Evaluating Interpretability Methods for DNNs in Game-Playing Agents

Aðalsteinn Pálsson[1] and Yngvi Björnsson[1][0000−0001−5366−2639]

Department of Computer Science, Reykjavik University

Abstract. There is a trend in game-playing agents to move towards an Alpha-Zero-style architecture, including using a deep neural network as a model for evaluating game positions. Model interpretability in such agents is problematic. We evaluate the applicability and effectiveness of several saliency-map-based methods for improving the interpretability of a deep neural network trained for evaluating game positions, using the game of Breakthrough as our testbed. We show that some of the more applicable methods provide valuable insights into the importance of the different game pieces and other domain-dependent knowledge learned by the model.

Keywords: game-playing · model-interpretability · deep neural-networks

1 Introduction

Over the past few decades, research into game-playing programs for abstract strategy board games has first-and-foremost concentrated on developing new techniques and algorithms for improving gameplay. That work has resulted in super-human strength game-playing agents for disparate games such as chess, checkers, Othello, Go, and many more. At the same time, other important aspects of intelligent systems have been mainly neglected, such as how to explain the rationality for one's actions in human-understandable terms. Moreover, recent advancements in the field where game-playing agents use deep neural networks (DNNs) to evaluate board positions and action selection in the think-ahead process make the decision-making process even more non-transparent.

The above-mentioned lack of transparency is not specific to game-playing agents. As computer-generated models in disparate fields such as healthcare and banking have become increasingly ubiquitous, the need for humans to understand their decisions becomes increasingly crucial for establishing trustworthiness. This has spurred research interest in *model interpretability*, that is, the development of approaches to make it less complicated for humans to understand the cause of models' decisions in terms of their inputs. Several such approaches now exist, for example, in the field of image recognition, which has hitherto been at the forefront of both deep neural network and model-interpretability research. In particular, *saliency maps* have become a popular way of visualizing which regions of an image are primarily responsible for a given classification decision.

In this work, we evaluate the applicability and effectiveness of several saliency-map-based methods for improving the interpretability of a neural network trained to evaluate game positions of a board game, using the game Breakthrough as our testbed. The paper's primary contributions are: (i) We evaluate several popular saliency-map-based methods within recently established paradigms and taxonomies for black-box interpretability methods; and (ii) show how they can be applied to interpret a deep neural network model for an abstract board-game — a domain they are not mainly intended for; and, finally; (iii) assiduously evaluate and rank the methods by their effectiveness in our domain.

The organization of the paper is as follows. Section 2 introduces the necessary terminology and preliminaries. Section 3 explains the game-playing agent, model, and evaluation methods used. In Section 4, which constitutes the main body of the work, we introduce and analyze the finding of the empirical evaluations of the different saliency-maps methods in our domain. Finally, in Section 5, we conclude and discuss future work.

2 Background

We start with a high-level overview of the model-interpretability methods we investigate, before explaining the rules of the game of Breakthrough.

2.1 Model Interpretability

The taxonomy of explanation methods of black-box models categorizes them as either *global* or *local* and *model-specific* or *model-agnostic*. Global methods create explanations valid across all input instances, while local methods' explanations are specific to individual input instances. Model-agnostic methods explain any black-box models, while model-specific methods may leverage the model's architecture to aid their explanations.

The most straightforward local method is *occlusion*, where the model's output sensitivity to leaving out (zeroing) arbitrary input parameters is investigated [12]. Such an approach, where applicable, is appealing as it is both model-agnostic and straightforward to implement.

A method that is local and model-specific but still requires no ad-hoc work is to analyze the gradient of the output with respect to individual input pixels [7]. Multiplying the input with the gradient is also often preferable because it leverages the strength of the input. A further extension on the gradient method is Integrated Gradients [9], which relies on a baseline and interprets the input feature attribution as the integration of gradients on the straight-line path between the input and the baseline. GradientShap [4] is an extension of Integrated Gradients that computes the expected gradient by sampling baseline values.

DeepLIFT [6] is a model-specific explanation method that does not rely on the gradient. It overcomes the limitations of loss of information when the gradient is zero because the signal might still be meaningful. It calculates the importance in a backward fashion by distributing attributions, or blame, in terms

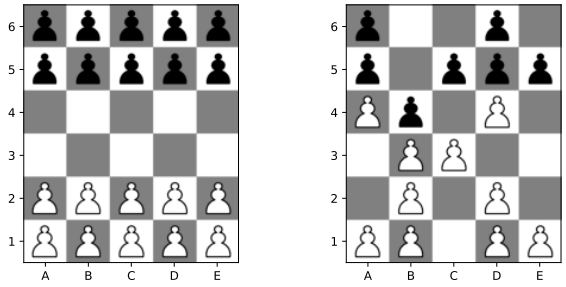

Fig. 1. Breakthrough: Initial board position (left); Example position (right)

of difference-from-reference. For all neurons, a difference-from-reference is calculated by passing through the input sample and the reference. Finally, it calculates the importance using predefined rules, such as the linear- or reveal cancel rule.[1]

LIME [5] is a local model-agnostic method, which uses an interpretable surrogate model to explain the black-box model. A local model, such as a linear model, is trained on a dataset derived from sampling noise around the input and using the model evaluation as a target. In the case of a linear model, the weights from the model serve as feature attributions.

The Shapley value [8] is a concept from cooperative game theory that can be used to calculate feature attribution. There are multiple ways to approximate the Shapley values [4], the one we use in this paper is Shapley Value Sampling. It takes random permutations of the input and adds them one by one to the baseline. This is repeated multiple times to approximate the Shapley values. The Shapley value satisfies the properties that if a feature does not affect the model prediction, its impact will be zero, and if two features similarly impact the model, their contribution will be similar.

2.2 Breakthrough

The game Breakthrough is an abstract strategy board game, originally played on a 7x7 board but later popularized to an 8x8 board. The game can be played on different-sized (not necessarily squared) boards. In this work we use a smaller variant of 5x6, mainly for the ease of demonstration.

The game is a two-player turn-taking game. The players are referred to as White and Black, respectively. The board is initially set up by placing White's pieces along the first two rows and the Black pieces on the last two rows, as shown in Fig. 1 (left). White goes first and then plays alternates, with each player moving one of their pieces per turn. A piece moves one square straight or diagonally forward (relative to the player). A straight forward move is allowed

[1] A baseline, or a reference, may be interpreted as a neutral state of a neural network, and is important for defining counterfactual arguments [9]. When assigning an attribution/blame to the input, it is done relative to the baseline. Most of the methods we consider in this paper rely on a baseline defined as all-zeros.

to an empty square only, but a diagonally forward moves may also capture an opponent's piece. For example, in Fig. 1 (right) the white pawn on $d2$ has two moves, to $d3$ or $e3$, and the piece on $d4$ also has two moves, to $c5$ or $e5$, both with a capture. The first player to get a piece across the board wins: White wins by moving a piece onto the last row, and likewise, Black wins by moving a piece onto the first row. If all pieces of a player are captured, that player loses. It follows from the rules that one of the players always wins (there are no draws).

3 Methods

An Alpha-Zero-like agent for playing the game of Breakthrough was developed for the paper. The following subsection providing details, whereas the next subsection gives an overview of our evaluation methodology.

3.1 The Model

We trained a AlphaZero-like model, $(p, v) = f_\theta(s)$ with parameters θ, where p is the policy and v is the value function. The value is the output from a tanh activation function, a scalar value between -1 and 1. And the policy is a tensor with three channels, where each channel is the same size as the board and encodes the three different move directions available for all the pieces, i.e., forward, and diagonally left, or -right.

The model consists of a body of 5 residual blocks with 56 filters, followed by the policy and value heads. The input to the model is a tensor with three channels, where each channel is the same size as the board. The first channel encodes the board positions of the active player, the second channel encodes the positions of the opponent, and the third channel is a binary encoding of the current player's color. If it is white to move, then the third channel is all ones. Otherwise, it is all zeros. In our case, we use a board with six rows and five columns. The search is performed with the Monte Carlo tree search algorithm like AlphaZero. The training procedure was via asynchronous self-play, where each move played used 200 simulations.

We deviate from AlphaZero as described in the paper because, unlike AlphaZero, we only feed into the model the current board position. We do this mainly to make the model easier to explain, and an explanation should not depend on previous board positions.

3.2 Evaluation Measures

The goal of this paper is to compare and evaluate model explanation methods in the domain of game-playing. Our model is returning a value estimate, and the goal is to explain the estimation. When evaluating the usefulness of the explanations, we will consider if we can use the explanations to gain trust in the trained model, if the explanation satisfies our human curiosity, and if we can interpret some meaning from the model [1].

The evaluation approaches are split into three categories [3]: (i) Functionally-grounded, where the explanation is evaluated without human, using a proxy as an indication of explainability; (ii) human-grounded, requiring a human with non-expertise to evaluate a simple explanation and; (iii) application-grounded, which requires a human-expert, evaluating an explanation for a real-world task.

At first, we will inspect the saliency map from a qualitative human-grounded perspective. We will evaluate and compare the saliency maps visually to see if we prefer one over another. Visual comparison sheds some light on the similarities and differences between the methods, which can be used as counterfactual arguments. We will briefly debate if the saliency maps match our human objectives by visualizing statistics from the saliency methods.

Quantitative evaluation will be in the form of functionally grounded experiments. We will define three tasks where the performance will be used as a proxy for explanation quality. First, we will assess if the saliency method assigns the highest saliency to a critical piece, and conversely, if the lowest saliency is assigned to a non-critical piece. We will analyze this as an ablation study, where we separately remove the least and most important piece and measure its impact on the game's outcome. The second proxy task is to analyze the saliency of the piece that ultimately secures the win in a self-play game. The third task is to find the smallest sufficient subset of players required to retain a winning position. Then we iteratively remove non-important players according to an explainability method. The explainability method that has the highest area under the curve has the highest explainability quality.

4 Results

We ran several experiments, both for contrasting the effectiveness of different model-interpretability methods and for gaining added insights into the domain-dependent knowledge captured by the learned model.

4.1 Experimental Setup

The model is implemented in Pytorch and trained using asynchronous self-play using two GeForce RTX 3090, AMD Ryzen 9 5950 with 64 GB of RAM, and using RAY [11]. It was trained while playing a total of 630,000 self-play games. The training used stochastic gradient descent with a batch size of 512 and a decaying learning rate that was re-initialized every few thousand iterations.

The saliency map methods in the paper used the implementations in the model interpretability library Captum [10].

4.2 Saliency Methods: Qualitative Evaluation

In image classification, a saliency map is a two-dimensional probability distribution over image pixels, representing their perceived importance for the model's output. The analogy for a board game would be a probability distribution over

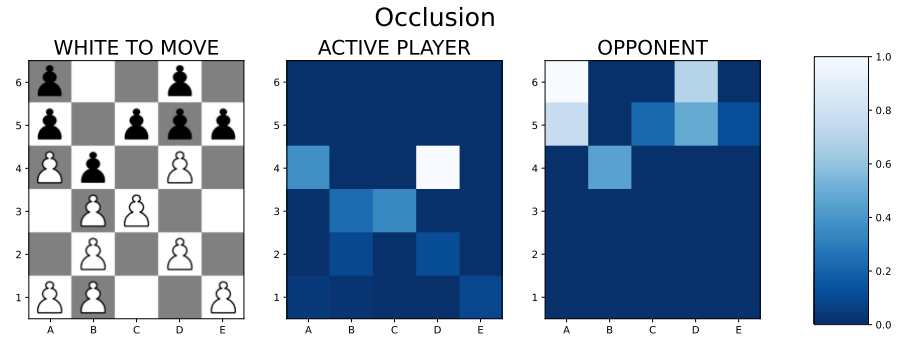

Fig. 2. The saliency map generated by Occlusion; the lighter a square is, the more important the piece occupying the square is.

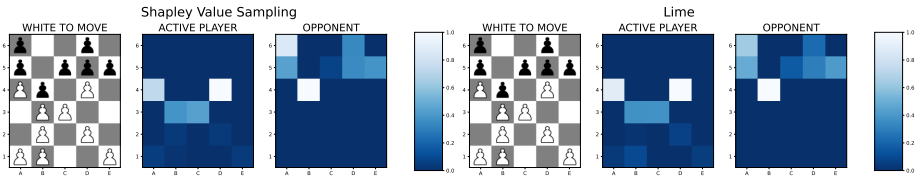

Fig. 3. Model-agnostic saliency methods: Shapley Value Sampling and Lime

the board's squares, representing the square's (or the piece on it) influence on the model's evaluation of the current game state.

Model-agnostic methods typically rely on modifying the input somehow and observing its effect on the model's output. That way, each input feature's attribution (or importance) to the output can be determined. One of the most straightforward of such ablation methods is that of *occlution*, where in our domain we remove a piece from the board and observe the effect of the model's output. Fig. 2 shows the saliency map from such an experiment. [2] It shows clearly that the attacking piece on $d4$ is White's primary asset along with the supporting piece on $c3$ (and the independently potential breakthrough piece on $a4$). Unsurprisingly, for Black, the defending pieces on $a6$, $a5$, $c6$ are $c5$ play the most crucial role. This assessment is in perfect consonance with human (expert-level) assessment: the white pieces on $d4$ and $c3$, with White to move, can collectively win the game on their own, while the piece on $a4$ is a valuable long-term asset severely restricting the mobility of two of the black pieces, thus potentially placing Black later in *zugzwang*, but a well-established expert-level strategy in playing Breakthrough is to force such situations.

We ran the same type of experiment for two additional model-agnostic algorithms, *Shapley Value Sampling (SVS)* and *LIME*, which also find attribution by perturbing the model input parameters. However, they do it in a more refined way, potentially detecting non-trivial input interactions (as described in Section

[2] For ease of comparison, the maps are scaled to be in the range [0.0-1.0].

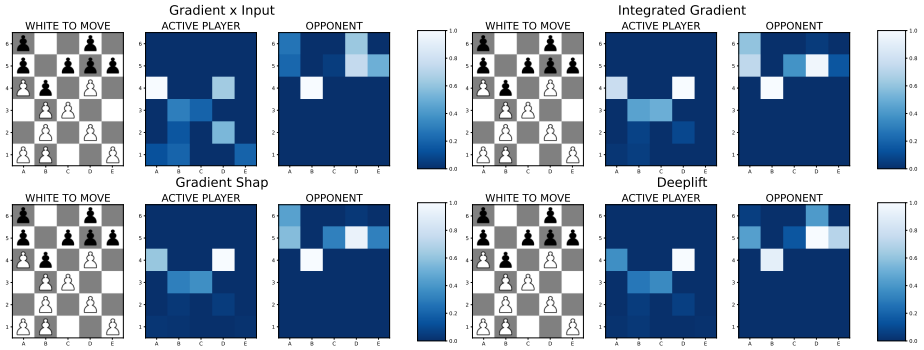

Fig. 4. Model-specific saliency methods.

2). The two methods give almost identical results, depicted in Fig. 3, with the result for the most part also consistent with the one from the occlusion method. The only significant difference is that for Black, the defending player, the piece on *b*4 gets much-added importance (without that defending piece, White will have additional ways to win by immediately playing a piece to that square).

We furthermore experimented with a few model-specific approaches as taking advantage of the model's internals may (in theory) yield further benefits. We looked at several methods that use the model's internal gradient in different ways to detect attribution, and one non-gradient-based, Deeplift. Fig. 4 shows the resulting saliency maps. Again, the more sophisticated methods, Integrated Gradient, GradientShap, and Deeplift, all give intuitively plausible results, whereas the straightforward gradient-based method is more indecisive and thus seems less reliable.

4.3 Saliency Methods: Quantitative Evaluation

Qualitative evaluation as in the previous subsection, albeit able to provide valuable insights, is not sufficient for determining the relative effectiveness of the different saliency map algorithms — a quantitative approach is needed for that.

We started by generating 10,000 game positions from self-play games by stopping play uniformly at random anywhere between 10 and 30 moves (plies) into the game. We played two games from each position in the test suite for each saliency-based approach: one without any intervention and one after removing the most important piece for the player to move as judged by the respective saliency method. The expectation is that for the more reliable indicators of the most-importance piece, the more profound drop we will see in winning ratio.

Table 1 summarizes the result. We see the expected effect in all cases but most profoundly for the *Occlusion* method followed closely by *LIME*, *SVS* and *DeepLift*. This gives us added confidence that these saliency methods are detecting the importance of the different pieces.

Table 1. The positions are placed into 9 bins depending on their evaluation score. The top-most rows shows average winning ratio of the player to move, for each bin, and the remaining rows the same information after removing the highest ranked piece. We also include the average and standard deviation of all methods after removing the lowest ranked piece.

Method	Importance	Proportion of games won								
Nothing deleted	-	**0.09**	**0.26**	**0.36**	**0.44**	**0.51**	**0.55**	**0.64**	**0.74**	**0.90**
Occlusion	Highest	0.08	0.21	0.26	**0.29**	**0.31**	**0.37**	**0.40**	**0.43**	**0.50**
LIME	Highest	**0.04**	0.18	0.26	**0.29**	0.34	0.39	0.45	0.47	0.51
SVS	Highest	**0.04**	0.19	**0.24**	0.32	0.38	**0.37**	0.44	0.45	0.51
Gradient	Highest	0.12	0.23	0.35	0.41	0.49	0.50	0.57	0.63	0.66
Integ. Gradients	Highest	0.06	0.18	**0.24**	0.32	0.38	0.41	0.44	0.47	0.56
Gradient Shap	Highest	0.05	**0.15**	0.27	0.31	0.36	0.41	0.45	0.49	0.57
Deeplift	Highest	**0.04**	0.18	**0.24**	0.34	0.35	0.40	0.46	0.47	0.51
Average	Lowest	0.10	0.27	0.37	0.44	0.50	0.56	0.62	0.75	0.91
Std Dev	Lowest	0.01	0.03	0.05	0.06	0.06	0.05	0.04	0.02	0.00
Mean bin value (before deletion)		**-0.90**	**-0.68**	**-0.45**	**-0.22**	**0.00**	**0.22**	**0.45**	**0.67**	**0.90**

In an attempt to further discriminate the effectiveness of the different approaches in detecting valuable pieces, we looked at how important a pawn reaching the opponent's back rank was judged a few moves earlier. One can think of that information as an indicator of how quickly a particular saliency method realizes the importance of such "breakaway" pieces. Fig. 5 shows that information, and apparently, $LIME$ and SVS seem to put much-added importance on such pieces, whereas *Occlusion* and *Gradient* do not.

It is also of interest to investigate how confidently the methods rank the less important pieces. For that, we use the (functionally grounded) method of smallest sufficient subsets [2], which in our domain translates into the set of pieces required to retain a winning position. To create a test-suite, we sampled positions from random games according to the model's value function to find positions in the current player being only a slight favorite. Then we gradually removed the pawns considered least important, one at a time, and recorded its

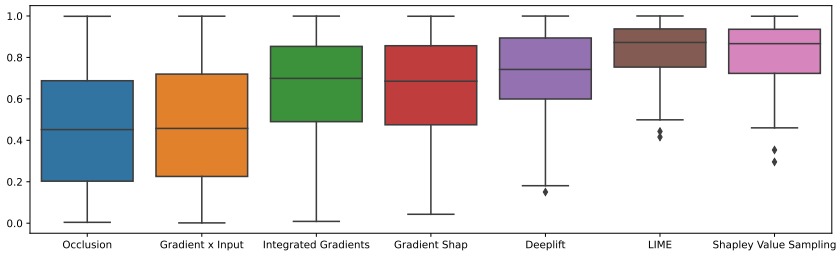

Fig. 5. Distribution of saliency values of a piece 4-ply prior to the winning move.

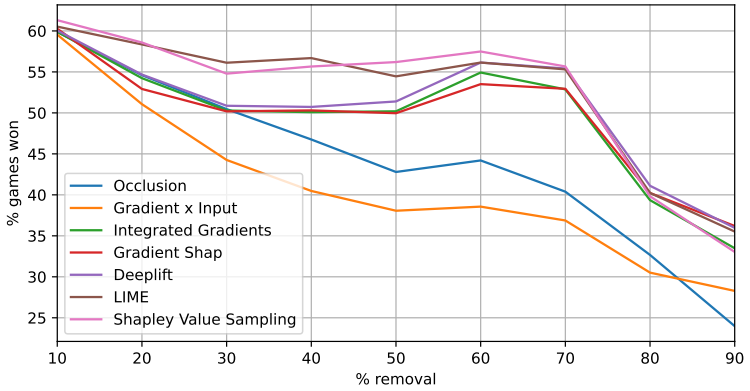

Fig. 6. Effect of gradually removing the least-important pieces.

effect on the winning ratio. Fig. 6 depicts the result. Essentially, the later a curve drops, the more effective the respective saliency method is in ranking pieces by importance. There are two clear winners, *LIME* and *SVS*, and two methods that do notably worse than the other methods, *Occlusion* and *Gradient*.

4.4 Explaining the Explanations

Finally, we were also interested in knowing common higher-level characteristics of pieces judged valuable. One way to unveil such characteristics is to build a surrogate model from hand-made higher-level features and then train the surrogate model to predict the saliency values.

We build such a surrogate model using a LightGBM regression tree. Fig. 7 shows the relative importance of higher-level features we defined for the model; it clearly shows how important it is in Breakthrough to have advanced pieces, but features such as a center-of-mass are also important.

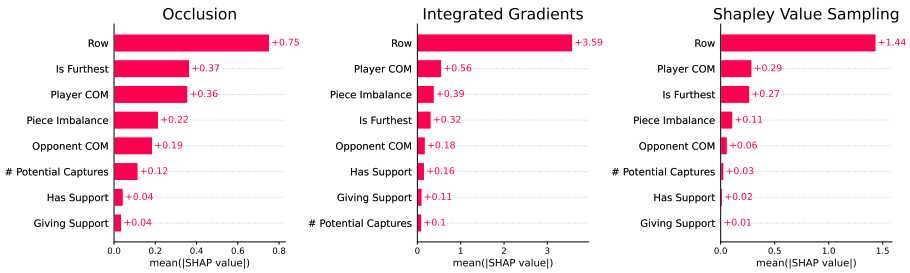

Fig. 7. The average Shapley Values of each of the inputs to the interpretable surrogate model. Here COM stands for center of mass. Has- and Giving Support indicate if the piece is supporting or giving support diagonally to a piece of the same color. # Potential Captures indicates is the number of available capture moves for the current player.

5 Conclusion and Future Work

This paper evaluated several popular saliency-based model interpretability methods on a DNN based game-playing agent, demonstrating their usefulness for identifying the most and least essential game pieces. The more sophisticated attribution methods, like Shapley Value Sampling and LIME, performed overall the best. One of the strengths of those methods is that they can capture non-trivial interactions between the inputs, which seems well suited to identify various in-game piece dynamics. Moreover, those methods are both model-agnostic, making them well-suited for a wide range of models.

As for future work, we plan to evaluate the applicability of those methods on other abstract board games to better establish their usefulness in the domain of abstract board games. Also, we only scratched the surface when looking at higher-level domain concepts (beyond piece importance), and further research into that direction holds promise.

References

1. Francesco Bodria, Fosca Giannotti, Riccardo Guidotti, Francesca Naretto, Dino Pedreschi, and Salvatore Rinzivillo. Benchmarking and Survey of Explanation Methods for Black Box Models. 51(5):1–33, 2021.
2. Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *Advances in Neural Information Processing Systems*, 2017-Decem:6968–6977, 2017.
3. Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning.
4. Scott M Lundberg, Paul G Allen, and Su-In Lee. A Unified Approach to Interpreting Model Predictions. Technical report.
5. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?". In *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 1135–1144. ACM Press, 2016.
6. Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *34th International Conference on Machine Learning, ICML 2017*, 7:4844–4866, 2017.
7. Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. pages 1–8, 2013.
8. Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41(3):647–665, 2014.
9. Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. Technical report, 2017.
10. The PyTorch Team. Model interpretability and understanding for PyTorch., 2021. Accessed on 20-09-2021, https://captum.ai.
11. The Ray Team. Ray provides a simple, universal API for building distributed applications., 2021. Accessed on 20-09-2021, https://docs.ray.io.
12. Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, 8689 LNCS(PART 1):818–833, 2014.