

POLY-CAM: HIGH RESOLUTION CLASS ACTIVATION MAP FOR CONVOLUTIONAL NEURAL NETWORKS

Anonymous authors

Paper under double-blind review

ABSTRACT

The need for Explainable AI is increasing with the development of deep learning. The saliency maps derived from convolutional neural networks generally fail in localizing with accuracy the image features justifying the network prediction. This is because those maps are either low-resolution as for CAM (Zhou et al., 2016), or smooth as for perturbation-based methods (Zeiler & Fergus, 2014), or do correspond to a large number of widespread peaky spots as for gradient-based approaches (Sundararajan et al., 2017; Smilkov et al., 2017). In contrast, our work proposes to combine the information from earlier network layers with the one from later layers to produce a high resolution Class Activation Map that is competitive with the previous art in term of insertion-deletion faithfulness metrics, while outperforming it in term of precision of class-specific features localization.

1 INTRODUCTION

We currently face unprecedented advances in the domain of artificial intelligence, primarily driven by the development of deep neural networks (DNNs). However, in contrast to techniques based on handcrafted features, DNNs often lack transparency and explainability (Adadi & Berrada, 2018). The need to assess a posteriori the behavior of a model has led to the development of explainable artificial intelligence (XAI) methods, ranging from more transparent models to post-hoc methods (explanation by example of black-box methods), see Samek et al. (2021) for a review.

Focusing on convolutional neural networks (CNN), saliency maps visualization has been adopted as a convenient approach to identify the image parts justifying the network prediction.

Those saliency maps are helpful to check that the predictions of a model are grounded on relevant information. It is indeed known that training convergence alone does not exclude undesired DNN predictions (Lapuschkin et al., 2019), typically because the model has learnt inputs/outputs correlations that do not correspond to the desired meaningful causal relationship. This case is illustrated in Appendix D.1, where a model trained to detect fractures in bone X-rays actually appear to rely on the plaster cast to make its decision rather than on a potential break in the bone.

Alternatively, when sufficiently accurate, the localization of salient features could convince a user that a model works properly, i.e. uses relevant cues, or could even help in identifying the parts of a signal that are relevant to solve a problem, e.g. help a medical doctor in identifying the X-ray visual cues that help to anticipate the evolution of a treatment (Appendix D.2).

Various techniques are available to define class-specific saliency, including perturbations based analysis (Zeiler & Fergus, 2014), gradient based techniques such as Integrated Gradients (Sundararajan et al., 2017) or SmoothGrad (Smilkov et al., 2017), and Class Activation Mapping techniques such as Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), or Score-CAM (Wang et al., 2020b). Class activation maps (Zhou et al., 2016) methods are limited in resolution, while gradient based techniques are generally subject to noise and thus produce saliency maps composed of a large number of widespread peaky spots. **In an attempt to get the best out of both strategies, recent solutions such as Zoom-CAM (Shi et al., 2021) and Layer-CAM (Jiang et al., 2021) have proposed to combine the gradients in earlier layers with activations to produce high resolution maps. As shown in our experiments, those maps however inherit some noise from the gradients.**

Our work introduces a new method to generate high resolution class activation maps, without relying on gradient backpropagation and thus limiting the noisy appearance of the map. In short, this is

obtained by multiplexing the high-resolution activation maps available in the early layers of the network with upsampled versions of the class-specific activation maps computed in the last layers of the network. It achieves state of the art performances on faithfulness metrics, and largely improves the localization accuracy of features that explain the network prediction.

2 RELATED WORK

Various strategies allow to visualize the image features contributing to the class prediction of a CNN.

Perturbation-based methods use multiple perturbations of the input image, and monitor the changes they induce at the output of the model, to build a saliency map. Those methods range from occlusion techniques, which recursively occlude patches of the input (Zeiler & Fergus, 2014), to more elaborated perturbations such as Randomized Input Sampling for Explanation of Black-box Models (RISE) (Petsiuk et al., 2018), which generate many random masks normalized between 0 and 1, to be weighted by the class-specific softmax output obtained when the input image is multiplied by the mask. These methods result in a smooth mask and are computationally intensive, especially when the resolution of the saliency map increases.

Gradient based methods use the back-propagated gradient of the neural network to identify regions in the input image that largely impact the prediction. Those methods however suffer from gradient shattering (Bahdanau et al., 2014), which results in a noisy saliency map. To mitigate this problem, a variety of approaches have been implemented to smooth out the gradient signal. They include Integrated Gradient (Sundararajan et al., 2017), which integrates the gradient for multiple interpolations between a baseline and the input image, or SmoothGrad (Smilkov et al., 2017), which averages gradient for multiple perturbed variations of the input image.

Class Activation Map (Zhou et al., 2016) was initially designed as a linear combination of activations from the last convolutional layer, weighted by the parameters of a fully connected classifier, taking as input the global average pooling of each channel in this last convolutional layer. Multiple methods were introduced based on alternative definitions of the weights. Grad-CAM (Selvaraju et al., 2017) and Grad-CAM++ (Chattopadhyay et al., 2018) define the weights based on gradients backpropagated upto the last convolutional layer. Score-CAM defines the weight of a channel based on the softmax output associated to the target class when probing the model with a version of the image masked by the activation channel. In a sense, it mixes CAM methods with perturbations methods (Wang et al., 2020b). Overall, Class Activation Map methods are less noisy than gradient-based methods but are coarser, with a resolution limited to the resolution of the last convolution layer.

To produce CAM at higher-resolution, Tagaris et al. (2019) propose to **train an expansion network**, but this has the drawback to require a specific training for each model to analyze (Ronneberger et al., 2015). At the time of writing, this expansion network is only defined for DenseNet (Huang et al., 2017) and trained on a subset of animal labels of ImageNet (Russakovsky et al., 2015), available at <https://github.com/djib2011/high-res-mapping>. **Very recently, Layer-CAM (Jiang et al., 2021) and Zoom-CAM (Shi et al., 2021) have proposed to generate high-resolution CAM by combining back-propagated gradients with activations from multiple layers, using element-wise multiplications. The two methods improve the resolution of the maps but also inherit some noise from the gradients.**

Our paper proposes an original method to compute high-resolution activation maps without using gradients, nor requiring to train a specialized network. Our primary contribution is a process to leverage the activation from multiple layers to increase the resolution of the Class Activation Map up to the one of the input image. Our second contribution is a new way to compute the weights of the activation maps, It introduces a dual strategy compared to the approach introduced by Score-CAM (Wang et al., 2020b), and proposes to get the best out of both strategies by merging them.

3 PROPOSED POLY-CAM APPROACH

This section presents the core contribution of our work. Section 3.1 introduces the notations and variables required in the rest of the text, while Section 3.2 reviews the formal definition of the conventional Class Activation Map method, which serves as a baseline to our work. Section 3.3 then introduces our Poly-CAM approach, which proposes to generate a high resolution class activation map by recursively multiplexing the high-resolution activation maps available in the early layers of

the network with upsampled versions of the class-specific activation maps computed in the last layers of the network. Eventually, Section 3.4 introduces three different methods to associate a weight to each layer activation channel. All methods measure how the output of the network is affected when masking/unveiling the input based on the channel activation.

3.1 NOTATIONS

Let $f_{\Theta}(X)$ denote the prediction of a CNN with parameters Θ when the image X is provided as input. In the following, for conciseness and because we are interested in analyzing a trained network (parameters Θ are fixed), we omit Θ , and just use $f(X)$ to refer to the CNN prediction associated to X . In the following, $f(X)$ is a vector, defined by the output of a softmax. $f_c(X)$ denotes the component of $f(X)$ corresponding to the class c .

\mathbf{A}_l denotes the activation tensor of the l^{th} convolutional layer, $1 \leq l \leq L$, while \mathbf{A}_l^k refers to the activations of the k -th channel of layer l .

s_l denotes the subsampling factor of layer l compared to the input. It corresponds to the product of stride and pooling factors encountered between the input and layer l .

$\uparrow_{bi}(\mathbf{M}, s)$ defines a bilinear upsampling of a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ by a factor $s \in \mathbb{N}$.

$\downarrow_{av}(\mathbf{M}, s)$ denotes a 2D average pooling on any matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ with a stride $s \in \mathbb{N}$.

$u(\mathbf{M})$ linearly maps the value range of the elements in matrix \mathbf{M} to the unit interval.

\odot denotes the element-wise division operator, while \odot denotes the element-wise product operator.

$LNorm(\mathbf{M}, s)$ is a local normalisation operator. It partitions the matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ in a set of non overlapping blocks of size $s \times s$, with $s \in \mathbb{N}$, and divides each matrix element by the mean value of its corresponding block. Formally, using the above notations,

$$LNorm(\mathbf{M}, s) = \mathbf{M} \odot (\uparrow_{bi}(\downarrow_{av}(\mathbf{M}, s), s)). \quad (1)$$

$ReLU$ denotes the rectified linear units (Dahl et al., 2013).

3.2 ACTIVATION MAPS IN PREVIOUS WORK

In (Zhou et al., 2016) CAM_l^c , the Class Activation Map associated to a target class c and a layer l is defined as

$$CAM_l^c = ReLU\left(\sum_k w_{l,k}(c) \mathbf{A}_l^k\right), \quad (2)$$

with \mathbf{A}_l^k denoting the k^{th} activation map of the l^{th} convolutional layer, $l \in 1, \dots, L$, and $w_{l,k}(c)$ being a scalar weighting factor. Most of the CAM-based methods (Selvaraju et al., 2017; Chattopadhyay et al., 2018; Smilkov et al., 2017; Wang et al., 2020b;a; Naidu et al., 2020) adopt this formula. They differ in the way they define the weighting factors, and generally only consider it for the last convolutional layer ($l = L$). **Alternatively, Zoom-CAM and Layer-CAM have proposed to combine activation maps from multiple layers, using gradients as dense weighting factors. Our work also combines multiple activation maps, but does it without back-propagated gradients, thereby managing to produce high-resolution saliency maps without inheriting the noise from the gradient.**

3.3 OUR PROPOSED POLY-CAM

Our method leverages information from multiple layers to produce a high resolution Class Activation Map. Similar to other CAM-based techniques, it builds on the linear combination of activation maps, but combines them through a backward recursive procedure, as depicted in Figure 1.

Letting P_l^c denote the class-specific saliency map associated to class c in the l^{th} layer, the recursive process works as follows. In the initial step, the saliency map P_L^c is defined to be equal to the conventional CAM_L^c saliency map, as derived from equation (2). Then, at each recursive step, an upsampled version of P_{l+1}^c is tuned (or modulated) by a locally normalized version of the activation map in the l^{th} layer. Mathematically, we have:

$$P_l^c = \begin{cases} ReLU(\sum_k w_{l,k}(c) \mathbf{A}_l^k) & \text{for } l = L \\ LNORM\left(ReLU\left(\sum_k w_{l,k}(c) \mathbf{A}_l^k\right), \frac{s_{l+1}}{s_l}\right) \odot \uparrow_{bi}\left(P_{l+1}^c, \frac{s_{l+1}}{s_l}\right) & \text{for } 1 \leq l \leq L - 1 \end{cases} \quad (3)$$

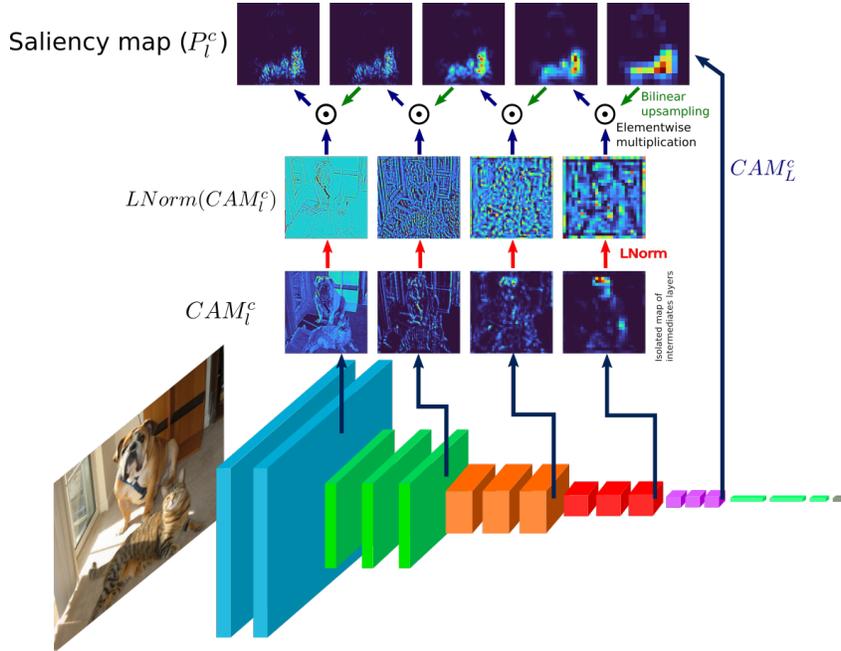


Figure 1: Our Poly-CAM process: the upsampled version of the saliency map in layer l is tuned based on the class activation map of layer $l - 1$. Image samples correspond to the ‘cat’ class, and are computed from VGG16.

with $w_{l,k}(c)$ denoting the class-specific weighting factor associated to A_l^k , as defined in Section 3.4. Intuitively, Equation (3) can be understood based on the following two observations. First, the element-wise multiplication, between the upsampled (and thus smooth) saliency map of layer $l + 1$ and the activation map in layer l , aims at restricting the large saliency values in layer $l + 1$ to the locations that are activated in layer l . Second, the local normalization ($LNorm$) of the activation map aims at preserving the spatial distribution of saliency across the layers. It ensures that an image block with large saliency in layer $l+1$ has also a large saliency in layer l , even if the level of activation in this block is small compared to the rest of the image. This is meaningful since the backpropagated saliency should be predominant to assign a saliency level to a spatial region in layer l , while the activation in layer l should simply control the increase in resolution, by tuning the smooth saliency signal inherited from coarser layers based on the local variations of the activation map. Our ablation study in Appendix B.2 confirms the critical role played by the $LNorm$ operator.

3.4 ACTIVATION MAP WEIGHT DEFINITION

This section presents different alternatives to define the weights $w_{l,k}(c)$, used in equation 3.

Channel-wise Increase of Confidence Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a) and IS-CAM (Naidu et al., 2020) define $w_{l,k}(c)$ based on the so-called Channel-wise Increase of Confidence (CIC), which estimates how the spatial support of the activation map A_l^k contributes to the softmax output f . Formally, the channel-wise increase is denoted $w_{l,k}^+$, and is defined as:

$$w_{l,k}^+ = f(X \odot u(\uparrow_{bi}(A_l^k, s_l))) - f(X_b), \quad (4)$$

with \odot denoting the pixel-wise product, and X_b referring to a baseline image. Previous works have considered baselines that are uniform black, uniform grey, or a blur version of X . In the following, $f_c(X_b)$ is set to zero in all experiments.

Our work proposes two extensions of equation 4, respectively to measure how the softmax output decreases when masking a fraction of the input, and to sum-up the increase and the decrease associated to the unveiling and the masking of the input. Those new weights are defined as follows.

Channel-wise Decrease of Confidence The Channel-wise Decrease of Confidence (*CDC*) is a dual notion compared to *CIC*. Instead of measuring the increase of softmax output when the part of the input corresponding to non-zero A_l^k is unveiled and the remaining is masked, *CDC* measures the decrease of softmax output when the part of the input corresponding to A_l^k is masked. The intuition is that an important part of the input for any class c not only increase the output when shown, but should also decrease it when hidden. Formally,

$$w_{l,k}^- = \text{ReLU}\left(f(X) - f\left(X \odot \left(1 - u\left(\uparrow_{bi}\left(A_l^k, s_l\right)\right)\right)\right)\right) \quad (5)$$

ReLU is applied to only keep the activation maps that decreases the output when removed.

Channel-wise Variation of Confidence By combining the *CIC* with the *CDC*, the Channel-wise Variation of Confidence (*CVC*) is defined. Formally,

$$w_{l,k}^\pm = \text{ReLU}\left(f(X) + f\left(X \odot u\left(\uparrow_{bi}\left(A_l^k, s_l\right)\right)\right) - f\left(X \odot \left(1 - u\left(\uparrow_{bi}\left(A_l^k, s_l\right)\right)\right)\right)\right) \quad (6)$$

The Channel-wise Variation of Confidence is either influenced by the ability of the activation map to increase the softmax output when inserted but also decrease it when removed.

4 EXPERIMENTS

Three variants of the Poly-CAM method introduced in Section 3.3 are considered, depending on whether $w_{l,k}$ is defined to be equal to $w_{l,k}^+$ (PCAM⁺), $w_{l,k}^-$ (PCAM⁻) or $w_{l,k}^\pm$ (PCAM[±]).

We follow the assessment method described in (Petsiuk et al., 2018) and Chattopadhyay et al. (2018) to evaluate our proposal. Datasets, networks, and baseline methods are presented in Section 4.1. Quantitative assessment of the saliency maps is considered in Section 4.2, while a qualitative and visual assessment is presented in Section 4.3.

4.1 EXPERIMENTAL SET-UP AND SALIENCY MAP BASELINES

For these evaluations, 2000 images were randomly selected from the 2012 ILSVRC validation set (Russakovsky et al., 2015). The images are scaled to 224x224x3 pixels and normalized to the same mean and standard deviation as the ImageNet (Russakovsky et al., 2015) training set (mean vector : [0.485, 0.456, 0.406], standard deviation vector [0.229, 0.224, 0.225]). The models used for faithfulness evaluation are VGG16 (Simonyan & Zisserman, 2014) and ResNet50 (He et al., 2016), both pretrained from the PyTorch model zoo. The analysis considers, as reference baselines, Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Smooth Grad-CAM++ (Omeiza et al., 2019), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Zoom-CAM (Shi et al., 2021), Layer-CAM (Jiang et al., 2021), Occlusion (Zeiler & Fergus, 2014), Input X Gradient (Shrikumar et al., 2016), Integrated Gradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017) and RISE (Petsiuk et al., 2018). The implementations for these methods are the ones from Captum (Kokhlikyan et al., 2020) for Integrated Gradient, SmoothGrad and Occlusion, from <https://github.com/eclique/RISE> for RISE, from <https://github.com/X-Shi/Zoom-CAM> for Zoom-CAM and from torchcam (Fernandez, 2020) for all the other CAM-based methods.

For SS-CAM, IS-CAM, LayerCAM and ZoomCAM, SmoothGrad and IntegratedGradient, the parameters recommended by the authors or set as default in the reference implementation have been used when available. ¹ For Occlusion, the size of occlusion patch was set to (64, 64) with a stride of (8, 8) as used by (Petsiuk et al., 2018). For RISE, 6000 masks were used.

For the Poly-CAM methods (PCAM⁺, PCAM⁻, PCAM[±]), the layers corresponding to a change in resolution were used. It corresponds to [block1_conv2, block2_conv2, block3_conv3, block4_conv3, block5_conv3] for VGG16, and [conv1_1, conv2_3, conv3_4, conv4_6, conv5_3] for ResNet50.

¹It means 35 input perturbations (with a $\sigma = 2$ Gaussian noise) for SS-CAM, 50 input perturbations (with a $\sigma = 1$ Gaussian noise) for SmoothGrad, 10 interpolation steps for IS-CAM, and 50 for IntegratedGradient. For Layer-CAM, the layers corresponding to a change in resolution were used, and recommended scaling has been applied to the first two layers. For Zoom-CAM, all the layers/blocks were fused for VGG16 and ResNet50.

4.2 FAITHFULNESS ASSESSMENT

There is still a lack of consensus regarding the metrics to assess the relevance of saliency maps for explainability (Poursabzi-Sangdeh et al., 2021). The ability to segment the semantic object justifying the class label was used as a proxy to evaluate saliency maps (Selvaraju et al., 2017), but it has been observed in Petsiuk et al. (2018) that the segmentation mask might not be correlated to the discriminant visual features justifying the class label. The same authors also introduced the insertion and deletion metrics to measure the increase (resp. decrease) in the class softmax output when inserting (resp. removing) pixels in decreasing order of saliency. These metrics are largely used in recent works (Wang et al., 2020b;a; Naidu et al., 2020). They are thus considered in this work, **even if they remain arguable, as discussed in the following sections.**

4.2.1 METRICS DEFINITION

Insertion Formalised by Petsiuk et al. (2018), the insertion metric measures how fast the model softmax output increases when adding the salient image pixels to a baseline image. The pixels are of a baseline (uniform black/grey or highly blurred version of the image) are replaced by the image ones in decreasing order of saliency map value, allowing to define the faithfulness metric as the area under the curve of the class softmax output, observed as a function of the proportion of pixels inserted. The higher the metric the better. A blurred version of the image is generally preferred to a uniform color baseline, since it prevents the introduction of sharp artificial edges that might disrupt the predictions.

Deletion The deletion metric (Petsiuk et al., 2018) measures the decrease in class softmax output while removing pixels. The pixels are progressively removed from the image and replaced by a baseline using the same logic as the insertion metric. The intuition behind the deletion metric is that removing the more important pixels in the saliency map should more rapidly decrease the softmax output of the target class prediction. The lower the metric the better.

Insertion - Deletion A good explanation map should have a good insertion map and a good deletion map. A combination of the two scores is thus proposed here by computing the difference between insertion (higher is better) and deletion (lower is better) to obtain a combined score that should be maximized.

It is worth noting that, despite they are widely used, those metrics have some drawbacks. In particular, the insertion and deletion procedures are likely to result in outliers, i.e. in images that do not match the distribution of natural images. To mitigate this problem, as most previous works, we have used blurred baselines when implementing the insertion/deletion process. However, it is not sufficient to ensure that a high correlation between the network prediction and the salient pixels means that those pixels exactly correspond to the whole set of class-discriminant visual features. Hence, those quantitative metrics should never be used as a substitute for a visual assessment of the saliency map, which remains the golden standard when comparing saliency map generation methods.

For the above metrics, 224 steps were performed with 224 pixels inserted or removed at each step. The used baseline for all the metrics is a blurred version of the input image using a Gaussian kernel of size 11x11 with a sigma of 5.

4.2.2 QUANTITATIVE ASSESSMENT

Table 1 compares the faithfulness metrics for all CAM-based methods. The results related to gradient and perturbation methods are presented in Appendix, Table 4. Plots of the softmax output as a function of the amount of inserted pixels are presented in Appendix E.2.

Among the three Poly-CAM variants, PCAM[±] gives the best results compared to PCAM⁺ and PCAM⁻ for all metrics on VGG16. On ResNet50, PCAM[±] gives a insertion metric similar to PCAM⁺ and superior to PCAM⁻, while PCAM⁻ gives a better deletion metric compared to PCAM⁺ and PCAM[±]. For insertion-deletion on ResNet50, PCAM[±] and PCAM⁻ are on par.

Interestingly, the insertion metrics of PCAM[±] is systematically better than all other CAM-based approaches. Compared to the non-CAM methods, the PCAM[±] method gives similar or better insertion results than perturbation or gradient methods, respectively.

In terms of deletion, PCAM[±] tends to perform better than most other CAM-based methods, but

Table 1: Faithfulness metrics for CAM-based methods (see Table 4 for full version with all methods)

Method	VGG16			ResNet50		
	Insertion	Deletion	Ins-Del	Insertion	Deletion	Ins-Del
GradCAM	0.58	0.18	0.40	0.65	0.31	0.35
GradCAM++	0.57	0.19	0.38	0.65	0.31	0.34
SmoothGradCAM++	0.54	0.21	0.33	0.63	0.32	0.30
ScoreCAM	0.59	0.19	0.40	0.65	0.31	0.34
SSCAM	0.50	0.23	0.27	0.59	0.36	0.24
ISCAM	0.59	0.19	0.40	0.65	0.32	0.33
ZoomCAM	0.60	0.14	0.46	0.66	0.29	0.37
LayerCAM	0.58	0.14	0.44	0.65	0.30	0.35
PCAM ⁺ (ours)	0.58	0.17	0.41	0.67	0.29	0.38
PCAM ⁻ (ours)	0.60	0.16	0.45	0.66	0.27	0.39
PCAM [±] (ours)	0.61	0.15	0.46	0.67	0.28	0.39

Insertion (higher is better), deletion (lower is better) and insertion-deletion (higher is better) with VGG16 and ResNet50 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Smooth Grad-CAM++ (Omeiza et al., 2019), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), **Zoom-CAM (Shi et al., 2021)**, **Layer-CAM (Jiang et al., 2021)**.

appears to be weaker than gradient-based methods. InputXGrad and IntegratedGradient achieve at the same time very poor results on the insertion metric and thus have a poor insertion-deletion. This is not surprising since gradient-based methods give lots of importance to the parts of the input that largely impact the loss and thus the output. As a consequence, the deletion metric is (trivially) good for those methods since this metric measures the decrease of output when important parts are removed from the input. The poor insertion metric however reveal that the parts that are considered as being important by gradient methods are not sufficient to explain the network prediction. **This observation reveals the limits of the metrics when applied to dissimilar kinds of techniques.**

4.3 VISUAL ASSESSMENT

This section assesses our method visually. Saliency maps were generated for all the baseline methods (see Section 4.1) on the 2000 selected images using VGG16 model. For the Poly-CAM methods, saliency maps were also generated for each target layer. An interactive interface is provided as a jupyter notebook in supplementary material, in addition to the source code, to allow an easy visualisation of any of the saliency maps generated in our experiments. Section 4.3.1 compares the three Poly-CAM variants, as a function of the layer index and targeted class. A comparison with previous works is shown in Section 4.3.2. In Section 4.3.3, PCAM[±] is considered to explain VGG16 misclassifications.

4.3.1 PCAM VARIANTS

PCAM produces saliency maps at various resolutions. Figure 1 shows how Poly-CAM progressively refines the last layer saliency map through a backward recursive strategy. More examples of intermediate maps obtained along this recursive process are presented in Appendix B.1. We observe that the structures are coarse at block5_conv3, to gain in accuracy when accounting for earlier network layers, doubling the resolution at each step. The elements highlighted by the three variants are similar on the majority of images. However, variations appear on some images, PCAM⁻ highlighting more frequently contextual elements compared to PCAM⁺ (and PCAM[±] sitting between the two). Intuitively, this can be understood by the fact that the appearance of those contextual features in a baseline image does not help in classifying the image correctly, while their removal from the original image penalizes the classification.

All Poly-CAM variants are class specific as displayed in Figure 2, where the saliency maps associated to the the Barn, Alps and the Ox classes are clearly distinct, with a level of accuracy close to segmentation. It is worth noting that PCAM⁺ is more specific in highlighting the part of the image

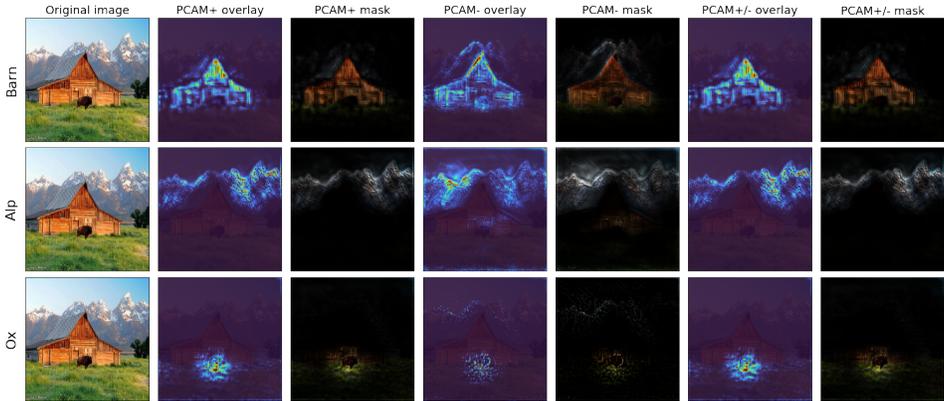


Figure 2: Multiple classes on Poly-CAM. The different classes as correctly determined by the three variants. We can note that $PCAM^-$ is less specific than the other methods (part of the mountain is attributed to the barn), while $PCAM^+$ is the most specific.

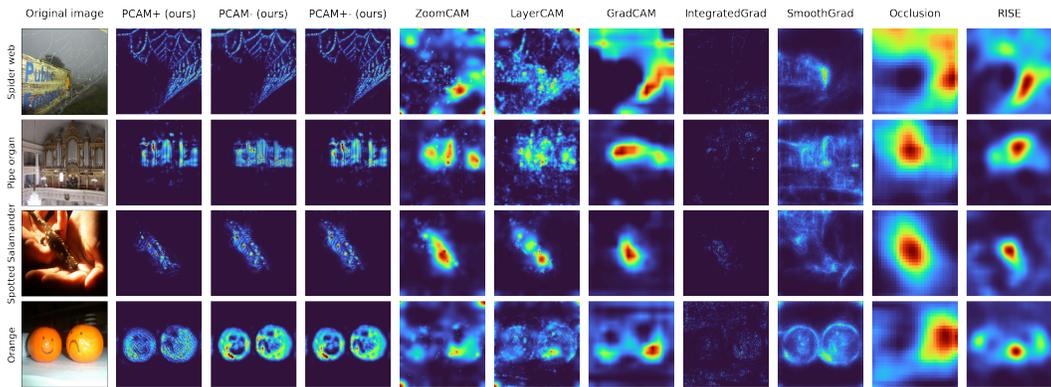


Figure 3: Visual comparison of methods (see Figure 5 in appendix for full version). The compared methods are our three variants of Poly-CAM ($PCAM^+$, $PCAM^-$, $PCAM^\pm$), **Zoom-CAM** (Shi et al., 2021), **Layer-CAM** (Jiang et al., 2021), Grad-CAM (Selvaraju et al., 2017), IntegratedGradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), Occlusion (Zeiler & Fergus, 2014) and RISE (Petsiuk et al., 2018). A description of this figure is available in Section 4.3.2

related to the class of interest. This is in line with the above observation that $PCAM^-$, and a bit less $PCAM^\pm$, are stronger in highlighting contextual information (see more examples in Appendix C).

4.3.2 COMPARISON WITH PREVIOUS WORKS

Saliency maps have been produced for all baseline methods listed in Section 4.1. A sample of this comparison is presented in Figure 3. For ease of view, this figure restricts to Grad-CAM, Score-CAM, Integrated Gradient, SmoothGrad, Occlusion, RISE and the three Poly-CAM variants. A comparison with all the methods listed in Section 4 can be found in Appendix A, Figure 5. We can see that Poly-CAM methods accurately identify quite relevant elements in the image like a spider net or the pipes of an organ. CAM and perturbation methods cannot achieve this level of precision, gradient base methods highlight elements of the image that are not related to the class, while **Zoom-CAM and Layer-CAM increase the resolution but are more noisy, halfway between gradient and more classical CAM-based methods**. The spotted salamander is highlighted by all methods but the Poly-CAM methods are the only ones to identify the spots. The oranges are identified by Poly-CAM methods while the smile sketched on them is correctly excluded. This is in contrast with other methods that are either too low resolution, or do not exclude the smile, while SmoothGrad seems to give more importance to the smile than to the texture of the orange.

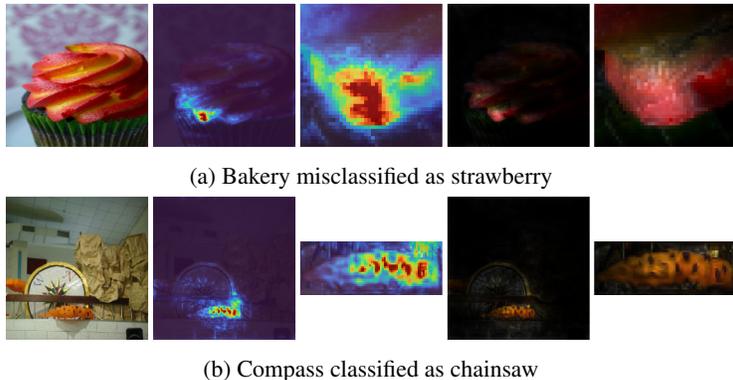


Figure 4: **Misclassification explanations.** From right to left: original image, attention map, masked image, and their zoomed versions. (a) Image is misclassified as strawberry. The saliency map explains the wrong classification by highlighting a visual stimuli looking like a strawberry. (b) Image is misclassified as chainsaw, as explained by the saliency map.

4.3.3 CLASSIFICATION FAILURE EXPLANATION

PCAM[±] is used to produce explanation maps for images misclassified by VGG16, to show the ability of Poly-CAM to explain the reasons why the model made a mistake. Images with misclassification were selected in the dataset. The image samples presented in Figure 4 have been chosen so that the misclassification is not related to a similar class (e.g. a golden retriever classified as a Labrador retriever) or to another object present in the image. As illustrated in Figure 4 looking at the image masked by the PCAM[±] saliency map helps in understanding the misclassification. In Figure 4(a), we can identify the strawberry seen by the model, when using the saliency map as a mask. In Figure 4(b), typical chainsaw features are also made visible by the overlaid saliency map.

4.4 SANITY CHECK AND ROBUSTNESS

As a sanity check, following the method in Adebayo et al. (2018), the PCAM saliency maps have been visualized at each step of a cascading randomization of a VGG16 network, from last to first layer. The purpose of this sanity check is to verify that the Poly-CAM methods do not work as edge detectors, and effectively relies on the actual weights of the model to derive class-specific saliency maps. All PCAM variants successfully passed the test, as shown in Appendix G. **To evaluate the robustness of our explanation method, a sensitivity analysis has been run, following the methodology introduced in (Ghorbani et al., 2019) and (Yeh et al., 2019). Results are presented in Appendix F. They reveal that PCAM has a small explanation sensitivity, similar to the ones obtained by other CAM-based methods, and one or two orders of magnitude below the sensitivities obtained by gradient-based and perturbation methods.**

5 CONCLUSION

This paper has introduced the Poly-CAM method, to produce high resolution saliency maps without relying on gradient backpropagation. Three variants of our Poly-CAM framework are investigated, depending on whether the values weighting the activation maps are obtained by masking or unveiling image pixels, or both. Our experiments reveal that the combined strategy, i.e. PCAM[±], provides the more versatile solution with state of the art performances in term of faithfulness insertion-deletion metrics and outperforming current available methods in term of precision of visualization. Despite our work is a valuable step towards a more explainable AI, there is still plenty of room for improvement in this domain. One of the questions raised by this work is related to the way the importance of a pixel should be quantified. Indeed, the importance of a group of pixels appears to be different when this group is removed or when it is inserted (for example the importance of contextual information is more important when removing it than when inserting it), which can not be properly reflected by a single saliency map.

REPRODUCIBILITY STATEMENT

Attention was ported on the reproductibility of this paper. The source code for Poly-CAM is provided as supplementary material. The list of images used from the 2012 ILSVRC validation set (Russakovsky et al., 2015) is also provided. The saliency maps used in this paper can be generated as npz files using provided scripts, or alternatively can be downloaded from <https://polycam.ddns.net>, an anonymous website made for the blind review process. One npz file is provided for each saliency method applied to one model. The files are named based on the model used ("vgg16" or "resnet50") and the saliency method used. Inside each npz file, the saliency maps can be retrieve in a numpy array format using the name of the image (with the ".JPEG" extension included) as a key. Similarly, measurement for the faithfulness metrics can either be generated as csv files or downloaded from <https://polycam.ddns.net>. The csv files are named as so '{del,ins}_{auc,details}_{vgg16,resnet50}_{saliency}.csv'. The prefix "del_" and "ins_" refer to files associated with the deletion and insertion metrics respectively, "auc" files contains the area under the curve for the above metrics while files named with "details" contains the class certitude for each step of insertion or deletion, "vgg16" and "resnet50" refers to the model and "saliency" is the name of the saliency method. Jupyter notebooks are also provided for easier handling. Practical information can be found in the README.md file inside the supplementary zip file.

REFERENCES

- Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.
- Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *arXiv preprint arXiv:1810.03292*, 2018.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- Aditya Chattopadhyay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision (WACV)*. IEEE, 2018.
- George E Dahl, Tara N Sainath, and Geoffrey E Hinton. Improving deep neural networks for lvsr using rectified linear units and dropout. In *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013.
- François-Guillaume Fernandez. Torchcam: class activation explorer. <https://github.com/frgfm/torch-cam>, March 2020.
- Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.
- 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.
- Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.
- Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1–8, 2019.

- Rakshit Naidu, Ankita Ghosh, Yash Maurya, Soumya Snigdha Kundu, et al. Is-cam: Integrated score-cam for axiomatic-based explanations. *arXiv preprint arXiv:2010.03023*, 2020.
- Daniel Omeiza, Skyler Speakman, Celia Cintas, and Komminist Weldermariam. Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. *arXiv preprint arXiv:1908.01224*, 2019.
- Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *British Machine Vision Conference (BMVC)*, 2018. URL <http://bmvc2018.org/contents/papers/1064.pdf>.
- Forough Poursabzi-Sangdeh, Daniel G Goldstein, Jake M Hofman, Jennifer Wortman Wortman Vaughan, and Hanna Wallach. Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021.
- Pranav Rajpurkar, Jeremy Irvin, Aarti Bagul, Daisy Ding, Tony Duan, Hershel Mehta, Brandon Yang, Kaylie Zhu, Dillon Laird, Robyn L Ball, et al. Mura: Large dataset for abnormality detection in musculoskeletal radiographs. *arXiv preprint arXiv:1712.06957*, 2017.
- Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.
- Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021.
- 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*, 2017.
- Xiangwei Shi, Seyran Khademi, Yunqiang Li, and Jan van Gemert. Zoom-cam: Generating fine-grained pixel annotations from image labels. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 10289–10296. IEEE, 2021.
- Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.
- Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*. PMLR, 2017.
- Thanos Tagaris, Maria Sdraka, and Andreas Stafylopatis. High-resolution class activation mapping. In *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019.
- Haofan Wang, Rakshit Naidu, Joy Michael, and Soumya Snigdha Kundu. Ss-cam: Smoothed score-cam for sharper visual feature localization. *arXiv preprint arXiv:2006.14255*, 2020a.
- 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, workshop on Fair, Data Efficient and Trusted Computer Vision*, 2020b.

Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32:10967–10978, 2019.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*. Springer, 2014.

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*, 2016.

A POLY-CAM VS PREVIOUS WORKS

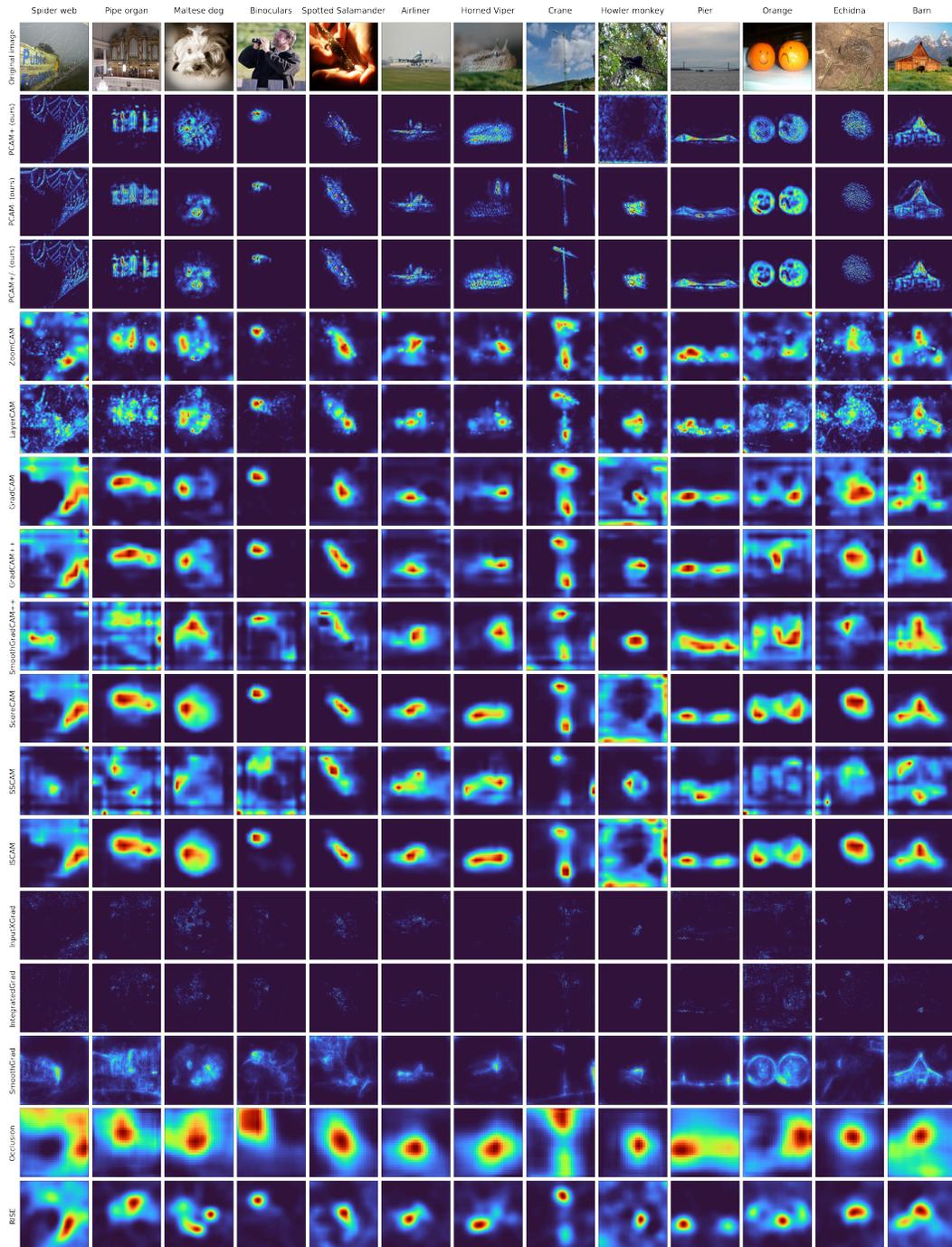


Figure 5: Visual comparison of methods. The compared methods are the three Poly-CAM variants proposed in this paper (PCAM⁺, PCAM⁻, PCAM[±]), Zoom-CAM (Shi et al., 2021), Layer-CAM (Jiang et al., 2021), Grad-CAM (Selvaraju et al., 2017), Grad-CAM⁺⁺ (Chattopadhyay et al., 2018), Smooth Grad-CAM⁺⁺ (Omeiza et al., 2019), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Input X Gradient (Shrikumar et al., 2016), IntegratedGradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), Occlusion (Zeiler & Fergus, 2014), RISE (Petsiuk et al., 2018).

B ABLATION STUDY

Section B.1 presents an ablation study designed to assess the importance of combining different layers (as proposed in equation 3) versus deriving the saliency map from a single layer (as defined by equation 2). vs using a specific layer in isolation in section B.1. Section B.2 reveals the critical importance of the LNorm operation in equation 1.

B.1 POLY-CAM VERSUS LAYER-SPECIFIC MAP

The interest of combining the activation maps from multiple layers, as proposed by equation 3, is demonstrated by comparing Poly-CAM with the saliency maps derived in individual layers, using equation 2. The three weighting factors presented in Section 3.4 are considered.

The set-up and models are the same as the ones used in Section 4.1 and the target layers are [block1_conv2, block2_conv2, block3_conv3, block4_conv3, block5_conv3] for VGG16, and [conv1_1, conv2_3, conv3_4, conv4_6, conv5_3] for ResNet50. A visual comparison of the PCAM methods vs intermediate CAM is shown in Figure 7 and Figure 6. A visual comparison of the three Poly-CAM variants for different layers is shown in Figure 8, Figure 9 and Figure 10. Two comparative graphs are provided in Figure 11 and Figure 12 for VGG16 and ResNet50 respectively. The results are shown in Table 2 for VGG16, and in Table 3 for ResNet50.

While the Poly-CAM methods improve when integrating more layers, the CAM dramatically loses in class-specificity when considering early layers in the networks. This highlights that combining the layers is a better approach than using CAM on isolated early layers using the proposed $w_{l,k}$ scores. This poor behavior of CAM saliency maps in early layers confirms the results provided by Shi et al. (2021) in their ablation study.

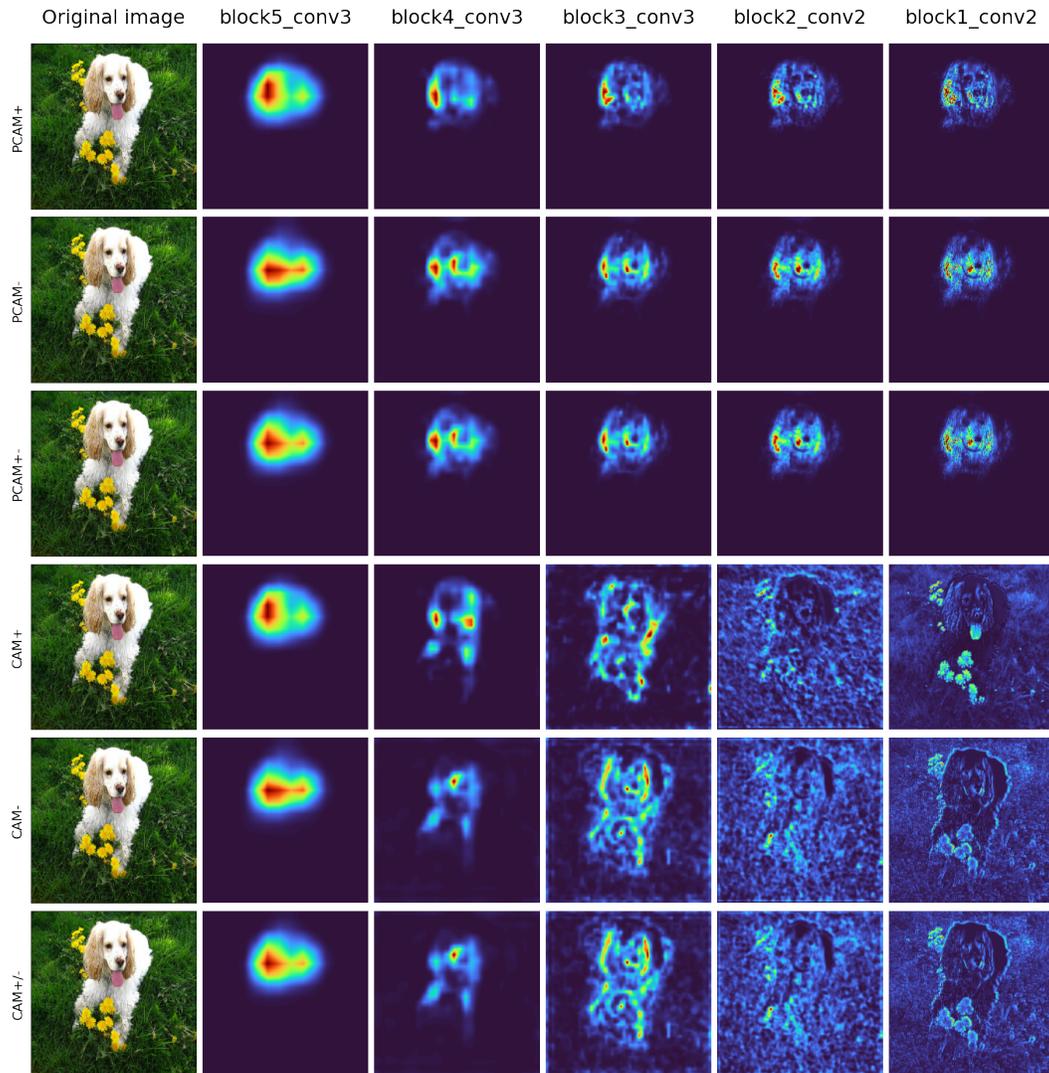


Figure 6: Comparison of Poly-CAM vs classical CAM-based results as a function of the layer, with the three proposed weights in Section 3.4 on VGG16. We can observe that computing the CAM saliency map from a single and early layer gives poor results, with maps highlighting elements all over the input image. In contrast, Poly-CAM methods increase the resolution while keeping the focus of saliency on class-specific relevant features.

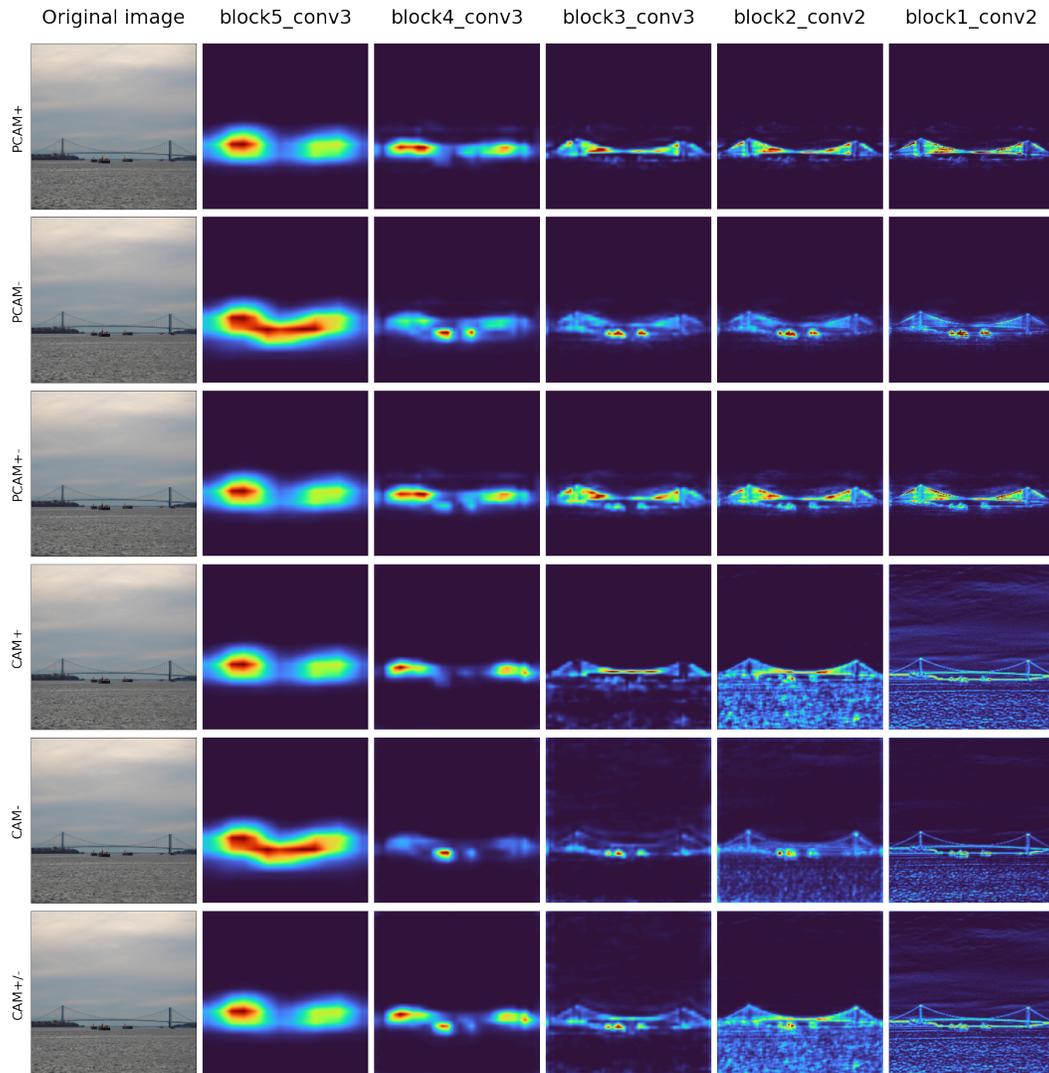


Figure 7: Comparison of Poly-CAM vs classical CAM-based results as a function of the layer, with the three proposed weights in Section 3.4 on VGG16. We can observe that computing the CAM saliency map from a single and early layer gives poor results, with maps highlighting elements all over the input image. In contrast, Poly-CAM methods increase the resolution while keeping the focus of saliency on class-specific relevant features.

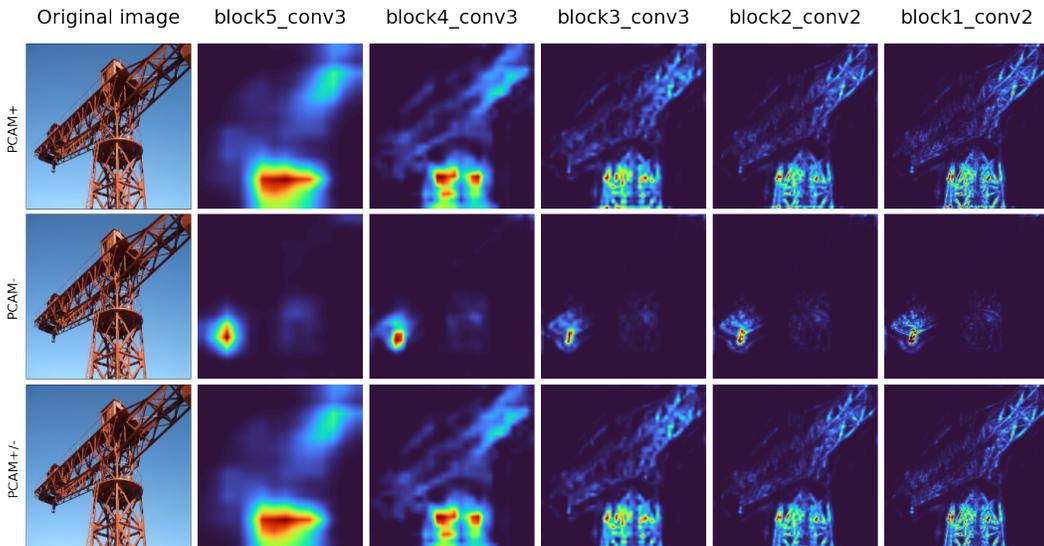


Figure 8: Comparison of Poly-CAM for different layers of VGG16 on a crane. The saliency map is refined as layers are added from left to right, showing more precise structures. We can also see that $PCAM^-$ gives more importance to the hook while $PCAM^+$ and $PCAM^\pm$ give more importance on the mast and jib of the crane. This reflects the fact that, given the metallic structure present in the image, removing the hook most significantly impact the crane functionality of this metallic structure. In contrast, all metallic parts are worth to be added when starting from a baseline image.

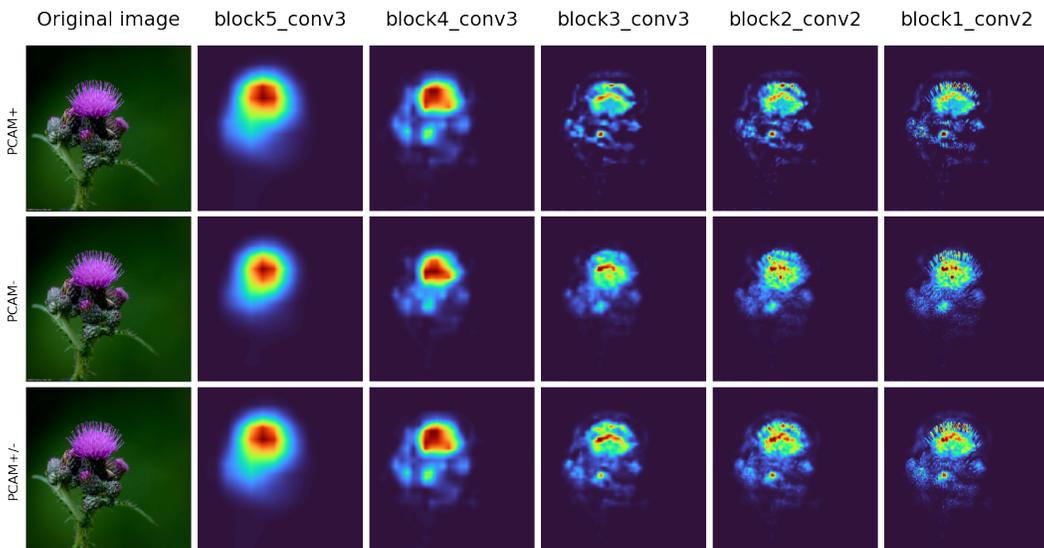


Figure 9: Comparison of Poly-CAM for different layers of VGG16 on a cardoon. The focus is progressively more intense in the spikes of the flowers, the most important part seems to be at the tips of the violet spikes for the three variants, which give similar results in this case.

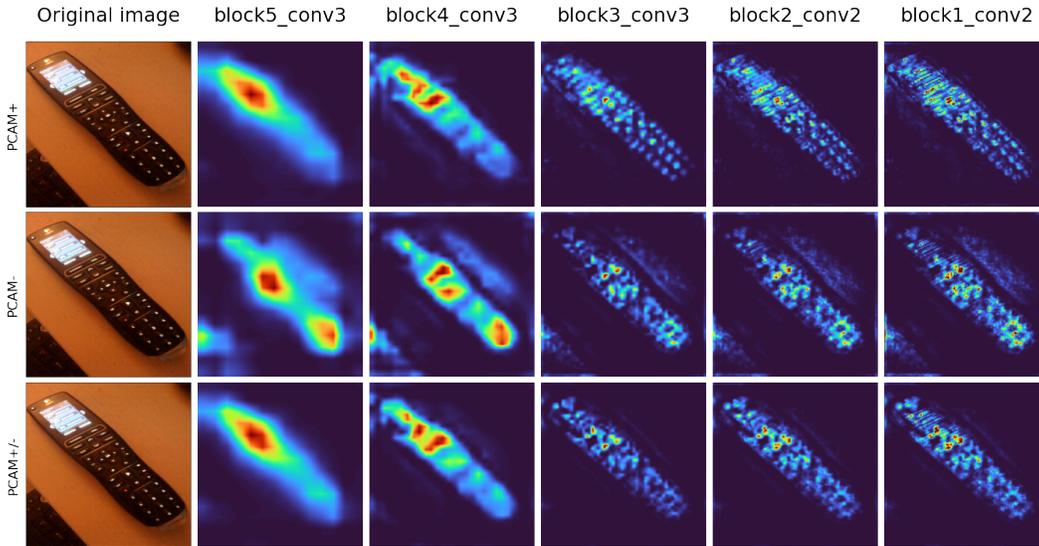


Figure 10: Comparison of Poly-CAM for different layers of VGG16 on a remote control. We can see the progressive focus on the buttons when we progress over the layers to produce a more precise map. In this case, the three variants produce similar results.

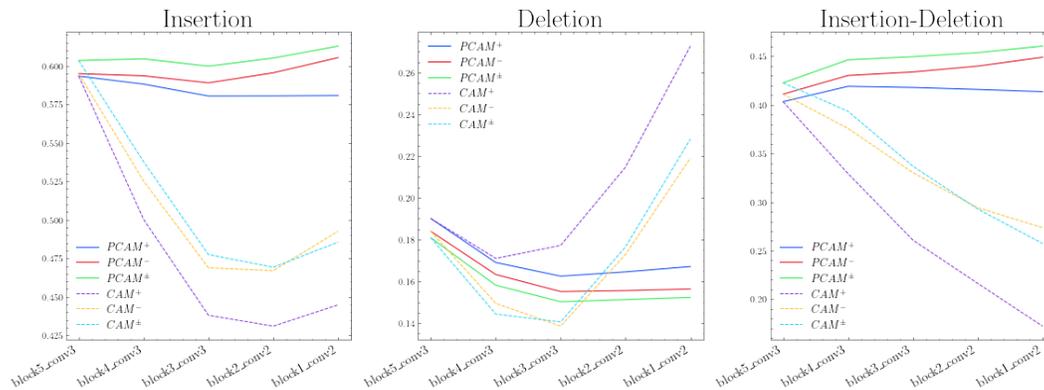


Figure 11: Comparison of metrics as a function of the layer index for Poly-CAM vs CAM on isolated layers - VGG16. Last to first layer is represented left to right. This show that using CAM saliency maps on early layers performs poorly.

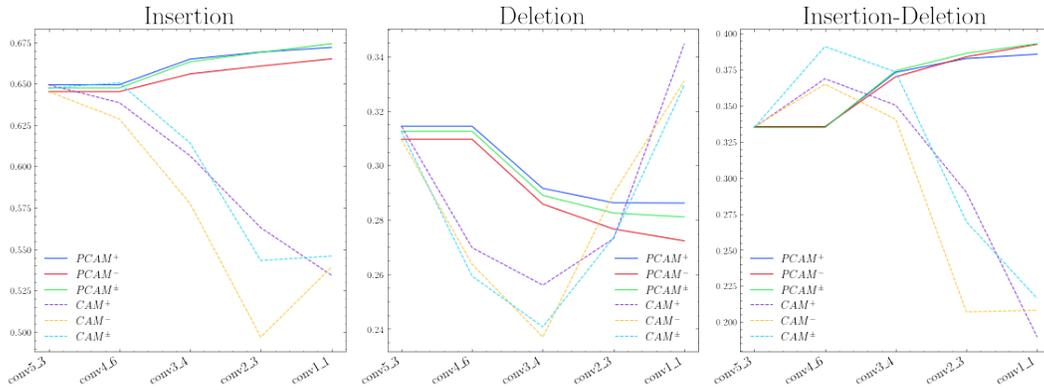


Figure 12: Comparison of metrics as a function of the layer index for Poly-CAM vs CAM on isolated layers - ResNet50. Last to first layer is represented left to right. This show that using CAM saliency maps on early layers performs poorly.

Table 2: Faithfulness metrics for Poly-CAM and CAM on isolated layers - VGG16

Method	VGG16			
	Layer	Insertion	Deletion	Ins-Del
<i>PCAM</i> ⁺	block5_conv3	0.59	0.19	0.40
	block4_conv3	0.59	0.17	0.42
	block3_conv3	0.58	0.16	0.42
	block2_conv2	0.58	0.16	0.42
	block1_conv2	0.58	0.17	0.41
<i>PCAM</i> ⁻	block5_conv3	0.59	0.18	0.41
	block4_conv3	0.59	0.16	0.43
	block3_conv3	0.59	0.16	0.43
	block2_conv2	0.60	0.16	0.44
	block1_conv2	0.60	0.16	0.45
<i>PCAM</i> [±]	block5_conv3	0.60	0.18	0.42
	block4_conv3	0.60	0.16	0.45
	block3_conv3	0.60	0.15	0.45
	block2_conv2	0.60	0.15	0.45
	block1_conv2	0.61	0.15	0.46
<i>CAM</i> ⁺	block5_conv3	0.59	0.19	0.40
	block4_conv3	0.50	0.17	0.33
	block3_conv3	0.44	0.18	0.26
	block2_conv2	0.43	0.21	0.22
	block1_conv2	0.44	0.27	0.17
<i>CAM</i> ⁻	block5_conv3	0.59	0.18	0.41
	block4_conv3	0.53	0.15	0.38
	block3_conv3	0.47	0.14	0.33
	block2_conv2	0.47	0.17	0.29
	block1_conv2	0.49	0.22	0.27
<i>CAM</i> [±]	block5_conv3	0.60	0.18	0.42
	block4_conv3	0.54	0.14	0.39
	block3_conv3	0.48	0.14	0.34
	block2_conv2	0.47	0.18	0.29
	block1_conv2	0.49	0.23	0.26

Table 3: Faithfulness metrics for Poly-CAM and CAM on isolated layers - ResNet50

Method	ResNet50			
	Layer	Insertion	Deletion	Ins-Del
<i>PCAM</i> ⁺	conv5_3	0.65	0.31	0.33
	conv4_6	0.65	0.31	0.33
	conv3_4	0.66	0.29	0.37
	conv2_3	0.67	0.29	0.38
	conv1_1	0.67	0.29	0.38
<i>PCAM</i> ⁻	conv5_3	0.65	0.31	0.34
	conv4_6	0.65	0.31	0.34
	conv3_4	0.66	0.29	0.37
	conv2_3	0.66	0.28	0.38
	conv1_1	0.66	0.27	0.39
<i>PCAM</i> [±]	conv5_3	0.65	0.31	0.33
	conv4_6	0.65	0.31	0.33
	conv3_4	0.66	0.29	0.37
	conv2_3	0.67	0.28	0.39
	conv1_1	0.67	0.28	0.39
<i>CAM</i> ⁺	conv5_3	0.65	0.31	0.33
	conv4_6	0.64	0.27	0.37
	conv3_4	0.61	0.26	0.35
	conv2_3	0.56	0.27	0.29
	conv1_1	0.53	0.34	0.19
<i>CAM</i> ⁻	conv5_3	0.65	0.31	0.34
	conv4_6	0.63	0.26	0.36
	conv3_4	0.58	0.24	0.34
	conv2_3	0.50	0.29	0.21
	conv1_1	0.54	0.33	0.21
<i>CAM</i> [±]	conv5_3	0.65	0.31	0.33
	conv4_6	0.65	0.26	0.39
	conv3_4	0.61	0.24	0.37
	conv2_3	0.54	0.27	0.27
	conv1_1	0.55	0.33	0.22

B.2 IMPORTANCE OF LNorm

The importance of including the LNorm operator in equation (3) is challenged this section. We produced saliency maps using both the complete method and a variant where LNorm has been ablated. Formally, the saliency map of the LNorm ablated method is

$$P_{withoutLNorm}_l^c = \begin{cases} ReLU(\sum_k w_{l,k}(c)A_l^k) & \text{for } l = L \\ ReLU(\sum_k w_{l,k}(c)A_l^k) \odot \uparrow_{bi} \left(P_{l+1}^c, \frac{s_{l+1}}{s_l} \right) & \text{for } 1 \leq l \leq L - 1 \end{cases} \quad (7)$$

Representative examples are presented in Figure 13 to compare the conventional and ablated PCAM, when considering their high resolution saliency maps. We clearly observe that the ablated method tend to ignore some of class-relevant features, and focuses on a limited set of highly contrasted features (such as eye, mouth, or beak).

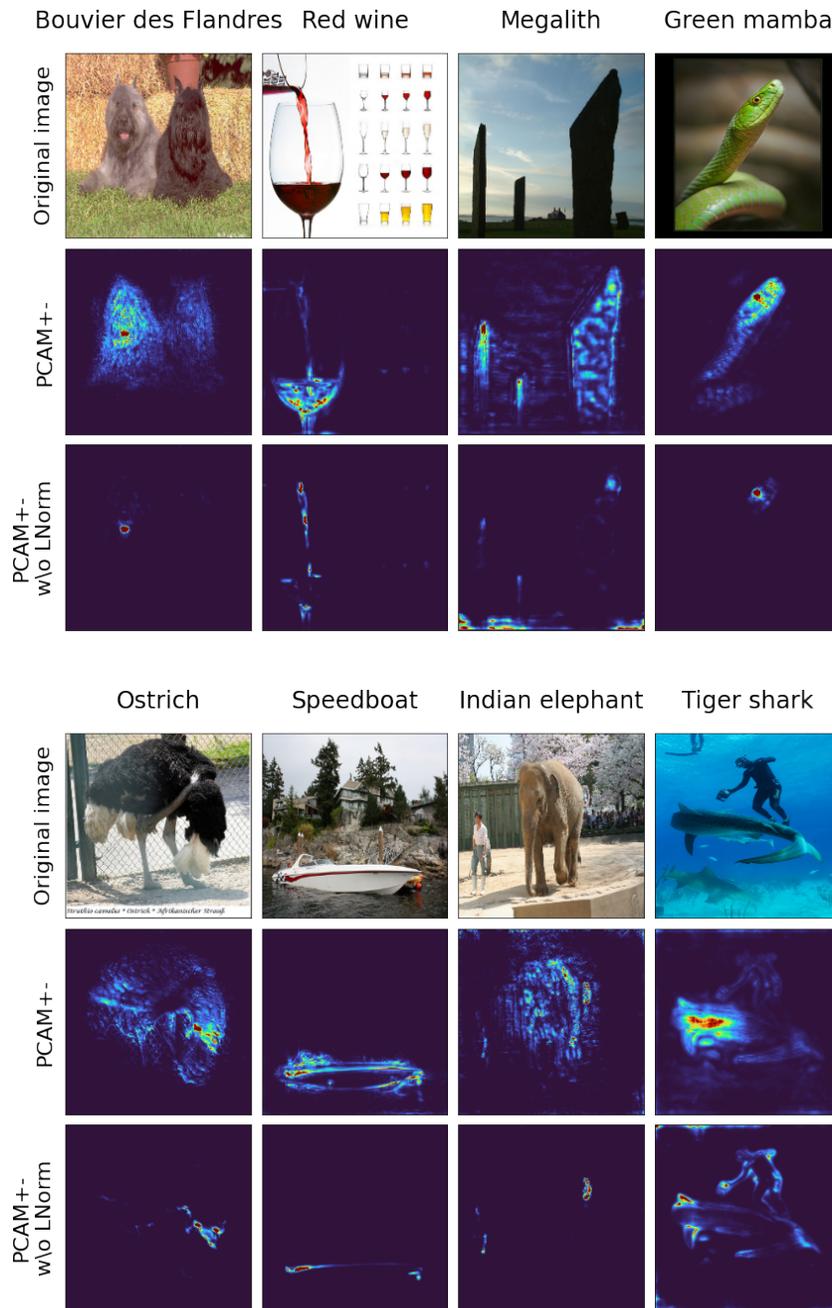


Figure 13: Visual comparison of PCAM[±] with and without LNorm. Without LNorm the visualisation tends to concentrate on very focal elements of the images like eyes, mouth,... Sometime some elements of the image that are not in object of the target class become also highlighted, like an object behind the elephant, or the diver next to the shark.

C MULTIPLE CLASSES COMPARISON

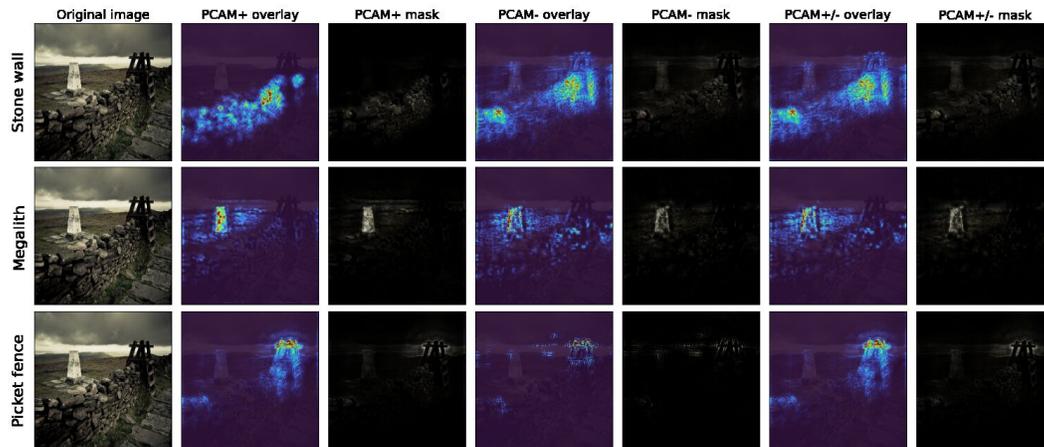


Figure 14: Comparison of multiple classes on an image with a stone wall, a megalith and pickets for the three Poly-CAM variants. The three classes are similarly separated for the three methods, while PCAM^+ being the more specific

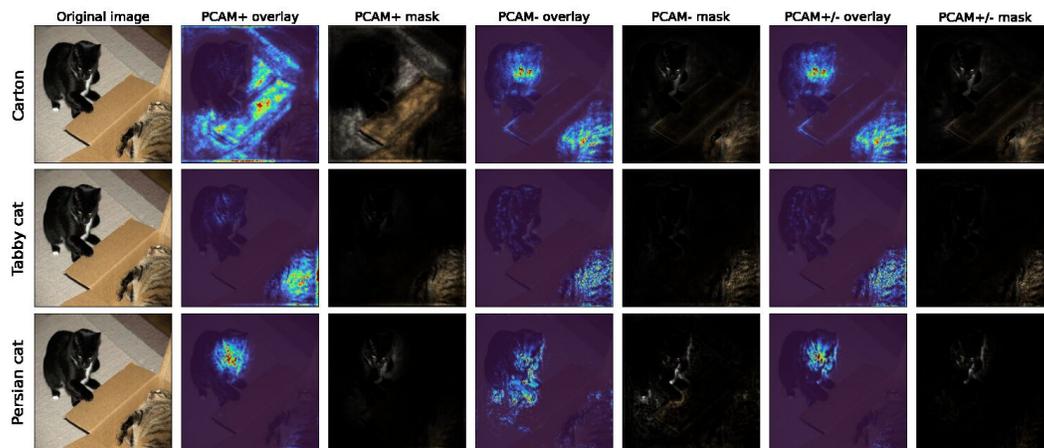


Figure 15: Comparison of multiple classes on an image with two different cats and a carton for the three Poly-CAM variants. The three methods correctly identify the two cats as different when using the Tabby cat and Persian cat classes, PCAM^+ correctly separate the carton while PCAM^- and PCAM^\pm fail for this class.

D EXAMPLES ON BONE X-RAY

D.1 CAST BIAS VISUALIZATION

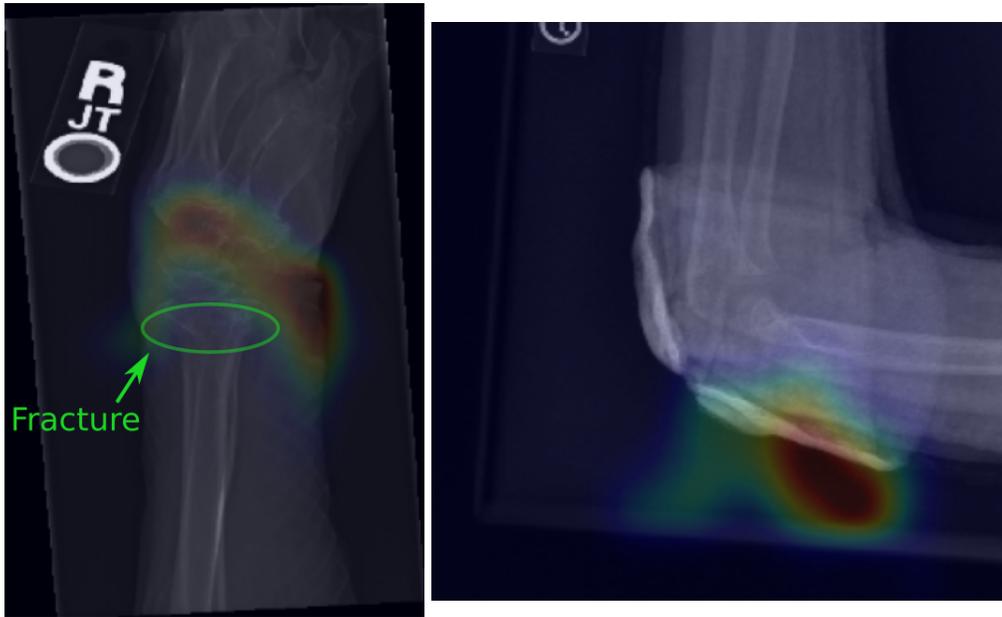


Figure 16: Visualization of a Class Activation Map (Zhou et al., 2016) for the pathological label on two bone X-Ray images from MURA dataset (Rajpurkar et al., 2017). Both images are labeled as pathological by the model, but only the left image show a bone fracture (manually annotated in green), the right image is clean of any fracture. The model seems to rely mostly on the plaster rather than on the absence of fracture to make a decisions. The model is a ResNet50 (He et al., 2016) initialised on ImageNet, trained on the MURA dataset (Rajpurkar et al., 2017) for 50 epochs with Adam optimizer, an initial learning rate of $6e-5$ with a cosine Annealing scheduler without restart, weight decay at $1e-5$. Images are resized to 320×320 with random rotation up to 15° during training.

D.2 IMPORTANCE OF BEING ACCURATE WHEN LOCALIZING SALIENCY

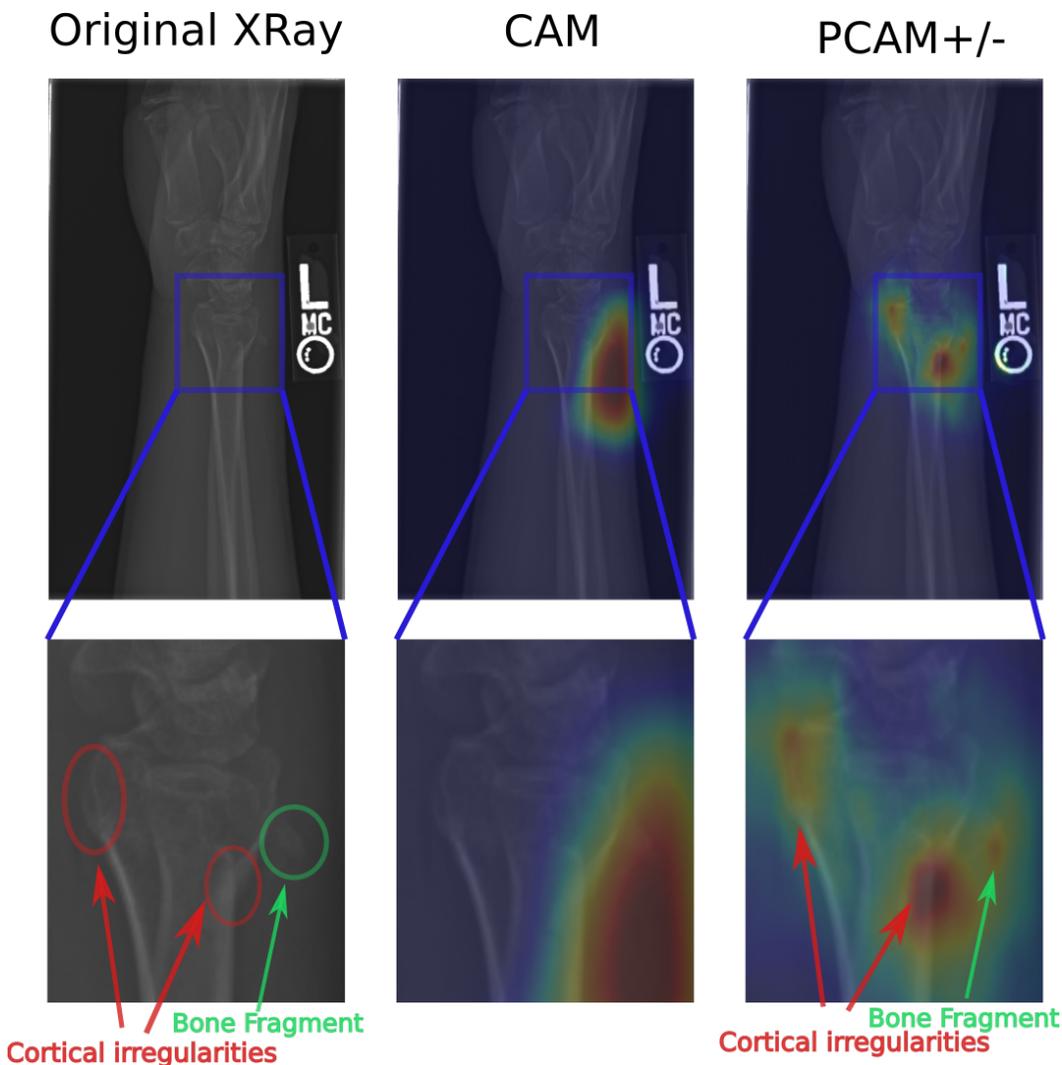


Figure 17: Visual comparison of Class Activation Map (Zhou et al., 2016) and $PCAM^{\pm}$ on a XRay of a bone fracture from MURA dataset (Rajpurkar et al., 2017), for the pathological class label. The bottom row is a zoom on the fracture area. Manual annotations for cortical irregularities and bone fragments (the main signs of the presence of a fracture on this XRay) are shown in red and green ovals. The Class Activation Map is not precise, it seems to include the bone fragment and the right cortical irregularity but due to the low resolution, the highlighted area is very large and go far from the fracture. In comparison, $PCAM^{\pm}$ highlight smaller structures and seems to identify correctly the cortical irregularities and the bone fragment on this image, being probably a greater help for a physician. The model is a ResNet50 (He et al., 2016) initialised on ImageNet, trained on the MURA dataset (Rajpurkar et al., 2017) for 50 epochs with Adam optimizer, an initial learning rate of $6e-5$ with a cosine Annealing scheduler without restart, weight decay at $1e-5$. Images are resized to 320×320 with random rotation up to 15° during training.

E FAITHFULNESS METRICS: SUPPLEMENTARY DATA

E.1 TABLE

Table 4: Faithfulness metrics for all methods: CAM-based, gradient and perturbation methods

Methods	VGG16			ResNet50		
	Insertion	Deletion	Ins-Del	Insertion	Deletion	Ins-Del
IntegratedGradient	0.41	0.10	0.31	0.52	0.16	0.36
InputXGrad	0.37	0.12	0.26	0.47	0.18	0.28
SmoothGrad	0.54	0.20	0.34	0.62	0.29	0.33
RISE	0.62	0.18	0.44	0.67	0.28	0.39
Occlusion	0.62	0.23	0.39	0.66	0.33	0.33
GradCAM	0.58	0.18	0.40	0.65	0.31	0.35
GradCAM++	0.57	0.19	0.38	0.65	0.31	0.34
SmoothGradCAM++	0.54	0.21	0.33	0.63	0.32	0.30
ScoreCAM	0.59	0.19	0.40	0.65	0.31	0.34
SSCAM	0.50	0.23	0.27	0.59	0.36	0.24
ISCAM	0.59	0.19	0.40	0.65	0.32	0.33
ZoomCAM	0.60	0.14	0.46	0.66	0.29	0.37
LayerCAM	0.58	0.14	0.44	0.65	0.30	0.35
PCAM ⁺ (ours)	0.58	0.17	0.41	0.67	0.29	0.38
PCAM ⁻ (ours)	0.60	0.16	0.45	0.66	0.27	0.39
PCAM [±] (ours)	0.61	0.15	0.46	0.67	0.28	0.39

Insertion (higher is better), deletion (lower is better) and Insertion-Deletion (higher is better) with VGG16 and ResNet50 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with gradient methods: Input X Gradient (Shrikumar et al., 2016), Integrated Gradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), perturbation methods: Occlusion (Zeiler & Fergus, 2014) and RISE (Petsiuk et al., 2018), and CAM methods: Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Smooth Grad-CAM++ (Omeiza et al., 2019), Zoom-CAM Shi et al. (2021) and Layer-CAM (Jiang et al., 2021). The grad methods are generally better in deletion but worst in insertion, translation by noisy saliency maps. Perturbations methods are similar to PCAM[±] in insertion, with a better result in VGG16 and lower result in ResNet50, but are less effective in deletion for both models. PCAM[±] give better average results, similar to Zoom-CAM for VGG16. For ResNet50 the results are better over all methods.

E.2 CURVES

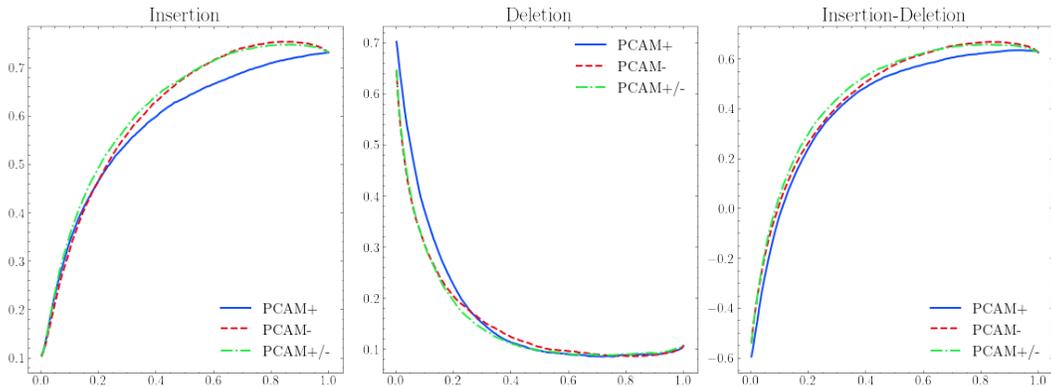


Figure 18: Faithfulness curves for Poly-CAM methods with VGG16 on the 2012 ILSVRC validation set. Comparison of the three variants of Poly-CAM.

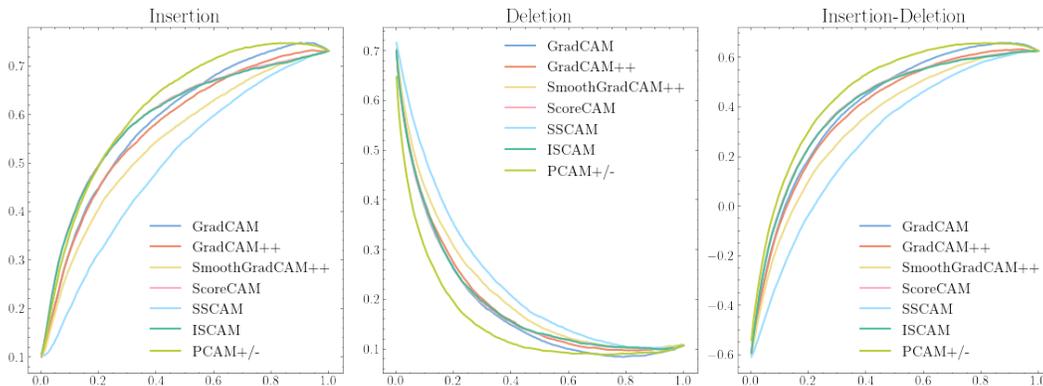


Figure 20: Faithfulness curves for CAM-based methods with VGG16 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Smooth Grad-CAM++ (Omeiza et al., 2019)

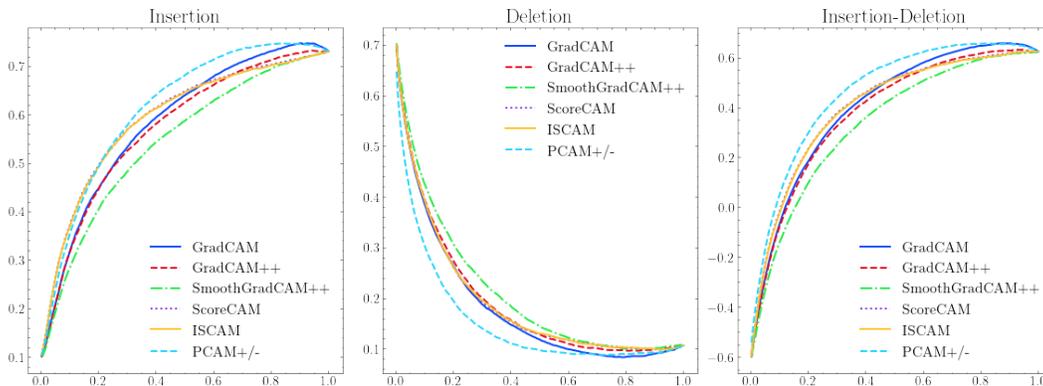


Figure 19: Faithfulness curves for CAM-based methods with VGG16 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Score-CAM (Wang et al., 2020b), IS-CAM (Naidu et al., 2020), Smooth Grad-CAM++ (Omeiza et al., 2019). SS-CAM (Wang et al., 2020a) was excluded in this graph for ease of view

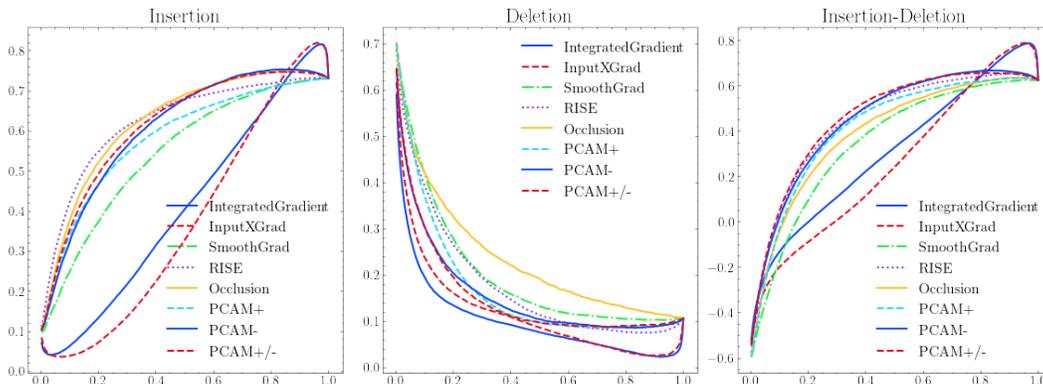


Figure 21: Faithfulness curves for gradient and perturbation methods with VGG16 on the 2012 ILSVRC validation set. Comparison of our PCAM[±] methods with gradient methods: Input X Gradient (Shrikumar et al., 2016), Integrated Gradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), and perturbation methods: Occlusion (Zeiler & Fergus, 2014) and RISE (Petsiuk et al., 2018)

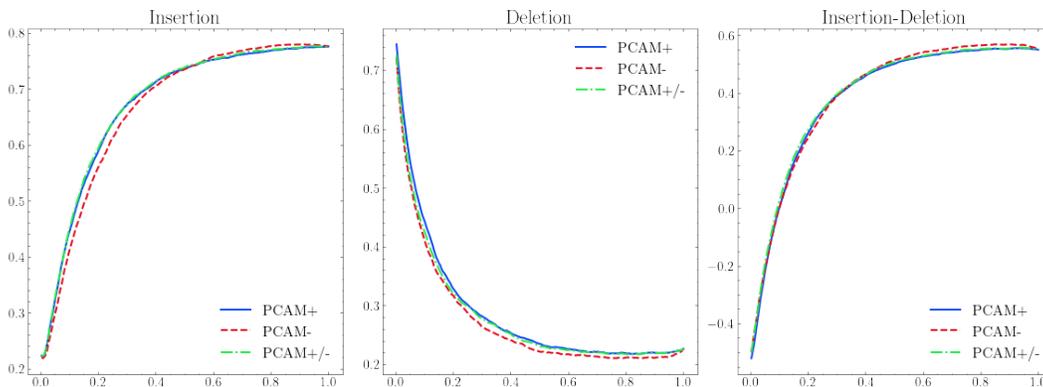


Figure 22: Faithfulness curves for Poly-CAM methods with ResNet50 on the 2012 ILSVRC validation set. Comparison of the three variants of Poly-CAM

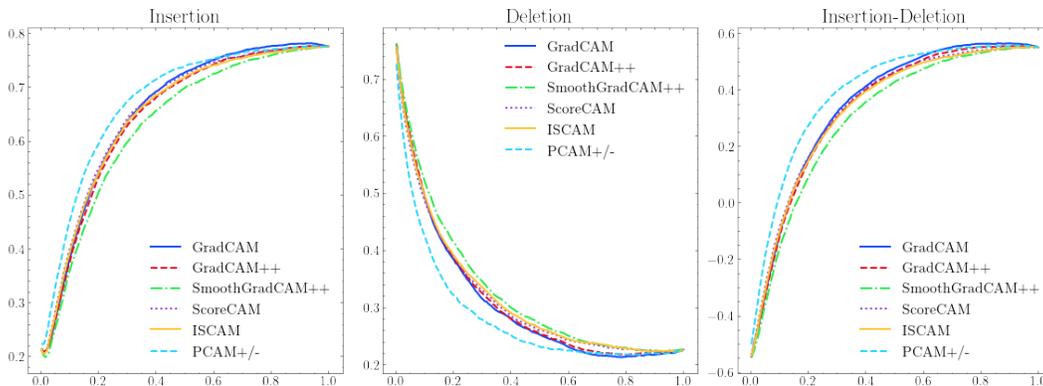


Figure 23: Faithfulness curves for CAM-based methods with ResNet50 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Score-CAM (Wang et al., 2020b), IS-CAM (Naidu et al., 2020), Smooth Grad-CAM++ (Omeiza et al., 2019). SS-CAM (Wang et al., 2020a) was excluded in this graph for ease of view

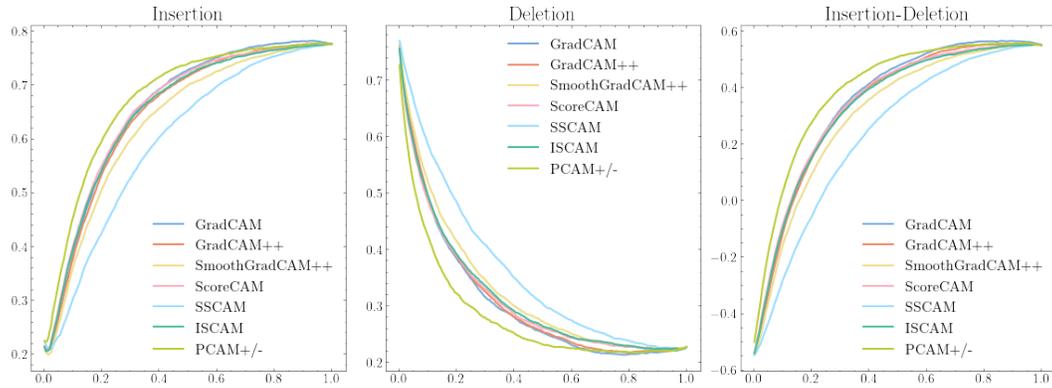


Figure 24: Faithfulness curves for CAM-based methods with ResNet50 on the 2012 ILSVRC validation set. Comparison of our Poly-CAM methods with Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Smooth Grad-CAM++ (Omeiza et al., 2019)

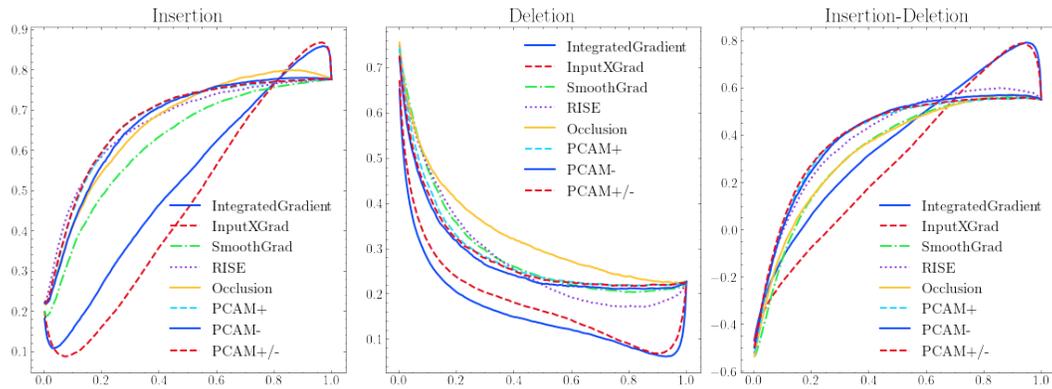


Figure 25: Faithfulness curves for gradient and perturbation methods with ResNet50 on the 2012 ILSVRC validation set. Comparison of our $PCAM^\pm$ methods with gradient methods: Input X Gradient (Shrikumar et al., 2016), Integrated Gradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), and perturbation methods: Occlusion (Zeiler & Fergus, 2014) and RISE (Petsiuk et al., 2018)

F ROBUSTNESS

Table 5: Sensitivity max table

Method	Sensitivity max	
	VGG16	ResNet50
IntegratedGradient	0.3576	0.5299
InputXGrad	0.6132	0.7225
SmoothGrad	5.6824	7.7777
RISE	0.7864	0.7841
Occlusion	2.5176	3.4378
GradCAM	0.0625	0.0212
GradCAM++	0.0525	0.0199
SmoothGradCAM++	0.5451	0.1594
ScoreCAM	0.0466	0.0193
ISCAM	0.0433	0.0334
ZoomCAM	0.0987	0.0485
LayerCAM	0.0937	0.0590
PCAM ⁺ (Ours)	0.0650	0.0262
PCAM ⁻ (Ours)	0.0837	0.0578
PCAM [±] (Ours)	0.0659	0.0659

Sensitivity max metric measures maximum sensitivity of an explanation using Monte Carlo sampling-base approximation (Yeh et al., 2019). Captum implementation was used with 10 perturbations per input and a epsilon radius of a L-Infinity ball set to 0.02 for sampling (defaults parameters from the implementation) (Kokhlikyan et al., 2020). The compared methods are the three Poly-CAM variants proposed in this paper (PCAM⁺, PCAM⁻, PCAM[±]), Zoom-CAM (Shi et al., 2021), Layer-CAM (Jiang et al., 2021), Grad-CAM (Selvaraju et al., 2017), Grad-CAM++ (Chattopadhyay et al., 2018), Smooth Grad-CAM++ (Omeiza et al., 2019), Score-CAM (Wang et al., 2020b), SS-CAM (Wang et al., 2020a), IS-CAM (Naidu et al., 2020), Input X Gradient (Shrikumar et al., 2016), IntegratedGradient (Sundararajan et al., 2017), SmoothGrad (Smilkov et al., 2017), Occlusion (Zeiler & Fergus, 2014), RISE (Petsiuk et al., 2018).

G SANITY CHECK

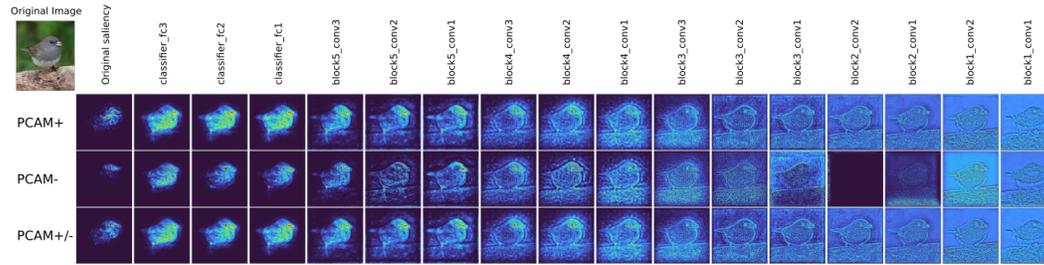


Figure 26: **Cascading randomization of VGG16.** Sanity check on Poly-CAM methods (Adebayo et al., 2018). Progression from left to right show a complete randomization of the VGG16 model, starting by the last layer up to the first layer. The methods are sensible to model randomization, which mean they pass this sanity check. It is interesting to note that the class specificity is lost rapidly after randomising the first classifier layer, then more and more features are lost while randomization progress up to the first layer of the network

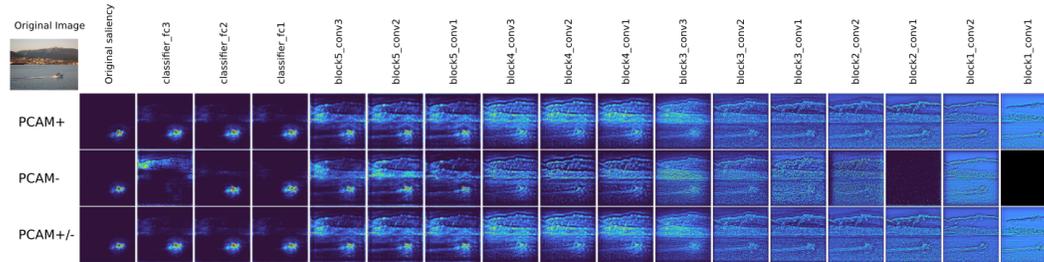


Figure 27: **Cascading randomization of VGG16.** Sanity check on Poly-CAM methods (Adebayo et al., 2018). Progression from left to right show a complete randomization of the VGG16 model, starting by the last layer up to the first layer.

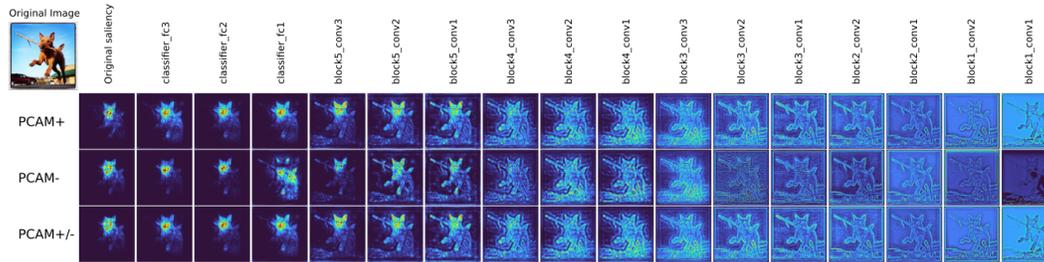


Figure 28: **Cascading randomization of VGG16.** Sanity check on Poly-CAM methods (Adebayo et al., 2018). Progression from left to right show a complete randomization of the VGG16 model, starting by the last layer up to the first layer.