

# Now you see me! A framework for obtaining class-relevant saliency maps

Nils Philipp Walter  
CISPA Helmholtz Center  
for Information Security  
nils.walter@cispa.de

Jilles Vreeken  
CISPA Helmholtz Center  
for Information Security  
jv@cispa.de

Jonas Fischer  
Max Planck Institute  
for Informatics  
jofische@mpi-inf.mpg.de

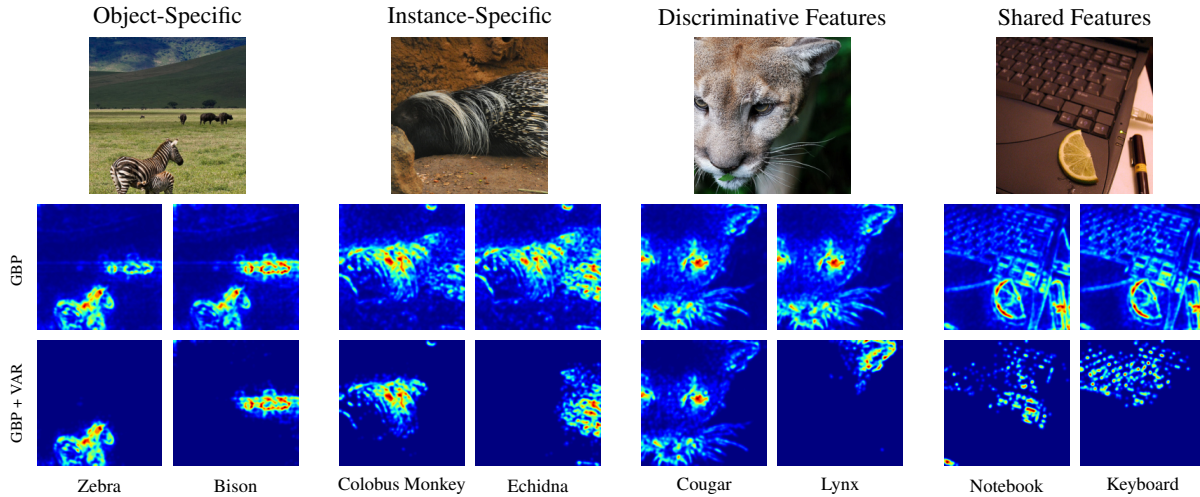


Figure 1. *Properties of VAR Attributions.* VAR attributions are **object-specific** and visually ground correct target objects. VAR attributions are **instance-specific**, identifying features that are relevant on a by-part-basis. VAR attributions are **class-discriminative**, yielding features that separate closely related classes. VAR attributions reveal **shared concepts** between closely related classes. In contrast, vanilla attribution methods (here GBP) do not show these properties.

## Abstract

Neural networks are part of daily-life decision-making, including in high-stakes settings where understanding and transparency are key. Saliency maps have been developed to gain understanding into which input features neural networks use for a specific prediction. Although widely employed, these methods often result in overly general saliency maps that fail to identify the specific information that triggered the classification. In this work, we suggest a framework that allows to incorporate attributions across classes to arrive at saliency maps that actually capture the class-relevant information. On established benchmarks for attribution methods, including the grid-pointing game and randomization-based sanity checks, we show that our framework heavily boosts the

performance of standard saliency map approaches. It is, by design, agnostic to model architectures and attribution methods and now allows to identify the distinguishing and shared features used for a model prediction.

## 1. Introduction

Neural Networks are prime models for decision-making, yet are inherently opaque when it comes to their reasoning. Especially in high-stakes prediction, but also in a more general context, there is, however, a growing need for more transparent reasoning that provides the user with an understanding of the model’s decision. In the context of Explainable Artificial Intelligence (XAI), explanations that describe *which input features* are used for a prediction, for example specific image regions, were

developed. This group of approaches is coined *attribution methods* as they attribute importance of input features to the output of a model and is among the most popular in XAI, already in use in high-stakes domains such as medical imaging [6].

However, it recently has been shown that these post-hoc explanation have severe shortcomings; while the features often seem sensible, they turned out to not properly model the class-relevant features used by the network [27]. Thus, the explanations failed in providing the desired information of which input features were actually *relevant* for the classification. Here, we propose an attribution method-agnostic as well as architecture-agnostic framework that adapts any given gradient-based attribution score to reflect the information that is relevant for distinguishing classes, providing saliency maps for image classification that reveal the *discriminative and locally shared features in the input* used by a network to make a prediction. In particular, we propose a simple yet powerful approach considering attributions across different classes at each spatial location.

Our framework for Visualizing Actually Relevant features (VAR) is easy to adapt to any gradient-based attribution score with little computational overhead. The resulting saliency maps on standard vision benchmarks qualitatively provide much more focused and class-relevant information across different models, including CNN and Transformer-based architectures (see Fig. 2). Quantitatively, VAR drastically improves the ability to retrieve correctly localized attributions in the grid pointing game, which was previously a culprit of existing work [27]. We further test our framework on insertion ablations [17], and show that VAR attributions are more robust to randomization-based sanity check [1]. Our framework thus equips attribution methods to identify the distinguishing features that a network uses for prediction, providing explanations that now properly describe *which input features* were class-relevant.

## 2. Related Work

Research in XAI gave rise to three main approaches to discover prediction-relevant input features. Perturbation techniques such as RISE [26], extremal perturbations [11], and SHAP [23] probe model behavior by modifying inputs. While effective, these methods are computationally expensive, often requiring multiple forward passes and significant processing time and often consider input features independently. Approximation techniques, including LIME [29] and FLINT [25], create interpretable surrogate models to mimic complex networks locally. Such surrogates can, however, largely dif-

fer from the target model reasoning, limiting their ability to accurately capture the prediction dependencies.

The third category, activation- and gradient-based methods, strikes a balance between efficiency and fidelity by leveraging the network’s internal computation graph. The simplest of these, standard gradient visualization [34], treats gradients of network outputs with respect to input features as importance scores, further multiplying by input magnitudes to receive more accurate saliency maps (Input×Gradient). Integrated Gradients [37] offers attributions with theoretical justifications by accumulating gradients along a straight-line path from baseline to input. For CNN-specific visualizations, GradCAM [31] generates activation maps by combining feature maps weighted by their average gradients and upsampling. Layer-wise Relevance Propagation (LRP) [3] defines custom activation-based propagation rules for distributing relevance backwards through the network and requires architecture-specific adaptations [24]. Similarly, DeepLift [32] uses reference activations to determine neuron importance through custom backpropagation procedures.

Counterfactual explanation techniques focus on differentiating between individual classes. Many approaches [2, 8, 16, 20, 22, 41] generate counterfactual inputs through complex generative models. Specific approaches generate [38] or use search algorithms to find [12] contrastive example inputs. Two approaches focus specifically on extracting what is the different, contrastive, information between classes [13, 39]. In contrast, here, we want to discover any input feature that is relevant for classification, including object- and instance-specific features, but also features that are shared across classes, which are commonly used in networks and would be lost with contrast alone.

## 3. Overview of Attribution Methods

Here we provide a brief formal description of our setting and attribution methods. We consider an input as a vector  $x \in \mathcal{I}$ , in our setting, the input is typically an image of height  $H$ , width  $W$  and  $d$  channels, so  $\mathcal{I} = \mathbb{R}^{H \times W \times d}$ . We describe a model as a function  $S : \mathcal{I} \rightarrow \mathbb{R}^C$ , where  $C$  is the number of classes in the classification problem. The final classification is usually performed as an argmax over  $S(x)$ . An attribution method provides a saliency map  $\mathcal{H} : \mathcal{I} \times S \times \{1, \dots, C\} \rightarrow \mathcal{I}'$  that for an input, a model, and optionally a target class provides an explanation of the same shape as the input, attributing scores to each feature  $i$  in the input describing in how far  $S$  uses  $x_i$  for the classification (or target class). With a slight abuse of notation, we use  $\mathcal{H}_c$

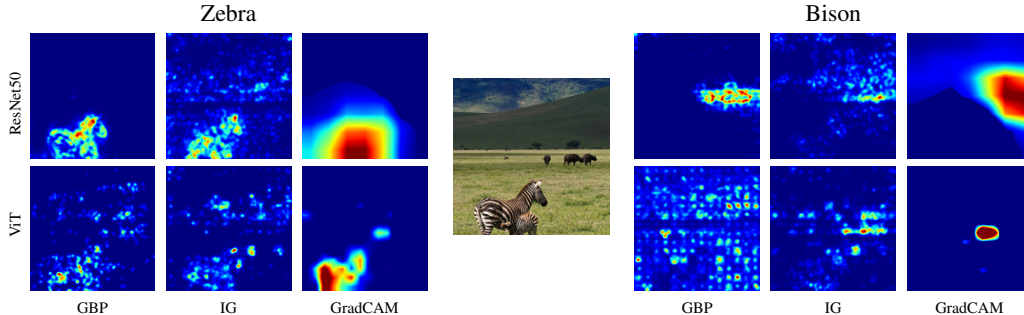


Figure 2. *VAR is generally applicable.* We show VAR augmenting different attribution methods (columns) and architectures (rows) for detecting zebras (left) resp. bison (right) in the original image (middle), using Gradient Backpropagation (GBP), Integrated Gradients (IG), and GradCAM, for respectively ResNet50 and ViT.

to denote both the explanation method and the resulting attribution map for class  $c$ . We next describe specific instantiations of  $\mathcal{H}$ .

**The gradient explanation** for an input  $x$  is  $\mathcal{H}_{\text{grad}}(x, S, c) = \frac{\partial S_c}{\partial x}$  [4, 10, 34]. The gradient quantifies how much a small change in each input feature would affect the predictions  $S_c(x)$ .

**Input  $\times$  Gradient (IxG)** addresses “gradient saturation” in the standard gradient explanation through the element-wise product of the input and the gradient, denoted  $x \odot \frac{\partial S_c}{\partial x}$ , which can reduce visual diffusion and highlight important features more effectively [32].

**Integrated Gradients (IG)** also addresses gradient saturation by integrating gradients along a path from a baseline to the input [37]. IG for an input  $x$  is defined as  $\mathcal{H}_{\text{IG}}(x, S, c) = (x - \bar{x}) \times \int_0^1 \frac{\partial S_c(\bar{x} + \alpha(x - \bar{x}))}{\partial x} d\alpha$ , where  $\bar{x}$  is a “baseline input” that represents the absence of a feature in the original input  $x$ .

**Guided Backpropagation (GBP)** [36] modifies standard backpropagation by introducing a non-negative constraint on gradients. As such, GBP prevents negative gradients from propagating.  $\mathcal{H}_{\text{GBP}}$  adjust the backpropagation for ReLU layers to  $\delta_{\text{GBP}}^l = \delta^{l+1} \cdot \mathbb{1}_{\delta^{l+1} > 0} \cdot \mathbb{1}_{x^l > 0}$  and for non-ReLU layers to  $\delta_{\text{GBP}}^l = \delta^{l+1} \cdot \mathbb{1}_{\delta^{l+1} > 0}$ , where  $\mathbb{1}$  is the indicator function. This filtering ensures that only signals that positively contribute to a classification is propagated.

**GradCAM** [31] computes class-specific attribution maps using the feature maps from the last convolutional layer. For a feature map  $A^k$  of the last convolutional layer, the attribution is defined as  $\mathcal{H}_{\text{GradCAM}}(x, S, c) = \text{ReLU}(\sum_k \alpha_c^k A^k)$ , where  $\alpha_c^k = \frac{1}{Z} \sum_i \sum_j \frac{\partial S_c}{\partial A_{ij}^k}$  are the importance weights computed by global average pooling of the gradients.

**Guided GradCAM** combines GBP and GradCAM i.e. the element-wise product  $\mathcal{H}_{\text{GuidedGradCAM}}(x, S, c) =$

$\mathcal{H}_{\text{GBP}}(x, S, c) \odot \mathcal{H}_{\text{GradCAM}}(x, S, c)$ , combining the localization capability of GradCAM with the fine-grained details of Guided Backpropagation.

**SmoothGrad (SG)** [35] seeks to alleviate noise and visual diffusion [32, 37] for saliency maps by averaging over explanations of noisy copies of an input. For a given explanation map  $\mathcal{H}$ , SmoothGrad is defined as  $\mathcal{H}_{\text{sg}}(x, S, c) = \frac{1}{N} \sum_{i=1}^N \mathcal{H}(x + g_i, S, c)$ , where noise vectors  $g_i \sim \mathcal{N}(0, \sigma^2)$  are drawn i.i.d. from a normal distribution.

## 4. The VAR framework

Post-hoc attribution methods have been shown to perform poorly in recovering the classification-relevant information from the network [7, 27] and arguably fail network perturbation based sanity checks [1]. This, however, means that the explanations obtained through these attribution methods are not suited for the typical application scenario, which is usually understanding what information in the input was relevant for a classification. In the following, we propose VAR for Visualizing Actually Relevant features. VAR is a general approach to refining saliency maps by considering attributions across classes to reveal information that is specifically relevant for distinguishing classes. By design, VAR is a framework that fits any attribution method that depend on the gradient of the network output with respect to input,  $\frac{\partial S_c}{\partial x}$ .

To elucidate the class-relevant information contained in a set of attribution maps  $\{\mathcal{H}_k | k \in K \subseteq \{1, \dots, C\}\}$ , we propose to compute the relative importance at each spatial location, here a pixel, between classes using softmax, analogous to how the final classification treats the different logits through softmax.<sup>1</sup>

<sup>1</sup>Attribution methods for outputs are usually applied to logits, as numerical issues caused by the flatness of the softmax function at the

In particular, we propose the following three-step procedure to turn any attribution method  $\mathcal{H} : \mathcal{I} \times S \times \{1, \dots, C\} \rightarrow \mathcal{I}$  into an attribution method revealing class-relevant features  $\mathcal{C}_{\mathcal{H}} : \mathcal{I} \times S \times K \rightarrow \mathcal{I}^d$ , where  $K \subseteq \{1, \dots, C\}$ .

### Class-relevant Attributions

**Step 1: Initial Attribution.** First, we compute attribution maps for each class  $c \in K$  as

$$\mathcal{H}_c = \mathcal{H}(x, S, c),$$

where  $x$  is the input,  $S$  is the model, and  $\mathcal{H}_c$  is any attribution map for class  $c$ .

**Step 2: Spatial Softmax.** To achieve attributions revealing the most class-relevant features, we calculate a softmax across classes for each pixel position  $(i, j)$  as

$$M_c(i, j) = \frac{e^{\hat{\mathcal{H}}_c(i, j)/t}}{\sum_{k \in K} e^{\hat{\mathcal{H}}_k(i, j)/t}},$$

where  $t = 0.1$  is the temperature. This produces an attribution  $M_c$  that emphasizes pixels where class  $c$  has higher attribution compared to most other classes, recovering distinguishing but also locally shared features.

**Step 3: Class-relevant Attribution.** The final attribution for class  $c$  is then computed as

$$\mathcal{C}_{\mathcal{H}_c} = \hat{\mathcal{H}}_c \odot M_c \odot \mathbb{1}_{M_c - \frac{1}{|K|} > \tau},$$

where  $\odot$  denotes element-wise multiplication,  $\mathbb{1}$  is the indicator function,  $\frac{1}{|K|}$  represents the uniform probability (chance level) across  $|K|$  classes, and  $\tau$  (set to 0.01 in our experiments) is a threshold parameter. The indicator function creates a binary mask that preserves only those pixels where the softmax probability  $M_c$  exceeds the uniform probability by at least  $\tau$ , effectively filtering out pixels that show only minimal preference for class  $c$  compared to other classes.

The resulting class-relevant attribution  $\mathcal{C}_{\mathcal{H}} : \mathcal{I} \times S \times K \rightarrow \mathcal{I}^{|K|}$  produces attribution maps that highlight features important for each of the selected classes while suppressing features that are common across most classes and hence not important for the decision of the network. Note that it is still possible to detect features that are shared and used between multiple classes (see Fig. 1), as long as they provide enough information to the network. Still detecting those shared features is not possible for methods that subtract feature maps or adjust the backpropagation rules for LRP.

---

(important) regions hinder using it directly as a target.

### Selecting the Set of Classes

Having defined our class-relevant attribution operator  $\mathcal{C}_{\mathcal{H}}$ , an important consideration is the selection of the set of classes  $K$  used for calculation.

We explore three approaches for class selection, each offering distinct advantages depending on the specific analysis goals and application context.

**Predefined Class Sets.** The canonical approach is to use a predefined set of classes  $K$  that are of particular interest. This is especially useful in contexts where specific class comparisons have natural interpretations. For example, in a grid-pointing game where users must identify the quadrant containing a particular object, the four quadrant classes directly correspond to the task structure. Similarly, in medical applications, contrasting disease subtypes can highlight discriminative features that aid differential diagnosis. This approach ensures that the resulting attributions focus on distinctions that are meaningful to the particular application domain. However, this approach requires specific knowledge about the task, which is often not available. The following approaches are data- and model-driven and, hence, do not require prior knowledge to select classes.

**Top- $k$  Most Probable Classes.** A model dependent approach to class selection involves choosing the  $k$  classes with highest predicted probabilities and the class with the lowest probability for a given input. This approach is particularly effective for highlighting the features that distinguish between the most plausible classifications for a given input, but also reveal information that is shared between highly related classes that are likely among the highest probabilities. As these classes represent the top candidates for the final classification, contrasting their attribution maps reveals the most decision-relevant features.

**Best-vs-Worst Classes.** The third approach compares the highest-probability class against the lowest-probability class:  $K = \{c_{\max}, c_{\min}\}$  where  $c_{\max} = \arg \max_c S_c(x)$  and  $c_{\min} = \arg \min_c S_c(x)$ . Such extreme can surprisingly reveal the most distinctive characteristics of the input as interpreted by the model, by showing which features are most critical for pushing the model toward or away from certain classifications.

## 5. Experiments

We evaluate VAR in three benchmark settings: ability to localize, insertion tests, and randomization-based sanity checks in combination with 5 different attribution methods. To assess localization ability, we consider the validation set of ImageNet [30], MS-COCO [19], and the Grid Pointing Game on ImageNet [27]. We as-



ness the quality of attributions by measuring how well these match annotated bounding boxes and segmentation masks.

For insertion tests, we quantitatively evaluate attributions using standard perturbation testing, which measures the importance of pixels. We employ the insertion method following the approaches of XRAI [17], which allows for systematic evaluation of how the addition of information impacts model confidence. Given the computational complexity, we use the first 1k. To validate robustness, we conduct sanity checks using randomization tests on 10k images on ImageNet [1].

We evaluate VAR on various architectures, including ResNet-50 [14], Vision Transformer B/16 (ViT) [9], and provide further results for VGG-16 [33], DenseNet-121 [15], Wide ResNet-50-2 [40], and ConvNeXt [21] in the Appendix. We consider models pre-trained on ImageNet, downloaded from PyTorch, and use these in their standard classification configuration. As attribution methods, we consider the most widely used model-agnostic attribution methods that can be readily applied to the above architectures. These include Gradient-weighted Class Activation Mapping (Grad-CAM) [31], Guided Backpropagation (GBP) [36], Integrated Gradients (IG) [37] with 50 steps and a blurred baseline, Input×Gradient (IxG) [32], and Guided Grad-CAM (Guided GC) [31]. For all methods we use Captum [18].

## 5.1. Localization

**Metrics** For our localization metrics, we assess attribution quality by measuring how well the attribution maps align with the actual object regions. The Region Attribution (RA) metric quantifies what portion of the total attribution weight falls within the target region, providing insight into attribution focus. The Intersection over Union (IoU) measures the spatial overlap between the attribution map and the ground truth region. Precision evaluates attribution specificity by calculating what fraction of the highlighted area corresponds to the target object, while Recall determines what proportion of the target region receives attribution. We also report the F1-score. Before evaluation, to prevent methods from being unduly rewarded for producing diffuse attributions, we apply a Gaussian blur to the attribution maps and so ensure a fair comparison across different approaches [28]. For both setups we use the target classes for  $\mathcal{C}_{\mathcal{H}}$ . We give further details in Appendix 7.1.

**Grid Pointing** For the grid-pointing game, we compile a  $2 \times 2$  grid of random images from ImageNet validation set, which we call Quad-ImageNet. This gives

us 12500 images. On VAR shows to greatly improve localization performance of all attribution methods (see Tab. 1). For ResNet50, we observe substantial gains in RA, with improvements ranging from +0.16 to +0.54 across different methods, and an average increase of +0.24. GradCAM with VAR shows a particularly strong performance, achieving an RA of 0.92 and F1 score of 0.81, along with an IoU improvement from 0.41 to 0.71. In terms of Precision, VAR improves the base methods with an average increase of +0.31 for ResNet50. This indicates that VAR produces more focused attribution maps that align better with the target object regions, which are also visually evident in Figure 3. We further observe that our framework is not only able to improve localization in terms of capturing the *distinguishing* features but is also able to recover the *common* features of closely related classes (cf. Fig. 3 second row). In particular, both GBP and Guided GC see strong precision improvements when enhanced with VAR (from 0.25 to 0.83 and from 0.39 to 0.84, respectively).

The decrease in recall with VAR (average -0.09) is explained by the ground truth encompassing entire quadrants, inherently favoring methods that highlight whole regions. While standard methods spread activation broadly, VAR focuses specifically on discriminative features that distinguish objects from others in the image. This targeted approach covers a smaller portion of the quadrant (lower recall) but identifies the most classification-relevant features (higher precision). The improved F1 scores confirm this precision-focused behavior is more valuable for practical applications than indiscriminately highlighting entire object regions. For the ViT architecture, we observe a similar pattern of improvements, though the magnitude is generally more modest. RA still shows consistent gains (average +0.12), while precision improvements are more moderate (+0.08 on average). The effect on IoU is minimal, and F1 scores show a slight decrease for some methods, suggesting that for transformers, the precision-recall trade-off is more balanced.

These results show that VAR enhances popular attribution methods in precisely localizing those features most relevant to classification. For these methods, the effect of VAR is strongest on convolutional architectures; likely since these methods are not tailored to ViT.

**MS-COCO** For MS-COCO, we use the whole validation set. We filter objects that are smaller than 1% of the image and objects for which the model has a confidence less than  $10^{-4}$ . In MS-COCO, VAR also improves localization performance, though the gains are more mod-

Method	Quad-ImageNet					COCO					
	RA	IoU	Prec.	Recall	F1	RA	IoU	Prec.	Recall	F1	
ResNet50	GradCAM	0.72	0.41	0.42	<b>0.98</b>	0.57	0.15	0.09	0.09	<b>1.00</b>	0.15
	w/ VAR	<b>0.91</b>	<b>0.71</b>	<b>0.81</b>	0.86	<b>0.81</b>	<b>0.18</b>	<b>0.11</b>	<b>0.12</b>	0.76	<b>0.17</b>
	Guided Backprop	0.35	0.20	0.25	<b>0.47</b>	0.32	0.15	0.09	0.09	<b>1.00</b>	0.15
	w/ VAR	<b>0.89</b>	<b>0.31</b>	<b>0.83</b>	0.33	<b>0.46</b>	<b>0.22</b>	<b>0.12</b>	<b>0.17</b>	0.45	<b>0.18</b>
	Guided-GradCAM	0.75	0.26	0.39	<b>0.45</b>	0.41	0.20	0.09	0.09	<b>0.99</b>	0.15
	w/ VAR	<b>0.92</b>	<b>0.28</b>	<b>0.84</b>	0.29	<b>0.42</b>	<b>0.26</b>	<b>0.13</b>	<b>0.20</b>	0.42	<b>0.20</b>
	Input×Gradient	0.40	0.20	0.25	<b>0.50</b>	0.33	0.12	0.09	0.09	<b>1.0</b>	0.15
	w/ VAR	<b>0.56</b>	<b>0.22</b>	<b>0.29</b>	0.49	<b>0.36</b>	<b>0.14</b>	0.09	0.09	0.99	0.15
	Integrated Gradient	0.42	0.21	0.26	<b>0.51</b>	0.34	0.13	0.09	0.09	1.00	0.15
	w/ VAR	<b>0.58</b>	<b>0.22</b>	<b>0.28</b>	0.50	<b>0.36</b>	<b>0.15</b>	0.09	0.09	1.00	0.15
<b>avg. change</b>	<b>0.24</b>	<b>0.13</b>	<b>0.31</b>	-0.09	<b>0.11</b>	<b>0.04</b>	<b>0.02</b>	<b>0.04</b>	-0.12	<b>0.02</b>	

Table 1. VAR improves a wide range of methods and scores. We show the quality of the attention maps of vanilla GradCAM, Guided Backpropagation, Guided-GradCAM, Input×Gradient, and Integrated Gradient, and augmented with VAR, measured using Region Attribution (RA), Intersection over Union (IoU), Precision, Recall, and F1. We see that by cleaning up the attention maps, Recall drops slightly, but all other metrics improve.

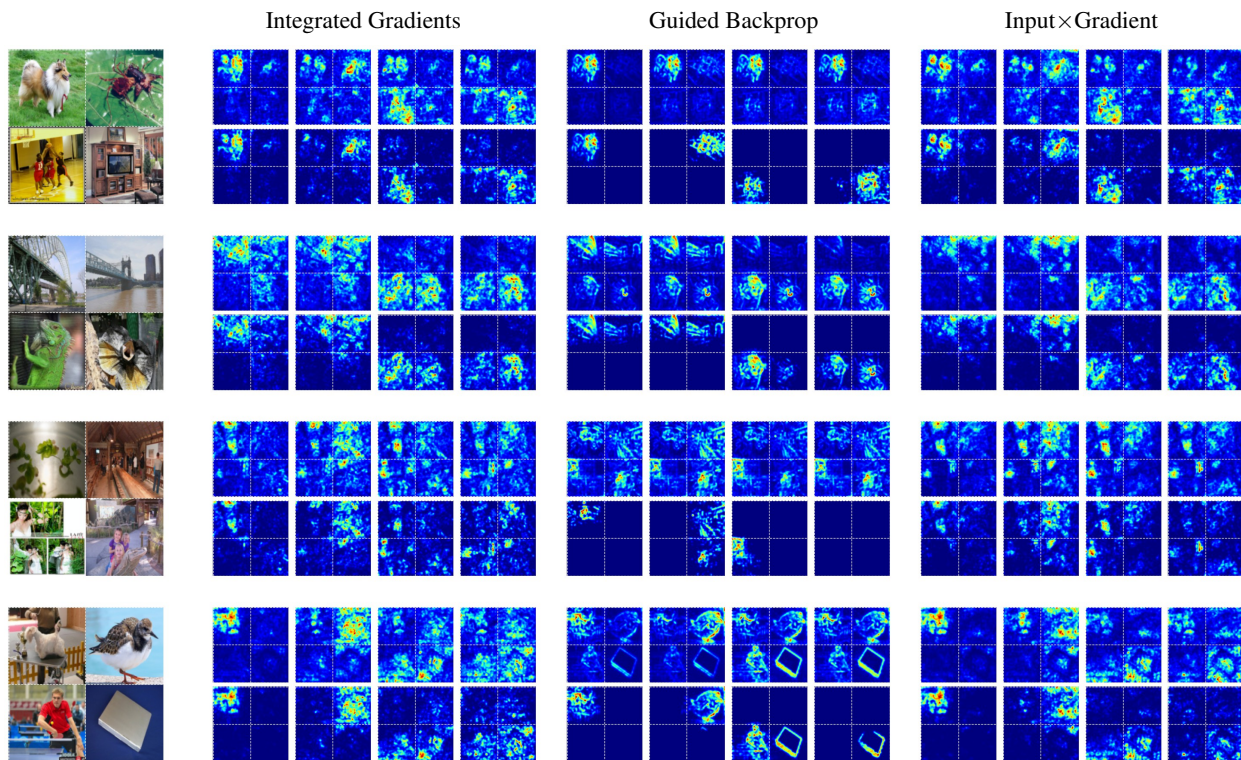


Figure 3. VAR on the Grid Pointing Game. We show examples from the grid pointing game for methods most affected by our framework (as columns: Integrated Gradient, Guided Backpropagation, Input×Gradient) for ResNet50. Input Images are given on the left, for each we provide vanilla attribution methods (top row) and augmented with VAR (bottom row). For each, we show the attribution for the four different classes in the grid as columns.

Method	SIC		AIC	
	ResNet50	ViT	ResNet50	ViT
GradCAM	0.73	<b>0.54</b>	0.76	<b>0.56</b>
w/ VAR	<b>0.74</b>	0.51	<b>0.78</b>	0.52
Guided Backprop	0.63	0.46	0.68	0.48
w/ VAR	<b>0.65</b>	0.46	<b>0.70</b>	0.48
Guided-GradCAM	0.69	<b>0.58</b>	0.73	<b>0.62</b>
w/ VAR	<b>0.7</b>	0.52	<b>0.74</b>	0.54
Input×Gradient	0.52	<b>0.56</b>	0.56	<b>0.59</b>
w/ VAR	<b>0.53</b>	0.51	<b>0.57</b>	0.53
Integrated Gradient	0.56	0.55	0.60	0.57
w/ VAR	<b>0.57</b>	0.55	<b>0.61</b>	0.57

Table 2. AIC and SIC scores for various attribution methods across different models, higher is better. While our method improves performance in some cases, it can also lead to lower scores due to its focus on discriminative features rather than the entire object.

est compared to Quad-ImageNet. Across all methods, we observe consistent improvements in Region Attribution (average +0.04), IoU (average +0.02), and precision (average +0.04). GradCAM and Guided GC with VAR show the strongest improvements, with RA increasing from 0.15 to 0.18 and 0.20 to 0.26 respectively. While recall decreases (average -0.12), this reflects VAR’s focus on discriminative features rather than entire object regions. COCO’s natural images contain multiple objects with complex backgrounds, making precise localization more challenging, yet VAR still manages to improve F1 scores across most methods on both ResNet50 and ViT, indicating better overall localization despite the more challenging context.

## 5.2. Insertion ablations

Following the benchmark evaluation scheme for saliency maps of Karpishnikov et al. [17], measuring how effectively an attribution method identifies the relevant image regions for a model’s decision. We follow the Performance Information Curve (PIC) framework, where we start with a blurred image, progressively restoring high-attribution pixels, measuring model confidence, and when the model returns to its initial prediction. This process produces Performance Information Curves that track how classification performance evolves as information is reintroduced.

To quantify overall performance, we report the Area under Accuracy Information Curve (AIC) and Softmax Information Curve (SIC), which summarizes accuracy across different information levels. We evaluate on ImageNet and report results in Table 2. We observe that our method improves scores for ResNet50 but provides similar or slightly decreased scores as the vanilla approaches on ViTs. These small differences in scores

could be explained by methods + VAR highlighting discriminative features rather than uniformly attributing importance across the object. For example, in an image of a cup, our method emphasizes the handle as a key distinguishing feature rather than the entire cup (cf. Appx. Fig. 6). While this better reflects the model’s decision-making, insertion tests favor methods that highlight *all* class-relevant pixels rather than indicating its class-specific features. In fact, we observe that VAR attributions, in contrast to the baselines, capture the specific class-relevant features, with ablated images (cf. Fig. 4) changing the output distribution without destroying all the information in the image, but rather surgically removing the information. For example for the porcupine, by removing the features of the one class we can change the prediction for the other class. While for the Cougar, we remove a very distinctive feature, namely the ear, which increases the uncertainty, showing that these features are crucial for the model.

## 5.3. Sanity Checks

To verify that attribution methods meaningfully reflect representation the model learned, we conduct cascading randomization tests following [1]. These tests progressively randomize model parameters from output to input layers, measuring how attribution maps change as model knowledge is systematically destroyed. We follow the same procedure as in the original paper. We measure the spearman correlation (Fig. 5), cosine similarity, and Pearson correlation (Appx. Fig. 7) between attributions before and after randomization. As the later parts of the network is randomized, ideally there is little information left about the target in the attribution maps. However, as discussed by Binder et al. [5], this randomization-based approach has shortcomings as it “preserves scales of forward pass activations with high probability”. Hence, we are primarily interested in the *relative change* between attributions of our framework and the respective original version, but we cannot compare the sanity-check results between different attribution methods.

We find that attribution maps + VAR give better results on the sanity checks, for all baseline methods and across randomization percentages (see Figure 5). For Guided-Backprop and Input × Gradient, the improvement is most pronounced, as well as for randomizing the latest layers, which carry most of the conceptual meaning for the classification.

## 6. Discussion & Conclusion

We presented VAR, a framework to enhance any attribution method in a plug-and-play manner to obtain bet-



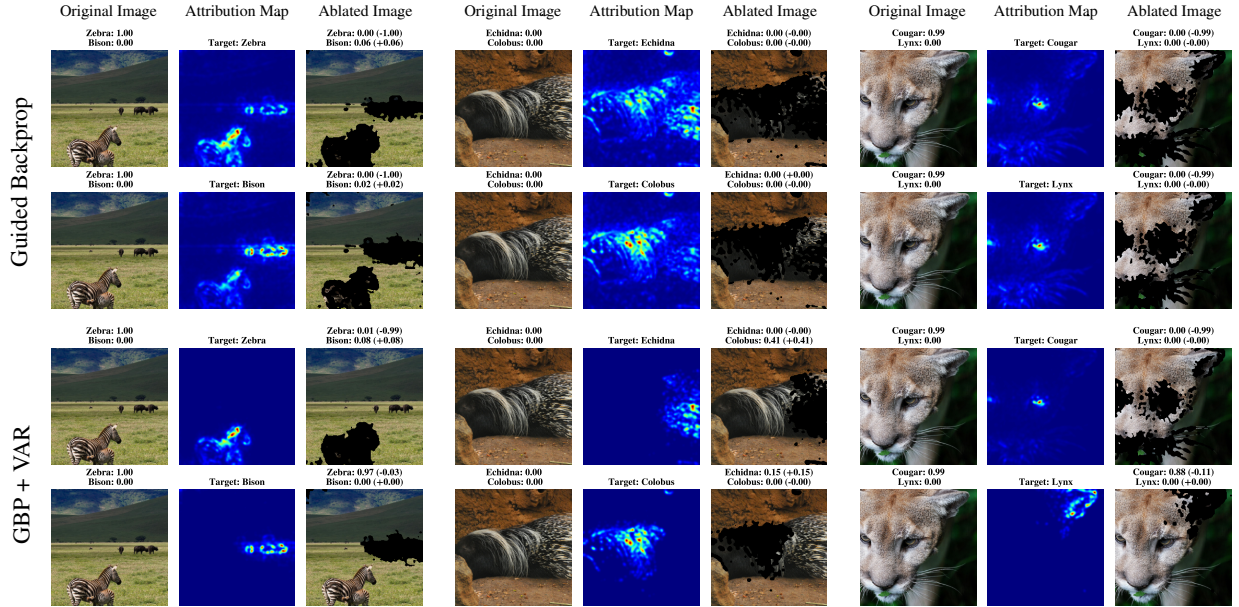


Figure 4. *Ablation study.* For GBP (top) and GBP with VAR (bottom) we provide three examples from the insertion/deletion ablation. For each example, we show the original image with associated class softmax scores for two classes associated with image features, the attribution map for each of the classes, and the attribution-based intervention mask on each of the classes with resulting changes in class softmax scores.

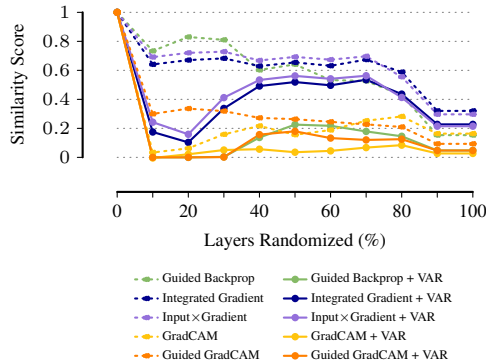


Figure 5. *Sanity check by network randomization.* We show similarity between attributions before and after randomization of  $x\%$  of network layers (ResNet50) for vanilla attribution methods (**dashed**) and when augmented with VAR (**solid**). **Lower is better.** Randomization is from back to front of the network following the strategy of Adebayo et al. [1].

ter class-specific saliency maps. VAR does not require any modification to the model or method by using a class-specific approach between attributions of classes to obtain attribution maps that are object- and instance-specific, reveal discriminative features for a class, but is also able to recover features that are shared for prediction of related classes.

To substantiate these claims, we provided extensive evaluation across five different attribution methods, convolutional as well transformer-based architectures, and different benchmarks for saliency maps, including the grid pointing game [27], sanity checks for saliency maps [1], and insertion tests [17].

Here, VAR showed to enhance standard attribution maps in terms of correct object localization. This comes at the cost of a bit of recall, a trade-off of methods augmented with VAR focusing on discriminative features rather than entire object regions, which is desirable for understanding but unfavorable for this particular metric. Attributions + VAR also showed to be capturing more important features for the model prediction, evident from the insertion deletion ablation, and are more robust to randomization sanity checks than their vanilla counterparts. While widely considered a good benchmark, the sanity checks has to be taken with a grain of salt when looking at absolute scores [5], hence we here focus on only interpreting relative differences of the same methodological approach to attributions.

Our evaluation of interpretability is by no means exhaustive, as a wide array of different benchmarks and metrics has been proposed over time. Instead, we here focused on the most common and widely employed evaluation protocols that are established in the literature. We



anticipate that our framework can be used as a general purpose tool to *understand the distinguishing features a model uses for prediction*.

## References

- [1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 9525–9536, 2018. 2, 3, 5, 7, 8
- [2] Chandan Agarwal and Anh Nguyen. Explaining image classifiers by removing input features using generative models. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2
- [3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS one*, 10(7):e0130140, 2015. 2
- [4] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun): 1803–1831, 2010. 3
- [5] Alexander Binder, Leander Weber, Sebastian Lapuschkin, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Shortcomings of top-down randomization-based sanity checks for evaluations of deep neural network explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16143–16152, 2023. 7, 8
- [6] Katarzyna Borys, Yasmin Alyssa Schmitt, Meike Nauta, Christin Seifert, Nicole Krämer, Christoph M. Friedrich, and Felix Nensa. Explainable ai in medical imaging: An overview for clinical practitioners – beyond saliency-based xai approaches. *European Journal of Radiology*, 162:110786, 2023. 2
- [7] Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos networks: Alignment is all we need for interpretability. 2022. 3
- [8] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems*, 2018. 2
- [9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5
- [10] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009. 3
- [11] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2950–2958, 2019. 2
- [12] Yash Goyal, Ziyang Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384, 2019. 2
- [13] Jindong Gu, Yinchong Yang, and Volker Tresp. Understanding individual decisions of cnns via contrastive backpropagation. In *Asian Conference on Computer Vision*, pages 119–134, 2018. 2
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5
- [15] 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 (CVPR)*, 2017. 5
- [16] Hong-Gyu Jung, Sin-Han Kang, Hee-Dong Kim, Dong-Ok Won, and Seong-Whan Lee. Counterfactual explanation based on gradual construction for deep networks. *Pattern Recognition*, 132:108958, 2022. 2
- [17] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. Xrai: Better attributions through regions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4948–4957, 2019. 2, 5, 7, 8
- [18] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*, 2, 2020. 5
- [19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 4
- [20] Shusen Liu, Bhavya Kailkhura, Donald Loveland, and Yong Han. Generative counterfactual introspection for explainable deep learning. In *2019 IEEE Global Conference on Signal and Information Processing*, pages 1–5, 2019. 2
- [21] Zhuang Liu, Hung Mao, Chongjian Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5
- [22] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In

- Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665, 2021. [2](#)
- [23] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017. [2](#)
- [24] Seitaro Otsuki, Tsumugi Iida, Félix Doublet, Tsubasa Hirakawa, Takayoshi Yamashita, Hironobu Fujiyoshi, and Komei Sugiura. Layer-wise relevance propagation with conservation property for resnet. In *European Conference on Computer Vision*, pages 349–364. Springer, 2024. [2](#)
- [25] Jay Parekh, Pavlo Mozharovskyi, and Florence d’Alché Buc. A framework to learn with interpretation. *Advances in Neural Information Processing Systems*, 34, 2021. [2](#)
- [26] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference*, 2018. [2](#)
- [27] Sukrut Rao, Moritz Böhle, and Bernt Schiele. Towards better understanding attribution methods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10223–10232, 2022. [2](#), [3](#), [4](#), [8](#)
- [28] Sukrut Rao, Moritz Böhle, and Bernt Schiele. Towards better understanding attribution methods. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 10223–10232, 2022. [5](#)
- [29] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016. [2](#)
- [30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. In *International Journal of Computer Vision*, 2015. [4](#)
- [31] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017. [2](#), [3](#), [5](#)
- [32] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153, 2017. [2](#), [3](#), [5](#)
- [33] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. pages 1–14, 2015. [5](#)
- [34] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*, 2014. [2](#), [3](#)
- [35] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. [3](#)
- [36] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015. [3](#), [5](#)
- [37] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3319–3328, 2017. [2](#), [3](#), [5](#)
- [38] Pei Wang and Nuno Vasconcelos. Scout: Self-aware discriminant counterfactual explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8981–8990, 2020. [2](#)
- [39] Yipei Wang and Xiaoqian Wang. ”why not other classes?”: Towards class-contrastive back-propagation explanations. *Advances in Neural Information Processing Systems*, 35, 2022. [2](#)
- [40] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016. [5](#)
- [41] Yifei Zhao. Fast real-time counterfactual explanations. *arXiv preprint arXiv:2007.05684*, 2020. [2](#)

# Now you see me! A framework for obtaining class-relevant saliency maps

## Supplementary Material

### 7. Evaluation Metrics

In our experimental setup, we evaluate attribution methods across several metrics to assess their efficacy in highlighting relevant features for model predictions. We define an input as a vector  $x \in \mathbb{R}^d$ , and a model as a function  $S : \mathbb{R}^d \rightarrow \mathbb{R}^C$ , where  $C$  is the number of classes in the classification problem. The final classification is performed via an argmax over  $S(x)$ . An explanation method provides an explanation map  $\mathcal{H} : \mathbb{R}^d \times S \times \{1, \dots, C\} \rightarrow \mathbb{R}^d$  that maps an input, a model, and optionally a target class to an attribution map of the same shape as the input.

#### 7.1. Localization metrics

We evaluate attribution methods using two datasets: a Grid Pointing Game based on ImageNet and COCO dataset with segmentation masks. For both evaluations, we apply the same set of metrics, treating both bounding boxes and segmentation masks as regions of interest  $R$  in the image. We match the region of interest with the correct attribution map  $\mathcal{H}_c$  i.e. for the first quadrant we also take the first attribution map. We only take the positive part of  $\mathcal{H}_c$ . Before evaluation, we apply a Gaussian blur with a kernel size of  $11 \times 11$  to the attribution maps:

$$\tilde{\mathcal{H}}_c = \mathcal{G}_\sigma * \mathcal{H}_c$$

where  $\mathcal{G}_\sigma$  is a Gaussian kernel with standard deviation  $\sigma$  and  $*$  denotes the convolution operation. This pre-processing step prevents methods from being unduly rewarded for producing diffuse attribution maps. We then compute the following metrics:

##### 7.1.1. Region Attribution

This metric quantifies what fraction of the total positive attribution falls within the region of interest:

$$\text{RA} = \frac{\sum_{i \in R} \tilde{\mathcal{H}}_c(i)}{\sum_i \tilde{\mathcal{H}}_c(i)}$$

##### 7.1.2. Intersection over Union (IoU)

We compute the overlap between the attribution map and the region of interest:

$$\text{IoU} = \frac{|(\tilde{\mathcal{H}}_c \cap R)|}{|\tilde{\mathcal{H}}_c \cup R|}$$

Method	ResNet50	ViT	VGG16	WRN50-2	DenseNet121	ConvNeXT
IG	0.56	0.55	<b>0.55</b>	0.57	0.55	0.47
w/ VAR	<b>0.57</b>	0.55	0.54	0.57	<b>0.56</b>	0.47
GBP	0.63	0.46	0.56	0.64	0.62	<b>0.46</b>
w/ VAR	<b>0.65</b>	0.46	<b>0.57</b>	<b>0.66</b>	<b>0.64</b>	0.44
IxG	0.52	<b>0.56</b>	0.51	0.53	0.51	<b>0.46</b>
w/ VAR	<b>0.53</b>	0.51	0.51	0.53	0.51	0.44
Guide-GC	0.69	<b>0.58</b>	0.54	0.69	<b>0.65</b>	<b>0.62</b>
w/ VAR	<b>0.7</b>	0.52	0.54	<b>0.7</b>	0.62	0.51
GradCam	0.73	<b>0.54</b>	0.52	0.72	<b>0.64</b>	<b>0.65</b>
w/ VAR	<b>0.74</b>	0.51	0.52	0.72	0.62	0.64

Table 3. SIC scores for the remaining architectures.

Method	ResNet50	ViT	VGG16	WRN50-2	DenseNet121	ConvNeXT
IG	0.6	0.57	0.58	0.61	0.61	0.64
w/ VAR	<b>0.61</b>	0.57	0.58	<b>0.62</b>	0.61	0.64
GBP	0.68	0.48	0.59	0.7	0.67	<b>0.62</b>
w/ VAR	<b>0.7</b>	0.48	0.59	<b>0.71</b>	<b>0.7</b>	0.52
IxG	0.56	<b>0.59</b>	0.54	0.57	0.57	<b>0.61</b>
w/ VAR	<b>0.57</b>	0.53	<b>0.55</b>	0.57	0.57	0.53
Guide-GC	0.73	<b>0.62</b>	0.57	0.74	<b>0.7</b>	<b>0.81</b>
w/ VAR	<b>0.74</b>	0.54	0.57	0.74	0.66	0.53
GradCam	0.76	<b>0.56</b>	0.54	0.77	<b>0.69</b>	<b>0.85</b>
w/ VAR	<b>0.78</b>	0.52	0.54	0.77	0.67	0.84

Table 4. AIC scores for the remaining architectures.

#### 7.1.3. Precision and Recall

$$\text{Precision} = \frac{|\tilde{\mathcal{H}}_c \cap R|}{|\tilde{\mathcal{H}}_c|}$$

$$\text{Recall} = \frac{|\tilde{\mathcal{H}}_c \cap R|}{|R|}$$

#### 7.1.4. F1 Score

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

## 8. Additional Result

### 8.1. Localization

We provide additional results of all the architectures mention in the Experiment section in Table 5. The trend remains the same for all architectures and methods; if they are augmented using VAR they improve the localization metrics and trade-off recall. Additionally we provide similar to Figure 3 plots for all other architectures in Figure 13-18.

### 8.2. Insertion test

We provide additional results for insertion tests in Table 3 and in Table 4. These results paint a similar picture to what we described in the section above.

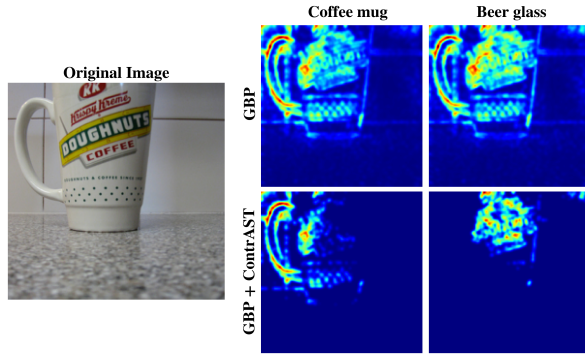


Figure 6. VAR highlights discriminative features. Standard Guided Backpropagation (GBP) produces nearly identical attributions for both "coffee mug" and "beer glass" classes (top row), while our approach (bottom row) clearly emphasizes the distinguishing features of each class—the handle for coffee mug and the cup shape for beer glass. This focus on discriminative features aids interpretability but can result in lower insertion test scores which reward highlighting the entire object.

### 8.3. Sanity Checks

We show the sanity check plots for the all the other architectures in Figure 7-12.



Method	Quad-ImageNet					COCO					
	RA	IoU	Prec.	Recall	F1	RA	IoU	Prec.	Recall	F1	
ViT	GradCAM	0.40	<b>0.26</b>	0.30	<b>0.80</b>	<b>0.40</b>	0.18	0.10	0.10	<b>0.97</b>	0.16
	w/ VAR	<b>0.43</b>	0.23	<b>0.39</b>	0.42	0.34	<b>0.21</b>	<b>0.11</b>	<b>0.16</b>	0.51	<b>0.17</b>
	GBP	0.27	<b>0.20</b>	0.25	<b>0.50</b>	<b>0.33</b>	0.09	<b>0.10</b>	0.10	<b>1.00</b>	0.15
	w/ VAR	<b>0.54</b>	0.20	<b>0.32</b>	0.42	0.32	<b>0.14</b>	0.09	<b>0.11</b>	0.77	0.15
	Guided-GradCAM	0.38	<b>0.20</b>	0.27	<b>0.48</b>	<b>0.34</b>	0.17	0.10	0.10	<b>0.98</b>	0.16
	w/ VAR	<b>0.49</b>	0.19	<b>0.38</b>	0.32	0.31	<b>0.21</b>	<b>0.12</b>	<b>0.16</b>	0.56	<b>0.19</b>
	Input×Gradient	0.33	0.20	0.25	<b>0.5</b>	0.33	0.10	0.10	0.10	<b>1.00</b>	0.15
	w/ VAR	<b>0.45</b>	<b>0.20</b>	<b>0.30</b>	0.41	0.33	<b>0.16</b>	0.10	<b>0.12</b>	0.79	<b>0.16</b>
DenseNet121	IG	0.36	0.20	0.25	<b>0.51</b>	0.34	0.12	0.10	0.10	<b>1.00</b>	0.15
	w/ VAR	<b>0.50</b>	<b>0.21</b>	<b>0.29</b>	0.44	0.34	<b>0.14</b>	0.10	0.10	0.98	<b>0.16</b>
	GradCam	0.52	<b>0.35</b>	0.39	<b>0.84</b>	<b>0.50</b>	<b>0.12</b>	<b>0.09</b>	<b>0.10</b>	<b>0.85</b>	<b>0.15</b>
	+ VAR	<b>0.55</b>	0.34	<b>0.51</b>	0.49	0.44	0.10	0.07	0.08	0.43	0.11
	GBP	0.33	0.20	0.25	<b>0.47</b>	0.33	0.14	0.09	0.09	<b>1.00</b>	0.15
	+ VAR	<b>0.86</b>	<b>0.33</b>	<b>0.74</b>	0.39	<b>0.49</b>	<b>0.21</b>	<b>0.13</b>	<b>0.15</b>	0.59	<b>0.19</b>
	Guide-GC	0.56	<b>0.23</b>	0.37	<b>0.40</b>	<b>0.37</b>	<b>0.17</b>	<b>0.09</b>	0.10	<b>0.79</b>	<b>0.14</b>
	+ VAR	<b>0.60</b>	0.12	<b>0.58</b>	0.12	0.19	0.15	0.06	<b>0.13</b>	0.21	0.10
ConvNext	IxG	0.35	0.20	0.25	<b>0.50</b>	0.33	0.12	0.09	0.09	<b>1.00</b>	0.15
	+ VAR	<b>0.47</b>	<b>0.21</b>	<b>0.27</b>	0.49	<b>0.35</b>	<b>0.14</b>	<b>0.09</b>	<b>0.09</b>	0.99	<b>0.15</b>
	IG	0.37	0.20	0.25	<b>0.51</b>	0.34	0.12	0.09	0.09	1.00	0.15
	+ VAR	<b>0.51</b>	<b>0.21</b>	<b>0.27</b>	0.50	<b>0.35</b>	<b>0.15</b>	<b>0.09</b>	<b>0.09</b>	1.00	<b>0.15</b>
	GradCam	0.96	<b>0.79</b>	0.85	<b>0.91</b>	<b>0.87</b>	0.24	0.13	0.13	<b>0.91</b>	0.19
	+ VAR	<b>0.98</b>	0.72	<b>0.94</b>	0.75	0.82	<b>0.26</b>	<b>0.14</b>	<b>0.16</b>	0.70	<b>0.22</b>
	GBP	0.41	0.20	0.25	<b>0.50</b>	0.33	0.13	0.09	0.09	<b>1.00</b>	0.15
	+ VAR	<b>0.59</b>	<b>0.24</b>	<b>0.42</b>	0.35	<b>0.37</b>	<b>0.17</b>	<b>0.09</b>	<b>0.10</b>	0.91	<b>0.15</b>
VGG16	Guide-GC	0.95	<b>0.34</b>	0.65	<b>0.43</b>	<b>0.51</b>	0.30	0.14	0.16	<b>0.80</b>	0.22
	+ VAR	<b>0.97</b>	0.19	<b>0.96</b>	0.19	0.31	<b>0.33</b>	<b>0.14</b>	<b>0.26</b>	0.39	0.22
	IxG	0.40	0.20	0.25	<b>0.50</b>	0.33	0.13	0.09	0.09	<b>1.00</b>	0.15
	+ VAR	<b>0.56</b>	<b>0.23</b>	<b>0.40</b>	0.35	<b>0.36</b>	<b>0.17</b>	<b>0.10</b>	<b>0.10</b>	0.92	<b>0.16</b>
	IG	0.47	0.21	0.26	<b>0.52</b>	0.34	0.13	0.09	0.09	<b>1.00</b>	0.15
	+ VAR	<b>0.67</b>	<b>0.27</b>	<b>0.37</b>	0.50	<b>0.42</b>	<b>0.16</b>	<b>0.09</b>	<b>0.09</b>	0.99	<b>0.15</b>
	GradCam	0.67	<b>0.43</b>	0.50	<b>0.79</b>	<b>0.59</b>	<b>0.16</b>	<b>0.10</b>	0.12	<b>0.57</b>	<b>0.16</b>
	+ VAR	<b>0.77</b>	0.33	<b>0.71</b>	0.40	0.45	0.15	0.08	<b>0.13</b>	0.27	0.12
WideResNet-50-2	GBP	0.27	<b>0.20</b>	0.25	<b>0.47</b>	<b>0.33</b>	0.14	<b>0.10</b>	0.10	<b>1.00</b>	<b>0.16</b>
	+ VAR	<b>0.48</b>	0.14	<b>0.40</b>	0.20	0.22	<b>0.16</b>	0.09	<b>0.12</b>	0.44	0.13
	Guide-GC	0.68	<b>0.26</b>	0.44	<b>0.41</b>	<b>0.41</b>	0.20	<b>0.09</b>	0.13	<b>0.50</b>	<b>0.15</b>
	+ VAR	<b>0.72</b>	0.08	<b>0.71</b>	0.09	0.14	<b>0.20</b>	0.06	<b>0.18</b>	0.12	0.09
	IxG	0.32	0.20	0.25	<b>0.50</b>	0.33	0.13	0.10	0.10	<b>1.00</b>	0.16
	+ VAR	<b>0.41</b>	<b>0.20</b>	<b>0.28</b>	0.45	<b>0.34</b>	<b>0.15</b>	<b>0.10</b>	<b>0.10</b>	0.99	<b>0.16</b>
	IG	0.35	0.21	0.26	<b>0.51</b>	0.34	0.14	0.10	0.10	<b>1.00</b>	0.16
	+ VAR	<b>0.47</b>	<b>0.21</b>	<b>0.28</b>	0.47	<b>0.35</b>	<b>0.17</b>	<b>0.10</b>	<b>0.10</b>	0.99	<b>0.16</b>
WideResNet-50-2	GradCam	0.74	0.42	0.42	<b>0.97</b>	0.58	0.15	0.10	0.10	<b>0.99</b>	0.15
	+ VAR	<b>0.91</b>	<b>0.71</b>	<b>0.82</b>	0.86	<b>0.81</b>	<b>0.18</b>	<b>0.11</b>	<b>0.12</b>	0.77	<b>0.17</b>
	GBP	0.42	0.20	0.25	<b>0.47</b>	0.32	0.16	0.10	0.10	<b>1.00</b>	0.15
	+ VAR	<b>0.90</b>	<b>0.35</b>	<b>0.80</b>	0.39	<b>0.51</b>	<b>0.23</b>	<b>0.12</b>	<b>0.16</b>	0.53	<b>0.18</b>
	Guide-GC	0.79	0.26	0.39	<b>0.45</b>	0.41	0.21	0.10	0.10	<b>0.98</b>	0.16
	+ VAR	<b>0.92</b>	<b>0.30</b>	<b>0.83</b>	0.32	<b>0.45</b>	<b>0.27</b>	<b>0.13</b>	<b>0.20</b>	0.47	<b>0.20</b>
	IxG	0.45	0.20	0.25	<b>0.50</b>	0.33	0.13	0.10	0.10	<b>1.00</b>	0.15
	+ VAR	<b>0.64</b>	<b>0.24</b>	<b>0.32</b>	0.49	<b>0.38</b>	<b>0.16</b>	<b>0.10</b>	<b>0.10</b>	0.99	<b>0.16</b>
IG	0.45	0.21	0.26	<b>0.51</b>	0.34	0.14	0.10	0.10	1.00	0.15	
+ VAR	<b>0.65</b>	<b>0.23</b>	<b>0.30</b>	0.50	<b>0.38</b>	<b>0.17</b>	<b>0.10</b>	<b>0.10</b>	1.00	<b>0.16</b>	

Table 5. VAR improves a wide range of methods and scores. We **bold** numbers where the relative improvements is either below or above 1 rounded to the second decimal. For ViT, we show the quality of the attention maps of vanilla GradCAM, Guided Backpropagation, Guided GradCAM, Input x Gradient, and Integrated Gradients, and augmented with VAR, measured using Region Attribution (RA), Intersection over Union (IoU), Precision, Recall, and F1. We see that by cleaning up the attention maps, Recall drops slightly, but all other metrics improve.

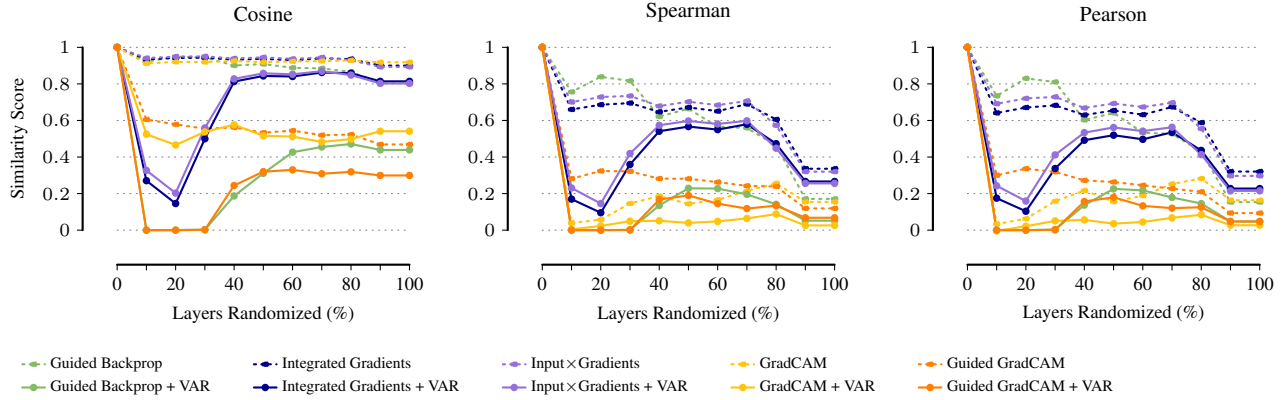


Figure 7. ResNet50: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which improve the corresponding baseline method.

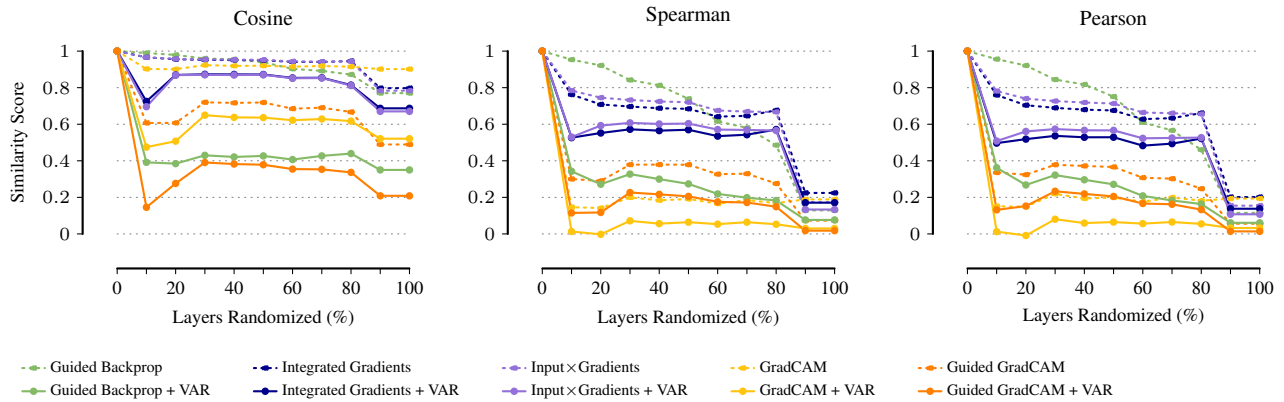


Figure 8. DenseNet121: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which always improve the corresponding baseline method.

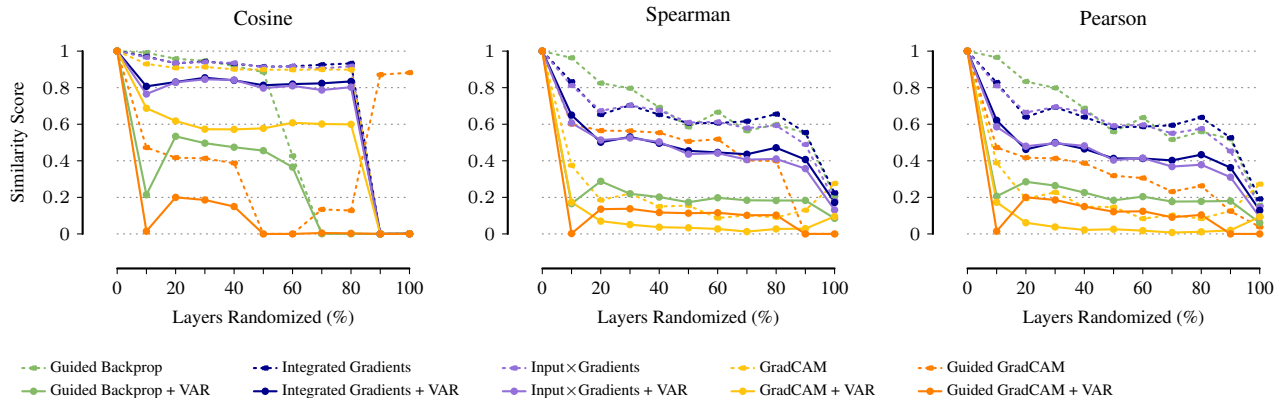


Figure 9. WRN50-2: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which always improve the corresponding baseline method.

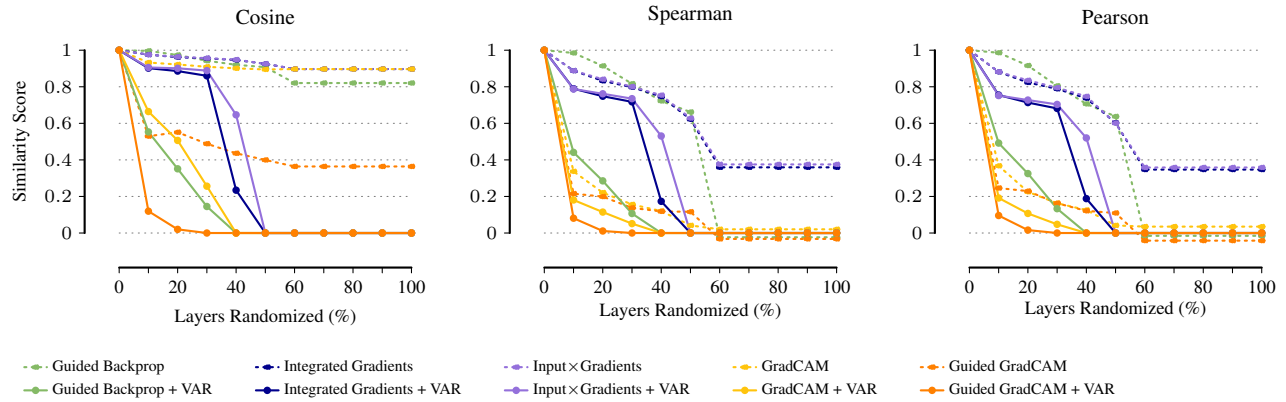


Figure 10. VGG16: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which improve the corresponding baseline method.

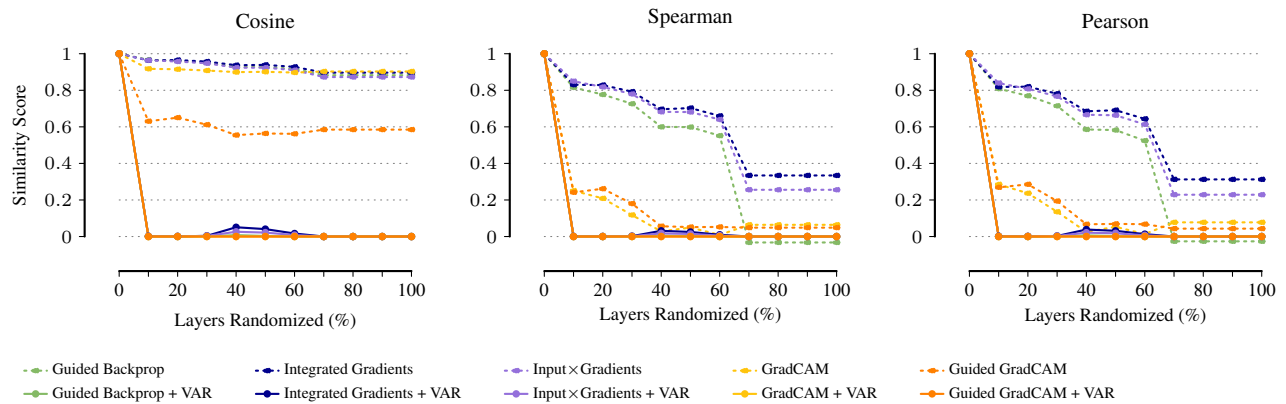


Figure 11. ConvNext: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which improve the corresponding baseline method.

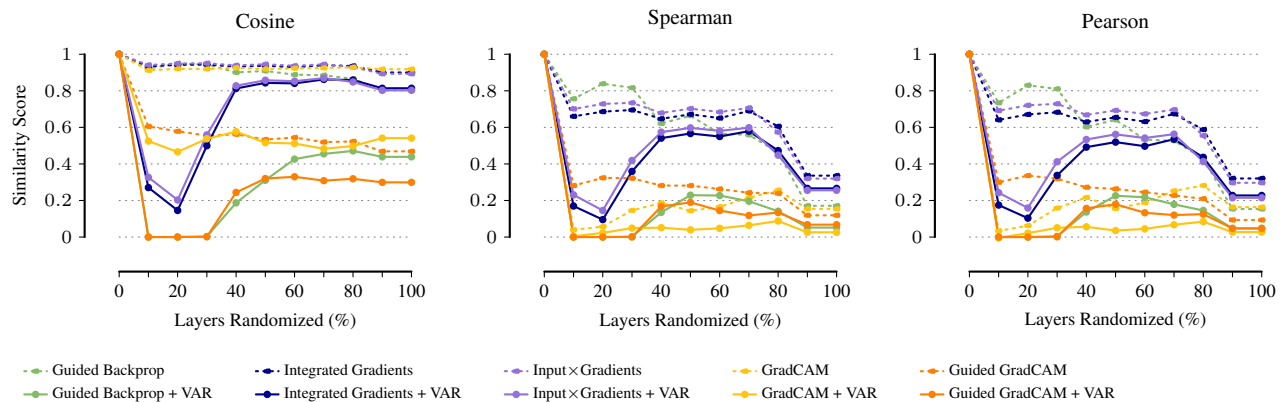


Figure 12. ViT: VAR improves all base methods under randomization [Lower is better]. For all methods and for varying level of randomization, we measure the similarity between the attention map for the unperturbed network and the randomized network. Dashed lines are base methods, solid lines when augmenting with VAR, which improve the corresponding baseline method.





Figure 13. **ResNet50**: VAR on the Grid Pointing Game. We show examples from the grid pointing game for methods most affected by our framework (as columns: Integrated Gradient, Guided Backpropagation, Input $\times$ Gradient). Input Images are given on the left, for each we provide vanilla attribution methods (top row) and augmented with VAR (bottom row). For each, we show the attribution for the four different classes in the grid as columns.



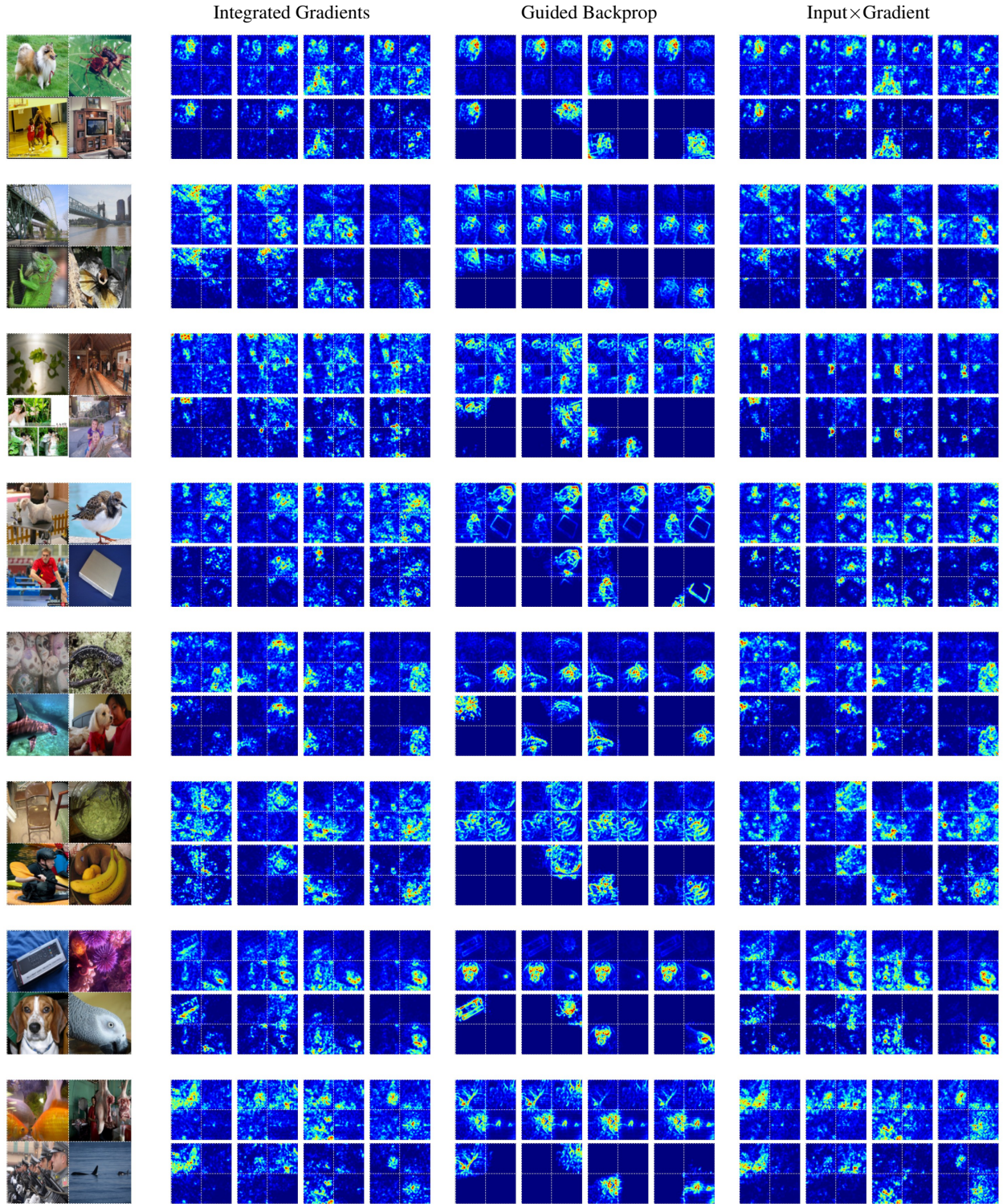


Figure 14. **DenseNet121**: *VAR on the Grid Pointing Game*. We show examples from the grid pointing game for methods most affected by our framework (as columns: Integrated Gradient, Guided Backpropagation, Input $\times$ Gradient). Input Images are given on the left, for each we provide vanilla attribution methods (top row) and augmented with VAR (bottom row). For each, we show the attribution for the four different classes in the grid as columns.









Figure 16. **VGG16: VAR on the Grid Pointing Game.** We show examples from the grid pointing game for methods most affected by our framework (as columns: Integrated Gradient, Guided Backpropagation, Input $\times$ Gradient). Input Images are given on the left, for each we provide vanilla attribution methods (top row) and augmented with VAR (bottom row). For each, we show the attribution for the four different classes in the grid as columns.









Figure 18. ViT: VAR on the Grid Pointing Game. We show examples from the grid pointing game for methods most affected by our framework (as columns: Integrated Gradient, Guided Backpropagation, Input $\times$ Gradient). Input Images are given on the left, for each we provide vanilla attribution methods (top row) and augmented with VAR (bottom row). For each, we show the attribution for the four different classes in the grid as columns.