Computing and evaluating saliency maps for image classification: a tutorial

Tristan Gomez and Harold Mouchère*
Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, Nantes, France

Abstract. Facing the black-box nature of deep learning models for image classification, a popular trend in the literature proposes methods to generate explanations in the form of heat maps indicating the areas that played an important role in the models' decisions. Such explanations are called saliency maps and constitute an active field of research, given that many fundamental questions are yet to be answered: how to compute them efficiently? How to evaluate them? What exactly can they be used for? Given the increasing rate at which papers are produced and the vast amount of literature that is already existing, we propose our study to help newcomers become part of this community and to contribute to the research field. First, the two existing approaches to generate saliency maps are discussed, namely post-hoc methods and attention models. Post-hoc methods are generic algorithms that can be applied to any model from a given class without requiring fine-tuning. On the contrary, attention models are ad-hoc architectures that generate a saliency map during the inference phase to guide the decision. We show that both approaches can be divided into several subcategories and illustrate each of them with one important model or method. Second, we present the current methodologies used to evaluate saliency maps, including objective and subjective protocols, depending on whether or not they involve users. Among objective methods, we notably detail faithfulness metrics and propose an implementation featuring the faithfulness metrics discussed in this paper (https://github.com/TristanGomez44/metrics-saliency-maps). [DOI: 10.1117/1.JEI.32.2.020801]

Keywords: saliency; explainability; interpretability; post-hoc; attention; faithfulness.

Paper 221232T received Nov. 3, 2022; accepted for publication Mar. 31, 2023; published online Apr. 24, 2023.

1 Introduction

The interpretability (or "explicability") of a system is its capacity to make a human understand its decisions, in particular with the help of explanations. The study of this type of system is a field that has experienced a renaissance with the recent surge of interest in deep learning. Indeed, deep neural networks (DNN) yield excellent performance but their complexity is a major obstacle to their deployment. It is not conceivable to let an algorithm make critical decisions alone, as these models are not robust and the decisions produced may reflect undesirable biases in the data on which they have been trained. In critical domains, it is therefore imperative that at least one expert verify the validity of the model's decisions. This verification is difficult in practice with a standard DNN because this class of model has millions of parameters and, unlike a linear regression model where there are only a few coefficients, it is not possible to directly understand how the model made its decision. A new branch of the study of deep learning has therefore emerged in recent years and focuses on the explainability of these models and proposes methods to improve and evaluate them. As this field is still young, the definition of interpretability is not well defined and there is no consensus on how to evaluate it. Despite these fundamental open questions, many approaches to improve DNN interpretability have been introduced, among which we distinguish between local and global approaches.

A global method sheds light on the behavior of a model on an entire dataset by showing which features of the model give a high weight to or in which areas of the input space the model has a limited performance.[1] In contrast, a local method seeks to explain a decision on a specific input. It shows the elements of the input that were important in the decision, to help the user

*Address all correspondence to Harold Mouchère, harold.mouchere@univ-nantes.fr

better understand why the model produced a certain decision and not another. It serves as an intermediary between the model and the user to make the decision made by the model interpretable. Global explanations are intended for model experts and are effective for an audit or global evaluation of the model. To facilitate human–model collaboration in routine use, local explanations are preferable because they gain the user's confidence in specific inputs to be processed.[1] Visual local explanations can take the form of examples,[2–5] concepts,[6] or saliency maps. Saliency maps are heat maps that indicate the areas of the image that played an important role in the model decision. In this tutorial, we focus on local explanations with saliency methods, because this approach is largely dominant in computer vision. We will first study the different approaches to computing saliency maps, which are divided into two categories: post-hoc methods and attention models. Next, we discuss existing methodologies for evaluating saliency maps, including objective and subjective protocols.

2 Saliency Map Generation

Two approaches have emerged in the literature to generate explanations based on saliency maps, as illustrated in Fig. 1. Post-hoc methods allow one to explain the decisions of any model from a certain class without requiring retraining. Attention models integrate an attention layer in the model to generate a saliency map (also called an attention map) during the inference to guide the decision. Such architectures are called attention models. First, we study the generic post-hoc methods that are used to explain the decisions made by any visual classification model. Second, we will discuss architectures that integrate attention modules.

2.1 *Post-hoc Explanation Methods*

There are three generic types of approaches for producing explanations of a model without having to retrain it: feature map weighting approaches, backpropagation approaches, and input image perturbation approaches, as illustrated in Fig. 2.

2.1.1 *Notations*

Let $x \in \mathbb{R}^{H \times W \times 3}$ be the image passed to the model, and $y_c(x)$ be the score of the class to be explained. We also note $(i, j) \in [0, \ldots, H] \times [0, \ldots, W]$, the spatial position at the level of the

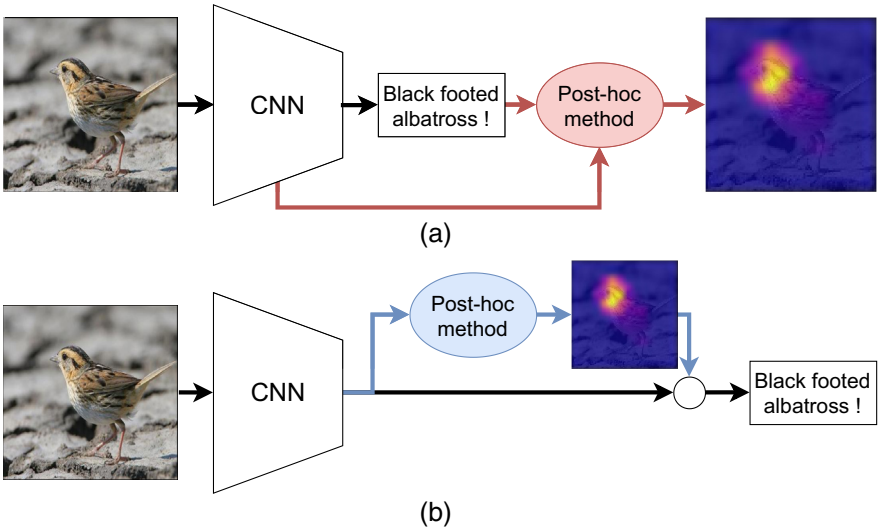

Fig. 1 The two approaches to generate saliency maps: (a) post-hoc methods and (b) attention models.

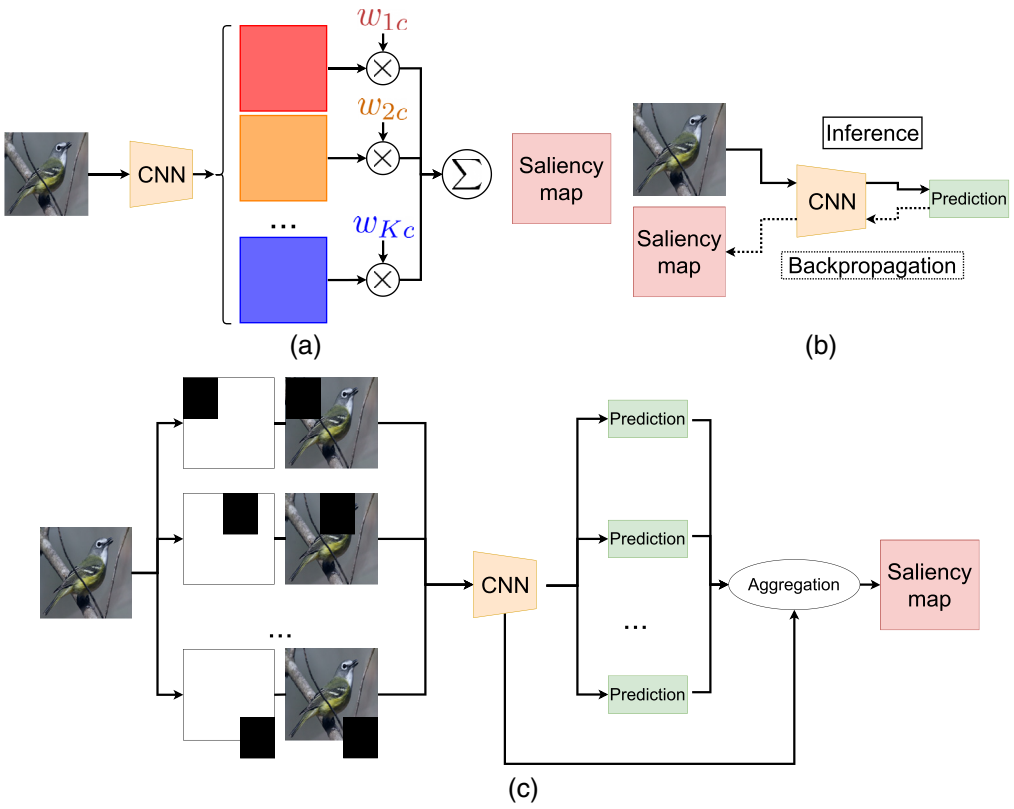

Fig. 2 The three post-hoc explanation approaches: (a) feature map weighting (e.g., CAM-based methods), (b) backpropagation (e.g., IG, SmoothGrad, but also LRP, DeepLIFT, etc.), and (c) input image perturbation (e.g., LIME, SHAP, etc.).

input image and $(i', j') \in [0, \ldots, H'] \times [0, \ldots, W']$, the spatial position at the level of the feature maps of the last layer.

2.1.2 *Perturbation approaches to the input image*

These methods consist in perturbing the input image to determine what contribution each part makes to the y_c score. One of the most popular perturbation-based post-hoc methods is local interpretable model-agnostic explanations (LIME).[7] This method splits the image into superpixels and estimates a weight for each superpixel. These weights correspond to the weights of a linear model G which approximates the model F locally. Let $\mathbf{z}$ be the input image and $\mathbf{z}'$ a simplified version of $\mathbf{z}$. More precisely, $\mathbf{z}'$ is a binary vector where $\mathbf{z}'_i$ indicates whether the i'th superpixel is present in $\mathbf{z}$ ($\mathbf{z}'_i = 1$) or whether it is replaced with gray pixels ($\mathbf{z}'_i = 0$). We note $G(\mathbf{z}')$ the prediction of the linear model by masking the superpixels indicated by $\mathbf{z}'$. The model F is approximated as follows:

$$G(\mathbf{z}') = \phi_0 + \sum_{i}^{P} \phi_i \mathbf{z}'_i, \tag{1}$$

where $\sum_i$ are the weights of each superpixel. The authors define the following cost function

$$\text{Loss} = \Omega(G) + \sum_{\hat{z} \in Z} \pi(\hat{\mathbf{z}}, \mathbf{z})(y_c(\hat{\mathbf{z}}) - G(\hat{\mathbf{z}}')), \tag{2}$$

where Z is a set obtained by sampling the neighborhood of $\mathbf{z}$, $\pi(x, \mathbf{z}) = \exp(-\|x - \mathbf{z}\|_2^2/\sigma^2)$ is a similarity function to give greater weight to the neighbors of $\mathbf{z}$ that are closest, and Ω is a regularization term to decrease the complexity in the explanation. A sample $\hat{\mathbf{z}}$ is generated by randomly masking superpixels of the image, which amounts to changing the values of $\mathbf{z}'$.

To reduce the complexity of the explanation computation, the following regularization term is chosen:

$$\Omega(G) = \begin{cases} \infty & \text{if } |\phi|_0 > C \\ 0 & \text{else} \end{cases}, \tag{3}$$

where $\|\phi\|_0$ is the number of coefficients whose value is nonzero. In practice, to respect this constraint, the authors choose the I most important superpixels with the least absolute shrinkage and selection operator algorithm[8] and determine their weight ϕ_i with a least squares regression. The other superpixels are assigned a zero weight.

Based on this work, Lundberg and Lee[9] use Shapley values as a solution to the Eq. (1) with a method called SHapley Additive exPlanations (SHAP). The authors show that Shapley values are the only solution to the Eq. (1) that can guarantee properties such as local fidelity or consistency: if the model changes such that the contribution of a superpixel increases or remains the same, the weight of the superpixel will not decrease.

Parallel to the development of LIME-inspired methods, Petsiuk et al.[10] introduced a method called "randomized input sampling for explanation" (RISE). With this method, instead of being split in superpixels, the image is sliced into a rectangular grid of size $H' \times W'$. The method consists to apply random binary masks on the input image to estimate which areas induce the largest drop in the score when masked. The authors then directly use the average score drop obtained when masking each spatial position as a saliency map.

A limitation of these methods is the number of inferences to be performed to get a good estimate of the saliency map, which is exponential with the resolution of the map/number of superpixels. Indeed, Petsiuk et al.[10] run $M = 8k$ inferences per image of standard size 224×224 with the RISE method in resolution 7×7 with a standard backbone residual network (ResNet)-50. Similarly, the authors of LIME perform $15k$ inferences per image to explain. Contrary to the two other families that we will discuss next, these methods can be applied to any type of image classifier and are therefore not restricted to convolutional neural networks (CNNs) or even DNNs.

2.1.3 *Feature map weighting approaches*

Another major approach to generating saliency maps is based on weighting the feature maps produced at the last layer of the CNN to explain the predicted class. These methods produce saliency maps at the resolution of the feature maps at the last layer, i.e., $S \in \mathbb{R}^{H' \times W'}$. A fundamental work from this category is the class activation map (CAM) method, which allows visualizing the areas that contributed most to the prediction of a specific class. To do so, the authors aggregate the feature maps of the last layer by weighing them by the weight they have on the score of the class to be explained, as illustrated in Fig. 3.

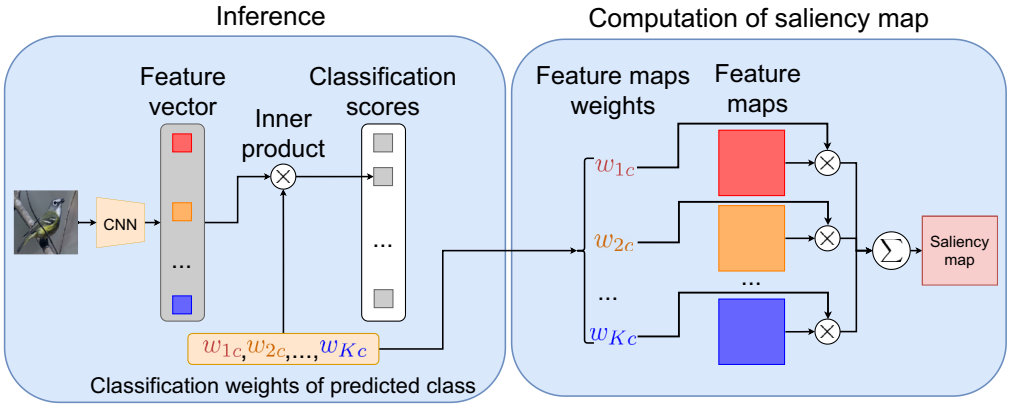

Fig. 3 Illustration of the CAM method. The weights map is a weighted average of the feature maps of the last convolution layer, using the weights of the classification layer.

Let $f^k \in \mathbb{R}^{H' \times W'}$, c, and w_{kc} be the k'th feature map of the last layer, the index of the class to be explained, and the weight of the linear classification layer connecting f_k and the score of the class c. The saliency map is defined as follows:

$$\text{CAM}(x) = \sum_k w_{kc} \times f^k(x). \tag{4}$$

One of the limitations of this method is that it requires the classification layer to be connected directly to the feature maps and therefore cannot be applied to architectures, such as Visual Geometry Group, where there are several dense layers between the maps and the classification scores, and cannot be applied to models designed for other tasks than classification, such as image description or the visual question answering task. Gradient-weighted CAM (Grad-CAM)[11] solves this problem by generalizing CAM to be applicable to all architectures where class scores are a continuous function of the feature maps. To do so, Grad-CAM computes the gradients of the classification score with respect to the feature maps to identify the maps that contributed the most to the decision.

Chattopadhyay et al.[12] note that Grad-CAM aggregates the partial derivatives all with the same weight and that as a result, the areas highlighted by the explanation do not correspond to the whole object, but only to parts of it. Also, when there are multiple occurrences of the same object, the Grad-CAM saliency maps do not highlight them all. This can be considered a problem since this situation is common outside the domain of single-label image classification. Chattopadhyay et al.[12] have therefore introduced Grad-CAM++, a variant of Grad-CAM where each partial derivative is assigned a different weight.

Wang et al.[13] have highlighted two problems concerning the gradient approaches of Grad-CAM and Grad-CAM++. First, gradient saturation/evanescence causes a noisy appearance of the saliency maps, and second, these approaches likely give too much weight to certain feature maps. As a solution, Wang et al.[13] proposed Score-CAM, a method that assigns a weight to each feature map based on the increase in observed score by masking areas of the input image that do not activate the map, as illustrated in Fig. 4.

Like the methods mentioned above, Score-CAM is formulated as follows:

$$\text{Score} - \text{CAM}(x) = \sum_k w^s_{kc} \times f^k(x). \tag{5}$$

The difference with the previous methods comes in the formulation of the weights attributed to each feature map, which are defined as the variation of the score resulting from masking the image areas not activating f_k

$$w^s_{kc} = y_c(x) - y_c(x \cdot \text{Norm}(\text{Upsample}(f^k))), \tag{6}$$

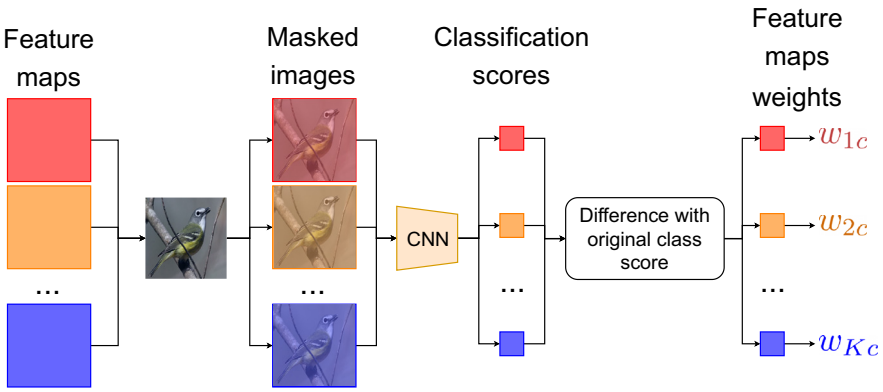

Fig. 4 Illustration of the Score-CAM method. The image is successively masked by each feature map and the class score variations are used as weights to aggregate the feature maps.

where Upsample interpolates f_k at the resolution of x and Norm is an operator that normalizes the map between 0 and 1 as follows: $\text{Norm}(x) = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$. A method similar to Score-CAM named Ablation-CAM was introduced in parallel by Desai and Ramaswamy[54] with motivations close to those of Score-CAM. Ablation-CAM evaluates the importance of a feature map by preventing the decision to exploit the information it contains and by measuring the relative variation of the y_c score. However, instead of hiding the input image, Ablation-CAM removes the map from the y_c calculation and measures the drop in score.

Recently, Jung and Oh[55] proposed learning important features class activation map (LIFT-CAM) by noting that Shapley values can be used to evaluate the saliency of feature maps and suggest approximating them with the deep learning of important features (DeepLIFT) algorithm. This idea is similar to the SHAP method, but instead of approximating the Shapley values of different parts of the input, Jung and Oh[55] approximate the Shapley values of the feature maps.

Fu et al.[14] also proposed aXiom-based Grad-CAM (Xgrad-CAM), an explanation method based on desirable properties that the authors introduce: sensitivity and conservation. The sensitivity property states that the weight assigned to a feature map is equal to the variation of the score when the map is removed from the calculation of the classification score. The conservation property states that the score must be explained exclusively by the contributions of the feature maps. To produce explanations that respect these properties as much as possible, the authors optimize a cost function composed of one term for each property. Other authors have also used the activation statistics of individual feature maps to compute feature map weights and detect high activation levels.[15,16]

2.1.4 *Backpropagation approaches*

Finally, the last family of approaches consists of backpropagating information from the output of the model to the input image to yield a high-resolution saliency map, which indicates the impact of each pixel on the decision, i.e., $S \in \mathbb{R}^{H \times W}$. The basic method of this type of approach is therefore to visualize these gradients and is simply called "gradient map" (GM), (also called "sensitivity map") which is equivalent to the "deconvolution" method in the case of a CNN with a rectified linear unit (ReLU) activation

$$\text{GM}_{ij}(x) = \frac{\partial y_c(x)}{\partial x_{ij}}. \tag{7}$$

To improve GM, Springenberg et al.[17] introduced a method called "guided backpropagation" with an idea also used by Grad-CAM++: negative gradients are suppressed with a ReLU activation, as they correspond to neurons that decrease the activity of the neuron that produces the score of the class to be explained, as illustrated in Fig. 5.

Next, Sundararajan et al.[18] proposed two properties that saliency maps should satisfy, namely sensitivity and invariance to implementation. The sensitivity property is respected if when a single input pixel x_{ij} is modified and when this affects the score $y_c(x)$, then the contribution of x_{ij} is changed too. The second property is satisfied if a method is not sensitive to the implementation of the model but only to the function implemented. Concretely, if two models implement the same function, i.e., if they produce the same predictions with the same inputs, an implementation-invariant assignment method should produce the same saliency maps for these two models. The authors introduce a method that respects these two properties called "integrated gradients" (IG)[18] by aggregating the GM resulting from the interpolation between the image we seek to explain and a reference image as follows:

$$\text{IG}_{ij}(x) = (x_{ij} - \tilde{x}_{ij}) \int_0^1 \text{GM}_{ij}(\tilde{x} + \alpha(x - \tilde{x}))\text{d}\alpha, \tag{8}$$

where $\tilde{x}$ is a reference image, which is an image that represents the absence of an object to be classified. The authors argue that $\tilde{x}$ should be chosen such that $y_c(\tilde{x}) \approx 0$ and propose to use a black image. Later, Smilkov et al.[56] argue that GMs vary chaotically, which contributes to the noise observed on these maps. In particular, they show that when interpolating from a reference

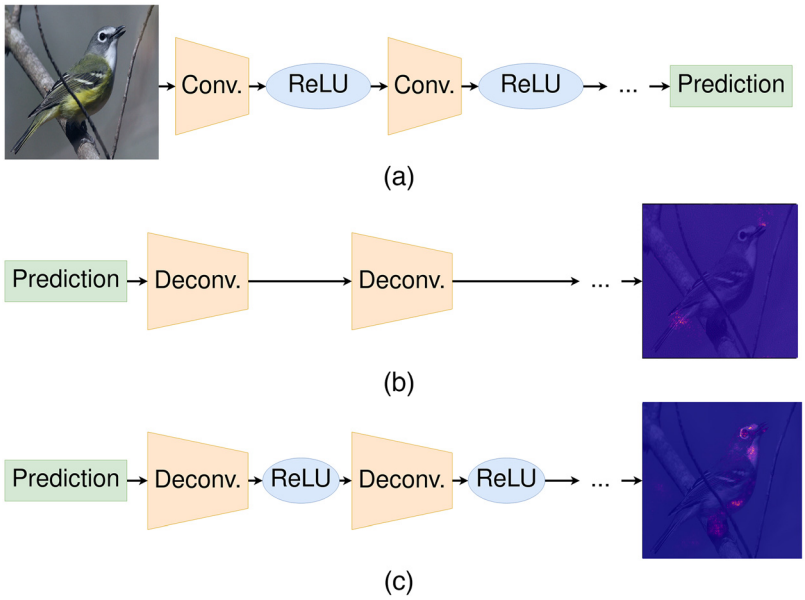

Fig. 5 The guided backpropagation method is equivalent to the deconvolution method except that it replaces the negative gradients with zero values to visualize only the pixels that participate in the activation of the class to be explained. (a) Inference, (b) deconvolution, and (c) guided backpropagation.

image to an image to be explained, the partial derivatives of y_c are abrupt and unpredictable. To overcome this problem, Smilkov et al.[56] proposed SmoothGrad to improve IG by aggregating the GM obtained by perturbing the input image with Gaussian noise. Next, Adebayo et al.[19] introduced a variant of SmoothGrad to address the same problem, called VarGrad, which uses the variance of the IG maps of the perturbed input as an explanation.

We end this section by discussing two gradient-free methods called "DeepLIFT"[20] and "layerwise relevance propagation" (LRP).[21] Unlike the methods mentioned earlier in this paragraph, DeepLIFT and LRP do not involve gradient computation. Nevertheless, these methods are close to gradient-based methods because they consist of backpropagating information from the model output to the input and constructing an explanation map at the input resolution. The DeepLIFT method postulates the "sum to difference" equation that states that the classification score can be explained by a sum of contributions from all the activations

$$\sum_{k=1}^{K} \sum_{ij} C_{\Delta x_{ij} \Delta y} = y_c(x) - y_c(\tilde{x}). \tag{9}$$

The authors of DeepLIFT demonstrate the existence of rules to estimate the impact of one layer's activation to the following and to backpropagate the estimates to the input image. According to the authors of DeepLIFT, the advantage of not using gradients is to propagate a signal of importance even in situations where the gradient is zero and to avoid artifacts caused by gradient discontinuities.

2.2 *Attention Architectures*

During the inference phase of a CNN with an averaging spatial aggregation layer (such as the ResNet model[22]), the activations of the maps all have the same weight in the calculation of the final feature vector. This can be a problem because some activations correspond to areas of the image that do not contain relevant information for classification, such as the background in the case of image classification. To overcome this problem, various attention modules have been proposed in the literature to allow the model to focus on specific parts of the image. These modules produce attention maps that give a different weight to different areas of the image,

allowing the model to ignore the background and highlight specific parts of the object of interest. These maps can therefore be used as visual explanations in the same way as the maps produced by the generic methods mentioned above, and will therefore also be called saliency maps. In the literature, these architectures fall into three categories: convolutional approaches that use convolution layers, prototypical approaches that compare feature vectors to learned prototypes, and nonparametric approaches that use nonparametric algorithms. The concept of attention has also led to the transformer approach, a deep architecture mostly constituted of particular attention layers called self-attention layers.

2.2.1 *Transformer approach*

Attention architectures have recently gained interest following the proposal by Vaswani et al.[23] of an architecture composed mainly of attention layers, named the transformer. This model has shown its interest first in natural language processing[23,24] but also more recently in computer vision.[25,26]

Due to the multiplicity of attention maps produced by a transformer, it can be argued that this type of architecture can be difficult to explain. However, it has been proposed to use the attention map of the classification token noted CLS of the last layer as an explanation[26]:

$$A = \text{softmax}\left(\frac{Q_{[\text{CLS}]}K^T}{\sqrt{d_k}}\right), \tag{10}$$

where $Q_{[\text{CLS}]}$, K^T, and d_k are the query of the last classification token, the keys of the last layer, and the dimension of the keys. The resolution of the attention map A is determined by the input image size $H \times W$ and the patch size $p_H \times p_W$: $A \in \mathbb{R}^{\frac{H}{p_H} \times \frac{W}{p_W}}$. Some more recent works have also proposed versions of post-hoc algorithms tailored for the transformer model.[27,28]

2.2.2 *Convolutional approaches*

Convolutional approaches use convolution layers to compute an attention map. Let $F \in \mathbb{R}^{H' \times W' \times C}$ be the tensor containing all the feature maps of the last layer of a CNN. During the inference phase of a CNN with an averaging spatial aggregation layer (such as the ResNet model[22]), the activations of the maps all have the same weight in the calculation of the final feature vector. This can be a problem because some activations correspond to areas of the image that do not contain relevant information for classification, such as the background in the case of image classification. To overcome this problem, various attention modules have been introduced in the literature to allow the model to focus on specific parts of the image. These modules produce attention maps that give a different weight to different areas of the image, allowing the model to ignore the background and highlight specific parts of the object of interest. These maps can therefore be used as visual explanations in the same way as the maps produced by the generic methods mentioned above, and will therefore also be called saliency maps.

The first attention module studied here was proposed by Hu and Qi[29] and is called "bilinear attention pooling" (BAP). First, a 1×1 convolution layer is applied on the tensor F to obtain an attention map $A^{H' \times W' \times 1}$. This module thus combines with a CNN that produces feature maps as follows:

$$F = \text{CNN}(x), \tag{11}$$

$$A = \text{Conv}_{1\times 1}(F). \tag{12}$$

The attention map is then multiplied with the feature maps, which lead to a set of weighted features $F_{\text{att}} \in \mathbb{R}^{H' \times W' \times C}$. Finally, an average pooling is applied on the spatial dimensions to obtain the final feature vector $f \in \mathbb{R}^C$, which is passed to the classification layer. The activations of the F_{att} tensor were amplified or reduced according to the weight indicated by the A map.

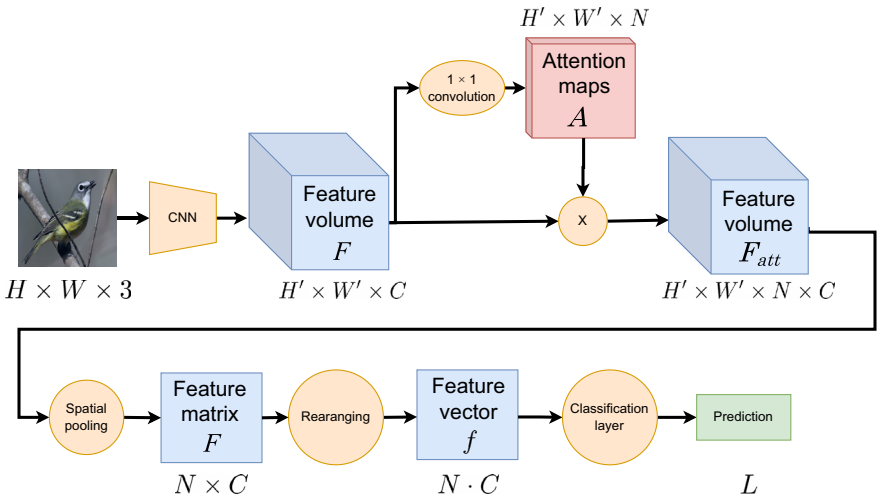

Fig. 6 Illustration of a bilinear CNN (B-CNN). A 1×1 convolution generates the attention maps. Then, the maps are multiplied by the feature maps before applying a spatial aggregation. Finally, the feature matrix is rearranged into a vector and is passed to the classification layer.

This mechanism allows the model to focus on the object of interest or on specific parts of it. To maximize the expressiveness of the model, the dominant approach in fine-grained classification is to extract one feature vector per important part of the object and concatenate the vectors before passing the final vector to the classification layer. Concretely, the 1×1 convolution has a kernel of size $N \times C \times 1 \times 1$ and produces $N > 1$ attention maps, i.e., a tensor of size $H' \times W' \times N$, as illustrated in Fig. 6. Each attention map is multiplied by the feature maps and the resulting tensor has dimension $H' \times W' \times N \times C$ and is reduced into a feature matrix F of dimension $N \times C$ using spatial average pooling. The matrix F is flattened into a vector $f \in \mathbb{R}^{NC}$ and passed to the linear classification layer. This mechanism allows the model to focus its attention on several different locations without losing information during spatial aggregation. The BAP module was later reused by other authors, notably in the context of transfer learning,[30] data augmentation strategies,[31] and causal learning.[32]

With a model named multiattention CNN, Zheng et al.[33] proposed an attention map defined as a weighted average of feature maps where weights are produced by a dense layer. Attention mechanisms with multiple convolution layers have also been introduced. For example, Fukui et al.[34] designed a model called attention branch network, which generates an attention map A with a convolution sequence interleaved with batch normalization layers and ReLU activations terminated by a sigmoidal activation. Instead of processing images one by one, Dubey et al.[35] use a cross-attention mechanism where images are processed in pairs, and the information extracted from one image is used as attention on channels of the features from the other image.

Some works also propose to use attention maps constructed from human gaze fixation density maps to improve the robustness and accuracy of attention models.[36,37] Convolutional attention architectures with multiple attention modules can also be found in the literature.[38–41] Note that these works develop architectural innovations to improve the accuracy of the model and are not designed to improve interpretability. Also, the attention mechanisms of these models can be seen as variants of the architectures mentioned in this section. For these reasons, these architectures are not detailed here.

2.2.3 *Prototypical approach*

Instead of using convolutional layers, Chen et al.[2] developed a model called ProtoPNet, which uses prototypes represented by parameter vectors learned during training. During inference, the similarity between these prototypes and the information extracted by the CNN is computed, as illustrated in Fig. 7.

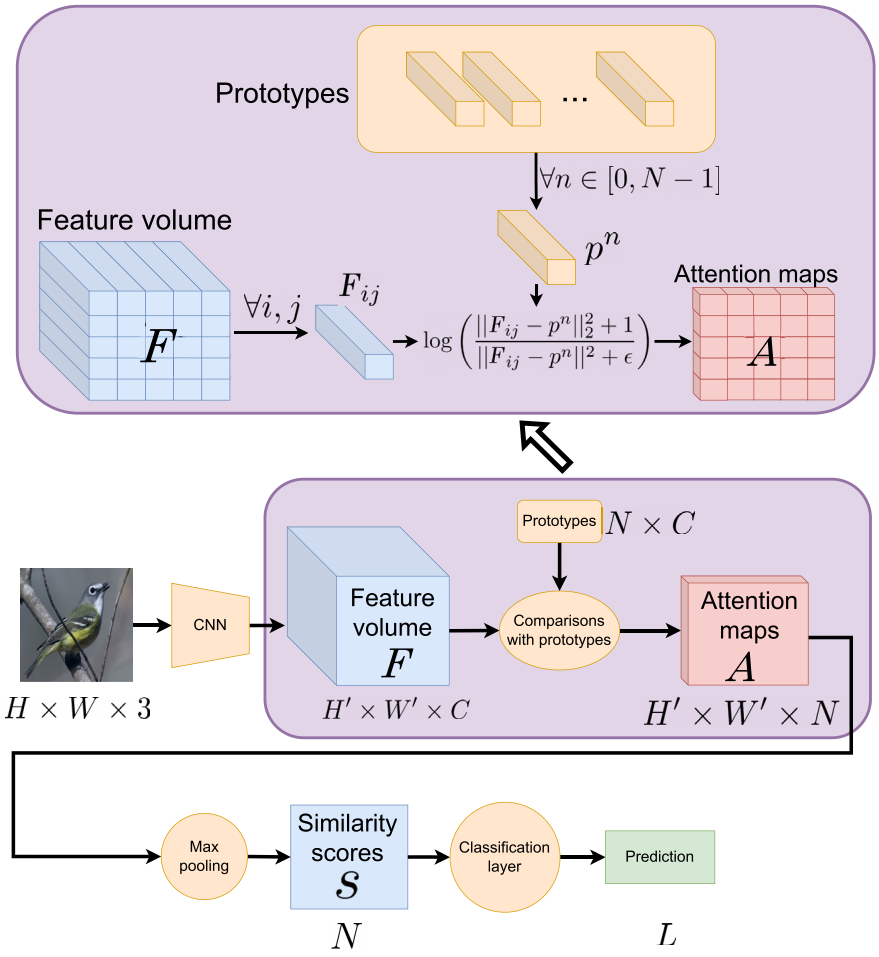

Fig. 7 Illustration of the prototypical architecture of ProtoPNet. The feature vectors extracted by the convolutional layers are compared to the prototypes, producing attention maps, and a similarity vector. This vector is then passed to the classification layer to produce a prediction.

More precisely, an attention map A^n is computed by comparing a prototype p^n to each feature vector F_{ij}

$$A_{ij}^n = \log\left(\frac{\|F_{ij} - p^n\|_2^2 + 1}{\|F_{ij} - p^n\|_2^2 + \epsilon}\right). \tag{13}$$

Then, a global similarity score s^k is computed with a spatial max pooling:

$$s_n = \max_{ij} A_{ij}^n. \tag{14}$$

The similarity s_n obtained represents how much the prototype p^n exists in the image. The linear classification layer y then takes the similarity vector s as input to produce a prediction. Chen et al.[2] also propose to assign each prototype to a class, with each class having $N//L$ prototypes, where L is the number of classes. We denote y_{gt} the label of image x and $\mathcal{P}_y$ the set of prototypes of class y. The training then proceeds in three phases. First, the CNN parameters are optimized so the extracted feature vectors are close to the prototypes of the right class and at the same time, the prototypes are optimized so that at least one prototype is found in each image with the following cost function:

$$\text{Loss} = \text{CE}(y(x), y_{gt}) + \lambda_1 C + \lambda_2 S, \tag{15}$$

where CE is the cross-entropy function and the terms C and S are respectively intended to move the feature vectors of a prototype closer to the right class and away from prototypes of other classes. These terms are defined as follows:

$$C = \min_{p^u \in \mathcal{P}_y} \min_{F_{ij}} \|p^u - F_{ij}\|_2^2, \tag{16}$$

$$S = -\min_{p^u \notin \mathcal{P}_y} \min_{F_{ij}} \|p^u - F_{ij}\|_2^2. \tag{17}$$

During this first phase, the parameters of the classification layer h are set at constant values. Let w_{nc} be the weight of the prototype p_n on the class c. Then, if p_n is a prototype of the class c, w_{nc} is set to 1 and otherwise, it is set to -0.5. This has the effect of forcing the network to learn prototypes that are representative of the class. During the second training phase, the prototypes are replaced by the feature vectors that are closest to them to interpret each prototype as a patch of an image. Indeed, each feature vector corresponds to a patch of size $H/H' \times W/W'$ on the input image. Let p_n be a prototype of class c. This vector is replaced as follows: $p_n = \underset{F_{ij}, x \in X_c}{\operatorname{argmin}} \|F_{ij} - p_n\|_2$, where X_c is the set of images of the class c. We then go through all the feature vectors extracted among all the images of the class c to find a vector that is as close to p as possible so as not to significantly alter the global behavior of the model. The third and last step of the training consists in optimizing only the $w_n c$ parameters of the h classification layer. The authors use the cross-entropy, to which they add a parsimony term on the w_{nc} weights as follows:

$$\text{Loss} = \text{CE}(y(x), y_{gt}) + \lambda_3 \sum_{nc} |w_{nc}|. \tag{18}$$

Chen et al.[2] argue that this regularization term penalizes models that reason by the negative, i.e., models with negative w_{nc} weights.

Various architectural modifications to the ProtoPNet architecture have been introduced. ProtoTree is a soft decision tree introduced by Nauta et al.[3] where the prototypes are arranged in a binary tree, and the image is directed to the left or to the right of a node according to its similarity with the prototype associated with the node. To facilitate the training, the images are directed smoothly through the tree and at each new node. It has also been proposed to use prototypes that are shared between classes[42] or deformable to enrich the expressiveness of the explanations.[43] Several authors also modify the loss function of the original ProtoPNet. Huang et al.[44] encourage the model to learn prototypes whose absence/presence is certain in each image, to generate more accurate attention maps. To improve the interpretability of the prototype space, Wang et al. force $N//L$ prototypes of the same class to form a subspace of dimension $N//L$ with a cost function term that promotes orthogonality.[45] Finally, Xiao et al. proposed to force the attention maps corresponding to the most present prototypes to be close to the saliency map produced by Grad-CAM with a term in the cost function, to provide meaningful and relevant explanations.[42] The ProtoPNet model has also inspired other variants that let the model reason negatively[46] or that use prototypes defined by not one but several vectors.[47]

2.2.4 *Non-parametric approaches*

In this category, we group the few approaches that we have identified generating attention maps without trainable parameters. For example, Choe et al. designed an architecture where the feature maps are simply averaged to obtain an attention map that correctly locates the objects of interest.[48] To generate multiple attention maps, Zheng et al. compute weighted averages of the feature maps, based on their similarity.[49] First, the f^k maps are spatially normalized with a softmax activation function

$$F_{\text{norm}} = \text{softmax}_{\text{spat}}(F). \tag{19}$$

We denote f^k_{norm} the maps after the normalization. Thus we have

$$\sum_{ij} f^k_{\text{norm}\,ij} = 1. \tag{20}$$

The maps are then flattened to have only one spatial dimension $F_{\text{flat-norm}} = \text{flat}(F_{\text{norm}}) \in \mathbb{R}^{H'W' \times K}$. We then compute a similarity matrix $M^{\text{sim}} \in \mathbb{R}^{K \times K}$, which indicates the similarity between the feature maps' activations:

$$M^{\text{sim}} = \text{softmax}(F^T_{\text{flat-norm}} \times F_{\text{flat}}), \tag{21}$$

where $\times$ denotes the matrix product, and the normalization softmax is applied on the second dimension of M^{sim}. The attention maps A^k are defined as follows:

$$A^k_{ij} = \sum_{k'} M^{\text{sim}}_{kk'} f^{k'}_{ij}. \tag{22}$$

We can also cite bilinear-representative non-parametric attention (BR-NPA),[50] which introduces a multipart nonparametric attention mechanism that groups feature vectors by similarity. This algorithm first consists to select N vectors among the feature vectors that have a high norm and a low cosine similarity with each other. Then each vector is refined by aggregating the feature vectors that have a high cosine similarity with it. The normalized cosine similarity maps can then be interpreted as attention maps.

2.3 *Visualization*

The Fig. 8 shows examples of saliency maps with one post-hoc method from each category (perturbation-based, feature map weighting-based, and backpropagation-based) and one attention model from each category (convolutional, prototypical, and nonparametric) on three datasets: Caltech-UCSD Birds-200-2011 (CUB-200-2011),[51] Fine-Grained Visual Classification of Aircraft (FGVC-Aircraft),[52] and Stanford cars.[53]

Visualizing saliency maps is a qualitative evaluation method used by all the previously cited works here.[2,3,7,9–14,17–21,43–45,54–57] This method is widely used because it is simple to set up. However, it can not be used to rank explanations and the conclusions of the analysis are largely dependent on the input images selected and on the researcher performing it. As a consequence, quantitative evaluation protocols have been proposed and are detailed in the next section.

3 Saliency Map Evaluation

Saliency map evaluation methods are divided into two categories: objective and subjective. Objective methods are automatic procedures that evaluate properties of the saliency map, such as fidelity, relevance, sensitivity, or invariance. Subjective methods confront users with saliency maps and are divided into two subcategories: direct and indirect methods. Direct methods ask users to evaluate properties of the saliency maps, such as relevance, quality, or consistency. Indirect methods ask questions to the users that are not directly about the explanation to determine what information they can extract about the model or the task. Objective and subjective methodologies are complementary as they do not evaluate the same aspects of the saliency maps. Note that assessing the interpretability of an attention model or post-hoc method is currently an open problem. Given that there is no widely accepted standard evaluation protocol yet, we review in the following all evaluation protocols that we have identified in the literature.

3.1 *Objective Evaluation*

Adebayo et al. proposed an evaluation of backpropagation approaches to determine what type of explanation these methods can or cannot produce.[58] Specifically, they measure the dependence of the maps produced by various methods on the parameters of the explained models. They show

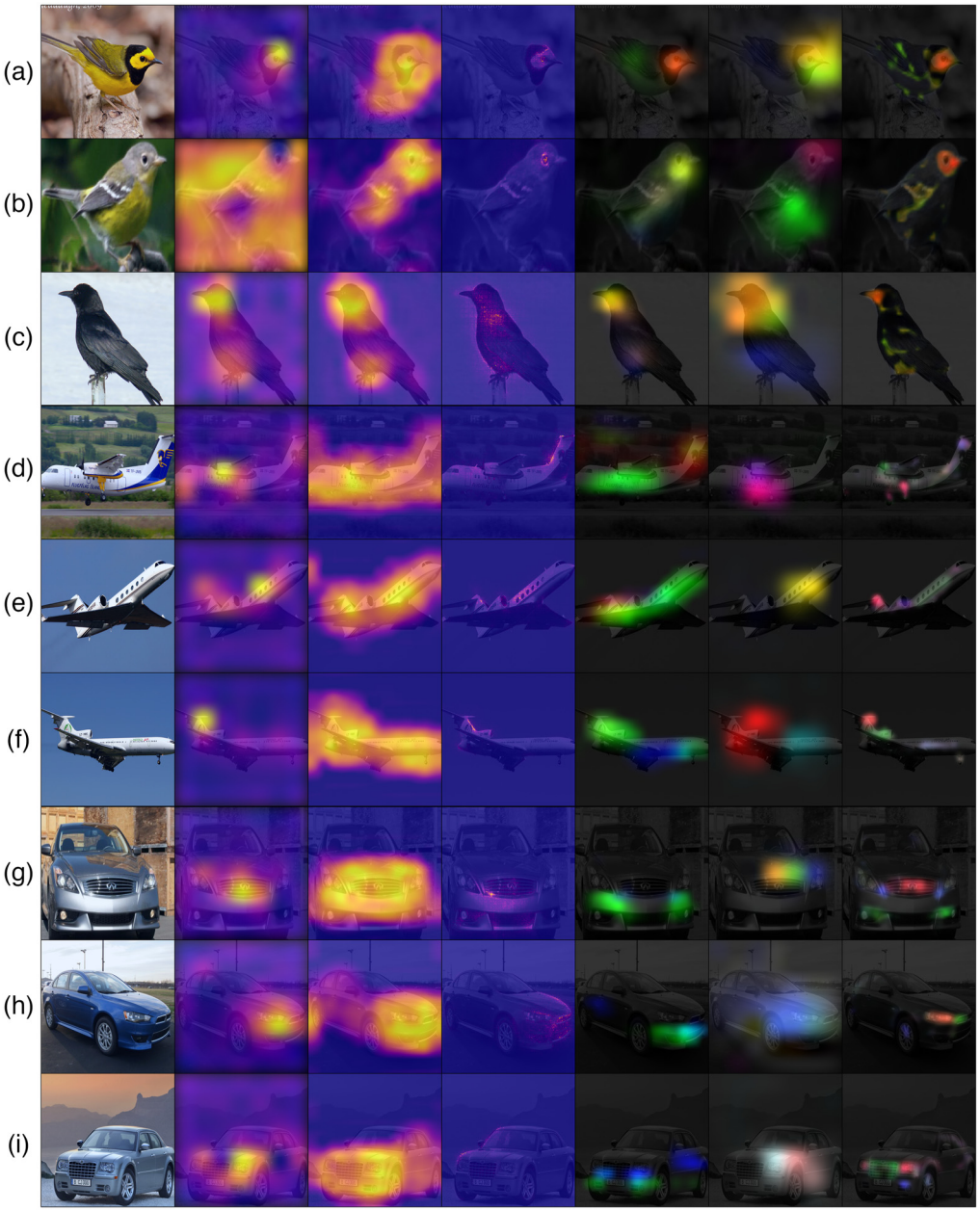

Fig. 8 Examples of saliency maps from each category of approach. From left to right: RISE (perturbation-based), Score-CAM (feature map weighting-based), SmoothGrad (backpropagation-based), B-CNN (convolutional), ProtoPNet (prototypical) et BR-NPA (nonparametric). The three attention models generate three attention maps, and the first, second, and third attention maps are represented in red, green, and blue colors, respectively. (a)–(c) CUB-200-2011 dataset,[51] (d)–(f) FGVC-Aircraft dataset,[52]and (g)–(i) Stanford cars dataset.[53]

that several methods produce maps that depend little on the parameters of the last layers of the model or the class of the image, notably guided backpropagation, questioning the effectiveness of these methods in this evaluation framework. Other work has also questioned the fidelity of explanation methods[59] or attention models.[60–62] On the contrary, recent work has shown that guided backpropagation is effective in detecting anomalies in input images.[63]

Some works evaluate how well the map highlights visual cues relevant to the task, i.e., the object to be classified in the case of image classification.[11–13] An important aspect of this method is that it does not only evaluate the explanation but also the model. Indeed, if a model uses bad visual cues in the image, such as the background in the case of object detection, the explanation should show this behavior, i.e., highlight the background. Thus, an explanation that is faithful to

the behavior of a model with a low accuracy would be penalized by this method. Other authors introduce important properties that an explanation should possess, such as consistency and local fidelity,[9] sensitivity and conservation,[14] or sensitivity and invariance to implementation.[18]

Similarly, the fidelity of saliency maps has been assessed using dedicated objective metrics,[10,12–14,55] which are called interchangeably fidelity, reliability, or faithfulness metrics. These metrics consist in perturbing the image processed by the model to determine whether the areas emphasized by the explanation contribute significantly to the class score. In the following, we discuss all the reliability metrics introduced so far in the literature, to the best of our knowledge. The input image is a tensor $x \in \mathbb{R}^{H\times W\times 3}$, and the saliency map is a 2D matrix $S \in \mathbb{R}^{H'\times W'}$ with a lower resolution, $H' < H$ and $W' < W$. The faithfulness metrics are divided into two groups: the single-step metrics that require one modification of the input image and the multistep metrics that require several. Given that there are multiple fidelity metrics proposed by various authors, we propose an implementation featuring all the faithfulness metrics discussed in this paper to simplify faithfulness evaluation (https://github.com/TristanGomez44/metrics-saliency-maps).

3.1.1 *Single-step metrics*

These metrics apply the saliency map on the input image as a soft mask to measure the impact that this masking procedure has on the score. The increase in confidence (IIC) metric[12] measures how often the confidence of the model in the predicted class increases when highlighting the salient areas. First, the input image is masked with the explanation map as follows:

$$I_m = \text{norm}(\text{upsamp}(S)) \cdot I, \tag{23}$$

where $\text{norm}(S)$ is the min–max normalization function, defined as $\text{norm}(S) = \frac{S-\min(S)}{\max(S)-\min(S)}$, $\text{upsamp}(S)$ is a function that upsample S to the resolution of I, and $\cdot$ is the element-wise product. The IIC metric is defined as

$$\text{IIC} = \mathbf{1}_{[c_I < c_{I_m}]}, \tag{24}$$

where c_I is the score of the predicted class with I as input and c_{I_m} is the score of the same class with I_m as input. The intuition is that a faithful saliency map S highlights areas such that when the nonsalient areas are removed, the class score increases. Therefore, maximizing this metric corresponds to an improvement. Note that this metric is a binary value and is only useful when computing its mean value over a large number of images.

Another example is the average drop (AD) metric,[12] which uses the same masking operation of the input image as IIC. Instead of measuring the frequency of score increase, AD measures the amplitude of the average score drop when highlighting the salient areas. Note that after the masking operation, the score is not supposed to decrease, which is why minimizing this metric corresponds to an improvement. Jung et al. proposed a variant of the AD metric called AD in deletion (ADD),[55] which consists to mask the salient areas instead of the nonsalient areas. To do this, the image is masked with the inverse of the saliency map, which highlights the nonsalient areas. Contrarily to AD, this metric removes the salient areas and the class score is expected to decrease, which means that maximizing this metric results in an improvement.

3.1.2 *Multi-step metrics*

Instead of using the saliency map as a mask, these metrics modify the input image incrementally, guided by the saliency map, and measure the impact of the modifications on the classification score. The deletion area under curve (DAUC) metric[10] evaluates the reliability of the saliency maps by progressively masking the image starting with the most important areas according to the saliency map and finishing with the least important. First, S is sorted and parsed from the highest element to its lowest element. At each element $S_{i'j'}$, we mask the corresponding area of I by multiplying it by a mask $M^k \in \mathbb{R}^{H\times W}$, where

$$M_{ij}^{k} = \begin{cases} 0, & \text{if } i'r < i < i^{'}(r+1) \quad \text{and} \quad j'r < j < j'(r+1) \\ 1, & \text{otherwise,} \end{cases} \tag{25}$$

and $r = H/H' = W/W'$. After each masking operation, the model m runs an inference with the updated version of I, and the score of the initially predicted class is updated, producing a new score c_k

$$c_k = m\left(I \cdot \prod_{\tilde{k}=1}^{\tilde{k}=k} M^{\tilde{k}}\right), \tag{26}$$

where $k \in \{1, \ldots, H' \times W'\}$. Second, once the whole image has been masked, the scores c_k are normalized by dividing them by the maximum $\max_k c_k$ and then plotted as a function of the proportion p_k of the image that is masked. Finally, the DAUC is defined as the AUC of this graph. The intuition behind this is that if a saliency map highlights the areas that are relevant to the decision, masking them would quickly result in a large decrease in the initially predicted class score, which in turn will minimize the AUC. Therefore, minimizing this metric corresponds to an improvement. Instead of progressively masking the image, the insertion AUC (IAUC)[10] metric starts from a blurred image and then progressively reveals it by replacing blurred patches with unmodified patches, starting from the most salient areas. Similarly, if the areas highlighted by the map are relevant for predicting the correct category, the score of the corresponding class is supposed to increase rapidly when revealing the original image. Maximizing this metric corresponds to an improvement.

Like DAUC, the deletion correlation (DC)[64] metric also consists in gradually masking the input image by following the order suggested by the saliency map, but instead of the AUC of the score curve, it is defined as the linear correlation of the class score variations and the saliency scores. As it measures the correlation between the saliency of a pixel and its impact on the class score, maximizing this metric is an improvement. Similarly, the insertion correlation (IC)[64] metric is inspired by IAUC and starts from a blurred image, and gradually reveals the image according to the saliency map. This correlation metric should also be maximized to indicate an improvement. Examples of masked images generated during the computation of the faithfulness metrics can be found in Fig. 9.

3.1.3 *Benchmark*

In Tables 1 and 2 can be found a benchmark performed on several explanation methods and attention models, using both single-step and multistep metrics. We also compute an Activation Map (AM) by averaging the feature maps as a simple baseline explanation. For each metric except IIC, we provide the mean and standard deviation over 100 random test images. For the IIC metric, we only provide the mean, as IIC is a binary value when computed on a single image. From this benchmark, we observe a limited consensus between metrics, which is consistent with previous observations.[65,66]

3.2 *Subjective Evaluation*

Subjective evaluations are user experiments, that we distinguish into two categories, direct and indirect, depending on whether the question asked to the users is directly related to the explanation or not. The direct category requires users to evaluate certain properties of the saliency maps, like their consistency,[67] the discriminability of the highlighted features,[68,69] how well it covers the object of interest,[12,54,67] the relevance,[67] or the overall quality.[70,71] On the other hand, the indirect category evaluates the saliency maps by asking users to extract information about the model with the help of a saliency map. This can take the form of model output prediction,[72–75] ground-truth (GT) class prediction,[76–81] recommendation,[7,82–84] or identification of the model objective.[83] The value of a saliency map is then determined by the difference in user performance with and without the saliency map.

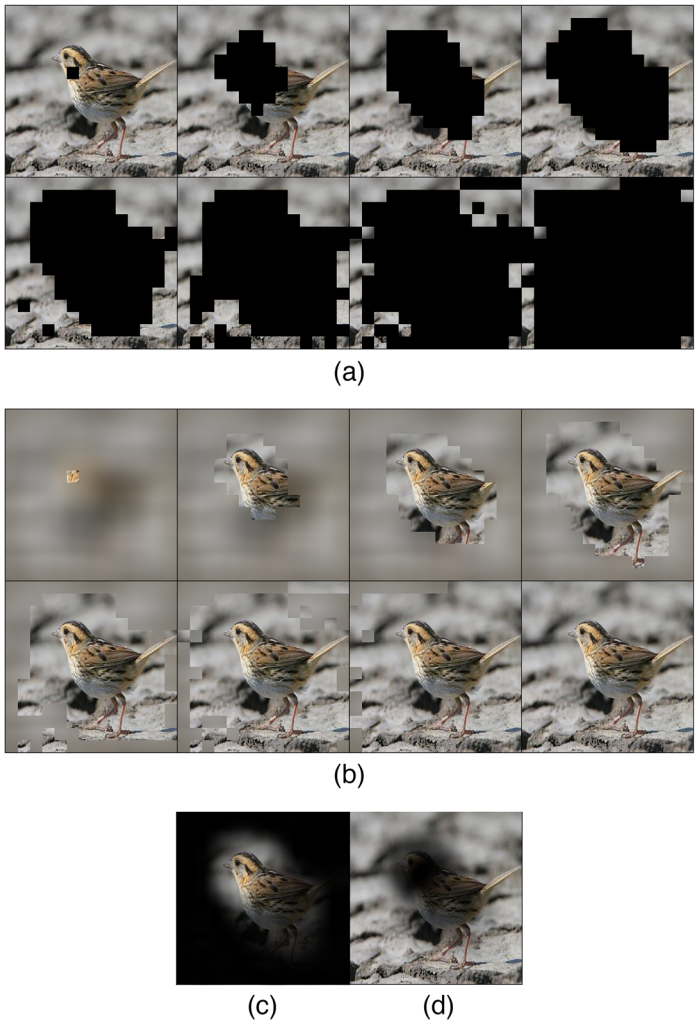

Fig. 9 Examples of image modifications applied during the computation of the faithfulness metrics: (a) DAUC, DC; (b) IAUC, IC; (c) IIC, AD; (d) ADD.

Table 1 Evaluation of explanations methods and attention models on multistep metrics.

Model	Viz. method	Accuracy	DAUC	IAUC	DC	IC
CNN	SmoothGrad	0.842	0.0398 ± 0.059	0.64 ± 0.271	0.429 ± 0.395	-0.009 ± 0.04
	AM		0.0362 ± 0.03	0.222 ± 0.213	0.313 ± 0.098	-0.088 ± 0.077
	RISE		0.0271 ± 0.066	0.427 ± 0.325	0.567 ± 0.296	0.227 ± 0.398
	Ablation-CAM		0.0215 ± 0.019	0.261 ± 0.229	0.358 ± 0.17	-0.04 ± 0.138
	Score-CAM		0.0207 ± 0.019	0.449 ± 0.295	0.316 ± 0.125	0.195 ± 0.125
	Grad-CAM++		0.0161 ± 0.013	0.454 ± 0.297	0.347 ± 0.096	0.192 ± 0.129
ProtoPNet	—	0.848	0.2964 ± 0.228	0.368 ± 0.257	0.095 ± 0.144	-0.06 ± 0.164
InterByParts	—	0.819	0.0811 ± 0.091	0.477 ± 0.191	0.232 ± 0.113	-0.042 ± 0.057
B-CNN	—	0.848	0.0208 ± 0.022	0.299 ± 0.274	0.266 ± 0.102	-0.023 ± 0.072
BR-NPA	—	**0.855**	0.0155 ± 0.014	0.493 ± 0.281	0.413 ± 0.12	-0.023 ± 0.057

Note: Bold value indicates best accuracy. Note that we do not highlight the best mean value for faithfulness metrics because of the large standard deviation that is observed.

Table 2 Evaluation of explanations methods and attention models on single-step metrics.

Model	Viz. method	Accuracy	IIC	AD	ADD
CNN	SmoothGrad	0.842	0.0	0.988 ± 0.039	0.033 ± 0.184
	AM		0.03	0.718 ± 0.302	0.562 ± 0.367
	RISE		0.18	0.38 ± 0.318	0.725 ± 0.35
	Ablation-CAM		0.11	0.42 ± 0.331	0.718 ± 0.365
	Score-CAM		0.2	0.255 ± 0.265	0.879 ± 0.186
	Grad-CAM++		0.16	0.285 ± 0.276	0.847 ± 0.218
ProtoPNet	—	0.848	0.09	0.604 ± 0.409	0.577 ± 0.424
InterByParts	—	0.819	0.73	0.005 ± 0.011	-0.004 ± 0.029
B-CNN	—	0.848	0.04	0.588 ± 0.331	0.712 ± 0.3
BR-NPA	—	**0.855**	0.01	0.884 ± 0.207	0.85 ± 0.241

Note: Bold value indicates best accuracy. Note that we do not highlight the best mean value for faithfulness metrics because of the large standard deviation that is observed.

3.2.1 *Indirect evaluation*

This type of evaluation ask questions to the user that are not directly about the explanation. As a consequence, the questions are independent of the explanation type and can be used to evaluate any type of local explanation. For this reason, we will also discuss user experiments where the explanation is not a saliency map.

Several researchers have conducted experiments asking users to predict the model output, i.e., to simulate the model. Alqaraawi et al.[72] measure how accurately users can simulate the model with the following experiment. First, users are shown examples of correct and incorrect predictions: one true positive, one false positive, one true negative, and one false negative. The positive and the negative examples belong to either one of two classes, which are balanced during the whole experiment. Each prediction is also accompanied by the corresponding input image and the model scores to illustrate each example. Second, another image is presented to users and they have to predict whether the system will recognize the positive class or not and rate their confidence in this prediction using a Likert item. This setup is illustrated in Fig. 10. To measure the impact of the saliency map and the model scores on the user prediction accuracy and confidence, the authors also tested three alternative setups where the map is masked, the scores are masked and both the map and the scores are masked. The results demonstrate that showing the saliency maps to users significantly improves their performance, but the effect size is small. On the other hand, the effect of showing scores to users was not significant. This method can be used to attribute a quality score to an explanation, where the score is defined as the user prediction accuracy when being helped by the explanation.

Due to the difficulty of users to predict the output of a complex model like a CNN, other authors focused on simpler tasks with models that are easier to explain, such as decision sets,[73] decision trees, and logistic regression.[75] Some authors proposed using prototypes instead of saliency maps to explain CNN decisions to users and help them simulate the model.[74] Counterfactual frameworks have also been introduced,[75,76] where users are asked to predict the impact of a modification of the input on the output.

Another type of experimental protocol consists to ask users to predict the GT class using the model's explanation[77–80] or to determine if the model prediction is correct or not.[81] Note that the methodology in this line of work implies that the models should have a negligible error rate so the explanations highlight features that are relevant to solve the task. Another task that has been introduced is to ask users to identify features that are important for the model using explanations.[72,83] Finally, other authors have also asked users to identify the features used by the model[72,83] or to ask them to estimate model performance based on explanations.[77–80] Note that with these protocols, one can also use user accuracy to rank explanations.

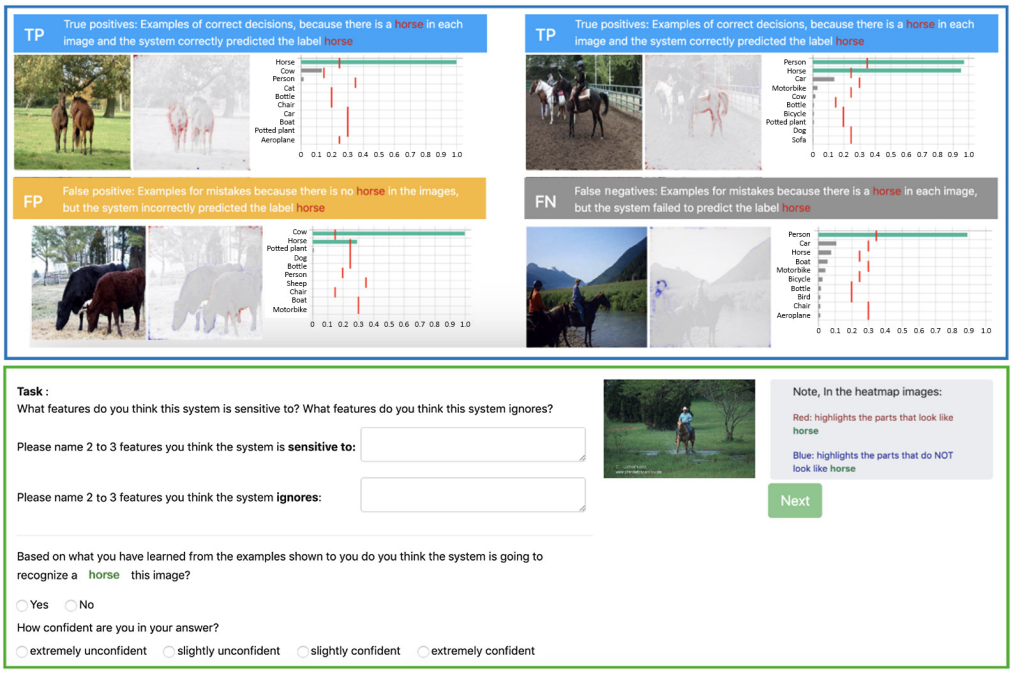

Fig. 10 The user interface used by Alqaraawi et al.[72] On top are shown examples of correct and incorrect classifications along with model scores and saliency maps.

3.2.2 *Direct evaluation*

This type of evaluation asks questions that are directly about the explanations to the users. Samuel et al.[67] showed users a series of incorrect predictions illustrated by the input image, the GT class, and the predicted class along with explanation maps for both the GT and the predicted class. Users then have to rate the degree to which the saliency maps justify the misclassifications, using the Likert scale item provided below, as illustrated in Fig. 11. This degree of justification is then used as a score to rank explanations.

Another line of work consists in directly asking users about the quality of explanations.[70,71] Jeyakumar et al. let users select the method that they consider to offer a better explanation among various explanation methods (saliency maps, examples).[70] Note that contrary to the previous protocols, this method does not generate an absolute quality score for each explanation but an average user preference rate. Also, Mohseni et al. asked users to review and evaluate heat

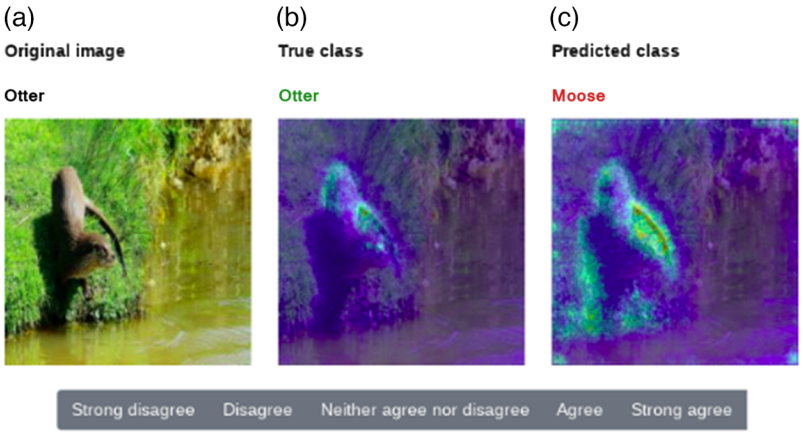

Fig. 11 The user interface used by Samuel et al.[67] The user must evaluate to what extent the saliency maps justify the classification errors using the scale provided below the images. (a) Original image, (b) true class, and (c) predicted class.

maps that highlighted the visual features used by the artificial intelligence (AI) to make its classification decision and were asked to rate the "quality" of the AI's decision on a scale of 1 to 10.[71] It is also possible to inquire users about less abstract properties of saliency maps. Other researchers let users choose which explanation map best highlights the object of interest.[12,54,67] As with Jeyakumar et al., this method does not generate a score but an average preference rate. Similarly, Samuel et al.[67] ask users to evaluate how consistently the saliency maps of an explanation method cover the parts of the object of interest. Finally, some experiments consist to ask users or experts to evaluate how discriminative the features highlighted by the saliency maps are.[68,69] These last two protocols use user quality estimations as scores to rank explanations.

4 Future Directions

Future work should first focus on improving the objective evaluation of the explanations protocols, as they are unreliable.[66,85] For example, Adebayo et al. proposed experiments showing that the explanation produced by guided-backpropagation (GP)[17] is independent of the weights of the model.[58] Yona et al.[85] later showed that the conclusions of Adebayo et al. were largely due to the considered task and demonstrated that GP could generate model-dependant explanations on tasks where the image contains multiple potential objects of interest.

Another example is Tomsett et al., who showed that there is little consistency in how faithfulness metrics are calculated in the literature and that such inconsistencies significantly affect the obtained fidelity value.[66] Furthermore, the authors used measures from the psychometric literature to show that the metrics have low statistical consistency and reliability.[66] Gomez et al. also showed that depending on the type of faithfulness metric considered, different types of explanation methods are favored.[65]

One property that may be a partial explanation for these issues is the fact that faithfulness metrics generate samples that lie outside of the training distribution (OOD).[64] In consequence, the explanations are generated based on a model that is in a specific OOD regime. This is not an intended feature of the metrics but rather an undesired side effect, and this might bias the results obtained. This demonstrates that another line of work should study the various experimental parameters used when evaluating explanations and how they impact the established benchmark.

As a consequence of the current issues that faithfulness metrics and more generally evaluation protocols have, it is still difficult today to compare explanation methods and to determine which one provides the best fidelity. For this reason, current progress on building more faithful methods is currently limited. Another line of work should further study the potential applications of saliency maps with user studies. In particular, we argue in favor of more research in the indirect evaluation domain. Indeed, these approaches are application-oriented and allow for the evaluation of the potential usefulness of saliency maps in a realistic use case. This is of great importance, given that user studies have yet to find experimental setups in which saliency maps are really useful to explain models trained on nontrivial tasks.[72]

5 Conclusion

In this tutorial paper, we first reviewed the state-of-the-art of saliency maps generation. We described the two approaches (post-hoc methods and attention models) and illustrated each category with several examples. One of the main differences between the post-hoc methods and the attention models is the computation cost. When considering only the training cost, post-hoc methods are more efficient because they do not require retraining a model. However, if one considers the inference cost, attention models can be more efficient, especially when compared to the perturbation-based methods. Indeed, post-hoc methods can require a number of samples that ranges from a few to several thousand, whereas attention models only require one inference. Practitioners desiring to generate saliency maps should then be guided by a compromise between training computation time and inference computation time. However, the most relevant criterion to compare these two types of approaches is their interpretability, both in terms of reliability/faithfulness and understandability by the users. Despite this, there currently are only a few works that compare these approaches[62,65] and no conclusion can be drawn yet.

The lack of consensus on the interpretability of attention models and post-hoc methods is also due to the difficulty to evaluate saliency maps. For this reason, we also described the currently existing objective and subjective evaluation protocols. Even though many frameworks were introduced, the community has yet to agree on which ones are relevant, and for what use. Some evaluation protocols also pose unsolved problems. For example, it has been shown that the multistep metrics are likely to generate OOD samples,[64] which questions the reliability of the obtained values and one can argue that single-step metrics could have the same issue as they also fill the input image with black pixels. Assessing and addressing this issue would allow the community to build more robust faithfulness evaluation methods.

Acknowledgments

This work was funded by ANR—Next grant DL4IVF (2017) (Grant No. 16-IDEX-0007). The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. M. Mitchell et al., "Model cards for model reporting," in *Proc. Conf. Fairness, Accountability, and Transparency, FAT* '19*, Association for Computing Machinery, New York, NY, USA, pp. 220–229 (2019).
2. C. Chen et al., "This looks like that: deep learning for interpretable image recognition," in *NeurIPS* (2019).
3. M. Nauta, R. van Bree, and C. Seifert, "Neural prototype trees for interpretable fine-grained image recognition," (2021).
4. R. Guidotti, "Counterfactual explanations and how to find them: literature review and benchmarking," *Data Min. Knowl. Discov.* (2022).
5. Q. Zhang et al., "Interpreting cnn knowledge via an explanatory graph," (2018).
6. B. Kim et al., "Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV)," in *ICML* (2018).
7. M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. and Data Min., KDD '16*, Association for Computing Machinery, New York, NY, USA, pp. 1135–1144 (2016).
8. B. Efron et al., "Least angle regression," *Ann. Stat.* **32**(2), 407–451 (2004).
9. S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Adv. in Neural Inf. Process. Syst.*, I. Guyon et al., Eds., Curran Associates, Inc., Vol. **30** (2017).
10. V. Petsiuk, A. Das, and K. Saenko, "RISE: randomized input sampling for explanation of black-box models," (2018).
11. R. R. Selvaraju et al., "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 618–626 (2017).
12. A. Chattopadhay et al., "Grad-CAM++ generalized gradient-based visual explanations for deep convolutional networks," in *IEEE Winter Conf. Appl. of Comput. Vis. (WACV)*, pp. 839–847 (2018).
13. H. Wang et al., "Score-cam: score-weighted visual explanations for convolutional neural networks," in *IEEE/CVF Conf. Comput. Vis. and Pattern Recognit. Workshops (CVPRW)*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 111–119 (2020).
14. R. Fu et al., "Axiom-based grad-cam: towards accurate visualization and explanation of CNNs," (2020).
15. K. Ahmed Asif Fuad et al., "Features understanding in 3D CNNs for actions recognition in video," in *Tenth Int. Conf. Image Process. Theory, Tools and Appl. (IPTA)*, pp. 1–6 (2020).
16. L. Bourroux et al., "Multi layered feature explanation method for convolutional neural networks," *Lect. Notes Comput. Sci.* **13363**, 603–614 (2022).

17. J. T. Springenberg et al., "Striving for simplicity: the all convolutional net," CoRR abs/1412.6806 (2015).
18. M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. 34th Int. Conf. Mach. Learn., ICML'17*, JMLR, Vol. **70**, pp. 3319–3328 (2017).
19. J. Adebayo et al., "Local explanation methods for deep neural networks lack sensitivity to parameter values," (2018).
20. A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *ICML'17*, JMLR, pp. 3145–3153 (2017).
21. S. Bach et al., "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS One* **10**, 1–46 (2015).
22. K. He et al., "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. and Pattern Recognit. (CVPR)*, pp. 770–778 (2016).
23. A. Vaswani et al., "Attention is all you need," in *Adv. in Neural Inf. Process. Syst.*, I. Guyon et al., Eds., Curran Associates, Inc., Vol. **30** (2017).
24. T. B. Brown et al., "Language models are few-shot learners," (2020).
25. A. Dosovitskiy et al., "An image is worth 16×16 words: transformers for image recognition at scale," (2020).
26. M. Caron et al., "Emerging properties in self-supervised vision transformers," (2021).
27. H. Chefer, S. Gur, and L. Wolf, "Transformer interpretability beyond attention visualization," in *Proc. IEEE/CVF Conf. Comput. Vis. and Pattern Recognit. (CVPR)*, pp. 782–791 (2021).
28. O. Barkan et al., "Grad-SAM: explaining transformers via gradient self-attention maps," in *Proc. 30th ACM Int. Conf. Inf. and Knowl. Manage., CIKM '21*, Association for Computing Machinery, New York, NY, USA, pp. 2882–2887 (2021).
29. T. Hu and H. Qi, "See better before looking closer: weakly supervised data augmentation network for fine-grained visual classification," CoRR abs/1901.09891 (2019).
30. A. Imran and V. Athitsos, "Domain adaptive transfer learning on visual attention aware data augmentation for fine-grained visual categorization," *Lect. Notes Comput. Sci.* **12510**, 53–65 (2020).
31. J. Chen et al., "Attention-based cropping and erasing learning with coarse-to-fine refinement for fine-grained visual classification," *Neurocomputing* **501**, 359–369 (2022).
32. Y. Rao et al., "Counterfactual attention learning for fine-grained visual categorization and re-identification," in *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 1005–1014 (2021).
33. H. Zheng et al., "Learning multi-attention convolutional neural network for fine-grained image recognition," in *IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 5219–5227 (2017).
34. H. Fukui et al., "Attention branch network: learning of attention mechanism for visual explanation," in *IEEE/CVF Conf. Comput. Vis. and Pattern Recognit. (CVPR)*, pp. 10697–2019) 10706).
35. A. Dubey et al., "Pairwise confusion for fine-grained visual classification," (2018).
36. A. M. Obeso et al., "Visual vs internal attention mechanisms in deep neural networks for image classification and object detection," *Pattern Recognit.* **123**, 108411 (2022).
37. A. Montoya Obeso et al., "Saliency-based selection of visual content for deep convolutional neural networks," *Multimedia Tools Appl.* **78**, 9553–9576 (2019).
38. F. Wang et al., "Residual attention network for image classification," in *IEEE Conf. Comput. Vis. and Pattern Recognit. (CVPR)*, pp. 6450–6458 (2017).
39. H. Zhang et al., "Resnest: split-attention networks," arXiv:2004.08955 (2020).
40. Y. Dai et al., "Attentional feature fusion," in *IEEE Winter Conf. Appl. of Comput. Vis. (WACV)*, pp. 3559–3568 (2021).
41. J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE/CVF Conf. Comput. Vis. and Pattern Recognit.*, pp. 7132–7141 (2018).
42. D. Rymarczyk et al., "Interpretable image classification with differentiable prototypes assignment," (2021).
43. J. Donnelly, A. J. Barnett, and C. Chen, "Deformable protoPNet: an interpretable image classifier using deformable prototypes," in *Proc. IEEE/CVF Conf. Comput. Vis. and Pattern Recognit. (CVPR)*, pp. 10265–10275 (2022).

44. Z. Huang and Y. Li, "Interpretable and accurate fine-grained recognition via region grouping," (2020).
45. J. Wang et al., "Interpretable image recognition by constructing transparent embedding space," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 895–904 (2021).
46. G. Singh and K.-C. Yow, "These do not look like those: an interpretable deep learning model for image recognition," *IEEE Access* **9**, 41482–41493 (2021).
47. G. Singh and K.-C. Yow, "An interpretable deep learning model for covid-19 detection with chest x-ray images," *IEEE Access* **9**, 85198–85208 (2021).
48. J. Choe, S. Lee, and H. Shim, "Attention-based dropout layer for weakly supervised single object localization and semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(12), 4256–4271 (2021).
49. H. Zheng et al., "Looking for the devil in the details: learning trilinear attention sampling network for fine-grained image recognition," in *Proc. IEEE Conf. Comput. Vis. and Pattern Recognit.*, pp. 5012–5021 (2019).
50. T. Gomez et al., "BR-NPA: a non-parametric high-resolution attention model to improve the interpretability of attention," (2022).
51. C. Wah et al., "*The Caltech-UCSD Birds-200-2011 dataset*," Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011).
52. S. Maji et al., "*Fine-Grained Visual Classification of Aircraft*," Tech. Rep. (2013).
53. J. Krause et al., "3D object representations for fine-grained categorization," in *4th Int. IEEE Workshop on 3D Represent. and Recognit. (3dRR-13)*, Sydney, Australia (2013).
54. S. Desai and H. G. Ramaswamy, "Ablation-CAM: visual explanations for deep convolutional network via gradient-free localization," in *IEEE Winter Conf. Appl. of Comput. Vis. (WACV)*, pp. 972–980 (2020).
55. H. Jung and Y. Oh, "Lift-CAM: towards better explanations for class activation mapping," ArXiv abs/2102.05228 (2021).
56. D. Smilkov et al., "Smoothgrad removing noise by adding noise," (2017).
57. W. Xiao, Z. Ding, and H. Liu, "Learnable visual words for interpretable image recognition," (2022).
58. J. Adebayo et al., "Sanity checks for saliency maps," in *Adv. in Neural Inf. Process. Syst.*, S. Bengio et al., Eds., Curran Associates, Inc., Vol. **31** (2018).
59. O.-M. Camburu et al., "The struggles of feature-based explanations: shapley values vs. minimal sufficient subsets," (2020).
60. S. Jain and B. C. Wallace, "Attention is not explanation," in *Proc. Conf. of the North Am. Chapter of the Assoc. for Comput. Linguistics: Hum. Lang. Technol., Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Minneapolis, Minnesota, pp. 3543–3556 (2019).
61. S. Wiegreffe and Y. Pinter, "Attention is not not explanation," (2019).
62. J. Bastings and K. Filippova, "The elephant in the interpretability room: why use attention as explanation when we have saliency methods?" in *Proc. Third BlackboxNLP Workshop on Anal. and Interpreting Neural Netw. for NLP*, Association for Computational Linguistics, pp. 149–155 (2020).
63. J.-S. Denain and J. Steinhardt, "Auditing visualizations: transparency methods struggle to detect anomalous behavior," (2022).
64. T. Gomez, T. Fréour, and H. Mouchère, "Metrics for saliency map evaluation of deep learning explanation methods," *Lect. Notes Comput. Sci.* **13363**, 84–95 (2022).
65. T. Gomez, T. Fréour, and H. Mouchère, "Comparison of attention models and post-hoc explanation methods for embryo stage identification: a case study," in *XAI Workshop (ICPR 2022)*, Montréal, Canada (2022).
66. R. Tomsett et al., "Sanity checks for saliency metrics," (2019).
67. S. Z. S. Samuel et al., "Evaluation of saliency-based explainability method," (2021).
68. I. Ahern et al., "Normlime: a new feature importance metric for explaining deep neural networks," (2019).
69. W. Jin, X. Li, and G. Hamarneh, "One map does not fit all: evaluating saliency map explanation on multi-modal medical images," (2021).

70. J. V. Jeyakumar et al., "How can I explain this to you? An empirical study of deep neural network explanation methods," in *Adv. in Neural Inf. Process. Syst.*, H. Larochelle et al., Eds., Curran Associates, Inc., Vol. **33**, pp. 4211–4222 (2020).
71. S. Mohseni, J. E. Block, and E. D. Ragan, "A human-grounded evaluation benchmark for local explanations of machine learning," (2018).
72. A. Alqaraawi et al., "Evaluating saliency map explanations for convolutional neural networks: a user study," in *IUI '20*, Association for Computing Machinery, New York, NY, USA, pp. 275–285 (2020).
73. M. Narayanan et al., "How do humans understand explanations from machine learning systems? An evaluation of the human-interpretability of explanation," (2018).
74. C.-K. Yeh et al., "On completeness-aware concept-based explanations in deep neural networks," in *Adv. in Neural Inf. Process. Syst.*, H. Larochelle et al., Eds., Curran Associates, Inc., Vol. **33**, pp. 20554–20565 (2020).
75. D. Slack et al., "Assessing the local interpretability of machine learning models," (2019).
76. I. Lage et al., "An evaluation of the human-interpretability of explanation," (2019).
77. D. Slack et al., "Reliable post hoc explanations: modeling uncertainty in explainability," in *Adv. in Neural Inf. Process. Syst.*, Vol. **34** (2021).
78. B. Kim, R. Khanna, and O. O. Koyejo, "Examples are not enough, learn to criticize! criticism for interpretability," in *Adv. in Neural Inf. Process. Syst.*, D. Lee et al., Eds., Curran Associates, Inc., Vol. **29** (2016).
79. I. Lage et al., "Human-in-the-loop interpretability prior," in *Adv. in Neural Inf. Process. Syst.*, S. Bengio et al., Eds., Curran Associates, Inc., Vol. **31** (2018).
80. S. Knapič et al., "Explainable artificial intelligence for human decision support system in the medical domain," *Mach. Learn. Knowl. Extraction* **3**(3), 740–770 (2021).
81. E. M. Kenny et al., "Explaining black-box classifiers using post-hoc explanations-by-example: the effect of explanations and error-rates in xai user studies," *Artif. Intell.* **294**, 103459 (2021).
82. J. Adebayo et al., "Debugging tests for model explanations," (2020).
83. T. Huber et al., "Local and global explanations of agent behavior: integrating strategy summaries with saliency maps," *Artif. Intell.* **301**, 103571 (2021).
84. R. El Shawi et al., "Interpretability in healthcare a comparative study of local machine learning interpretability techniques," in *IEEE 32nd Int. Symp. Comput.-Based Med. Syst. (CBMS)*, pp. 275–280 (2019).
85. G. Yona and D. Greenfeld, "Revisiting sanity checks for saliency maps," in *eXplainable AI Approaches for Debugging and Diagnosis* (2021).

Tristan Gomez received his engineering degree in computer science from Polytech Nantes, France, in 2019. He is currently pursuing a PhD in Digital Science Laboratory of Nantes (LS2N), France. His research interest includes image classification, interpretable deep models, attention-based architectures, and the application of deep learning to *in vitro* fertilization (IVF).

Harold Mouchère has been a full professor at the University of Nantes since 2018. His research is mainly focused on the analysis of handwritten graphical languages. The application domains are mainly the recognition of handwritten mathematical expressions and more recently the analysis of old handwritten documents, machine vision, and medical imaging.