RL-CAM: Visual Explanations for Convolutional Networks using Reinforcement Learning

Soumyendu Sarkar*† Ashwin Ramesh Babu † Sajad Mousavi † Sahand Ghorbanpour
Vineet Gundecha Antonio Guillen Ricardo Luna Avisek Naug

Hewlett Packard Enterprise
USA

Abstract

Convolutional Neural Networks (CNNs) are state-of-the-art models for computer vision tasks such as image classification, object detection, and segmentation. However, these models suffer from their inability to explain decisions, particularly in fields like healthcare and security, where interpretability is critical. Previous research has developed various methods for interpreting CNNs, including visualization-based approaches (e.g., saliency maps) that aim to reveal the underlying features used by the model to make predictions. In this work, we propose a novel approach that uses reinforcement learning to generate a visual explanation for CNNs. Our method considers the black-box CNN model and relies solely on the probability distribution of the model's output to localize the features contributing to a particular prediction. The proposed reinforcement learning algorithm has an agent with two actions, a forward action that explores the input image and identifies the most sensitive region to generate a localization mask, and a reverse action that fine-tunes the localization mask. We evaluate the performance of our approach using multiple image segmentation metrics and compare it with existing visualization-based methods. The experimental results demonstrate that our proposed method outperforms the existing techniques, producing more accurate localization masks of regions of interest in the input images.

1. Introduction

Convolutional Neural Networks have become the state-of-the-art technique for many computer vision tasks, such as image classification, object detection, segmentation, and many others. They are very effective at learning complex visual features from images, allowing them to accurately classify images with a high level of accuracy. However, these models when compared to traditional machine learning techniques suffer from issues such as the "generalization vs interpretability" tradeoff. For example, a decision tree might be an easy-to-interpret algorithm but might tend to generalize poorly on the provided dataset, while a hard to interpret algorithms usually generalize better with millions of parameters such as CNNs.

Figure 1. Output of RL framework for visual explanation via localization

Interpretability is an important concern in many applications where deep learning is currently being used, particularly in the fields such as healthcare and finance, where the impact caused by incorrect decisions might have severe adverse effects. It is important to understand the reasoning behind a model's predictions and to ensure that the model is making decisions based on relevant features. When it comes to architectures such as CNNs, there are several reasons due to which they might lack interpretability. One reason is the use of a large number of complex nonlinear operations, making it difficult to understand how the features are extracted from the images. Additionally, the use of a high number of parameters makes it difficult to trace the contributions of each feature during the final decision. Researchers in the past have developed various methods for interpreting CNNs. These approaches aim to reveal the underlying features that the model is using to make its pre-

*Corresponding author. Email: soumyendu.sarkar@hpe.com
†Equal contribution.

dictions with techniques such as visualization, attribution methods, and so on.

Visualization-based methods aim to visualize the regions of an image that the model is focusing on. For example, one of the first approaches to provide visual explanations for CNN was proposed by Zeiler et. al [22] using Deconvnets and Zhou et. al as Class Activation Mapping (CAM) [24]. Furthermore, there were numerous other incremental works such as Grad-CAM, Grad-CAM++ and many others.

In this work, we propose RL-CAM a reinforcement learning approach that can learn a policy to generate a visual explanation for Convolutional Neural Networks as shown in figure 1 . This approach considers the CNN model as a black-box and completely relies on the probability distribution of the model's output without requiring any other information to localize features that contribute towards a particular prediction. The RL framework is composed of an agent with two actions, the forward action, and the reverse action. The forward action attempts to add distortion to regions in the input image that is most sensitive at a given step which builds the segmentation mask while the reverse action attempts to remove distortion that was previously added and is insignificant, essentially clearing up the segmentation mask. This process is performed until the model misclassifies the input image creating an adversarial sample. The objective is to extract the area where the perturbations were added creating a mask that exposes the most significant regions that contributed towards a particular decision. As a result, a finer localization of the Region of Interest (ROI) is generated. Multiple image segmentation metrics such as Intersection over Union (IOU) and Dice Coefficient were computed to compare the performance of the proposed method with the existing techniques. Unlike the majority of the recent research that uses bounding boxes to evaluate the performance of their proposed approach, we use a segmentation map to accurately measure the localization performance with metrics such as IOU and Dice Coefficient as shown in figure 2. Our approach achieves over 20 percent performance gains over existing popular approaches when evaluated on imagenet dataset.

The main Contribution of the work can be summarized as,

1. A novel Reinforcement Learning agent that can generate an accurate localization mask of ROI for the input images with a gradient-free approach.

2. The learned policy outperforms the existing popular visual explanation techniques from the recent literature.

2. Related Works

One of the earliest and most well-known visual explanation techniques is Class Activation Mapping (CAM) introduced by Zhou et al. (2016) [24], which highlights the regions of an image that are most important for a given class. Grad-CAM (Selvaraju et al., 2017) [15] is another popular technique that extends CAM by using gradients to weight the contribution of each feature map to the final prediction. Other visual explanation techniques include Deconvolutional Networks (Zeiler and Fergus, 2014) [22], guided backpropagation (Springenberg et al., 2015) [18], and saliency maps (Simonyan et al., 2013) [17], which highlight the pixels that are most relevant for a given class.

More recent works have focused on improving the accuracy and interpretability of these visual explanation techniques. For example, Kim et al. (2018) [5] proposed CAM2, an improved version of CAM that incorporates attention mechanisms to improve localization accuracy. Wang et al. (2022) [20] introduced Layer-Wise Co-Activation Mapping (LCAM), which uses a layer-wise co-activation matrix to visualize the feature maps of a CNN. Akhtar and Mian (2022) [2] proposed a spatial-spectral complementarity-based approach for weakly supervised object localization, which combines spectral and spatial attention mechanisms to improve the accuracy of visual explanations.

In recent times, researchers identified the unreliability of gradient-based approaches to generate visual explanations/localization maps [1, 19]. In addition to these visual explanation techniques, there has been a growing interest in developing methods for generating textual explanations of CNN predictions like eigenCAM [21] and many others. Similarly, Hendricks et al. (2016) [4] proposed a method for generating natural language explanations of image classification decisions. Recent works have extended this approach to generate more detailed and informative textual explanations (e.g., Zhang et al., 2021) [23]. Similarly, work done by Muhammad et. al [8] proposed a technique that uses the principal components of the learned representations from the convolutional layers to create visual explanations.

Another approach that has been explored recently is perturbation-based explainability to understand the behavior of convolutional neural networks (CNNs) [3, 6, 7]. This involves making small changes to the input data and observing the corresponding changes in the output of the network. By analyzing the changes in output, insights can be gained as to how the network makes decisions and what features are focused on.

Overall, the development of explainability techniques for image classification CNNs is an active area of research, with many new and innovative approaches being proposed. These techniques can potentially improve the interpretability and trustworthiness of CNN models and facilitate their deployment in real-world applications. Furthermore, reinforcement learning has shown tremendous success in applications such as healthcare, sustainable energy, controls,

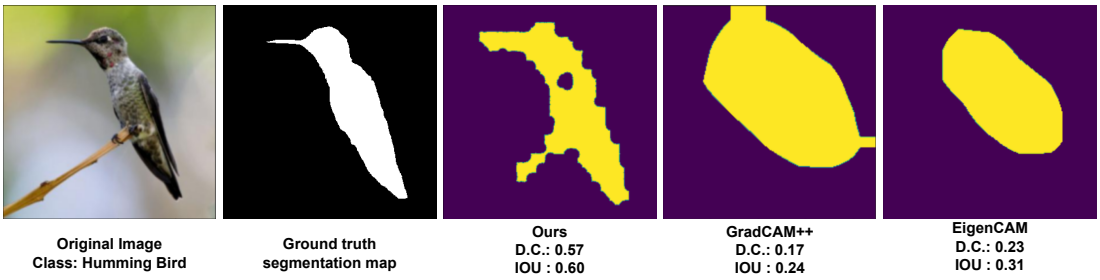

Figure 2. Compare the performance of the proposed method and the competitors with the ground truth segmentation map

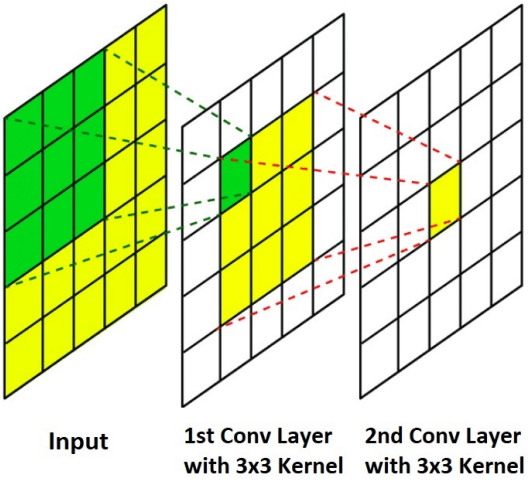

Figure 3. Receptive Field of Convolutional Neural Networks

and many others [10–14]. This work, to the best of our knowledge, is the first use of reinforcement learning for explainability.

Receptive fields in CNNs refer to the region of the input image that a single neuron in a given network layer is sensitive to. Each layer consists of a set of filters that convolve over the input image and produce a set of feature maps as shown in figure 3. The receptive fields of a neuron in a particular layer are determined by the size of the filter in that layer. As information passes through the network, the receptive fields of the neurons become larger and more complex, allowing the network to capture increasingly abstract and high-level features in the input image. For example, for two sequential convolutional layers l_1 and l_2 with kernel size k and stride s, the receptive field can be defined as,

$$r(i-1) = s_i \cdot r_i + (k_i - s_i) \tag{1}$$

A more generalized equation that applies the above operation recursively for L layers can be defined as,

$$r0 = \sum_{i=1}^{L}((K_i - 1)\prod_{j=1}^{l-1} s_j) + 1 \tag{2}$$

Here, r_0 denotes the desired RF of the architecture.

The size of the receptive field in convolutional neural networks (CNNs) can have a significant impact on the performance of saliency map methods. Saliency maps are used to identify the most important features or regions in an image and are commonly used in computer vision applications such as object recognition and scene understanding.

Large receptive fields in CNNs tend to capture more global information and context, while small receptive fields capture more local details. One potential negative effect of using large receptive fields for gradient-based visualization is that it can lead to reduced spatial resolution in the resulting heatmap. This is because the receptive field of a neuron in a CNN determines the size of the local region of the input image that it is sensitive to. When the receptive fields become large, the neurons respond to a broader range of input features, which can make it difficult to pinpoint the exact location of the salient features in the input image. This can lead to a loss of spatial resolution in the CAM heatmap, which can make it harder to interpret the results.

3. RL-CAM: Reinforcement Learning Framework

3.1. Problem Formulation

A trained Deep Neural Network (DNN) model under evaluation can be represented as $y = argmax f(x;\theta)$, where x denotes the input image, y represents the prediction and θ represents the model parameters. A non-targeted black-box attack without access to the θ generates a perturbation δ such that, $y \neq argmax f(x+\delta;\theta)$. The distance between the original and the adversarial sample, $D(x, x+\delta)$ will be any function of the l_p norms where the agent learns to keep this factor minimum. The objective is to extract the areas where the perturbation is added creating a mask that exposes the most significant regions that contributed towards a particular decision.

3.2. Overall Architecture

In our approach, the input image is divided into square patches of size $n \times n$ and, then the sensitivity of the ground

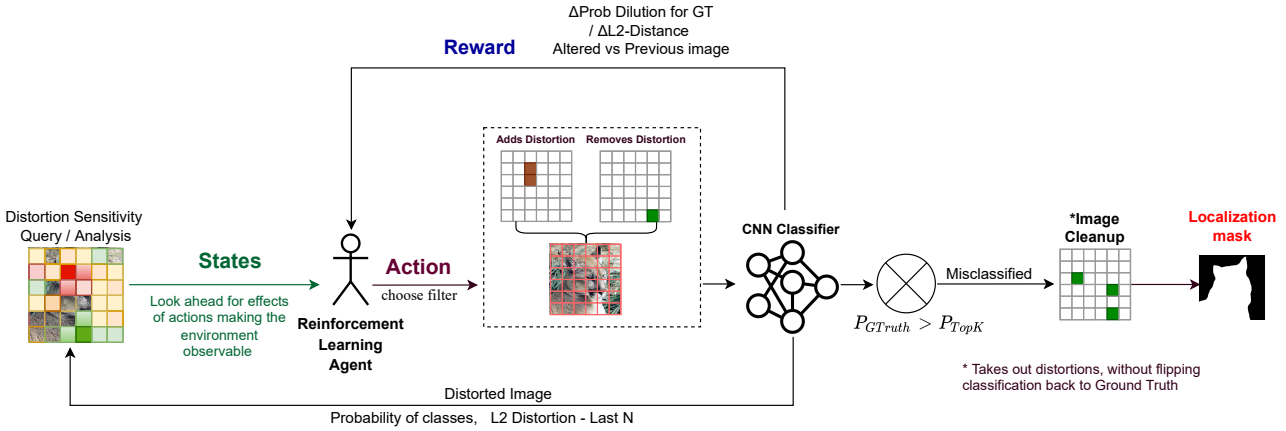

Figure 4. Reinforcement Learning architecture for RLAB.

truth probability P_{GT}, to addition and removal of distortion, is computed for each patch. Based on this sensitivity information, the RL agent takes two actions,

1. Patches to which distortion are added
2. Patches from which the distortions are removed

This process is done iteratively until the model misclassifies the image or until the budget for the number of maximum allowed steps is reached. Finally, once the adversarial sample is generated (model misclassifies), we perform an iterative image cleanup as a post-processing step to further minimize D. The difference between the adversarial sample generated and the original image highlights the regions where the distortions were added. This mask provides a more accurate localization of features that contributed towards the particular prediction. The overall flow of the proposed method is represented in Figure 4 and 5.

4. RL for Localization map

4.1. State Design

We created a state space that provides the necessary visibility to the RL agent, while remaining simple enough for efficient training. To achieve this, we conducted a sensitivity analysis to pinpoint the most responsive areas of a given image.

4.1.1 Sensitivity Analysis

We utilized distortion filters (masks) of the same size as the square patches $n \times n$ for the sensitivity analysis. Each filter had hyperparameters such as noise levels and brightness levels, which remained fixed throughout the experiment. At each step of the training and validation, we applied the mask across all the square patches to assess the shift in the ground truth classification probability P_{GT}. We chose to keep the hyperparameters associated with the distortion filters (such as noise intensity and amount of blurring) to a minimum to allow for more precise distortion addition in successive steps and to control the L_p norm. Additionally, we constrained the distorted samples to values within the range of $[0, 1]^d$.

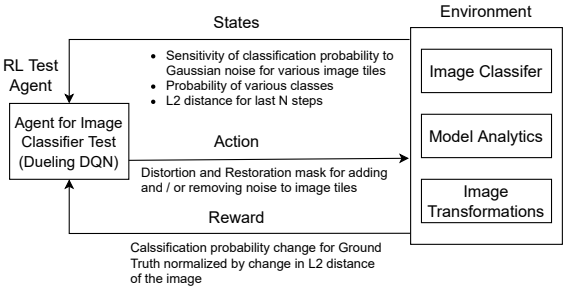

Figure 5. Reinforcement Learning agent for RLAB

$LIST_{ADD}$	Square patches in descending order of normalized sensitivity to addition of distortion
$LIST_{REMOVE}$	Square patches in ascending order of normalized sensitivity to removal of distortion
$LIST_{PROB}$	Classification probability of various classes at this step
$LIST_{L2}$	L2 distance from original for the last $N_{steps} = 4$ steps

Figure 6. RL states

4.1.2 State Vector

We constructed the state vector using the results of the image sensitivity analysis. The state vector is ordered based on the degree of drift in P_{GT} for patches during the addition ($LIST_{ADD}$) and removal ($LIST_{REMOVE}$) of distortions. Additionally, the state vector includes the classification probabilities of each class at each step ($LIST_{PROB}$) and the L_p norm, as shown in Figure 6.

4.2. Action

During each step, the RL agent selects the number of patches (N_{ADD_DIST}) to which distortion will be added from $LIST_{ADD}$ and the number of patches (N_{REM_DIST}) from which distortions will be removed from $LIST_{REMOVE}$, as illustrated in Figure 7. The RL action space was designed to be discrete and straightforward to allow for easy learning of the RL policy. We made sure that $N_{REM_DIST} < N_{ADD_DIST}$ so that distortions are progressively added at each step to keep the number of queries optimally low. However, it is possible for the patch from which we are removing the distortion to have previously undergone multiple distortions, which would only lower the net increase of the L_2 distance for the step. Also, to ensure computational efficiency, we limited the action space with $N_{ADD_DIST} \in [1, N_{max}]$, where N_{max} is a hyperparameter set to 8 for ImageNet (224×224) image size with a 2×2 patch size, to balance effectiveness and computation.

4.2.1 Alternate to Tree Search

The inspiration for having two actions (addition and removal) comes from the application of reinforcement learning in board games. In board games, the most effective moves are determined through a computationally expensive process called Deep Tree Search (DTS), which involves searching multiple layers ahead to anticipate future moves and outcomes. Even with approximations such as Monte Carlo Tree Search (MCTS), DTS remains computationally expensive. However, in this problem, we have the ability to reset previous moves when we realize that a less optimal move was made. In the RLAB platform, this is achieved by removing distortions from patches where distortions were added in the previous step and adding distortions to other patches based on the current state of the modified image. This approach is akin to replaying all the moves in a single step and limiting the sensitivity analysis to the current state of the image, without the need for a tree search. Our method simplifies the complexity from $O(N^d)$ to $O(N)$, where N is the computation complexity of one level of evaluation corresponding to the image size, and d represents the depth of the tree search. In other words, d signifies how many queries and actions we consider looking ahead if we were performing a tree search. The value of d can range from 1 to the maximum number of steps (max_steps).

4.3. Reward

We introduce a probability dilution (PD) metric that gauges how much the classification probability deviates from the ground truth towards other classes. The difference between the PD of the original image and that of the altered image resulting from an action (ΔPD) measures the efficacy of the action, while the change in L_2-distance (ΔL_2) quantifies the added distortion and serves as the cost for the action. We use the normalized PD (as given in Equation 3)

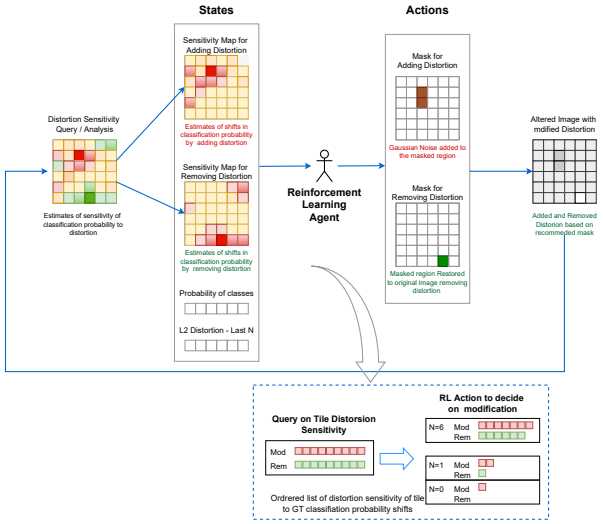

Figure 7. Details of the Reinforcement Learning Action step (addition and removal of distortion) for RLAB

Algorithm 1: RLAB: Reinforcement Learning Training

Initialization: Policy parameters
Input: Validation set, number of iterations Max_{iter} = 3500
Output: Optimized policy for Dueling DQN
for *image* ***in*** *validation set* **do**
    Load the image;
    Calculate reward R_t and advantage $\hat{A}_t$ based on current value function;
    Calculate sensitivity of ground truth classification probability P_{GT} to change in distortion for square patches;
    $i \leftarrow 0$;
    $Pred_{fstep} \leftarrow 1 - P_{GT}$;
    while $Pred_{GT} == Pred_{fstep}$ ***and*** $i < Max_{iter}$ **do**
        Collect set of trajectories (state, action) by running policy $\pi_k = \pi(\theta_k)$ in the environment $\rightarrow$ action (N_{add_dist}, N_{rem_dist}) ;
        Calculate reward R_t and TD error;
        Update the DQN policy;
        Compute/take action and perform prediction $Pred_{fstep}$;
        $i \leftarrow i + 1$;
    end
end

as the reward metric (R_t) for our RL agent.

$$R_t = \Delta PD_{normalized} = -\Delta PD/\Delta L_2 \quad (3)$$

At each step, the probability distribution is updated in the state vector ($LIST_{prob}$), allowing the agent to select the optimal action while maintaining L_p and the number of steps/queries. We set the discount factor (γ) to 0.95 via hyperparameter tuning, where it determines the extent to which the RL agent values future rewards compared to those at the current step.

4.4. RL Algorithm

We used the Dueling DQN [16] algorithm-based Reinforcement Learning (RL) agent for RLAB as a localization agent. The Dueling DQN algorithm splits the Q-values into two parts: the value function V(s) and the advantage function A(s, a). The same neural network splits its last layer into two parts, one of them to estimate the state value function for states (V(s)) and the other one to estimate state-dependent action advantage (A (s, a)). It then combines both parts into a single output, estimating the Q-values. This change is helpful because sometimes it is unnecessary to know the exact value of each action. So just learning the state-value function can be enough in some cases. The main benefit is generalizing learning across actions without imposing specific changes to the underlying reinforcement learning algorithm. The Dueling DQN model fits well with the discrete action space of a limited number of possible values of N_{ADD_DIST} and N_{REM_DIST} and has the suitable complexity for effective prediction with a reasonably bounded training. Algorithm 1 explains the overall training procedure for the proposed approach.

5. Experimental Setup

We trained our RL approach on image classification datasets ILSVRC2012 [9]. 80 percent of the original validation set was used to train the RL algorithm, and 20 percent of the original validation set was used for evaluation. We experimented with three off-the-shelf pre-trained Convolution-based Neural Network architectures: ResNet-50, Inception-V3, and VGG-16. We set a maximum iteration of 3500 or until the model misclassifies a sample.

All experiments were performed for a patch size of 2×2 and a gaussian noise-based distortion filter as we got the best results for this configuration.

We evaluate the performance of our approach on ImageNet dataset where 1000 samples were randomly sampled along with their corresponding ground truth segmentation maps. All compared methods were evaluated on the same set of images for a fair comparison. To generate the segmentation map from class activation maps, we directly binarize them with a threshold of 40% of maximum intensity. 40% yielded the best results when compared with the ground truth map after extensive evaluation of multiple thresholds. The localization map from the RL framework is extracted by computing the difference between the perturbed image and the original image.

The majority of the work in the literature focus on using bounding box for object localization, specifically to evaluate the performance of interpretability methods. To generate more accurate localization, we use ground truth segmentation maps, unlike other works. We used two metrics to evaluate the performance of the competitors and the proposed method, Intersection over Union (IOU) and Dice Coefficient which are the two most common methods used in image segmentation. The IOU metric is a measure of the overlap between the predicted bounding box and the ground truth bounding box of an object. It is computed as the ratio of the intersection of the predicted and the ground truth bounding boxes to the union of the two boxes. In other words, it measures how much of the predicted bounding box overlaps with the ground truth bounding box. The values are expressed between the range of 0 and 1, where a value of 0 indicates no overlap between the predicted and the ground truth bounding boxes, while a value of 1 indicates a perfect match between the two.

Similarly, the Dice coefficient is defined as the ratio of the intersection of two sets to the average size of the two sets. In the context of object localization, the two sets are the predicted bounding box and the ground truth bounding box. The Dice coefficient measures how well the predicted bounding box aligns with the ground truth bounding box.

$$\text{Dice coefficient} = \frac{2 \times |A \cap B|}{|A| + |B|} \quad (4)$$

The Dice coefficient ranges from 0 to 1, with a value of 1 indicating a performance match between the predicted and ground truth bounding boxes. A value of 0 indicates no overlap between the predicted and ground truth boxes.

The computation for the complete pipeline is GPU-dependent and is efficiently batched, and scaled on GPUs. Caching techniques were used for pre-computed information such as the noise masks for improved efficiency. Apollo servers with $8 \times V100$ 32 GB GPUs were used for training and validation. We processed 16(images per GPU) x 8(GPUs) = 128 images in a batch for the complete pipeline.

5.1. Results and Discussion

Results presented in Table 1 represent the dice coefficient values averaged over 1000 samples considered. We achieve an improvement in the performance of approximately 22 percent for ResNet-50 architecture, 29 percent improvement for VGG-16 and 22 percent improvement for Inception-v3 architecture. It can be observed that the performance of the localization is dependent on the perfor-

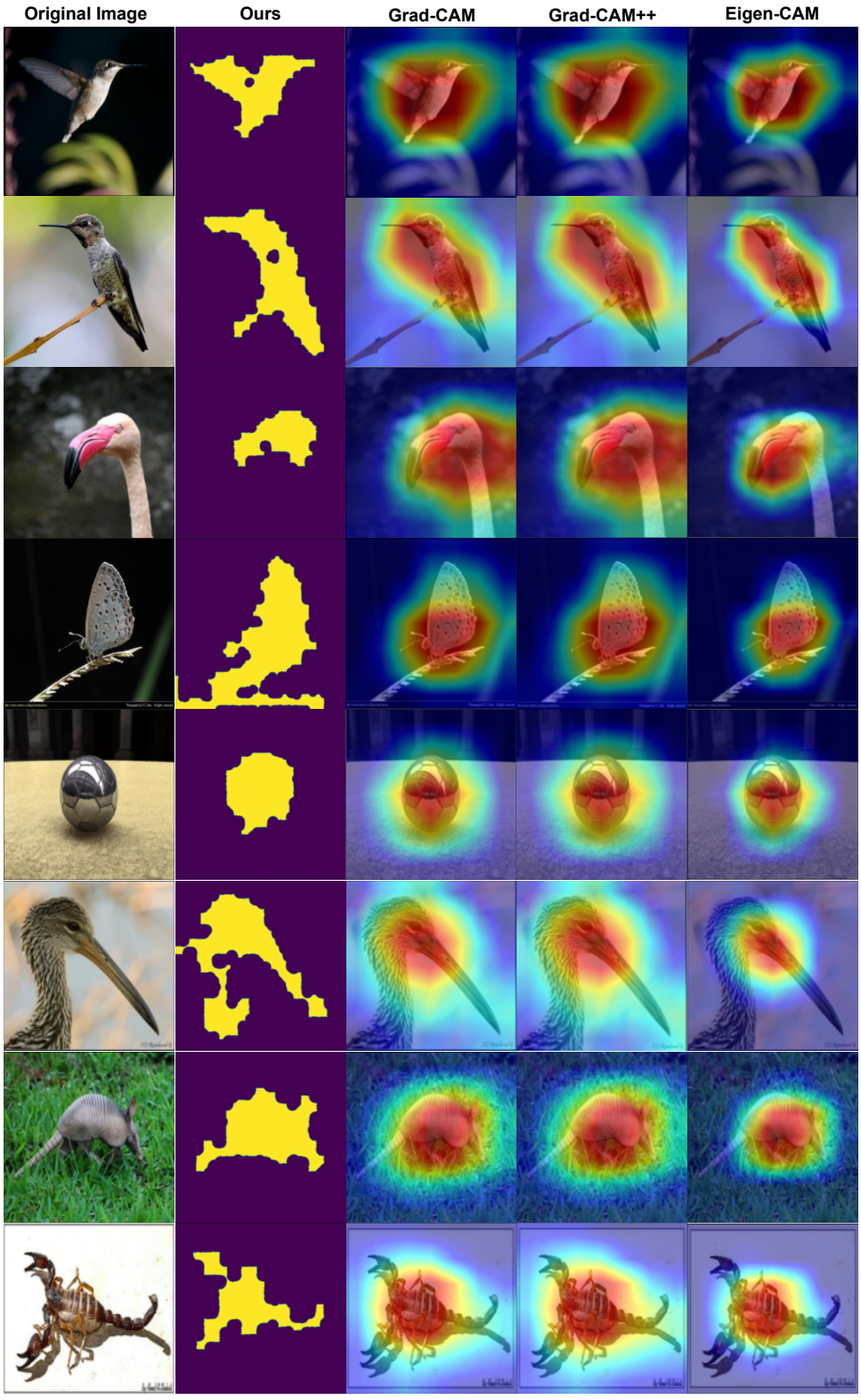

Figure 8. Example output of our approach compared with our competitors

Table 1. Localization results for various architectures. Comparing RL framework with other popular visualization/localization approaches. **Metric: Dice Coefficient**

Architectures	GradCAM	GradCAM++	EigenCAM	AblationCAM	Ours
ResNet-50	0.21	0.28	0.39	0.36	0.61
VGG-16	0.16	0.19	0.21	0.29	0.58
Inception-V3	0.24	0.18	0.4	0.37	0.59
Averaged	0.20	0.23	0.33	0.34	**0.59**

Table 2. Localization results for various architectures. Comparing RL framework with other popular visualization/localization approaches. **Metric: Intersection over Union (IOU)**

Architectures	GradCAM	GradCAM++	EigenCAM	AblationCAM	Ours
ResNet-50	0.26	0.24	0.43	0.51	0.64
VGG-16	0.19	0.26	0.33	0.32	0.69
Inception-V3	0.21	0.5	0.36	0.29	0.73
Averaged	0.22	0.33	0.37	0.37	**0.68**

mance of the architecture. This means an architecture that has been well-trained can localize better when compared to a weakly trained architecture. Figure 8 shows some sample output of our proposed approach compared to the other popular visualization approaches.

Similarly, Table 2 compares the performance of the proposed method against popular visualization approaches from the literature. It can be observed that our approach outperforms other competitor approaches by a significant margin.

6. Conclusion

Convolutional Neural Networks have revolutionized computer vision tasks with their high accuracy in classifying complex visual data. However, due to their complex architecture and large number of parameters, they lack interpretability, which is very essential. Many researchers have developed methods for interpreting CNNs, including visualization-based methods such as CAM, Grad-CAM, and others. In this work, we proposed a novel RL-CAM approach that uses reinforcement learning to generate visual explanations for CNNs without requiring any prior information. The proposed method outperforms existing popular techniques in terms of localization accuracy, with the contributions of this work being the novel Reinforcement Learning agent that generates accurate localization masks of ROI for input images and the improvement over existing visual explanation techniques from the literature. The proposed method is expected to pave the way for further research to enhance the interpretability of CNNs in other domains such as medical and satellite imaging.

References

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pages 9505–9515, 2018.

[2] Naveed Akhtar and Ajmal Mian. Exploiting spatial-spectral complementarity for weakly supervised object localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[3] Minshu Cheng, Wei Huang, Dongyu Yang, and Yueting Zhuang. Interpretable convolutional neural networks for text classification. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[4] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19. Spring, 2016.

[5] Byeongho Kim, Martin Wattenberg, and Justin Gilmer. Improving interpretability of deep learning models: Cam 2.0. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5954–5963, 2018.

[6] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rusty Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *Journal of Machine Learning Research*, 22(168):1–37, 2021.

[7] Jangho Kim, Yoonho Lee, and Heeyoul Kim. Learning disentangled representations via perturbation-based explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[8] Mohammed Bany Muhammad and Mohammed Yeasin. Eigen-cam: Class activation map using principal components. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy,

Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[10] Soumyendu Sarkar, Vineet Gundecha, Sahand Ghorbanpour, Alexander Shmakov, Ashwin Ramesh Babu, Alexandre Pichard, and Mathieu Cocho. Skip training for multi-agent reinforcement learning controller for industrial wave energy converters. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 212–219. IEEE, 2022.

[11] Soumyendu Sarkar, Vineet Gundecha, Sahand Ghorbanpour, Alexander Shmakov, Ashwin Ramesh Babu, Alexandre Pichard, Mathieu Cocho, and Hewlett Packard Enterprise. Function approximations for reinforcement learning controller for wave energy converters.

[12] Soumyendu Sarkar, Vineet Gundecha, Alexander Shmakov, Sahand Ghorbanpour, Ashwin Ramesh Babu, Paolo Faraboschi, Mathieu Cocho, Alexandre Pichard, and Jonathan Fievez. Multi-objective reinforcement learning controller for multi-generator industrial wave energy converter. In *NeurIPs Tackling Climate Change with Machine Learning Workshop*, 2021.

[13] Soumyendu Sarkar, Vineet Gundecha, Alexander Shmakov, Sahand Ghorbanpour, Ashwin Ramesh Babu, Paolo Faraboschi, Mathieu Cocho, Alexandre Pichard, and Jonathan Fievez. Multi-agent reinforcement learning controller to maximize energy efficiency for multi-generator industrial wave energy converter. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12135–12144, 2022.

[14] Soumyendu Sarkar, Sajad Mousavi, Ashwin Ramesh Babu, Vineet Gundecha, Sahand Ghorbanpour, and Alexander K Shmakov. Measuring robustness with black-box adversarial attack using reinforcement learning. In *NeurIPS ML Safety Workshop*.

[15] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[16] Mohit Sewak. Deep q network (dqn), double dqn, and dueling dqn. In *Deep Reinforcement Learning*, pages 95–108. Springer, 2019.

[17] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[18] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. In *International Conference on Learning Representations*, 2015.

[19] Hendrik Strobelt, Sebastian Gehrmann, Bernd Huber, Hanspeter Pfister, and Alexander M. Rush. The (un)reliability of saliency methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2051–2054. IEEE, 2018.

[20] Shuhui Wang, Qingxin Zhu, Qingsong Wen, and Feng Liang. Deep layer-wise co-activation mapping for interpretable feature visualization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[21] Haofu Wu, Zhe Chen, Raghuveer Mukkamala, and Ming-Hsuan Yang. Eigencam: Class activation mapping with eigenvector regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13383–13392, 2020.

[22] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[23] Xinyun Zhang, Tengyu Ma, Jason Wei, and Zachary C. Lipton. Generating high-quality textual explanations for neural predictions. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4361–4373, 2021.

[24] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.