Rethinking Positive Aggregation and Propagation of Gradients in Gradient-based Saliency Methods

Ashkan Khakzar [1] **Soroosh Baselizadeh** [1] **Nassir Navab** [1 2]

Abstract

Saliency methods interpret the prediction of a neural network by showing the importance of input elements for that prediction. A popular family of saliency methods utilize gradient information. In this work, we empirically show that two approaches for handling the gradient information, namely positive aggregation, and positive propagation, break these methods. Though these methods reflect visually salient information in the input, they do not explain the model prediction anymore as the generated saliency maps are insensitive to the predicted output and are insensitive to model parameter randomization. Specifically for methods that aggregate the gradients of a chosen layer such as GradCAM++ and FullGrad, exclusively aggregating positive gradients is detrimental. We further support this by proposing several variants of aggregation methods with positive handling of gradient information. For methods that backpropagate gradient information such as LRP, RectGrad, and Guided Backpropagation, we show the destructive effect of exclusively propagating positive gradient information.

1. Introduction

The quest for understanding the basis of neural networks' predictions is gaining momentum as the need for interpretability intensifies especially in sensitive application domains such as medicine, finance, policy, and law. One approach for understanding the predictions is using saliency methods (aka attribution methods) which assign each input element (*e.g.* pixels) its importance for the corresponding output prediction. There exist a plethora of proposed saliency methods, and thus raising the question "which explanation method to trust?". Visual evaluation of saliency maps has been shown to be unreliable as the method can generate human interpretable saliency maps but not explaining model behavior (Adebayo et al., 2018; Kindermans et al., 2019; Nie et al., 2018; Hooker et al., 2019). One way to evaluate saliency methods is by checking whether they satisfy certain desirable properties. Some of these properties are formulated as axioms (Sundararajan et al., 2017; Lundberg & Lee, 2017), such as efficiency (completeness), dummy (null-player), symmetry, sensitivity, and implementation invariance. It is therefore plausible to theoretically show whether methods abide by the axioms, though there is the caveat that while a method's formulation satisfies axioms, the required practical assumptions and approximations can break them (Sundararajan & Najmi, 2019).

It is also possible to formulate desirable properties as experiments (sanity checks) and empirically test whether a method violates those properties. Two important properties that a saliency method is required to satisfy are the sensitivity of the saliency map to the predicted output class (Nie et al., 2018), aka class-sensitivity, and the sensitivity of the saliency map to model parameter randomization (Adebayo et al., 2018), i.e. the saliency map for the model before randomizing its weights should be different from the saliency map after the model parameters are randomized. Not satisfying one of these two properties signifies that the method cannot explain model behavior.

In this work, we empirically show that two common approaches for handling gradient information in gradient-based saliency methods, methods that utilize gradient information to explain the model, can result in violating class-sensitivity and randomization-sensitivity. We show that in methods which aggregate gradients in a layer, if this aggregation is done only on positive gradients (e.g. via using a ReLU function) the desired properties are not satisfied. We show this phenomenon for FullGrad (Srinivas & Fleuret, 2019) and GradCAM++ (Chattopadhay et al., 2018), and other positive aggregation methods that we propose: positive aggregated GradCAM (Selvaraju et al., 2020) for various layers, positive aggregated Gradients for various layers, and a version of both where results from all layers are aggregated. Another approach that we investigate is inher-

[1]Technical University of Munich, Germany [2]Johns Hopkins University, USA. Correspondence to: Ashkan Khakzar <ashkan.khakzar@tum.de>.

2020 ICML Workshop on Human Interpretability in Machine Learning (WHI 2020).

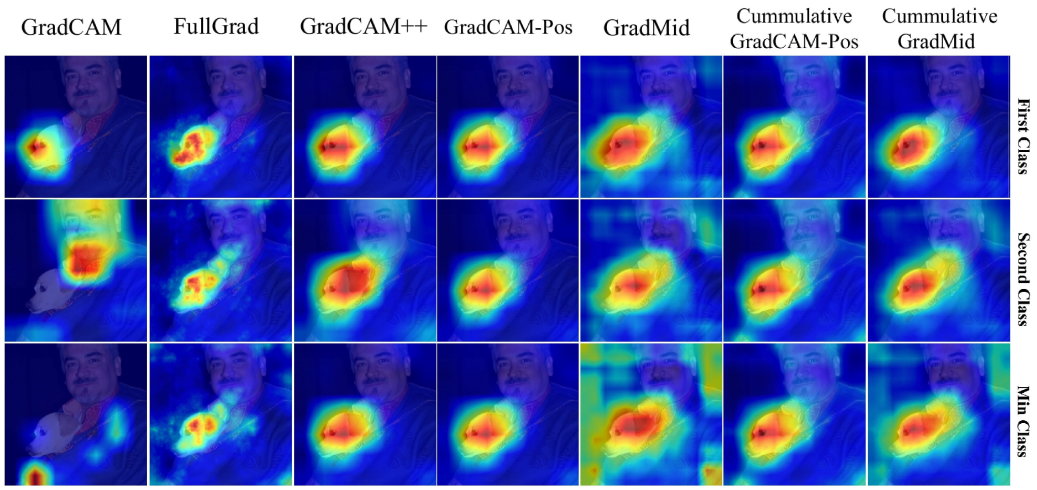

Figure 1. Saliency maps derived from different aggregation methods (columns) for different output class predictions (rows). All methods except GradCAM use positive aggregation and are not class-sensitive.

ent in methods that propagate gradient information such as LRP (Montavon et al., 2017; Bach et al., 2015) (and Excitation Backpropagation (Zhang et al., 2018) as it is equivalent to LRP-$\alpha 1\beta 0$), RectGrad (Kim et al., 2020), and Guided Backpropagation (Springenberg et al., 2015). We show that, when only positive information is backpropagated, the properties are not satisfied. We investigate these hypotheses empirically via ablative experiments. For sensitivity to model parameter randomization, we use the sanity checks of (Adebayo et al., 2018). For evaluating class sensitivity we use the class sensitivity metric proposed by (Rebuffi et al., 2020) and the pointing game of(Zhang et al., 2018; Fong et al., 2019). We also propose the restricted pointing game for evaluating class sensitivity.

(Rebuffi et al., 2020) propose a general formulation for aggregation methods, however, do not discuss how the choice of filtering functions (such as $|\cdot|$ or ReLU) affect them. It is generally observed that these aggregations are not class sensitive. In this work, we reveal the culprit and also show positive filtering affects the propagation methods.

2. Experimental Setup

The goal in experiments is to evaluate class-sensitivity and sensitivity to model parameter randomization. For this purpose we use the following experiments (we use the VGG-16 network (Simonyan & Zisserman, 2014)):

Pointing game. (Zhang et al., 2018; Fong et al., 2019). Saliency maps are generated for each class present in the image. For each class, if the maximum value in the generated saliency map is inside the ground truth mask the method has correctly pointed to the class. The accuracy is the ratio between correctly pointed classes and all pointing trials. As we aim to study class sensitivity we use a subset of PASCAL VOC07 (Everingham et al., 2015) dataset where there are *multiple object classes present in each image* (as defined in (Zhang et al., 2018)).

Restricted pointing game. As we are using a dataset with multiple object classes in each image, class-sensitive methods can generally achieve better accuracy in pointing game. However, the major problem with the pointing game experiment is that it does not disentangle class-sensitivity from localization. Therefore, comparing different methods with each other is difficult. We propose the restricted pointing game to tackle this problem. For each image and all its object classes, we only generate the saliency map for the max output class. The ground truth masks of all objects in the image are all compared against the max saliency map. If the resulting accuracy for the restricted pointing game is similar to the accuracy of the original pointing game, it can show that the saliency maps for different object classes are similar and are pointing to the same object class.

Class-sensitivity. (Nie et al., 2018) showed that for Guided Backpropagation and deconvolution methods, the generated saliency maps for different output classes in the same image are visually indiscernible. Motivated by this experiment, (Rebuffi et al., 2020) proposed a quantitative evaluation for class-sensitivity. For a given attribution method, saliency maps are generated for the max and min classes, and the correlation between them is computed. We report the average on all images in PASCAL VOC07.

Parameter randomization. (Adebayo et al., 2018) proposed a sanity check experiment for evaluating the methods' sensitivity to model parameter randomization. The similarity between saliency maps is measured by Spearman on HOG. The model parameters are increasingly randomized by reinitializing ($\sim \mathcal{N}(0, 0.01)$) the parameters of layers in a cascading fashion from the last layer to the first. We

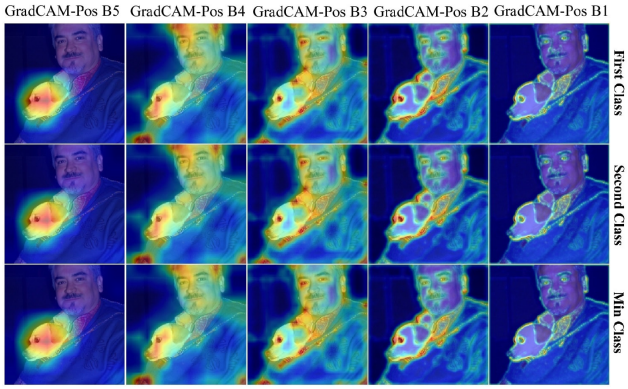

Figure 2. Saliency maps for different output predictions (rows) using GradCAM with positive aggregation (GradCAM_Pos) on different layers of VGG-16. Left column being the final layer.

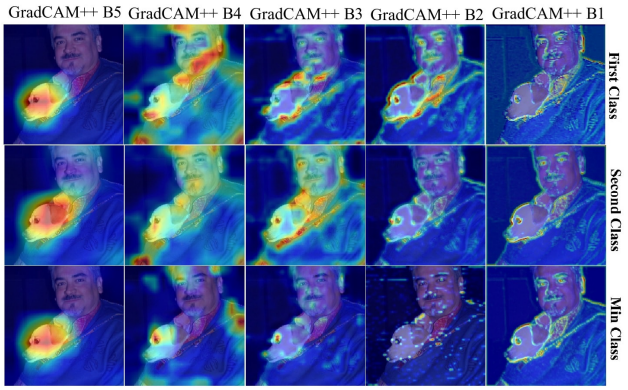

Figure 3. Saliency maps generated for different output predictions (rows) using GradCAM++ on different layers (columns) of VGG-16. Earlier layers presented on the right.

use a set of 1000 images of ImageNet (Deng et al., 2009).

3. Positive Aggregation

The locally connected structure of convolutions results in internal feature maps that retain the spatial information of the input, and therefore an attribution of contributions to these feature maps can be scaled to input space. Several methods utilize the gradient of the output with respect to activation units (GradCAM, GradCAM++) or biases (FullGrad) in a chosen layer. The gradient information in the chosen layer is aggregated across the channel dimension. In this section, we show how *positive aggregation* — using an absolute function (FullGrad) or using ReLU (GradCAM++) before the aggregation — can result in feature maps that recover salient objects in the image without being related to the output prediction being explained.

We also show that simply adding an absolute function to the formulation of GradCAM (Selvaraju et al., 2020) before aggregation can result in the same behavior. The positive aggregation allows for GradCAM to be applicable to earlier layers as well, but we show that it is only recovering salient feature information without any regard for the prediction. The original formulation of GradCAM is as follows:

$$S_{\text{GradCAM}} = \mathrm{ReLU}(\sum_k w_k A^k) \tag{1}$$

where A^k is the activation map at channel k, f is the output being explaind, and

$$w_k = \frac{1}{Z} \sum_i \sum_j \frac{\partial f}{\partial A^k_{ij}} \tag{2}$$

where Z is the number of activation units in A^k. This method is well justified for the final convolutional layer, as such linear approximation of output with respect to activations, which is the basis for the aggregation of feature maps in GradCAM holds stronger for the last layer than previous layers (in a network where CAM (Zhou et al., 2016) is applicable, linear aggregation exactly derives the output). However, if GradCAM is computed for earlier layers, the resulting saliency maps are arbitrary as reported in (Selvaraju et al., 2020; Rebuffi et al., 2020). During the aggregation phase, GradCAM++ uses ReLU on gradients before the summation directly in the main formulation of the method as follows:

$$S_{\text{GradCAM++}} = \sum_k w_k A^k \tag{3}$$

where,

$$w_k = \sum_i \sum_j \alpha^k_{ij} \, \mathrm{ReLU}(\frac{\partial f}{\partial A^k_{ij}}) \tag{4}$$

and α^k_{ij} is a coefficient computed for each activation unit (please refer to (Chattopadhay et al., 2018)).

Fullgrad's (Srinivas & Fleuret, 2019) main formulation distributes the contribution to the output prediction between all inputs to the network (input and mid-level biases) and there is no ReLU or absolute summation. However such formulation cannot be visualized as a saliency map. Thus an aggregation phase is added for CNN visualizations, and an absolute function is used within the post-processing ψ:

$$S_{\text{FullGrad}} = \psi(\frac{\partial f}{\partial x} \odot x) + \sum_{l \in L} \sum_k \psi(\frac{\partial f}{\partial A^k} \odot b^l) \tag{5}$$

where x is the input image, and b^l stands for the bias parameters of layer l. The absolute operation is added to visualize only the magnitude of importance while ignoring the sign. However, ignoring the sign results in aggregating the gradients related to all features in the image (compare this

Table 1. Pointing game on aggregation methods

	ORIGINAL	RESTRICTED
GRADCAM	74.1	48.9
FULLGRAD	58.4	50.5
GRADCAM++	66.2	54.1
GRADCAM++ B4	36.3	37.1
GRADCAM++ B3	35.2	36.4
GRADCAM++ B2	38.0	39.2
GRADCAM++ B1	40.4	40.3
GRADCAM_POS	60.4	58.1
GRADCAM_POS B4	37.6	37.8
GRADCAM_POS B3	31.8	31.6
GRADCAM_POS B2	40.6	40.7
GRADCAM_POS B1	40.7	40.6
GRADMID	63.9	56.3
GRADMID B4	65.4	56.3
GRADMID B3	60.0	53.8
GRADMID B2	53.8	48.8
GRADMID B1	51.8	44.5
CUMULATIVE_GRADCAM	58.1	56.2
CUMULATIVE_GRADMID	64.1	56.3

to CAM/GradCAM where the summation is done without $|\cdot|$ because the linear combination of activations in the last layer recovers the output).

Initially, we show that if we change the formulation of GradCAM by adding an absolute function during aggregation, which we call GradCAM_Pos, the resulting saliency maps for all layers show salient image features, however, it seems that these are merely silhouettes of salient image features and we observe that these features are not relevant to the predicted class. A visual example is presented in Fig. 2, 1. The maps are computed for different layers of VGG. We use the convolutional maps at different resolution blocks, and represent them as GradCAM_Pos plus block number, where B1 for instance represents the first block. It is observed that as we move toward earlier layers the generated maps recover low-level features of objects such as edges. *The maps on earlier layers should not be interpreted as 'high resolution', as they are not attribution or saliency maps anymore.* It is also reported by (Rebuffi et al., 2020) that earlier layers result in class-insensitive maps in aggregation methods, however, we are considering the role positive aggregation plays. *With positive aggregation, even in the final layer, maps highlight salient image features but for both classes present in the image.* This observation is further supported in the pointing game experiments in Table 1 as the original and restricted games accuracies are similar. Class-sensitivity metric in Table 2 shows that for all layers, except the final layer, the correlation is close to 1. For the final layer the correlation is lower but still *significantly higher than GradCAM with absolute function (0.69 vs. 0.03), showing how detrimental the effect of positive aggregation is.* The positive aggregation in this case also has

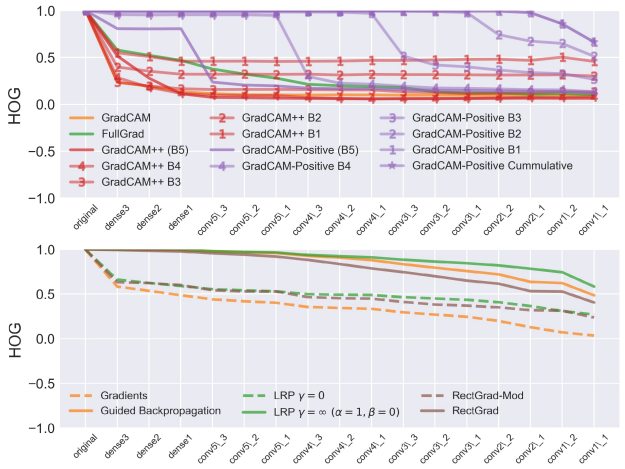

Figure 4. Sanity checks for sensitivity to parameter randomization for aggregation (Up) and propagation (Down) methods.

a significant effect on the method's sensitivity to randomization. Fig. 5 and 4 shows that *these method's saliency maps do not change after randomization, further supporting the claim that methods are recovering image features.*

As stated, the ability to generate saliency maps that look interpretable on earlier layers for GradCAM can be achieved via positive aggregation. GradCAM++ also uses positive aggregation and its visual examples for the last layer and preceding layers are presented in Fig 3 and Fig. 1. The observations for GradCAM_Pos are visible here as well. The results on the restricted pointing game and original are equivalent when experiments are done on all layers except the final layer. However, correlations (Fig. 2) are better than the GradCAM_Pos case, but still significantly high compared to GradCAM. These lower correlations are expected when looking at the visual results as there are arbitrary highlighted areas in the maps as well, but the method still highlights all the salient features in the image, which is confirmed by pointing game experiments (Table 1). In parameter randomization in Fig. 4 and 5 *the method gets more insensitive when applied to earlier layers.*

Fullgrad has one major difference in formulation compared to the other methods. The method has an extra aggregation step where maps from all *layers* are aggregated. It is expected that is aggregations compounds the destructive effect of positive aggregation within each layer. In this section, we study the effect of aggregation on all layers. First, we propose an all-layer aggregation on GradCAM_Pos and call the method CumulativeGradCAM. As can be seen in Table 2 and 1 the method is not class sensitive. The result seems to be bounded by the worst-case early layer and best case final layer for GradCAM_Pos. This also explains why FullGrad is less class sensitive than GradCAM++. Cumu-

Table 2. Class-sensitivity metric on aggregation methods

	SPEARMAN HOG
GRADCAM	0.03
FULLGRAD	0.48
GRADCAM++	0.37
GRADCAM++ B4	0.23
GRADCAM++ B3	0.44
GRADCAM++ B2	0.58
GRADCAM++ B1	0.65
GRADCAM_POS	0.69
GRADCAM_POS B4	0.94
GRADCAM_POS B3	0.98
GRADCAM_POS B2	0.99
GRADCAM_POS B1	0.99
GRADMID	0.29
GRADMID B4	0.35
GRADMID B3	0.32
GRADMID B2	0.34
GRADMID B1	0.43
CUMULATIVE_GRADCAM	0.92
CUMULATIVEMID	0.36

Table 3. Pointing game on propagation methods

	ORIGINAL	RESTRICTED
GUIDED BACKPROP	48.7	47.5
GRADIENTS	52.7	45.1
RECTGRAD	51.2	50.0
RECTGRAD_MOD	59.5	53.8
LRP-$\gamma = \infty$	50.6	50.0
LRP-0	51.1	44.2

lativeGradCAM uses the model's activations, therefore we also propose a variant where we only aggregate gradients. When done on one layer we call this as GradMid and when done on all layers we call it CumuluativeGradMid. This is to show that *positive aggregation on gradients alone without any weighting can also generate visually interpretable feature maps, though similar to other cases the visualization only shows salient image features irrespective of the model and the output.* The results for these all-layer summation methods are presented alongside all the aforementioned methods in Fig. 1. *It is also observed in Fig. 4 that Fullgrad is insensitive to randomization considerably, and the CumulativeGradCAM is the least sensitive.*

4. Positive Propagation

Gradient information backpropagation towards the input from the the output being explained is the underlying principle of a class of gradient-based saliency methods. In this section we study the effect of rules that propagate only positive gradient information. Considering a ReLU based neural network, let a_i^l denote an activation unit i in layer l and

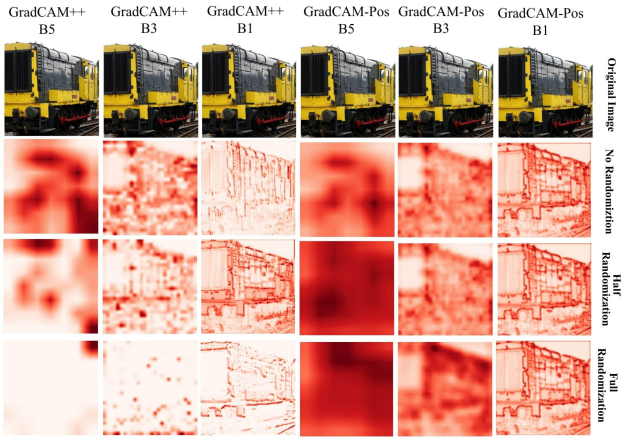

Figure 5. From left: Saliency maps for GradCAM++ B5, B3 and B1 layers. followed by GradCAM_Pos B5, B3, B1. From top: Original image, saliency map on original model, half randomized model and fully randomized model.

the flowing gradient into a_i^l be R_i^{l+1} and let the gradient backpropagated from a_i^l be R_i^l. In normal gradient backpropagation $R_i^l = \mathbb{I}(a_i^l)R_i^{l+1}$, where $\mathbb{I}(.)$ is the indicator function. Guided Backpropagation rule is defined as:

$$R_i^l = \mathbb{I}(a_i^l R_i^{l+1} > 0)R_i^{l+1} \tag{6}$$

and RectGrad uses the following propagation rule:

$$R_i^l = \mathbb{I}(a_i^l R_i^{l+1} > \tau)R_i^{l+1} \tag{7}$$

where τ is a threshold value.

Layerwise Relevance Propagation (LRP) has several propagation rules and a widely adopted rule is the LRP-$\alpha 1\beta 0$, which is equivalent (Samek et al., 2019) to Deep Taylor $\alpha 1\beta 0$ (Montavon et al., 2017), LRP z+- rule and Excitation-Backprop (Zhang et al., 2018) methods. This rule is also equivalent to LRP-γ with $\gamma = \infty$. LRP-γ is introduced to favor the effect of positive contributions over negative ones and is defined as follows:

$$R_j = \sum_k \frac{a_j(w_{jk} + \gamma w_{jk}^+)}{\sum_j a_j(w_{jk} + \gamma w_{jk}^+)} R_k \tag{8}$$

where R_j is the relevance of neuron j for prediction and R_k is relevance of neuron k in the next layer. The controlling parameter for considering positive contributions is γ. As γ is increased the effect is more pronounced. In this work we consider the limits $\gamma = \infty$ and $\gamma = 0$ (LRP-0). For all these methods, we show the effect of positive gradient information backpropagation by running the experiments in section 2 on these methods and their counterpart versions where there is no bias towards positive gradients. The counterpart version for Guided Backpropagation is changing the rule to $R_i^l = \mathbb{I}(a_i^l > 0)R_i^{l+1}$ so that

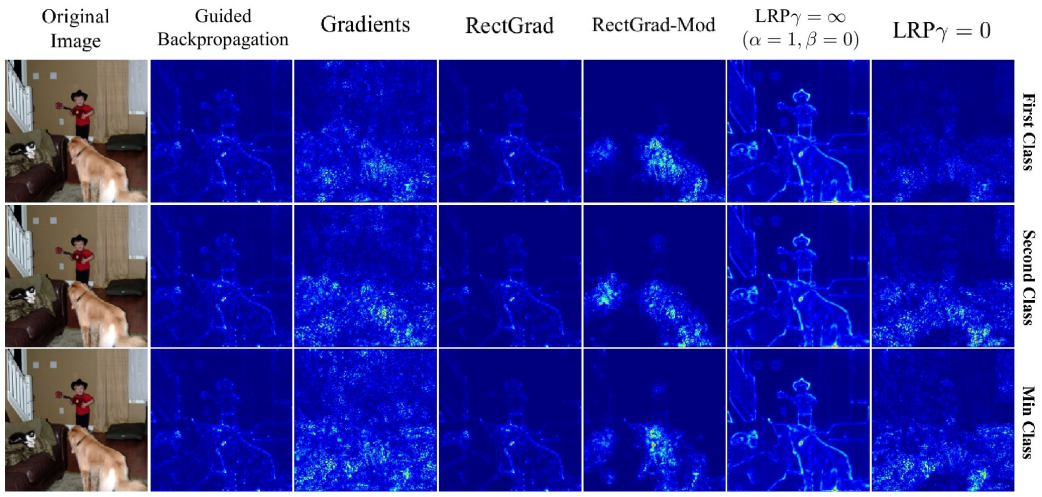

Figure 6. Saliency maps generated for different output predictions (rows) using different propagation methods.

Table 4. Class-sensitivity metric on propagation methods

	SPEARMAN HOG
GUIDED BACKPROP	0.99
GRADIENTS	0.44
RECTGRAD	0.99
RECTGRAD_MOD	0.72
LRP-$\gamma = \infty$	0.99
LRP-0	0.59

both positive and negative information is propagated and evidently this is equivalent to unmodified gradients. The counterpart version for RectGrad is achievable by using an absolute function in the formulation so that large positive and negative gradients can both flow backwards, therefore the rule is modified to $R_i^l = \mathbb{I}(\left|a_i^l R_i^{l+1}\right| > \tau) R_i^{l+1}$, which we call RectGrad_Mod. Setting $\gamma = 0$ in LRP-γ removes the bias towards positive information backpropagation.

We first investigate the effect of positive backpropagation on the sensitivity of the methods to model parameter randomization. (Adebayo et al., 2018; Nie et al., 2018) show that the Guided Backpropagation method is insensitive to parameter randomization. (Khakzar et al., 2019) report insensitivity to randomization of RectGrad using sanity checks. Our sanity check experiment reported in Fig. 4 further shows that *RectGrad and LRP-$\gamma = \infty$ are also insensitive to the same extent as Guided Backpropagation*. As the resulting saliency maps resemble the images before and after randomization, it points to the fact that such *positive gradient information backpropagation recovers salient features in the image regardless of the model*. The results for the counterpart versions of all methods signify that these counterpart methods are sensitive to randomization.

The pointing game experiment in Table 3 shows that the resulting accuracy on the original is similar to accuracy on restricted pointing games are for the positive propagation methods. *This signifies that these methods are pointing to the same salient object class in the images for different output predictions*. On the other hand, for the counterpart versions, the accuracy drops in the restricted version, implying that indeed the saliency maps for different outputs differ. However, it still seems that these counterpart methods (e.g. Gradients) are also class-insensitive for many images (Fig. 6), and effect which needs to be studied in future, especially when other works (Rebuffi et al., 2020; Nie et al., 2018) have contradicting statements in this regard.

The class-sensitivity metric in Table 4 shows that there is a significant similarity between saliency maps of different outputs for the positive backpropagation methods. This further confirms the observation that *these methods are recovering salient image features rather than explaining the output prediction.* It is also observed that for counterpart methods, the saliency maps change for different output predictions (Table 3, 4), though it does not mean that they are class sensitive (Fig. 6). *Nevertheless, positive propagation makes these methods class-insensitive.*

5. Conclusion

In this work, we empirically showed that positive aggregation or propagation of gradients in gradient-based saliency methods results in saliency maps that recover salient image features regardless of the model and output prediction, and changes the methods towards being class-insensitive and insensitive to randomization.

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 2018.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 2015. ISSN 19326203. doi: 10.1371/journal.pone.0130140.

Chattopadhay, A., Sarkar, A., Howlader, P., and Balasubramanian, V. N. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018*, 2018. ISBN 9781538648865. doi: 10.1109/WACV.2018.00097.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, January 2015.

Fong, R., Patrick, M., and Vedaldi, A. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2950–2958, 2019.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 9737–9748, 2019.

Khakzar, A., Baselizadeh, S., Khanduja, S., Kim, S. T., and Navab, N. Explaining neural networks via perturbing important learned features. *arXiv preprint arXiv:1911.11081*, 2019.

Kim, B., Seo, J., Jeon, S., Koo, J., Choe, J., and Jeon, T. Why are Saliency Maps Noisy? Cause of and Solution to Noisy Saliency Maps. 2020. doi: 10.1109/iccvw.2019.00510.

Kindermans, P. J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. The (Un)reliability of Saliency Methods. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2019. doi: 10.1007/978-3-030-28954-6_14.

Lundberg, S. M. and Lee, S. I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017.

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K. R. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 2017. ISSN 00313203. doi: 10.1016/j.patcog.2016.11.008.

Nie, W., Zhang, Y., and Patel, A. B. A theoretical explanation for perplexing behaviors of backpropagation-based visualizations. In *35th International Conference on Machine Learning, ICML 2018*, 2018. ISBN 9781510867963.

Rebuffi, S.-A., Fong, R., Ji, X., and Vedaldi, A. There and Back Again: Revisiting Backpropagation Saliency Methods. apr 2020. URL http://arxiv.org/abs/2004.02866.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Muller, K.-R. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. *Lecture Notes in Computer Science*, 2019. doi: 10.1007/978-3-030-28954-6.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *International Journal of Computer Vision*, 2020. ISSN 15731405. doi: 10.1007/s11263-019-01228-7.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. In *3rd International Conference on Learning Representations, ICLR 2015 - Workshop Track Proceedings*, 2015.

Srinivas, S. and Fleuret, F. Full-Gradient Representation for Neural Network Visualization. Technical report, 2019.

Sundararajan, M. and Najmi, A. The many shapley values for model explanation. *arXiv preprint arXiv:1908.08474*, 2019.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *34th International Conference on Machine Learning, ICML 2017*, 2017. ISBN 9781510855144.

Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Learning Deep Features for Discriminative Localization. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.319.