EVALUATING EXPLAINABLE CNN TECHNIQUES : CAM,GRADCAM,GRADCAM++

PROJECT REPORT

Nikhil Bisen*
Department of Electronics System Engineering
Indian Institute of Science
Bengaluru, Karnataka
nikhilbisen@iisc.ac.in

Monalisa Bakshi
Department of Electrical Communication Engineering
Indian Institute of Science
Bengaluru, Karnataka
mmonalisab@iisc.ac.in

ABSTRACT

In CAM, the global average pooling layer in convolutional neural networks (CNNs) enables remarkable localization ability even with image-level labels. Despite being initially proposed as a regularization technique, the authors find that it builds a generic deep representation that can be used for various tasks. The study shows that this technique achieves impressive results in object localization on ILSVRC 2014 and can also identify discriminative image regions in other tasks, even without specific training for those tasks.Grad-CAM produces visual explanations for decisions made by convolutional neural network models. It uses the gradients of target concepts flowing into the final convolutional layer to produce a coarse localization map highlighting important image regions for predicting the concept. The authors combine Grad-CAM with existing fine-grained visualizations to create Guided Grad-CAM, which is applicable to a wide range of CNN model families without architectural changes or retraining. The proposed method is demonstrated to be more transparent, explainable, and faithful to the underlying model than previous approaches, and it helps to identify important neurons for providing textual explanations for model decisions. The paper also presents a new method called Grad-CAM++ that provides better visual explanations than state-of-the-art techniques, including explaining multiple object instances in a single image, and is applicable to various tasks such as classification, image caption generation, and 3D action recognition.

1 Introduction

Deep learning models are reckoned as "black-box" unlike classical machine learning. There is a need for Explainable Techniques for Convolutional Neural Networks which can tell us what our neural network is looking at while giving the output .Explainability become very crucial in AI for HealthCare domains , where doctors would be able to diagnose the disease more accurately if they have the model interpretability available . In 2016 we had the first explainability methods coming out - Class Activation Maps for Networks with Global Average Pooling and then soon after GradCAM which solve some issues of CAM . In 2017 , the modified version of gradCAM , gradCAM++ comes out and after that there have been series of development in Explainable CNN with different CAM's model coming out. We are going to see the working of CAM , gradCAM and gradCAM++ today , understand the experimental setup that we did to evaluate this models .

2 CAM

CAM is a procedure for generating class activation maps (CAM) using global average pooling (GAP) in CNNs. These maps indicate the image regions used by the CNN to identify a particular category. The procedure involves performing global average pooling on the feature maps of the last convolutional layer, and using those as features for a

*Use footnote for providing further information about the author (webpage, alternative address)—*not* for acknowledging funding agencies.

fully-connected layer that produces the desired output. The importance of the image regions is identified by projecting back the weights of the output layer onto the convolutional feature maps, resulting in the class activation maps. By upsampling these maps to the size of the input image, the relevant image regions can be identified. The text also highlights the difference between GAP and global max pooling (GMP), and shows that GAP outperforms GMP for localization.

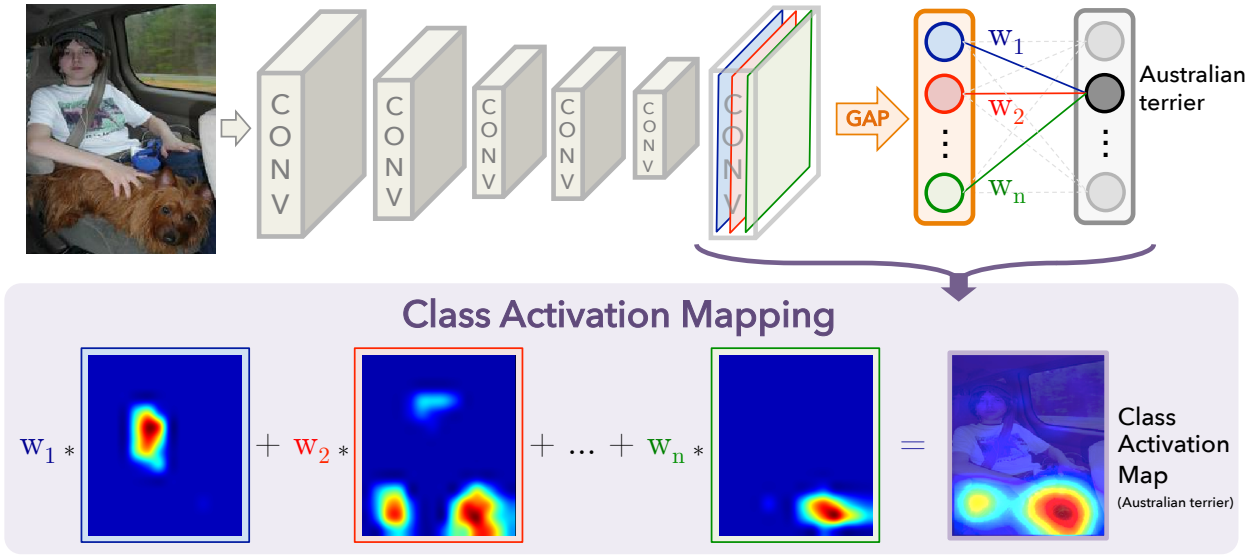

Figure 1: Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.

For a given image, let $f_k(x, y)$ represent the activation of unit k in the last convolutional layer at spatial location (x, y). Then, for unit k, the result of performing global average pooling, F^k is $\sum_{x,y} f_k(x, y)$. Thus, for a given class c, the input to the softmax, S_c, is $\sum_k w_k^c F_k$ where w_k^c is the weight corresponding to class c for unit k. Essentially, w_k^c indicates the *importance* of F_k for class c. Finally the output of the softmax for class c, P_c is given by $\frac{\exp(S_c)}{\sum_c \exp(S_c)}$. Here we ignore the bias term: we explicitly set the input bias of the softmax to 0 as it has little to no impact on the classification performance.
By plugging $F_k = \sum_{x,y} f_k(x, y)$ into the class score, S_c, we obtain

$$\begin{aligned} S_c &= \sum_k w_k^c \sum_{x,y} f_k(x, y) \\ &= \sum_{x,y} \sum_k w_k^c f_k(x, y). \end{aligned} \tag{1}$$

We define M_c as the class activation map for class c, where each spatial element is given by

$$M_c(x, y) = \sum_k w_k^c f_k(x, y). \tag{2}$$

Thus, $S_c = \sum_{x,y} M_c(x, y)$, and hence $M_c(x, y)$ directly indicates the importance of the activation at spatial grid (x, y) leading to the classification of an image to class c.

2.1 Limitation of CAM

1) Using class activation maps involves the overhead of learning N linear models to learn the weights w1,w2,w3....wn for each of the N classes.

2) The introduction of the global average pooling (GAP) layer after the last convolutional layer imposes a restriction on the ConvNet architecture.

3) CAM can only be used in Classification Task and can not be used in more complex tasks like Visual question answering(VQA) and Image Captioning .

3 Grad-CAM

we utilized a Convolutional Neural Network (CNN) to classify images. Previous research has shown that deeper layers in a CNN capture higher-level visual features and retain spatial information. The last convolutional layers are particularly useful as they provide a balance between high-level semantics and detailed spatial information. To explain the decisions made by the output layer of our CNN, we used the technique of Grad-CAM, which assigns importance values to each neuron by analyzing the gradient information flowing into the last convolutional layer. While this technique can be applied to any layer of a deep network, we chose to focus on explaining output layer decisions specifically. Overall, the use of Grad-CAM allowed us to better understand the decisions made by our CNN and provide insight into the high-level semantics and detailed spatial information used in image classification.

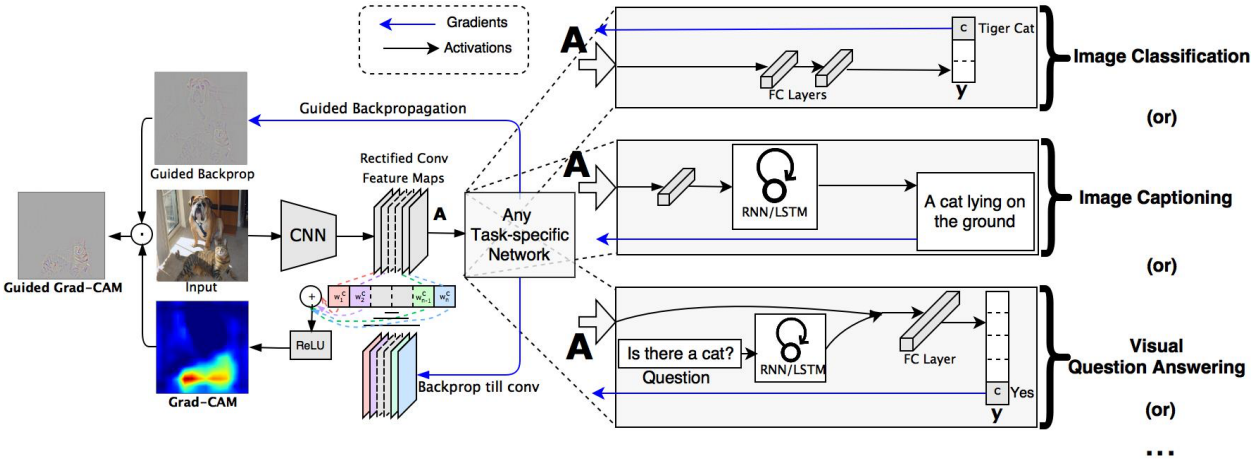

Figure 2: Grad-CAM overview: Given an image and a class of interest (, 'tiger cat' or any other type of differentiable output) as input, we forward propagate the image through the CNN part of the model and then through task-specific computations to obtain a raw score for the category. The gradients are set to zero for all classes except the desired class (tiger cat), which is set to 1. This signal is then backpropagated to the rectified convolutional feature maps of interest, which we combine to compute the coarse Grad-CAM localization (blue heatmap) which represents where the model has to look to make the particular decision. Finally, we pointwise multiply the heatmap with guided backpropagation to get Grad-CAM visualizations which are both high-resolution and concept-specific.

3.1 Grad-CAM generalizes CAM

In this section, we discuss the connections between Grad- CAM and Class Activation Mapping and formally prove that Grad-CAM generalizes CAM for a wide variety of CNN-based architectures. Recall that CAM produces a localization map for an image classification CNN with a specific kind of architecture where global average pooled convolutional feature maps are fed directly into softmax. Specifically, let the penultimate layer produce K feature maps, $A^k \in \mathbb{R}^{u \times v}$, with each element indexed by i, j. So A^k_{ij} refers to the activation at location (i, j) of the feature map A^k. These feature maps are then spatially pooled using Global Average Pooling (GAP) and linearly transformed to produce a score Y^c for each class c,

$$Y^c = \sum_k \underbrace{w_k^c}_{\text{class feature weights}} \overbrace{\frac{1}{Z} \sum_i \sum_j}^{\text{global average pooling}} \underbrace{A^k_{ij}}_{\text{feature map}} \tag{3}$$

Let us define F^k to be the global average pooled output,

$$F^k = \frac{1}{Z} \sum_i \sum_j A_{ij}^k \tag{4}$$

CAM computes the final scores by,

$$Y^c = \sum_k w_k^c \cdot F^k \tag{5}$$

where w_k^c is the weight connecting the k^{th} feature map with the c^{th} class. Taking the gradient of the score for class c (Y^c) with respect to the feature map F^k we get,

$$\frac{\partial Y^c}{\partial F^k} = \frac{\frac{\partial Y^c}{\partial A_{ij}^k}}{\frac{\partial F^k}{\partial A_{ij}^k}} \tag{6}$$

Taking partial derivative of (4) A_{ij}^k, we can see that $\frac{F^k}{A_{ij}^k} = \frac{1}{Z}$. Substituting this in (6), we get,

$$\frac{\partial Y^c}{\partial F^k} = \frac{\partial Y^c}{\partial A_{ij}^k} \cdot Z \tag{7}$$

From (5) we get that, $\frac{\partial Y^c}{\partial F^k} = w_k^c$. Hence,

$$w_k^c = Z \cdot \frac{\partial Y^c}{\partial A_{ij}^k} \tag{8}$$

Summing both sides of (8) over all pixels (i, j),

$$\sum_i \sum_j w_k^c = \sum_i \sum_j Z \cdot \frac{Y^c}{A_{ij}^k} \tag{9}$$

Since Z and w_k^c do not depend on (i, j), rewriting this as

$$Z w_k^c = Z \sum_i \sum_j \frac{Y^c}{A_{ij}^k} \tag{10}$$

Note that Z is the number of pixels in the feature map (or $Z = \sum_i \sum_j \mathbf{1}$). Thus, we can re-order terms and see that

$$w_k^c = \sum_i \sum_j \frac{Y^c}{A_{ij}^k} \tag{11}$$

Up to a proportionality constant $(1/Z)$ that gets normalized-out during visualization, the expression for w_k^c is identical to α_k^c used by Grad-CAM. Thus, Grad-CAM is a strict generalization of CAM. This generalization allows us to generate visual explanations from CNN-based models that cascade convolutional layers with much more complex interactions, such as those for image captioning and VQA .

3.2 Limitation of Grad-CAM

1) One limitation of Grad-CAM is that when there are multiple occurrences of the target class within a single image, the spatial footprint of each occurrence can be substantially lower. This means that the technique may not be able to accurately localize all instances of the target class in the image. This can be a problem, especially when the target class is relatively small or when it is partially occluded by other objects in the image. In such cases, Grad-CAM may provide unsatisfactory localization performance, leading to incorrect or incomplete interpretations of the image.

2) Another limitation of Grad-CAM is its sensitivity to the choice of CNN architecture and layer. Different CNN architectures may have different levels of abstraction and may represent visual features in different ways. Similarly, different layers in the same CNN architecture may capture different levels of detail and may have different levels of spatial resolution. These differences can affect the interpretability and generalizability of Grad-CAM results.

4 Grad-CAM++

Consider a saliency map L^c , and a binary object classification task, with output 0 if object is absent or 1 if present. A^k represents the visualization of the k^{th} feature map. According to previous work , each A^k is triggered by an abstract visual pattern. In this example, $A^k_{ij} = 1$ if a visual pattern is detected else 0. (the dark regions correspond to $A^k_{ij} = 1$.) The derivative $\frac{\partial y^c}{\partial A^k_{ij}}$ is expected to be high for feature map pixels that contribute to the presence of the object. Without loss of generality, let us assume the derivative map to be:

$$\begin{aligned} \frac{\partial y^c}{\partial A^k_{ij}} &= 1 \qquad if \quad A^k_{ij} = 1 \\ &= 0 \qquad if \quad A^k_{ij} = 0 \end{aligned} \tag{12}$$

Plugging in values from Eqn 12 into Eqn 11, we obtain the following feature map weights in the case of Grad-CAM for the given input image I, $w^c_1 = \frac{15}{80}$, $w^c_2 = \frac{4}{80}$ and $w^c_3 = \frac{2}{80}$ for the three feature maps.

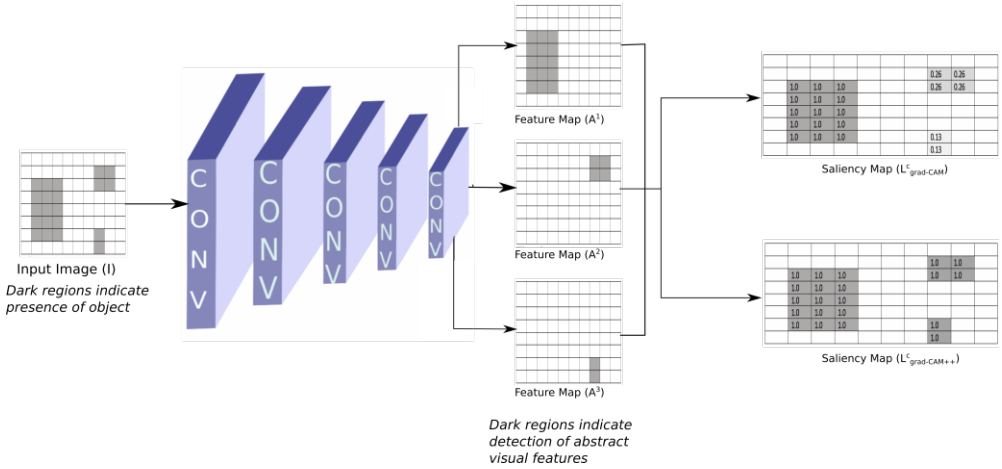

Figure 3: A hypothetical example elucidating the intuition behind grad-CAM++. The CNN task here is binary object classification. Clearly taking a weighted combination of gradients $L^c_{grad-CAM++}$ provides better salient features (all the spatially relevant regions of the input image are equally highlighted) than its unweighted counterpart $L^c_{grad-CAM}$ (some parts of the object are paled out in the saliency map). The values in the pixels of each saliency map indicates the intensity at that point.

Comparing with the input image I, it is evident that the spatial footprint of an object in an image is important for Grad-CAM's visualizations to be strong. Hence, if there were multiple occurrences of an object with slightly different orientations or views (or parts of an object that excite different feature maps), different feature maps may be activated with differing spatial footprints, and the feature maps with lesser footprints fade away in the final saliency map.

This problem can be fixed by taking a weighted average of the pixel-wise gradients. In particular, we reformulate Eqn 11 by explicitly coding the structure of the weights w^c_k as:

$$w^c_k = \sum_i \sum_j \alpha^{kc}_{ij} . relu(\frac{\partial Y^c}{\partial A^k_{ij}}) \tag{13}$$

where *relu* is the Rectified Linear Unit activation function. Here the α^{kc}_{ij}'s are weighting co-efficients for the pixel-wise gradients for class c and convolutional feature map A^k. In the above example, by taking

$$\begin{aligned} \alpha^{kc}_{ij} &= \frac{1}{\sum_{l,m} \frac{\partial y^c}{\partial A^k_{lm}}} \qquad if \quad \frac{\partial y^c}{\partial A^k_{ij}} = 1 \\ &= 0 \qquad \text{otherwise} \end{aligned} \tag{14}$$

presence of objects in all feature maps are highlighted with equal importance.

The idea behind considering only the positive gradients in Eqn 13 is similar to works such as Deconvolution **?** and Guided Backpropogation . w^c_k captures the importance of a particular activation map A^k, and we prefer positive gradients to indicate visual features that increase the output neuron's activation, rather than suppress the output neuron's activation.

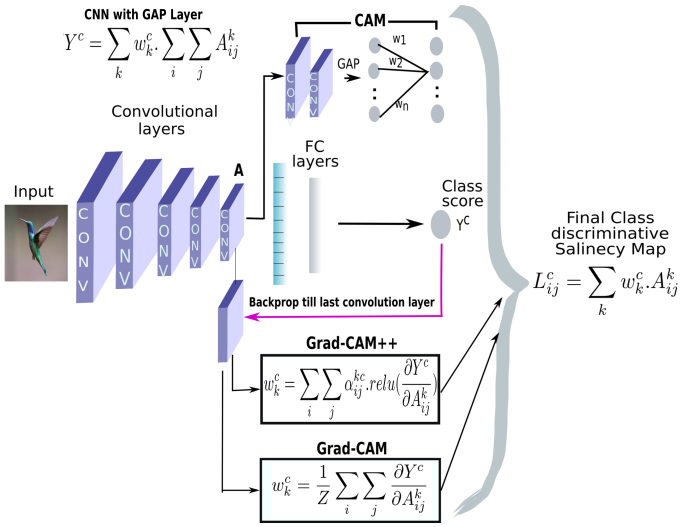

Figure 4: An overview of all the three methods – CAM, Grad-CAM, Grad-CAM++ – with their respective computation expressions.

4.1 Methodology

We derive a method for obtaining the gradient weights α_{ij}^{kc} for a particular class c and activation map k. Let Y^c be the score of a particular class c.

$$Y^c = \sum_k \{ \sum_a \sum_b \alpha_{ab}^{kc} . relu(\frac{\partial Y^c}{\partial A_{ab}^k}) \} [\sum_i \sum_j A_{ij}^k] \tag{15}$$

Here, (i, j) and (a, b) are iterators over the same activation map A^k and are used to avoid confusion. Without loss of generality, we drop the $relu$ in our derivation as it only functions as a threshold for allowing the gradients to flow back. Taking partial derivative w.r.t. A_{ij}^k on both sides:

$$\frac{\partial Y^c}{\partial A_{ij}^k} = \sum_a \sum_b \alpha_{ab}^{kc} . \frac{\partial Y^c}{\partial A_{ab}^k} + \sum_a \sum_b A_{ab}^k \{ \alpha_{ij}^{kc} . \frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} \} \tag{16}$$

Taking a further partial derivative w.r.t. A_{ij}^k:

$$\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} = 2 . \alpha_{ij}^{kc} . \frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \{ \alpha_{ij}^{kc} . \frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3} \} \tag{17}$$

Rearranging terms, we get:

$$\alpha_{ij}^{kc} = \frac{\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2}}{2 \frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \{ \frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3} \}} \tag{18}$$

Substituting Eqn 18 in Eqn 19, we get the following Grad-CAM++ weights:

$$w_k^c = \sum_i \sum_j [\frac{\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2}}{2 \frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \{ \frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3} \}}] . relu(\frac{\partial Y^c}{\partial A_{ij}^k}) \tag{19}$$

Evidently, comparing with Eq 11, if $\forall i, j, \quad \alpha_{ij}^{kc} = \frac{1}{Z}$, Grad-CAM++ reduces to the formulation for Grad-CAM. Thus, Grad-CAM++, as its name suggests, can be (loosely) considered a generalized formulation of Grad-CAM.

In principle, the class score Y^c can be any prediction; the only constraint being that Y^c must be a smooth function. For this reason, unlike Grad-CAM (which takes the penultimate layer representation as their class score Y^c), we pass the penultimate layer scores through an exponential function, as the exponential function is infinitely differentiable.

A bird's eye view of all the three methods – CAM, Grad-CAM, and Grad-CAM++ – is presented in Fig. 9.

What makes a good visual explanation? Consider image classification, a 'good' visual explanation from the model for justifying any target category should be (a) class-discriminative (i.e. localize the category in the image) and (b) high-resolution (i.e. capture fine-grained detail).

In contrast, localization approaches like CAM or our proposed method Gradient-weighted Class Activation Mapping (Grad-CAM),are highly class-discriminative.

In order to combine the best of both worlds, we show that it is possible to fuse existing pixel-space gradient visualizations with Grad-CAM to create Guided Grad-CAM visualizations that are both high-resolution and class-discriminative. As a result, important regions of the image which correspond to any decision of interest are visualized in high-resolution detail even if the image contains evidence for multiple possible concepts.

5 Experimental Setup

We have multiplied the image (before image-net normalization) by the explanation. Only regions that score high will still be visible. Then we have run the new modified "dark" image through the model, and check the new category scores.

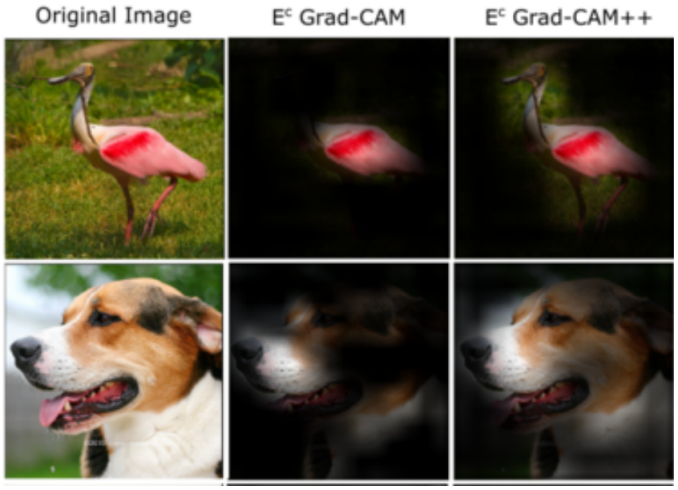

Figure 5:

The metrics are:

1) Drop in Confidence(Smaller value is better): What's the percentage drop of the condience ? (or 0 if the confidence increased). The confidence is assumed to drop a bit since we're removing details.

2) Increase in confidence(Larger value is better): In how many of the cases did the confidence increase. You might ask: why do we need two complementory metrics, why not just measure the average change in confidence. I'm not sure, I suspect that would be better.

This is a way of measuring the "fidelity" or "faithfulness" of the explanation. We want a good explanation to reflect the actual regions that the model is using.

Specifications :

Model : Resnet50 Model Pretrained on ImageNet DataSet which consists of 1000 different Classes . Number of Features at the end of last Convolution layers are 2048 . Performed the Experiments on Class No. 295 of Image Net (American black bear, black bear, Ursus americanus, Euarctos americanus) .

5.1 METHOD 1 :

Evaluating the explanations by using them to pertubate the image and predicting again to measure the change in confidence value .
You multiply the original image by the activation map(point wise product). Only regions that score high will still be visible as show below :

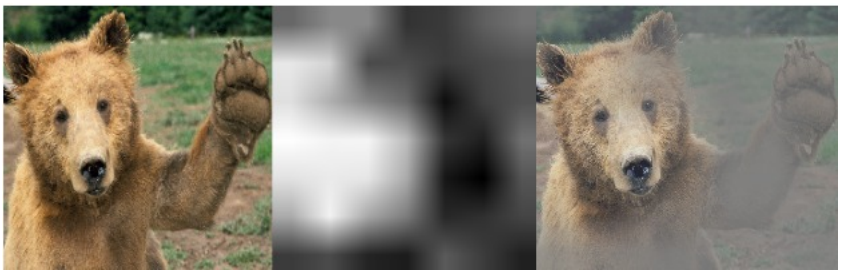

Figure 6:

Then you run the new modified "dark" image through the model, and check the new category scores.
Measure the Increase in confidence(Larger value is better) because the CAM reduced noise from other parts of the image and retains the information that triggers the category output.

5.2 METHOD 2 :

Completely remove the highest scoring 25%, and see the how much model confidence drops now . Now this time the more negative is the value the better will be our CAM method , because we are removing the important portion from the image(according to CAM model) and then calculating the model accuracy , so if the CAM has really predicted that what is important so confidence value must drop significantly . This works better than method 1 .

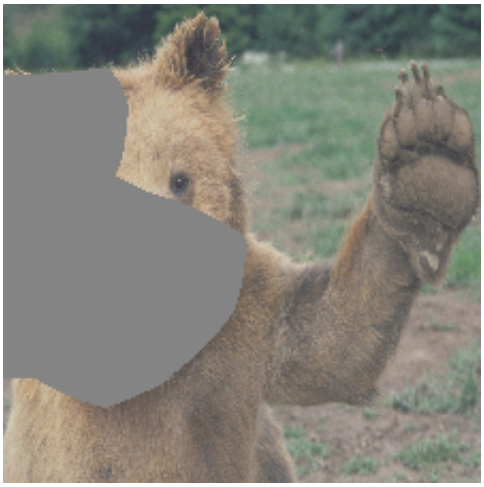

Figure 7:

6 Results

6.1 For Single Object Case :

First Lets Consider the case where my image contains only single bear and compare the results .

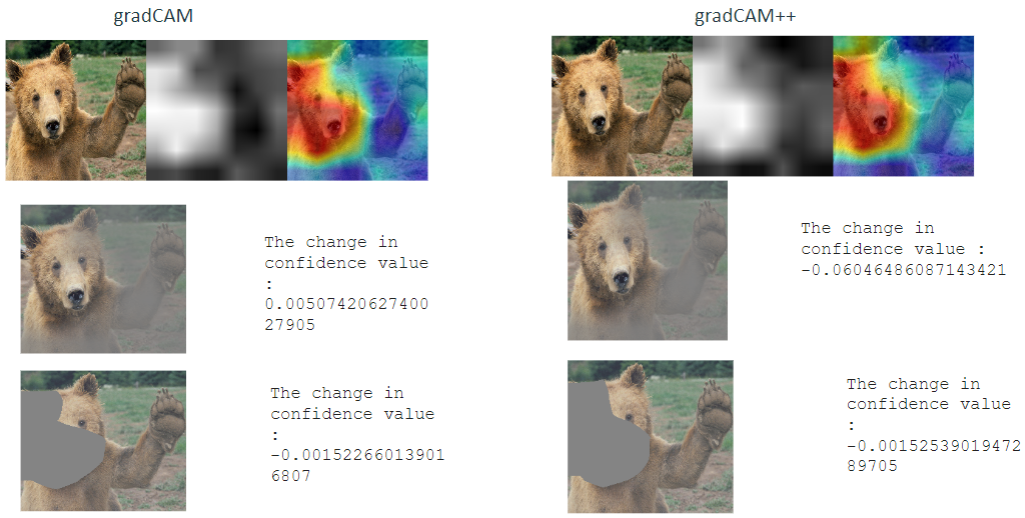

Figure 8: Experimental Results for Single Object Case

6.2 For Multiple Object Case :

Now Let's Consider the Image containing multiple bears, now the gradCAM ++ will outperform the gradCAM by a significant margin.

Figure 9: Experimental Results for Multiple Object Case

7 Conclusion

Different CAM Models performed differently on some images. GradCAM++ outperforms the CAM and gradCAM models in case where there are multiple objects present in the same image. Although there was no significant difference between gradCAM++ and gradCAM in the case of images having a single object. Explainability in CNN's widely used for many applications and hence need the method to evaluate them too . Method 1 was not performing well, in some cases the classification score decreased, this may be due to the fact that the CAM explanation wasn't telling us the whole story in the first place, and there are other parts that were important as well that is was missing.

References

[1]Zhou, Bolei, et al. "Learning deep features for discriminative localization." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[2]Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE international conference on computer vision. 2017.

[3]Chattopadhyay, Aditya, et al. "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks." 2018 IEEE winter conference on applications of computer vision (WACV). IEEE, 2018.

[4]Kakogeorgiou, Ioannis, and Konstantinos Karantzalos. "Evaluating explainable artificial intelligence methods for multi-label deep learning classification tasks in remote sensing." International Journal of Applied Earth Observation and Geoinformation 103 (2021): 102520.

[5]Tomsett, Richard, et al. "Sanity checks for saliency metrics." Proceedings of the AAAI conference on artificial intelligence. Vol. 34. No. 04. 2020.

[6]Hooker, Sara, et al. "A benchmark for interpretability methods in deep neural networks." Advances in neural information processing systems 32 (2019).