visual input. With our approach, we aim to generate saliency maps that reflect this property by focusing on the most relevant parts of the input instead of giving too many details. For this purpose, we propose to use an $argmax$ function to find the most contributing neurons in each convolutional layer.

This idea is inspired by Mopuri et al. [14], who generated visualizations for neural networks solely based on the positions of neurons that provide evidence in favor of the prediction. During this process, they follow only the most contributing neurons in each convolutional layer. Our method adds relevance values to the positions of those neurons and therefore expands the approach of Mopuri et al. by an additional dimension of information. Since those relevance values follow the LRP concept, they also possess the advantageous properties of the LRP concept like conservation of the prediction value.

As we have seen in the foundations section 3.1, a LRP method is defined by its messages $R_{j\leftarrow k}^{l,l+1}$ which propagate the relevance from a layer $l + 1$ to the

preceding layer l. If $l+1$ is a fully connected layer fc_i of the DQN (see section 3.1 for our notation of the DQN architecture), we use the same messages that are used in the z^+-rule. In the case that l and $l+1$ are convolutional layers $conv_{i-1}$ and $conv_i$, we propose new messages based on the $argmax$ function. To define those messages we analyze how the activation of a neuron $conv_i(x)_k$ was calculated during the forward pass. Let W and A denote the weight kernel and part of $conv_{i-1}(x)$ respectively that were used to calculate $conv_i(x)_k$ during the forward pass. If we write W and A in appropriate vector form, we get

$$conv_i(x)_k = \sigma(\sum_j w_j a_j + b), \tag{7}$$

where σ denotes the activation function of $conv_i$ and b the bias corresponding to W. Analogously to the z^+-rule we assume that the activation function and the bias can be neglected when determining the relevance values of the inputs a_i. We propose to use an $argmax$ function to find the most relevant input neurons by defining the messages in the following way

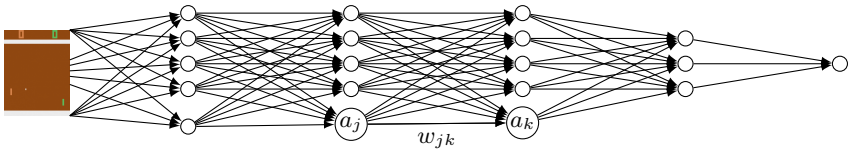
$$R^{l,l+1}_{j\leftarrow k} := \begin{cases} R^{l+1}_k & \text{if } j = argmax\{w_j a_j\} \\ 0 & \text{if not.} \end{cases} \tag{8}$$

This definition satisfies the LRP condition given by equation 2 because the only non vanishing summand of the sum

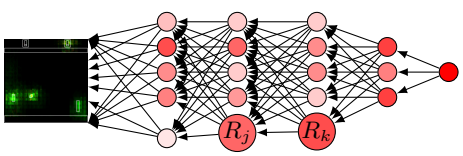
$$\sum_{j\in\{j \text{ is input for neuron } k\}} R^{l,l+1}_{j\leftarrow k} \tag{9}$$

is R^{l+1}_k.

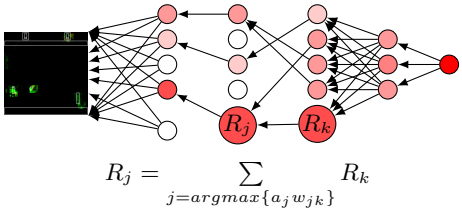
If we use the same $argmax$ approach to propagate relevance values from $conv_1$ to the input $conv_0$, then we get very sparse saliency maps where only a few neurons are highlighted. If we highlight the whole areas of the input $conv_0$ that were used to calculate relevant neurons of $conv_1$, then we lose information about the relevance values inside those areas. Therefore we draw inspiration from the guided Grad-CAM approach introduced in [16]. Guided Grad-CAM uses one throughout relevance analysis for the neurons of the last convolutional layer to get relevant areas for the specific prediction and another throughout relevance calculation for the input pixels to get fine granular relevance values inside those areas. We already did a throughout analysis of the neurons of the last convolutional layer by using the z^+-rule on the fully connected layers. By following the most relevant neurons through the convolutional layers we keep track of the input areas that contributed the most to those values. Mimicking the second throughout analysis of the Guided Grad-CAM approach we propose to use the z^+-rule to propagate relevance values from $conv_1$ to $conv_0$. This generates fine granular relevance values inside the areas identified by following the most contributing neurons and ascertains that those relevance values follow the LRP concept.



Forward pass.



Relevance propagation using the z^+-Rule.



Relevance propagation using the *argmax* approach.

Fig. 1: A visualization of how our *argmax* approach differs from the z^+ Rule.

Figure 1 visualizes the differences between our *argmax* approach and the z^+-rule. We implemented our proposed algorithm for the OpenAi baselines library [5] and plan to integrate it in the iNNvestigate framework [2].

3.3 LRP on Dueling Q-Networks

The dueling Q-network is a neural network architecture first introduced by Wang et al. in [20] as an improvement of the neural network architecture used in the DQN algorithm [12]. Because it is only changing the architecture of the neural network, it is independent of the training algorithm. Therefore it can easily be combined with other improvements of the DQN algorithm. This can be seen in the rainbow algorithm, the current state of the art deep Q-learning algorithm [8], which combines many different improvements of the DQN algorithm. We chose Dueling DQN because the LRP concept only depends on the neural network architecture. Therefore applying LRP to the Dueling DQN architecture suffices to apply LRP on all currently used versions of the DQN algorithm.

Instead of using a single fully connected network after the convolutional part of the DQN, the Dueling DQN architecture uses two fully connected networks A and S which both use the output of the last convolutional layer as input. These two fully connected networks share the same architecture apart from their output layer. For an input state s, the state value network S has only one single output neuron $S(s)$ which measures the value of the state s. The network A has an output neuron $A(s, a)$ for each action a which describes the advantage of choosing the action a in the state s. The Q-Value (the prediction of the whole

model) for an input state s and an action a is then calculated by

$$Q(s,a) = S(s) + A(s,a) - \frac{1}{N}\sum_{i=1}^{N} A(s,a_i), \tag{10}$$

where N denotes the number of available actions a_i.

One way of using LRP on this architecture would be to use LRP methods on each of the networks S and A separately, but then we would lose the conservation property because the relevance values would not add up to $Q(s,a)$. Therefore we have to define a way to propagate the relevance value of the output $Q(s,a)$ to $S(s)$ and $A(s,a)$. Because equation 10 is already a linear decomposition, the main question is how we handle the summand $-\frac{1}{N}\sum_{i=1}^{N} A(s,a_i)$. For this we follow the original thought process of Wang et al. in [20], where they treat $(A(s,a) - \frac{1}{N}\sum_{i=1}^{N} A(s,a_i))$ as the modified contribution of $A(s,a)$ to $Q(s,a)$. Analogously to the z^+-rule we only propagate those values if they are positive since we want to exclusively highlight evidence in favor of the chosen action a . That is we set

$$S(s)^+ \coloneqq max(0, S(s)) \tag{11}$$

$$A(s,a)^+ \coloneqq max\big(0, A(s,a) - \frac{1}{N}\sum_{i=1}^{N} A(s,a_i)\big). \tag{12}$$

If we would use these values as LRP messages, then the LRP equation 2 would not hold if either of $S(s)$ or $A(s,a)$ are negative. Therefore we set the LRP messages analogously to the z^+-rule as:

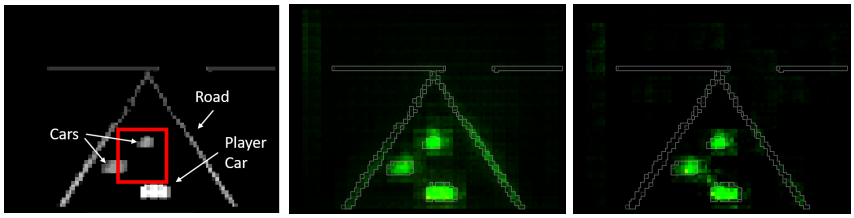
$$R_{S(s)\leftarrow Q(s,a)} \coloneqq \frac{S(s)^+}{S(s)^+ + A(s,a)^+} Q(s,a) \tag{13}$$

$$R_{A(s,a)\leftarrow Q(s,a)} \coloneqq \frac{A(s,a)^+}{S(s)^+ + A(s,a)^+} Q(s,a). \tag{14}$$

If both $S(s)$ and $A(s,a)$ are negative, then there is no evidence in favor of the prediction. Consequently, it is justified that we do not propagate any relevance values in this case.

4 Results and Discussion

In order to verify that our *argmax* approach, described in section 3.2, creates more selective saliency maps then the r^+-rule (see section 3.1), we tested our approach on three different Atari 2600 games and will present the results of those experiments in this section. For all games, we trained an agent using the DQN implementation of the OpenAi baselines framework [5]. Since this implementation utilizes the Dueling DQN architecture [20], we used the approach described in 3.3 to apply LRP to this architecture.

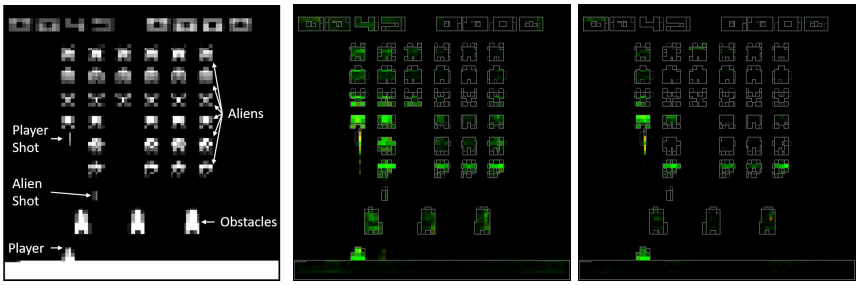


(a) screen (b) z^+-rule (c) *argmax*

Fig. 2: A comparison of action advantage analysis: The left image (a) shows a screen from the Atari game Enduro with additional descriptions. The red area was identified as relevant by gradient-based saliency maps in [20]. While the z^+-rule (b) highlights the cars and the edge of the road even though it is not important in this situation, our *argmax* approach (c) selects only the relevant cars.

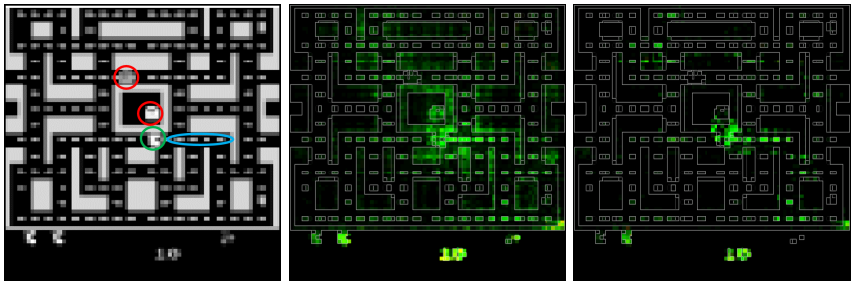
We keep track of which relevance values correspond to the state value and the action advantage values and differentiate them by coloring them red and green respectively. This allows us to compare our saliency maps with the ones generated by gradient-based methods in [20] for a Duelling DQN agent trained on the Atari game Enduro. In this simple driving game, the Player controls a car and has to avoid hitting other cars while overtaking as many of them as possible. The left image of Figure 2 shows a screen from this game in the preprocessed form that the agent received. The area that was identified as relevant for the action advantage value in similar game-states by the gradient-based saliency maps in [20] is marked in red. To facilitate readability, we added descriptions of the important game objects and cut off the lower part of the screen which only contains the score. The middle and right images show saliency maps generated by the z^+-rule and our *argmax* approach respectively for the game-state shown in the left image. All three saliency maps identified the area in front of the player car as the most relevant area. The gradient-based saliency map in [20] focused strongly on this region but was not fine-grained enough to select individual cars. The z^+-rule, on the other hand, emphasizes all the relevant cars but does not focus on the area in front of the agent. Instead, it also highlights the general course of the road which is not particularly important in this situation. Our *argmax* approach is the most selective and only highlights the relevant cars inside the area which was also identified by the gradient-based approach.

The second game we trained our agent on is called Space Invaders. In this game, the agent controls a cannon, which can move horizontally along the bottom of the screen, and has to destroy descending waves of aliens. Additionally, the player needs to evade incoming projectiles fired from the aliens or to take cover behind three floating obstacles. In contrast to purely reactive games like Enduro, Space Invaders requires the agent to develop long-term strategies, as it has to determine an order in which it destroys the aliens in each wave and also has to decide when to hide behind obstacles. While this does not necessarily imply that



(a) screen (b) z^+-rule (c) *argmax*

Fig. 3: The first image (a) shows a screen of the Atari game Space Invaders with additional descriptions. The saliency map created for this game-state by the z^+-rule (b) highlights most of the aliens and all the obstacles while our *argmax* approach (c) focuses on the first row of aliens which the agent can actually hit.



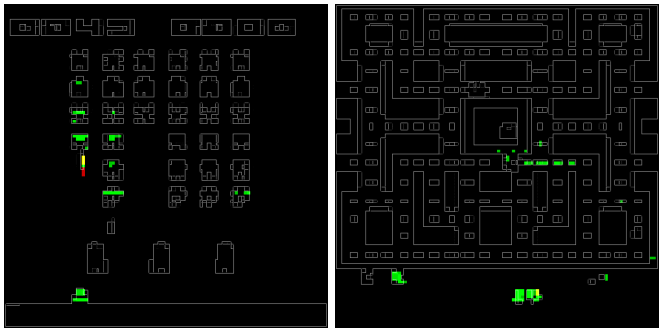
(a) screen (b) z^+-rule (c) *argmax*

Fig. 4: The left image (a) shows a screen of Pacman. The player (green circle) has to collect pellets (blue area) while avoiding ghosts (red circles). The saliency map created for this game-state by the z^+-rule (b) highlights a huge area as relevant while our *argmax* approach (c) focuses on the vicinity of the player.

the game is harder to learn for an agent, analyzing the trained model might lead to a better understanding of an optimal strategy to solve this game. Figure 3 shows a comparison of the two different saliency map approaches for a specific game-state of space invaders. Both the z^+-rule, as well as the *argmax* approach are showing that the agent mostly considers the aliens positioned on the outline of the grid as relevant. However the *argmax* approach does so more clearly by only highlighting aliens on the outline of the grid. This selection makes sense since the other enemies cannot be hit by the agent. Our selective *argmax*-rule further shows that the agent is not paying attention to the obstacles. Given a certain performance level of our model, this suggests that they might not be a necessary component of an optimal strategy for Space Invaders. In this way, our selective saliency maps enable us not only to find errors in our model but also to pass on the learned knowledge to human players.

The last game we used to verify our approach is MsPacman, where the player has to navigate through a maze and collect pellets while avoiding enemy ghosts. Because this game contains many important objects and gives the agent a huge variety of possible strategies, DQN agents struggle in this environment and perform worse than the average human player [12]. Transparency methods are especially desirable in environments like this, where the agent is struggling because they help us to understand where the agent had difficulties. The saliency map created by the z^+-rule (see figure 4 b) reflects the complexity of MsPacman by showing that the agent tries to look at nearly all of the objects in the game. This information might be helpful to optimize the DRL agent, but it also distracts from the areas which influenced the agents' decision the most. Figure 4 shows that the saliency map created by the *argmax* approach is more focused on the vicinity of the agent and makes it clearer what the agent is focusing on the most. Figure 4 further illustrates that a fine-granular saliency map in the vicinity of the agent is necessary to see that the agent will most likely decide on moving to the right as his next action.

For the sake of completeness, we want to mention that a similar selective effect can be obtained by using the z^+-rule and implementing some kind of threshold, for example only showing the highest 1% of all relevance values. However, this approach comes with its own set of challenges. While a threshold might be suited for one environment it might be too high or low for other environments, presenting too much or too less information (see for example Figure 5). Our pro-



(a) Space Invaders (b) Pacman

Fig. 5: Only showing the top 42 relevance values created by the z^+-rule produces a saliency map (a) which is similar to the one created by our *argmax*-approach for Space Invaders in Figure 3(c). Using the same threshold for Pacman (b) we lose some relevant information since, in contrast to 4(c), the position of the player is no longer highlighted.

posed approach is independent of the environment which eliminates the need to empirically determine a specific threshold for each new problem. Furthermore, the conservation property of LRP is lost by simply removing relevance values.

Therefore the generated saliency maps are not proportional to the prediction which makes it harder to compare different saliency maps.

In total, our experiments have shown that our approach can be used on three games, each of which presents a different challenge, and that it generates informative saliency maps that are more selective than the ones generated by the z^+-rule.

5 Conclusion

In this paper, we presented two adjustments to the LRP concept which enable compatibility with state of the art deep reinforcement learning approaches and increase the selectivity of the generated saliency maps while maintaining all desired properties of the original algorithm. For one, we have shown a way to use LRP on the Dueling DQN architecture, which makes it possible to use LRP on all current versions of the DQN algorithm. Secondly, we introduced an adjustment to an existing LRP variant, which generates saliency maps that focus more on the important objects inside the input image.

We tested our approach on three different Atari 2600 games and verified that the saliency maps generated by our system are more selective than the ones created by existing LRP methods, while still including the information expected from visual explanations. Since this selectiveness is an important property of inter-human explanations we argue that our approach might prove beneficial, when it comes to explain the actions of a trained agent to people without a machine-learning background. Understanding an agents reasoning process is especially interesting since DQN agents are already outperforming human players in many Atari games. In the game Space Invaders, for example, the analysis of our selective saliency maps helped us to formulate the hypothesis that the obstacles, which protect the agent from enemy projectiles, are not relevant for an optimal strategy. In the future, we would like to further investigate the potential of our approach to impart the learned knowledge, which leads to such achievements, to a human user.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018). https://doi.org/10.1109/ACCESS.2018.2870052
2. Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K.T., Montavon, G., Samek, W., Müller, K.R., Dähne, S., Kindermans, P.J.: iNNvestigate neural networks! arXiv preprint arXiv:1808.04260 (2018)
3. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. PLOS ONE **10**(7) (Jul 2015). https://doi.org/10.1371/journal.pone.0130140
4. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: Openai gym. arXiv preprint arXiv:1606.01540 (2016)
5. Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., Zhokhov, P.: Openai baselines. https://github.com/openai/baselines (2017)
6. Goel, V., Weng, J., Poupart, P.: Unsupervised video object segmentation for deep reinforcement learning. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada. pp. 5688–5699 (2018), http://papers.nips.cc/paper/7811-unsupervised-video-object-segmentation-for-deep-reinforcement-learning
7. Greydanus, S., Koul, A., Dodge, J., Fern, A.: Visualizing and understanding atari agents. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. pp. 1787–1796 (2018), http://proceedings.mlr.press/v80/greydanus18a.html
8. Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M.G., Silver, D.: Rainbow: Combining improvements in deep reinforcement learning. In: Proceedings of the 32nd Conference on Artificial Intelligence, AAAI 2018. pp. 3215–3222 (2018)
9. Iyer, R., Li, Y., Li, H., Lewis, M., Sundar, R., Sycara, K.P.: Transparency and explanation in deep reinforcement learning neural networks. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018. pp. 144–150 (2018). https://doi.org/10.1145/3278721.3278776
10. Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking clever hans predictors and assessing what machines really learn. Nature Communications **10**(1), 1096 (2019)
11. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artif. Intell. **267**, 1–38 (2019). https://doi.org/10.1016/j.artint.2018.07.007
12. 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.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
13. Montavon, G., Samek, W., Müller, K.: Methods for interpreting and understanding deep neural networks. Digital Signal Processing **73**, 1–15 (2018). https://doi.org/10.1016/j.dsp.2017.10.011
14. Mopuri, K.R., Garg, U., Babu, R.V.: Cnn fixations: an unraveling approach to visualize the discriminative image regions. IEEE Transactions on Image Processing **28**(5), 2116–2125 (2019)

15. Schiller, D., Huber, T., Lingenfelser, F., Dietz, M., Seiderer, A., André, E.: Relevance-based feature masking: Improving neural network based whale classification through explainable artificial intelligence. In: 20th Annual Conference of the International Speech Communication Association INTERSPEECH (2019), in press
16. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 618–626 (2017). https://doi.org/10.1109/ICCV.2017.74
17. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. CoRR **abs/1312.6034** (2013)
18. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A.: Striving for simplicity: The all convolutional net. CoRR **abs/1412.6806** (2014)
19. Such, F.P., Madhavan, V., Liu, R., Wang, R., Castro, P.S., Li, Y., Schubert, L., Bellemare, M.G., Clune, J., Lehman, J.: An atari model zoo for analyzing, visualizing, and comparing deep reinforcement learning agents. CoRR **abs/1812.07069** (2018)
20. Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., de Freitas, N.: Dueling network architectures for deep reinforcement learning. In: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016. pp. 1995–2003 (2016), http://jmlr.org/proceedings/papers/v48/wangf16.html
21. Weitkamp, L., van der Pol, E., Akata, Z.: Visual Rationalizations in Deep Reinforcement Learning for Atari Games. CoRR **arXiv:1902.00566** (Feb 2019)
22. Weitz, K., Hassan, T., Schmid, U., Garbas, J.U.: Deep-learned faces of pain and emotions: Elucidating the differences of facial expressions with the help of explainable ai methods. tm-Technisches Messen **86**(7-8), 404–412 (2019). https://doi.org/10.1515/teme-2019-0024
23. Zahavy, T., Ben-Zrihem, N., Mannor, S.: Graying the black box: Understanding dqns. In: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016. pp. 1899–1908 (2016), http://jmlr.org/proceedings/papers/v48/zahavy16.html
24. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I. pp. 818–833 (2014). https://doi.org/10.1007/978-3-319-10590-1_53