CNN-based small object detection and visualization with feature activation mapping

Medhani Menikdiwela*‡, Chuong Nguyen*†‡, Hongdong Li*‡ and Marnie Shaw‡
*Australian Centre of Excellence for Robotic Vision
‡Research School of Engineering, College of Engineering and Computer Science
Australian National University
Canberra ACT 2601, Australia
Email: firstname.lastname@anu.edu.au
†Quantitative Imaging, CSIRO Data61
Canberra ACT 2601, Australia
Email: chuong.nguyen@csiro.au

***Abstract*—Object detection is a well-studied topic, however detection of small objects still lacks attention. Detecting small objects has been difficult due to small sizes, occlusion and complex backgrounds. Small objects detection is important in a number of applications including detection of small insects. One application is spider detection and removal. Spiders are frequently found on grapes and broccolis sold at supermarkets and this poses a significant safety issue and generates negative publicity for the industry. In this paper, we present a fine-tuned VGG16 network for detection of small objects such as spiders. Furthermore, we introduce a simple technique called "feature activation mapping" for object visualization from VGG16 feature maps. The testing accuracy of our network on tiny spiders with various backgrounds is 84%, as compared to 72% using fined-tuned Faster R-CNN and 95.32% using CAM. Even though our feature activation mapping technique has a mid-range of test accuracy, it provides more detailed shape and size of spiders than using CAM which is important for the application area. A data set for spider detection is made available online.**

***Keywords*—object detection, heat map, CNN, R-CNN, feature activation map**

I. Introduction

Although object detection has long been studied in computer vision[7], [15], detecting small-sized objects in an image remains a challenging task [4], [11]. The difficulties involved in small-object detection are multi-fold, but the main challenge comes from the relatively-small size of an object in an image, compared with its background, e.g. only 1–5% of pixels in an image are occupied by the small object of interest. In this paper we propose to adapt a simple network such as the VGG16 for small object detection. We also introduce a new visualization technique called "feature activation mapping" for object localization once the object is detected.

Small object detection is important in various scenarios including autonomous vehicles, mobile robots, agriculture and tele-operation. In the agriculture scenario, we are particularly interested in detecting redback spider on grapes or broccoli. In the grape industry around Australia, screening of redback spiders, one of the most venomous spiders to humans, remains a major challenge. Grape inspection is mostly done while grapes are being picked on farms. Although grape pickers are trained to identify and remove grape bunches having spiders or spider webs, there is still a significant chance of missing them due to their small size and occlusion. Therefore, it would be highly desirable to find an effective solution for removing redback spiders from grapes and broccoli, without causing damage to crops. A tentative method is to identify and separate spider and non-spider grape bunches at the inspection or the packaging stage by using deep networks [10]. Here we use the well-known VGG16 network [20]. The network faces the same challenges as humans when detecting spiders. As the insects can be very small, i.e. a few pixels wide, their features are likely to get lost after several convolution and pooling layers. Their appearance changes depending on viewing angle and distance and spiders can be occluded partially or completely when deep inside grape bunches.

II. Related works

Object detection has been significantly improved thanks to recent advances in deep-learning. These include the Fast and Faster R-CNN [7], [15]. A recent extension of R-CNN [5] focused on detecting small objects. The algorithm reached a mean average precision accuracy of 23.5%. Although these networks perform well, the training is very expensive and requires a large set of image data. Contextual action recognition also describes how to use R-CNN for more than one region for classify objects [8]. Furthermore, Jianan et al [11] recently introduced method called "perceptual GAN" for detecting small objects including pedestrians and traffic signs. VGG16 network [20] was designed for object classification and has been extended to perform various additional tasks [6], [12]. In this paper, we aim to see how far this network can be adjusted for this small object detection task.

One additional task is to locate the classified object by visualizing what a network perceives. There are a number of works on visualizing deep neural networks [13], [23], [24]. Zeiler et al [23] used a deconvolutional network to visualize layer outputs. Zhou et al [24] used a network that can perform both scene recognition and object localization in a single forward-pass. Neither includes any fully connected layers to

978-1-5386-4276-4/17/$31.00

the network structure. Therefore, conventional networks like VGG [20], AlexNET [9], GoogLeNet [21] need to replace fully connected layers with a GAP (Global Average Pooling) layer for the visualization. However, removing those layers can reduce the classification accuracy. A recent visualization method called "gradient-weight class activation mapping" does not require any changes in the network [18].

Heat map is a visualization method that highlights features learned by a deep neural network. There are several methods to compute heat maps including "sensitivity analysis" based on neural network partial derivatives [2], [19], deconvolutional method [16] and "layer-wise relevance propagation" algorithm [1], [17]. Maxime et al [14] treated the last fully connected layers as convolutions and introduced a global max pooling layer. Alessandro et al [3] introduced an object localization method which has negative examples generated from the positive examples by covering the object with a bounding box. The sensitivity analysis method by Wojciech et al [16] does not consider the direction of the gradient flow. The deconvolution method network [16] predicts neighboring pixels at deconvolutional layers. Finally, the global average pooling layer by Zhou et al [25] has also performed well on object visualization.

III. Methodology

A. Network architecture

Our network architecture is based on VGG16, type of a convolutional neural network(CNN) [20] as shown in Figure 1. This network has 13 convolution layers with rectified linear units, 5 pooling layers and 3 fully connected layers. VGG16's last fully connected layer of 1000 outputs is replaced by a new layer with only 2 binary outputs for spider and non-spider classes. An existing VGG16 model trained on the ImageNet data set [9] was fine-tuned on our new spider data set.

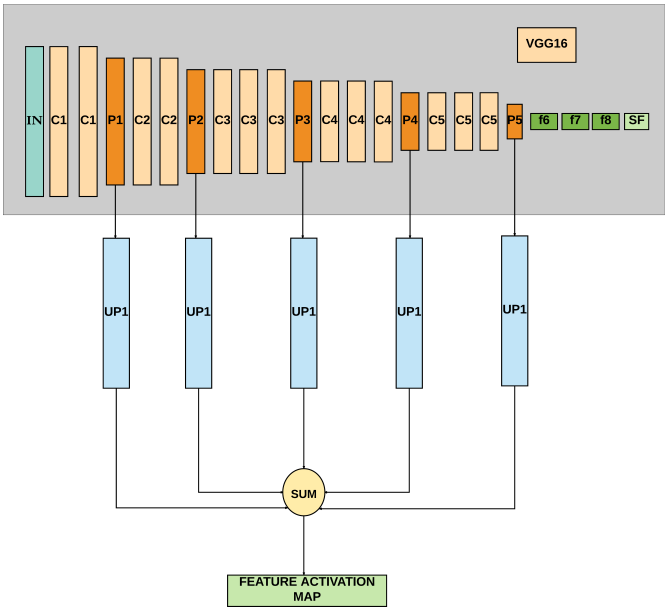


Figure 1: Modified VGG16 network with additional up-sampling layers to create a "feature activation map".

B. Feature activation mapping

Our technique "feature activation mapping" is slightly different from other visualization methods. The technique generates a feature activation map by using both the high-level and low-level feature maps. The standard VGG16 network has five pooling layers so we up-sampled each pooling layer output to the same resolution as the input image and summed them to reconstruct a heat map. The neural network learns different types of features at each layer. Low-level features like edges are learned by top layers of the network and high-level features like objects are learned by the bottom layers of the network. Figure 2 displays the first nine feature maps of every final convolutional and pooling layer [13], [22]. The features of the spider are clearly displayed in the first convolutional layers until the pool3 layer. Thereafter it shows a yellow colored bright spot which represents the center point of the spider. In addition, some of the spider feature maps are almost blank at higher convolutional layers, which may indicate the disappearing of the features of tiny objects. Therefore combining feature maps from all pooling layers is necessary to reconstruct a high resolution heat map. Figure 1 shows the VGG16 network with modifications to generate the heat map.

Feature maps H_k^l from a single layer l are summed to create a layer heat map H^l. The layer heat maps of all layers are upsampled with corresponding factor R^l to the same resolution of input images and then summed together to produce the overall heat map as expressed in Equation 1. L is 5 in our network since we have selected all five pooling layers.

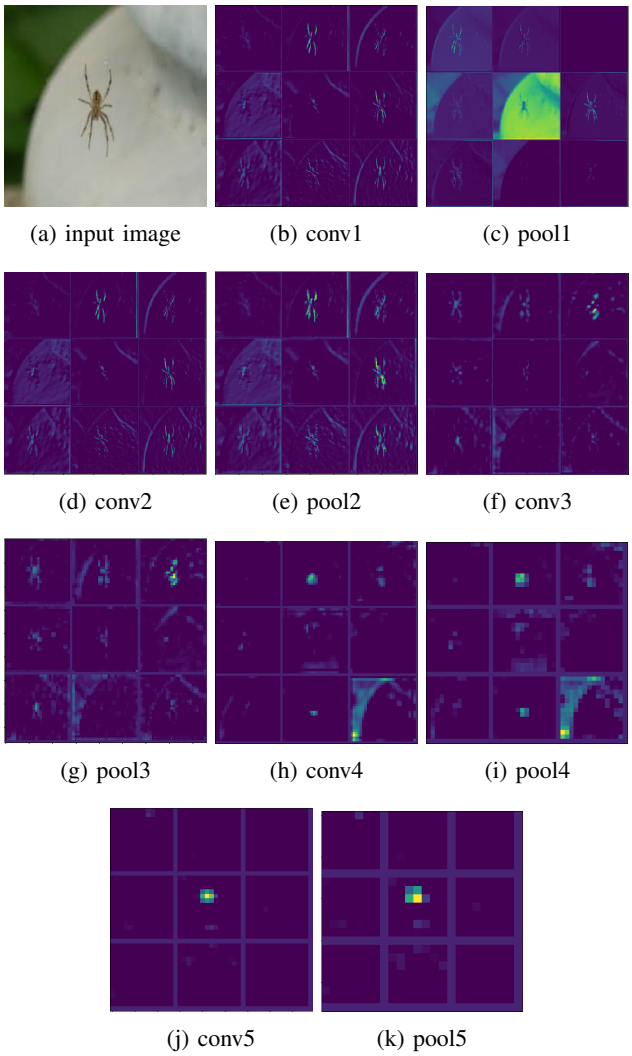
$$H = \sum_{l=1}^{L} H^l * R^l = \sum_{l=1}^{L} \left(\sum_{k=1}^{K} H_k^l \right) * R^l \qquad (1)$$

When an image is classified as a spider image, the heat map is then used to visualize the spider location.

IV. New Spider detection dataset

Our dataset consists of spider images collected from Google and Flicker. Given that one of the objectives was the detection of tiny spiders, we used images where the scale of the spider was 1-10% of the image size. We did not use any spider images from the ImageNet dataset because in those images, the scale of the spider is quite large compared with the background. Before data augmentation, the total number of images was 400, of which 200 are spider images and 200 are non-spider images. Non-spider images contain grapes, broccoli, green vegetables and other backgrounds where spiders often inhabit. Due to the relatively small number of spider images, data augmentation was applied to generate more images. The data augmentation was done by flipping and rotating the images. For the task of small object detection, we used only 27 spider images out of 200 which had a scale of 1-5% of the image size, as well as a similar amount of non-spider images with various backgrounds. Sample images from the small-scale spiders dataset are shown in the left column of figure 5.

We make spider detection dataset available at https://tinyurl.com/ybzaycsf.



(a) input image (b) conv1 (c) pool1

(d) conv2 (e) pool2 (f) conv3

(g) pool3 (h) conv4 (i) pool4

(j) conv5 (k) pool5

Figure 2: First nine feature maps of some VGG16 layers.

V. Experiments

Our initial training was only with spider and non-spider images, irrespective of the scale of spiders. Therefore the network was initially fine-tuned and tested on differently scaled spider images, including 1-5% of the image size. At the testing stage, some spider images were not detected as spiders for two reasons: first, the scale of the spider was very small compared to the image size, and second, the images had complex backgrounds. After cropping the images, the network was able to correctly detect spiders in those images. Figure 3 shows such an image before and after cropping. The size of the spider in the original image is quite small compared to the background (about 2%) and the background is complex. But in the image on the right, the size of the spider is increased relative to the background (about 7%) and the background is also not as complex as that on the left. Therefore we concluded that scale and background complexity are major factors for CNN based small object detection.

As a result, network fine-tuning was then performed with different strategies to identify the most suitable strategy to obtain improved results. We tried fine-tuning various layers, in particular the final fully connected layer, all fully connected layers and all layers including the convolutional layers. After comparing training losses we found that the network performs best when all the layers are fine-tuned because the training loss approaches zero.

Figure 3: The spider is unsuccessfully detected on original image is on the left, but successfully detected from the cropped image on the right.

We also identified an issue that the network had reduced capability to identify spiders with a scale of less than 5% compared to the image size. Therefore we moved on to fine tune our network on spider images with a scale 1-5%, on top of the original weights which were trained on ImageNet.

After fine-tuning all the layers, we used the network to classify small spiders and generated heat maps as a post-processing stage.

For comparison, different networks with different methods of localization and visualization were also fine tuned and tested on the same dataset. Faster R-CNN [7], [15] classifies spiders and locates them with a bounding box. Class activation mapping (CAM) [25] classifies spiders and also generates heat maps to localize them.

VGG16 was trained with spider and non spider images with a scale of 1-5%. Input image size is 224×224 pixels. After data augmentation, the number of spider images was 108 which were taken as positive examples and a similar number of negative examples were used for the training. The training was performed with 10000 iterations, or nearly 470 epochs. The faster R-CNN network and CAM based network were also fine tuned for two classes with above dataset. Those networks were also fine tuned with a similar number of iterations as VGG16. Since faster R-CNN performed well on object detection, we decided to choose faster R-CNN for our comparison.

A. *Classification results*

The training and testing accuracy percentage comparison with faster R-CNN and CAM is shown in Table 1. Testing was based on 120 spider and non-spider images. The network was validated with 20 spider and non-spider images. The classification accuracy of our VGG16 network lies between Faster R-CNN and CAM. Inferring a single image takes 0.25-0.27 sec to get a binary output using VGG16 and 10-15 sec to

Table I: Training and testing accuracy comparison with Faster R-CNN and CAM

Network	Training Accuracy	Testing Accuracy
VGG16 fine tuned on spiders	95.37%	84%
Faster R-CNN fine tuned on spiders	92%	72%
CAM with global pooling	98%	95.32%

visualize a feature activation map. Faster R-CNN takes 0.04-0.6 sec to get a bounding box, while CAM takes 5-7 sec to generate a heat map. At the testing stage, even though average time to get a classification output is faster than faster R-CNN, visualization is slower compared to CAM.

B. Visualization results

Visualization is useful to find the location of the spiders when images are classified as spider images. Figures 4 and 5 shows our feature activation maps of spider and non-spider images respectively. The feature maps of our method in Figure 4 clearly show not only the center point of the spider but also the features and the boundary lines of the spiders. While CAM's heat maps show a single broad peak near the actual location of the spiders, they do not indicate what size and shape of the spider is being detected. According to the Figure 5, most of the time faster R-CNN has detected spiders accurately but sometimes other objects with similar features were also detected as a spiders.

Non spider feature activation maps in Figure 5 show boundary lines of objects in the image (grapes, green leaves, grass etc). Note that CAM's non-spider images are not clear compared to ours for two reasons: first, those images do not show any boundary lines or features and, second, non spider images also have several peaks which could be misunderstood as spiders.

VI. Conclusion

We have shown that a simple network like VGG16 can be fined-tuned to detect small objects, and that feature activation mapping is a useful simple technique to visualize objects detected in an image. This method has the ability to depict the shapes of the objects. We used the object scale of 1-5% of image size and trained our network with relatively few training examples. We compared our network performance with CAM and Faster R-CNN qualitatively and quantitatively. Although the accuracy of our method is inferior to CAM, the feature activation maps for our method show more detail of shape and size of the spiders. The comparison provides useful information for practitioners to select a suitable method for their detection problem. We also make available our spider detection dataset to facilitate future research work.

Acknowledgment

Authors would like to thank Australian Government Research Training Program for funding this research. This research was conducted by the Australian Research Council Center of Excellence for Robotic Vision (CE140100016) http://www.roboticvision.org.

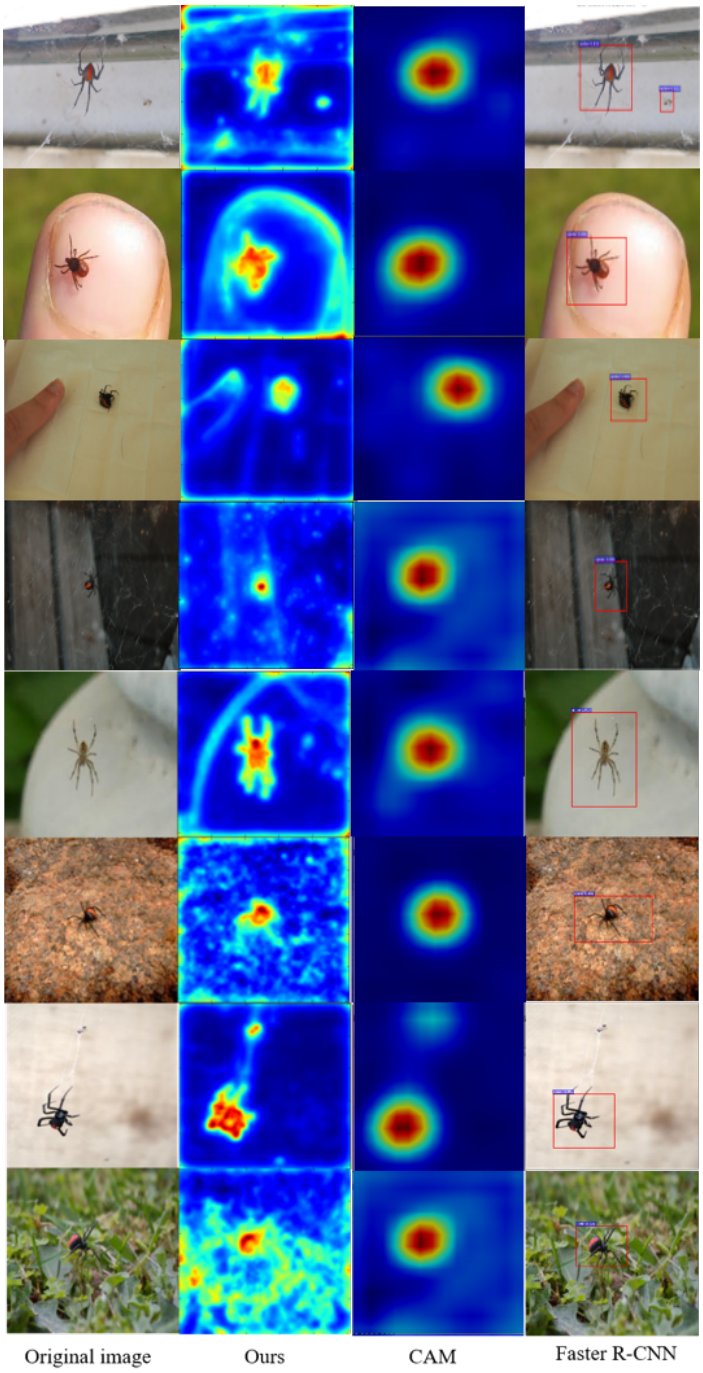


Figure 4: Our positive (spider) feature activation maps in comparison with CAM and Faster R-CNN. Compared with CAM our feature activation maps have more details including shapes and the boundary lines of the spiders.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[2] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. How to explain individual classification decisions. *Journal of Machine Learning Research*,

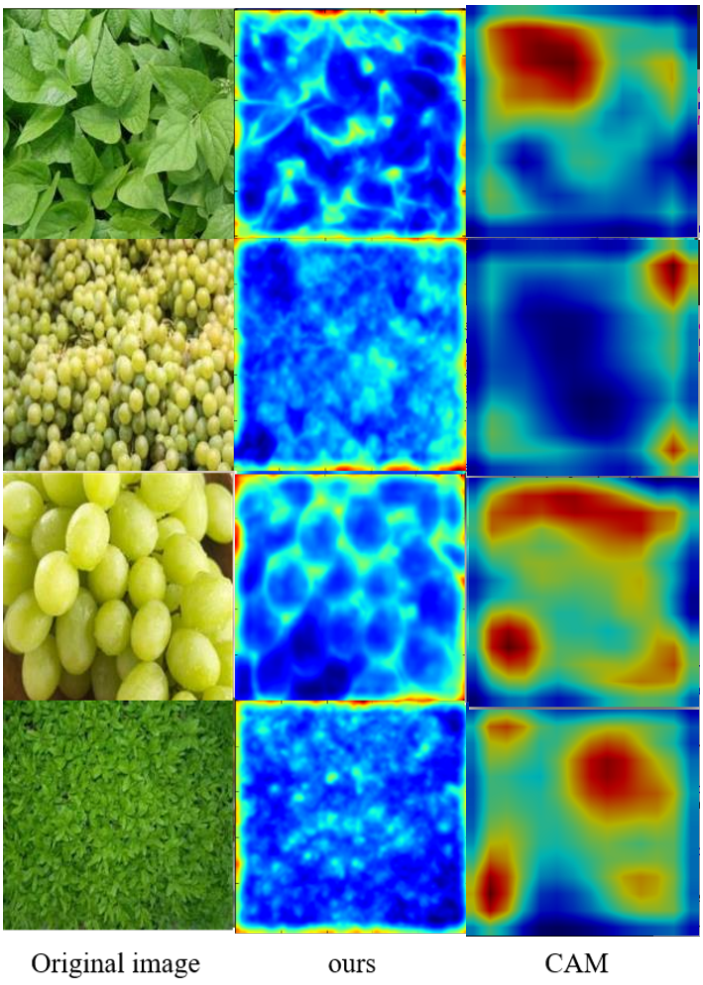


Figure 5: Our negative (non-spider) feature activation maps in comparison with CAM. CAM's non spider heat maps also have peaks similar to spider heat maps and does not have any object boundaries compared to ours

11(Jun):1803–1831, 2010.

[3] Loris Bazzani, Alessandra Bergamo, Dragomir Anguelov, and Lorenzo Torresani. Self-taught object localization with deep networks. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.

[4] Sean Bell, C Lawrence Zitnick, Kavita Bala, and Ross Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2874–2883, 2016.

[5] Chenyi Chen, Ming-Yu Liu, Oncel Tuzel, and Jianxiong Xiao. R-cnn for small object detection. In *Asian Conference on Computer Vision*, pages 214–230. Springer, 2016.

[6] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015.

[7] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[8] Georgia Gkioxari, Ross Girshick, and Jitendra Malik. Contextual action recognition with r* cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1080–1088, 2015.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[11] Jianan Li, Xiaodan Liang, Yunchao Wei, Tingfa Xu, Jiashi Feng, and Shuicheng Yan. Perceptual generative adversarial networks for small object detection. *arXiv preprint arXiv:1706.05274*, 2017.

[12] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[13] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5188–5196, 2015.

[14] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 685–694, 2015.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[16] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 2017.

[17] Wojciech Samek, Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, and Klaus-Robert Müller. Interpreting the predictions of complex ml models by layer-wise relevance propagation. *arXiv preprint arXiv:1611.08191*, 2016.

[18] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2016.

[19] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[20] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[21] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[22] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[23] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[24] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

[25] 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.