Course Project - CS 689

Github Repository

# Visualising Convolutional Neural Networks

Niraj Mahajan
{180050069}

Advisor : Prof Harish Guruprasad Ramaswamy

**Abstract**

In this project, we have implemented and explored the paper on Learning Deep Features for Discriminative Localization [1]. We revisit the global average pooling layer proposed in [2], and explore it's applications for visualisation of Convolutional Neural Networks. The authors propose Class Activation Mappings (CAMs) as a tool for CNN visualisation, and further demonstrate it's effectiveness in Weakly Supervised Object Localisation (WSOL). In this project, we have analysed the CAM matrices on several applications and trained a weakly supervised digit detector for localising numbers present in the wild. We have also contrasted the performance of CAMs with saliency maps proposed by [3]. The code for this project is provided in a public github repository found herein.

## 1 Introduction

Visualisation techniques for CNNs are an important area to understand the "blackbox" our conventional CNNs are. [1] propose a widely used visualisation technique known as Class Activation Mappings (or CAMs). The authors propose a simple but effective technique to identify the object pixels in an image by extracting information from the conv layers output that is normally lost when passed through the FC layers.

The outputs generated by the convolutional blocks are commonly known as the "features" since these contain a "simplified" version of the information present in the images. These features are passed on the the fully connected layers, which destroy the spatial information. So basically the degree of acttivation of any feature can be given by the magnitude of the logit in the conv-output. For the output generated by the convolutional layers, each logit produced can be traced back to a region/field of view in the input image. Thus we can assign importance to the pixels in the input image by considering the logits generated by the conv layers.

This method can help us determine the important pixels in the input images. Such information can be deployed in several applications beyond explainability like object detection and scene understanding. Despite the simplicity of the method, CAMs achieve comparable results with SOTA methods in object detection on popular datasets like ILSVRC as cited by [1].

# 2 Related Work

This project combines two popular areas of modern Deep Learning - Visualisation and Object Detection. This paper [1] was one of the first papers that researched on visualisation. Several upgrades and other methods have been proposed subsequently. [4] introduced grad-CAMs which augmented the traditional CAM method with a gradient based weighting. [5] proposed score-CAMs which, unlike previous CAM based approaches, eliminate the dependence on gradients by obtaining the weight of each activation map through its forward passing score on target class. [3] propose the use of saliency maps by approximating the entire convolutional computation by a linear transformation. [6] propose LayerCams that aim to generate more fine-grained object localization information from the class activation maps to locate the target objects more accurately.

Considerable work has also been done in object detection with the publishing of methods like RCNN [7], Fast-RCNN [8], Faster-RCNN [9] and YOLO [10]. But all these methods require annotated data for object detection which is a costly task. To tackle this problem, several weakly supervised object localisation methods have been proposed, surveyed in [11].

# 3 Method

For our experiments, we have implemented CAM as well as saliency maps as our base methods. In this section we have discussed both these methods as well as the architectures we used.

## 3.1 Class Activation Mappings

[1] propose a pipeline of conv layers followed by a Global Average Pooling layer and a Single Linear Layer to generate classification scores (logits). As shown in Figure 1, each of the channel generated by the conv layers produces a CAM matrix which highlights the locations used by the conv layers in the input image. Using the weights of the linear layer, we take the weighted combination of these CAM matrices to produce a single mapping which is displayed as our final class activation mapping. This mapping is upsampled to match the input image and overlayed for comparison.

## 3.2 Class Saliency Maps - Backprop

*Simonyan et al* [3] propose a more specific method to get the contribution of each pixel to the pre softmax classification score. To uderstand this, let us first assume a CNN without any activation functions, ie, a linear score generator. For any class c, the pre-Softmax score $S_c$ is given by:

$$S_c(I) = w_c^T I + b_c \tag{1}$$

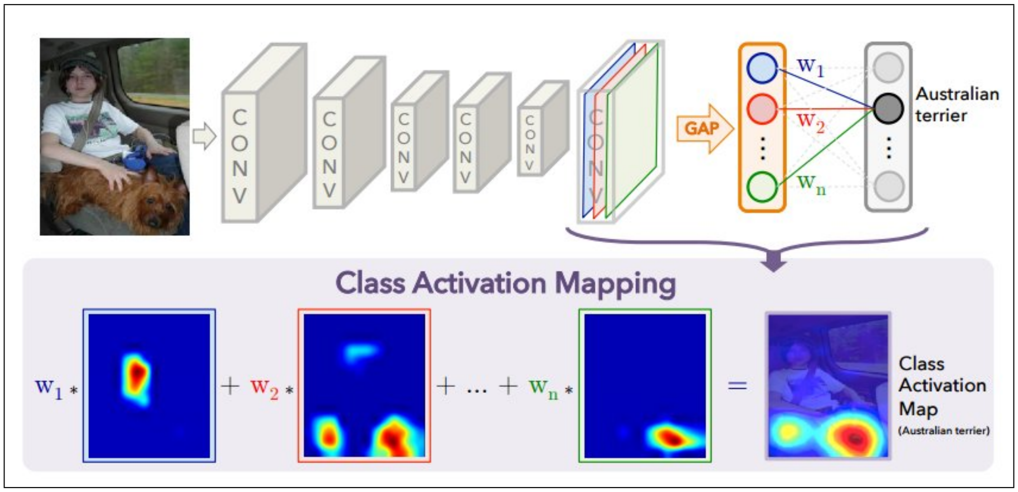

Figure 1: Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.

But since the output generated by any CNN is highly non-linear, the above score can be approximated again with a linear function (by the first order Taylor expansion).

$$S_c(I) \sim w_c^T I + b_c \tag{2}$$

Note that in our experiments, and in almost all popular CNN architectures, the activation function being used is the ReLU [12] operation. For any input image, the modes of the ReLUs (on/off) are known/fixed. The ”on” ReLU nodes simply pass on the input value and so, at the end, there is no approximation in case of ReLUs and we get the exact weights using the above equation.

Here, w is the derivative of the pre-Softmax score of class c, with respect to the input image pixel-values, given by:

$$w = \left.\frac{\delta S_c}{\delta I}\right|_{I_0} \tag{3}$$

This derivative can easily be computed using the backprop operation implemented in the deep learning frameworks.

### 3.3 Architecture used

We have used the vgg16 [13] architecture as the skeleton for all experiments. The input is expected to be 224x224 in size. The conv-5 layer of VGG-16 produce a 512x7x7 output. These logits are reduced to 512x1x1 by a Global Average Pooling Layer. These 512 logits are then passed on to a Linear Layer (512xC), where C is the number of classes. The architecture is shown in Figure 2

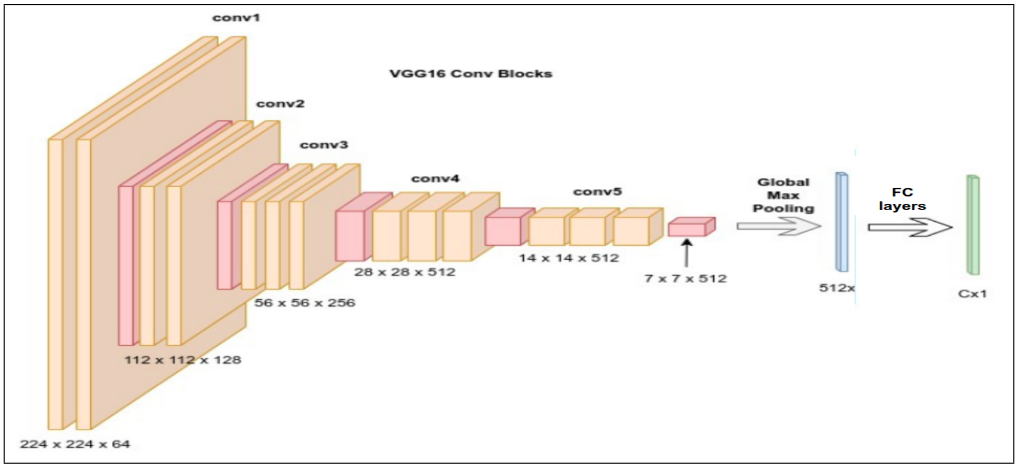

Figure 2: The architecture used in all our experiments. The Fully connected block has been replaced by a sequential global average pooling layer (GAP) followed by a single linear layer to produce the pre-softmax score.

# 4 Experiments and Results

In this section, we summarise the results obtained for Class Activation Mapping for CIFAR10 [14] with VGG16. We also have summarised the WSOL results for detection of numbers in images taken in the wild.

## 4.1 Class Activation Mappings on CIFAR10

We trained a VGG16 model on CIFAR10 with a learning rate of $10^{-3}$ with the Adam Optimizer. This training was done for 25 epochs. For extrapolating the generated CAMs, we used bicubic interpolation. The CAM results for various classes are shown in Figure 3.

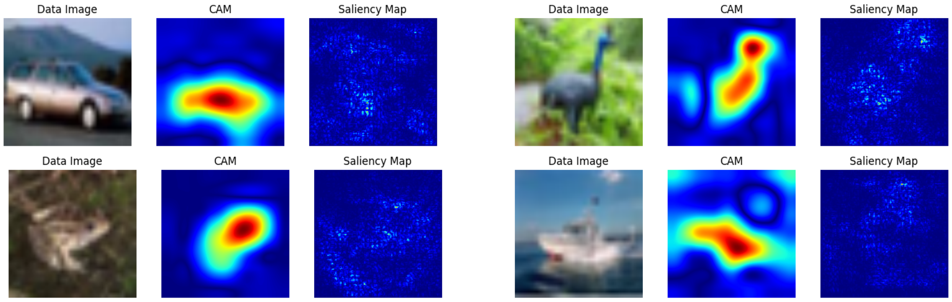

Figure 3: Examples of localization from CAM using VGG16 on CIFAR10

### 4.2 WSOL - Digit Detection

In this section, we have attached the results for Weakly Supervised Object Localisation - specifically Digit recognition in the wild. We have trained a binary classifier to distinguish between images with and without text in the wild. For this purpose, we have used the SVT dataset [15] and the House Price dataset [16] for detecting the presence of numbers in images. Such a classifier will need to focus on instances of text to perform classification. A few results are attached in Figure 4.

We can see that the WSOL results are suboptimal and the CAM detection is not exactly on the text. This is because the dataset is really small ( 300 images) and the model sees to be overfitting on the data. Also there is a stark difference in the dataset colours. The model can simply look at the top region of the images (ie the sky) and can easily distinguish the classes by the colors.

## 5 Future Work

We are planning to train an "eye detector" using this method by training a binary classifier to distinguish glasses vs no-glasses. Both these classes need to look at the information near the eyes and naturally the CAMs will focus on the eyes, leading to WSOL.

## 6 Conclusion

In this project, we surveyed and explored the applications of Class Activations Mappings for visualisation as well as object detection. Although we were not able to get optimal results for WSOL due to scarcity of data, it is evident from the paper [1] that optimal results are easily achieved using CAMs in such applications. We have also shown the superiority of CAM results over saliency maps in terms of visualisation. CAMs are a really effective way to help explain the CNN blackbox and to answer the very basic question "What exactly is the CNN looking for in an image".

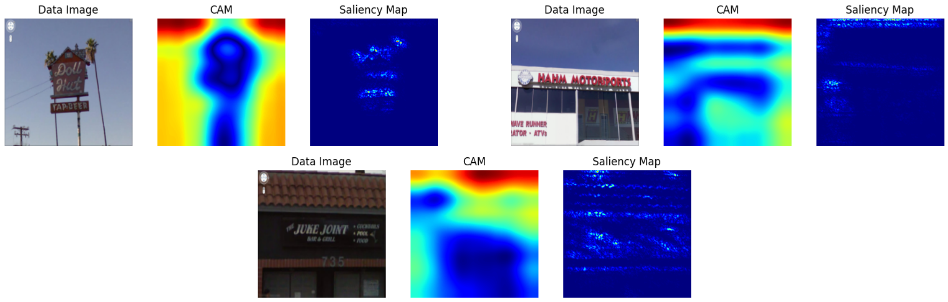

Figure 4: Examples of localization from CAM using VGG16 on CIFAR10

## References

[1] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. 12 2016.

[2] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2014.

[3] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *preprint*, 12 2013.

[4] 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 *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[5] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 24–25, 2020.

[6] Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888, 2021.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 11 2013.

[8] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. pages 779–788, 06 2016.

[11] Dingwen Zhang, Junwei Han, Gong Cheng, and Ming-Hsuan Yang. Weakly supervised object localization and detection: A survey. *CoRR*, abs/2104.07918, 2021.

[12] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

[14] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[15] Kai Wang. The street view text dataset (svt). 2016.

[16] Eman Ahmed and Mohamed Moustafa. House price estimation from visual and textual features. 11 2016.