Pattern Recognition 65 (2017) 211–222

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Explaining nonlinear classification decisions with deep Taylor decomposition

Grégoire Montavon[a,*], Sebastian Lapuschkin[b], Alexander Binder[c], Wojciech Samek[b,*], Klaus-Robert Müller[a,d,**]

[a] Department of Electrical Engineering & Computer Science, Technische Universität Berlin, Marchstr. 23, Berlin 10587, Germany
[b] Department of Video Coding & Analytics, Fraunhofer Heinrich Hertz Institute, Einsteinufer 37, Berlin 10587, Germany
[c] Information Systems Technology & Design, Singapore University of Technology and Design, 8 Somapah Road, Building 1, Level 5, 487372, Singapore
[d] Department of Brain & Cognitive Engineering, Korea University, Anam-dong 5ga, Seongbuk-gu, Seoul 136-713, South Korea

ARTICLE INFO

Keywords:
Deep neural networks
Heatmapping
Taylor decomposition
Relevance propagation
Image recognition

ABSTRACT

Nonlinear methods such as Deep Neural Networks (DNNs) are the gold standard for various challenging machine learning problems such as image recognition. Although these methods perform impressively well, they have a significant disadvantage, the lack of transparency, limiting the interpretability of the solution and thus the scope of application in practice. Especially DNNs act as black boxes due to their multilayer nonlinear structure. In this paper we introduce a novel methodology for interpreting generic multilayer neural networks by decomposing the network classification decision into contributions of its input elements. Although our focus is on image classification, the method is applicable to a broad set of input data, learning tasks and network architectures. Our method called deep Taylor decomposition efficiently utilizes the structure of the network by backpropagating the explanations from the output to the input layer. We evaluate the proposed method empirically on the MNIST and ILSVRC data sets.

1. Introduction

Nonlinear models have been used since the advent of machine learning (ML) methods and are integral part of many popular algorithms. They include, for example, graphical models [1], kernels [2,3], Gaussian processes [4], neural networks [5–7], boosting [8], or random forests [9]. Recently, a particular class of nonlinear methods, Deep Neural Networks (DNNs), revolutionized the field of automated image classification by demonstrating impressive performance on large benchmark data sets [10–12]. Deep networks have also been applied successfully to other research fields such as natural language processing [13,14], human action recognition [15–17], or physics [18,19].

Although these models are highly successful in terms of performance, they have a drawback of acting like a *black box* in the sense that it is not clear *how* and *why* they arrive at a particular classification decision. This lack of transparency is a serious disadvantage as it prevents a human expert from being able to verify, interpret, and understand the reasoning of the system. In this paper, we consider the problem of explaining classification decisions of a machine learning model in terms of input variables. For instance, for image classification problems, the classifier should not only indicate whether an image of interest belongs to a certain category or not, but also explain what structures (e.g. pixels in the image) were the basis for its decision.

Linear models readily provide explanations in terms of input variables [20–22], however, due to their limited expressive power, they cannot be applied to complex tasks such as explaining image classifications. Explanation methods for complex nonlinear models such as convolutional neural networks can be categorized as follows: (1) *functional approaches* [23] where the explanation results from the local analysis of the prediction function, for example, sensitivity analysis or Taylor series expansion, and (2) *message passing approaches* [24,25] that view the prediction as the output of a computational graph, and where the explanation is obtained by running a backward pass in that graph.

A main goal of this paper is to reconcile in the context of deep neural networks the functional and message passing approaches for producing these explanations.[1] Specifically, we view each neuron of a

* Corresponding authors.
** Corresponding author at: Department of Electrical Engineering & Computer Science, Technische Universität Berlin, Marchstr. 23, Berlin 10587, Germany
E-mail addresses: gregoire.montavon@tu-berlin.de (G. Montavon), sebastian.lapuschkin@hhi.fraunhofer.de (S. Lapuschkin), alexander_binder@sutd.edu.sg (A. Binder), wojciech.samek@hhi.fraunhofer.de (W. Samek), klaus-robert.mueller@tu-berlin.de (K.-R. Müller).

[1] Similarly, error backpropagation [26] used for training neural networks also offers both a function-based interpretation (gradient evaluation) and a message passing interpretation (chain-rule for derivatives).

http://dx.doi.org/10.1016/j.patcog.2016.11.008
Received 11 May 2016; Received in revised form 8 August 2016; Accepted 12 November 2016
Available online 30 November 2016
0031-3203/

deep network as a function that can be expanded and decomposed on its input variables. The decompositions of multiple neurons are then aggregated or propagated backwards, resulting in a "deep Taylor decomposition". Furthermore, we will show how the propagation rules derived from deep Taylor decomposition relate to those heuristically defined by [25].

Because of the theoretical focus of this paper, we do not perform a broader empirical comparison with other recently proposed methods for explanation, however, we refer to [27] for that matter.

1.1. Related work

There has been a significant body of work focusing on the analysis and understanding of nonlinear classifiers. Some methods seek to provide a *global* understanding of the trained model, by measuring important characteristics of it, such as the noise and relevant dimensionality of its feature space(s) [28–30], its invariance to certain transformations of the data [31], the role of particular neurons [32], or its global decision structure [33,34]. Other methods focus instead on the interpretation of *individual* predictions. The method proposed in [35] explains predictions in terms of input variables by locally evaluating the gradient of the decision function. Simonyan et al. [23] incorporate saliency information into the explanation by multiplying the gradient by the actual data point. To determine the importance of input variables for a particular prediction, Landecker et al. [36] proposed a contribution propagation approach for hierarchical networks, applying at each unit of the network a propagation rule that obeys a conservation property.

Recent work has focused on the problem of understanding of state-of-the-art GPU-trained convolutional neural networks for image classification [23–25,37], offering new insights into these highly complex models. The deconvolution method proposed by Zeiler and Fergus [24] was designed to visualize and understand the features of state-of-the-art convolutional neural networks with max-pooling and rectified linear units. The method performs a backpropagation pass on the network, where a set of rules is applied uniformly to all layers of the network, resulting in an assignment of values onto pixels. The method however does not aim to attribute a defined meaning to the assigned pixel values, except for the fact that they should form a visually interpretable pattern. For the same convolutional neural network models, the layer-wise relevance propagation method of Bach et al. [25] applies at each neuron of the network a propagation rule with a conservative property, resulting in an assignment of values onto pixels which is directly interpretable as their importance for the classification decision. While scoring high quantitatively [27], the choice of propagation rules was mainly heuristic and lacked a strong theoretical justification.

A theoretical foundation to the problem of measuring the importance of input variables for a prediction, can be found in the Taylor decomposition of a nonlinear function. The approach was described by Bazen and Joutard [38] as a nonlinear generalization of the Oaxaca method in econometrics [21]. The idea was subsequently introduced in the context of image analysis [23,25] for the purpose of explaining machine learning classifiers.

As an alternative to propagation methods, spatial response maps [39] build heatmaps by looking at the neural network output while sliding the neural network in the pixel space. Attention models based on neural networks, trained to classify an image from only a few glimpses of it [40], readily provide a spatial interpretation for the classification decision. Similar models can also visualize what part of an image is relevant at a given time in some temporal context [41]. However, these dynamical models can be significantly more complex to design and train.

2. Pixel-wise decomposition of a function

In this section, we describe the general concept of explaining a neural network decision by decomposing the function value (i.e. neural network output) onto the input variables in an amount that matches the respective relevance of these input variables to the function value. After enumerating a certain number of desirable properties of a decomposition, we will present in Sections 2.1 and 2.2 two simple solutions to this problem. Because all subsequent empirical evaluations focus on the problem of image recognition, we call the input variables "pixels", and use the letter p for indexing them. Also, we employ the term "heatmap" to designate the set relevance scores assigned to pixels of an image. Despite the image-related terminology, the concept is applicable to other input domains such as vector spaces, time series, or more generally any type of input domain whose elements can be processed by a neural network[2].

Let us consider a positive-valued function $f: \mathbb{R}^d \to \mathbb{R}^+$. In the context of image classification, the input $\boldsymbol{x} \in \mathbb{R}^d$ of this function is an image. The image can be viewed as a set of pixel values $\boldsymbol{x} = \{x_p\}$ where p denotes a particular pixel. The function $f(\boldsymbol{x})$ quantifies the presence of a certain type of object(s) in the image. A function value $f(\boldsymbol{x}) = 0$ indicates an absence of it. On the other hand, a function value $f(\boldsymbol{x}) > 0$ expresses its presence with a certain degree of certainty, or in a certain amount.

We would like to associate to each pixel p in the image a *relevance score* $R_p(\boldsymbol{x})$, that indicates for an image $\boldsymbol{x}$ to what extent the pixel p contributes to explaining the classification decision $f(\boldsymbol{x})$. The relevance of each pixel can be stored in a heatmap denoted by $\boldsymbol{R}(\boldsymbol{x}) = \{R_p(\boldsymbol{x})\}$ of same dimensions as $\boldsymbol{x}$ and can be visualized as an image. A heatmapping should satisfy properties that we define below:

Definition 1. A heatmapping $\boldsymbol{R}(\boldsymbol{x})$ is *conservative* if the sum of assigned relevances in the pixel space corresponds to the total relevance detected by the model:

$$\forall\, \boldsymbol{x}\colon f(\boldsymbol{x}) = \sum_p R_p(\boldsymbol{x}).$$

Definition 2. A heatmapping $\boldsymbol{R}(\boldsymbol{x})$ is *positive* if all values forming the heatmap are greater or equal to zero, that is:

$$\forall\, \boldsymbol{x}, p\colon R_p(\boldsymbol{x}) \geq 0$$

The first property ensures that the total redistributed relevance corresponds to the extent to which the object in the input image is detected by the function $f(\boldsymbol{x})$. The second property forces the heatmapping to assume that the model is devoid of contradictory evidence (i.e. no pixels can be in contradiction with the presence or absence of the detected object in the image). These two properties can be combined into the notion of *consistency*:

Definition 3. A heatmapping $\boldsymbol{R}(\boldsymbol{x})$ is *consistent* if it is conservative *and* positive. That is, it is consistent if it complies with Definitions 1 and 2.

In particular, a consistent heatmap is forced to satisfy $(f(\boldsymbol{x}) = 0) \Rightarrow (\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{0})$. That is, in absence of an object to detect, the relevance is forced to be zero everywhere in the image (i.e. empty heatmap), and not simply to have negative and positive relevance in same amount. We will use Definition 3 as a formal tool for assessing the correctness of the heatmapping techniques proposed in this paper. It was noted by [25] that there may be multiple heatmapping techniques that satisfy a particular definition. For example, we can consider a heatmapping that assigns for all images the relevance uniformly onto the pixel grid:

$$\forall\, p\colon \quad R_p(\boldsymbol{x}) = \frac{1}{d} \cdot f(\boldsymbol{x}), \tag{1}$$

[2] See [42,43] for the application of decomposition techniques to text and EEG data.

where d is the number of input dimensions. Alternately, we can consider a heatmapping where all relevance is assigned to the first pixel:

$$R_p(\boldsymbol{x}) = \begin{cases} f(\boldsymbol{x}) & \text{if } p = \text{1st pixel} \\ 0 & \text{else.} \end{cases} \tag{2}$$

Both (1) and (2) are consistent in the sense of Definition 3, however they lead to different relevance assignments. In practice, it is not possible to specify explicitly all properties that a heatmapping technique should satisfy. In the following, we give two meaningful examples of decompositions that comply with the definitions above.

2.1. Natural decomposition

A natural decomposition can be defined as a decomposition that is obtained directly from the structure of the prediction. Consider, for example, the prediction function

$$f(\boldsymbol{x}) = \sum_p \sigma_p(x_p), \tag{3}$$

where $\{\sigma_p\}$ is a set of positive nonlinear functions applying to each pixel. The relevance of each input variable can be identified as elements of the sum [22]:

$$R_p(\boldsymbol{x}) = \sigma_p(x_p).$$

If there exists for each pixel a deactivated state $\tilde{x}_p$ such that $\sigma_p(\tilde{x}_p) = 0$, then, the relevance score $R_p(\boldsymbol{x})$ can be interpreted as the effect on the prediction of deactivating pixel p. A pixel whose deactivation would cause a large drop in function value is therefore modeled as relevant.

Fig. 1 illustrates on a simple two-dimensional function the difference between this decomposition technique and another frequently used explanation technique called sensitivity analysis [44], which explains the prediction as locally evaluated squared partial derivatives. We can observe that sensitivity analysis is not related to the function value and is discontinuous in some regions of the input space. The natural decomposition, on the other hand, is continuous and also incorporates the function value, as evidenced by the continuously varying size and direction of the arrows.

While this example motivates the importance of distinguishing between decomposition and sensitivity, functions like the one of Eq. (3) are typically not expressive enough to model the high complexity of input-output relations observed in real data.

2.2. Taylor decomposition

Moving to the general case of arbitrary differentiable functions $f(\boldsymbol{x})$, we introduce a decomposition method based on the Taylor expansion of the function at some well-chosen *root point* $\tilde{\boldsymbol{x}}$. A root point is a point where $f(\tilde{\boldsymbol{x}}) = 0$. The first-order Taylor expansion of $f(\boldsymbol{x})$ is given by

$$f(\boldsymbol{x}) = f(\tilde{\boldsymbol{x}}) + \left(\frac{\partial f}{\partial \boldsymbol{x}}\Big|_{x=\tilde{x}}\right)^\top \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}}) + \varepsilon = 0 + \sum_p \underbrace{\frac{\partial f}{\partial x_p}\Big|_{x=\tilde{x}} \cdot (x_p - \tilde{x}_p)}_{R_p(\boldsymbol{x})} + \varepsilon, \tag{4}$$

where the sum $\sum_p$ runs over all pixels in the image, and $\{\tilde{x}_p\}$ are the pixel values of the root point $\tilde{\boldsymbol{x}}$. We identify the summed elements as the relevances $R_p(\boldsymbol{x})$ assigned to pixels in the image. The term ε denotes second-order and higher-order terms. Most of them involve several pixels and are therefore more difficult to redistribute. Thus, for simplicity, only the first-order terms are considered. The heatmap (composed of all identified pixel-wise relevances) can be written as the element-wise product "$\odot$" between the gradient of the function $\partial f/\partial \boldsymbol{x}$ at the root point $\tilde{\boldsymbol{x}}$ and the difference between the image and the root $(\boldsymbol{x} - \tilde{\boldsymbol{x}})$:

$$\boldsymbol{R}(\boldsymbol{x}) = \frac{\partial f}{\partial \boldsymbol{x}}\Big|_{x=\tilde{x}} \odot (\boldsymbol{x} - \tilde{\boldsymbol{x}}). \tag{5}$$

For a given classification function $f(\boldsymbol{x})$, the Taylor decomposition approach has one free variable: the choice of the root point $\tilde{\boldsymbol{x}}$ at which the Taylor expansion is performed. A good root point is one that removes what in the data point $\boldsymbol{x}$ causes the function $f(\boldsymbol{x})$ to be positive (e.g. an object in an image that is being detected), but that minimally deviates from the original point $\boldsymbol{x}$ for the Taylor expansion to be still valid. In mathematical terms, it is a point $\tilde{\boldsymbol{x}}$ with $f(\tilde{\boldsymbol{x}}) = 0$ that lies in the vicinity of $\boldsymbol{x}$ under some distance metric, for example the nearest root. If $\boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathbb{R}^d$, one can show that for a continuously differentiable function f the gradient at the nearest root always points to the same direction as the difference $\boldsymbol{x} - \tilde{\boldsymbol{x}}$, and their element-wise product is always positive, thus satisfying Definition 2. Relevance conservation in the sense of Definition 1 is however not satisfied for general functions f due to the possible presence of non-zero higher-order terms in ε. The nearest root $\tilde{\boldsymbol{x}}$ can be obtained as a solution of an optimization problem [37], by minimizing the objective

$$\min_{\boldsymbol{\xi}} \; \| \boldsymbol{\xi} - \boldsymbol{x} \|^2 \quad \text{subject to} \quad f(\boldsymbol{\xi}) = 0 \quad \text{and} \quad \boldsymbol{\xi} \in \mathcal{X},$$

where $\mathcal{X}$ is the input domain. The nearest root $\tilde{\boldsymbol{x}}$ must therefore be obtained in the general case by an iterative minimization procedure. It is time consuming when the function $f(\boldsymbol{x})$ is expensive to evaluate or differentiate, although some fast approximations do exist [45]. It is also not necessarily solvable due to the possible non-convexity of the minimization problem. A further problem with the Taylor-based approach comes from the observation in [37] that for large deep neural networks, nearest root points $\tilde{\boldsymbol{x}}$ are often imperceptibly different from the actual data point $\boldsymbol{x}$. In particular, the difference $(\boldsymbol{x} - \tilde{\boldsymbol{x}})$ is hardly interpretable visually, and thus, cannot properly play its role in Eq. (5) for supporting a pixel-wise decomposition.

Relation to sensitivity analysis: Sensitivity analysis can be viewed as a special instance of Taylor decomposition where one expands the function $f(\boldsymbol{x})$ not at a root point $\tilde{\boldsymbol{x}}$, but at a point $\boldsymbol{\xi}$, taken at an infinitesimally small distance from the actual point $\boldsymbol{x}$, in the direction of maximum gradient (i.e. $\boldsymbol{\xi} = \boldsymbol{x} - \delta \cdot \partial f/\partial \boldsymbol{x}$ with δ small). On these infinitesimal scales, the function is locally linear and the gradient is constant, and rewriting the Taylor expansion of $f(\boldsymbol{x})$ at $\boldsymbol{\xi}$ in a way that the first-order terms can be identified,

$$f(\boldsymbol{x}) = f(\boldsymbol{\xi}) + \left(\frac{\partial f}{\partial \boldsymbol{x}}\Big|_{x=\xi}\right)^\top \cdot (\boldsymbol{x} - \boldsymbol{\xi}) + \varepsilon = f(\boldsymbol{\xi}) + \sum_p \underbrace{\delta \left(\frac{\partial f}{\partial x_p}\right)^2}_{R_p(\boldsymbol{x})} + 0,$$

the direct relation between identified relevances and the squared local derivatives used in sensitivity analysis becomes clear. The resulting heatmap is positive, but not conservative since almost all relevance is absorbed by the non-redistributed zero-order term.

3. Deep Taylor decomposition

In this section, we introduce the main contribution of this paper: a novel method for explaining nonlinear predictions that we call "deep Taylor decomposition". It is applicable to a much larger class of functions than those considered in Section 2.1. It also overcomes the multiple technical limitations of the simple Taylor-based method described in Section 2.2. We will assume that the function $f(\boldsymbol{x})$ is implemented by a deep neural network, composed of multiple layers of representation, where each layer is composed of a set of neurons. Each neuron performs on its input an elementary computation consisting of a linear projection followed by a nonlinear activation function. Deep neural networks derive their high representational power from the interconnection of a large number of these neurons, each of them, realizing a small distinct subfunction.

The deep Taylor decomposition method is inspired by the divide-

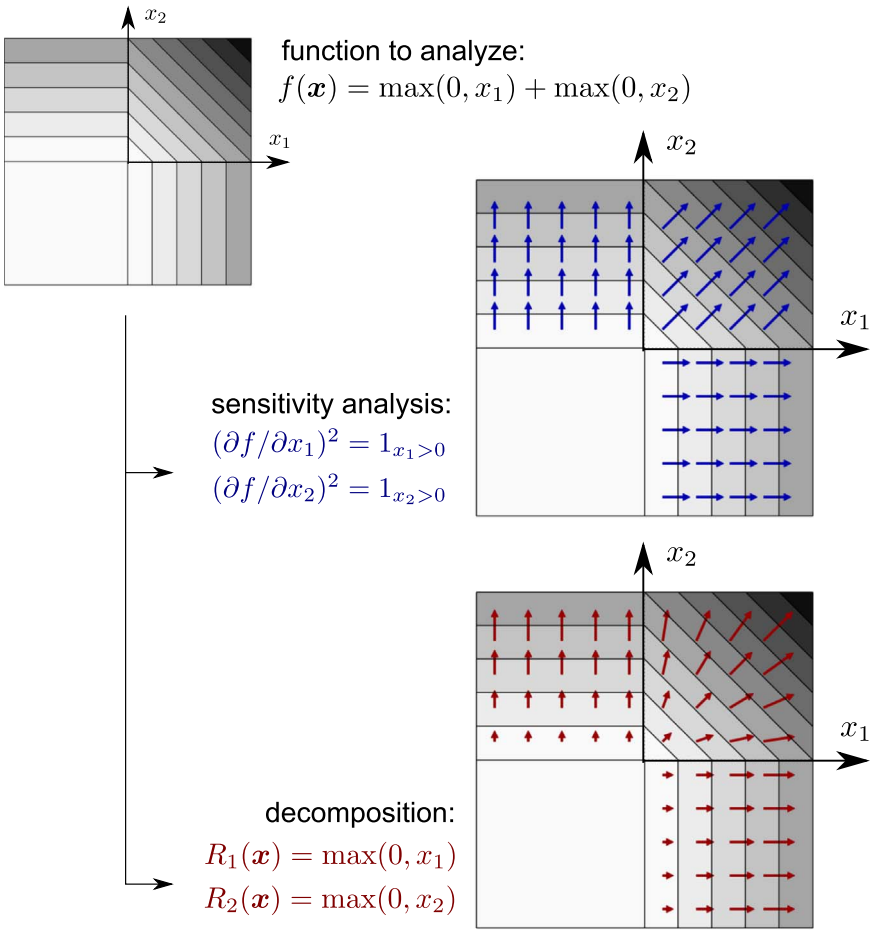

Fig. 1. Difference between sensitivity analysis and decomposition for an exemplary two-dimensional function $f(\boldsymbol{x})$. The function value is represented with contour lines. Explanations are represented as a vector field.

and-conquer paradigm, and exploits the property that the function learned by a deep network is decomposed into a set of simpler subfunctions, either enforced structurally by the neural network connectivity, or occurring as a result of training. These subfunctions can, for example, apply locally to subsets of pixels, or they can operate at a certain level of abstraction based on the layer at which they are located in the deep network. An example of neural network mapping an input image to some score indicating the presence of an object of a certain class is given in Fig. 2 (top).

Let us assume that the function $f(\boldsymbol{x})$ encoded by the output neuron x_f has been decomposed on the set of neurons at a given layer. Let x_j be one such neuron and R_j be its associated relevance. We would like to decompose R_j on the set of lower layer neurons $\{x_i\}$ to which x_j is connected. Assuming that $\{x_i\}$ and R_j are related by a function $R_j(\{x_i\})$, such decomposition onto input neurons can be obtained by Taylor decomposition. It should be noted, however, that the relevance function may in practice depend on additional variables in the neural network, for example, the relevances of upper-layer neurons $\{x_k\}$ to which x_j contributes. These top-down dependencies include the necessary information to determine whether a neuron x_j is relevant, not only based on the pattern it receives as input, but also based on its context. For now, we will take for granted, that these top-down dependencies in the relevance function are such, that one can always decompose R_j exclusively in terms of $\{x_i\}$. Practical relevance models that satisfy this property will be introduced in Section 5.

We define a root point $\{\tilde{x}_i\}^{(j)}$ of this function. Note that we choose a different root point for each neuron x_j in the current layer, hence the superscript $^{(j)}$ to identify them. The Taylor decomposition of R_j is given by:

$$R_j = \left(\frac{\partial R_j}{\partial \{x_i\}}\Big|_{\{\tilde{x}_i\}^{(j)}}\right)^{\top} \cdot (\{x_i\} - \{\tilde{x}_i\}^{(j)}) + \varepsilon_j = \sum_i \underbrace{\frac{\partial R_j}{\partial x_i}\Big|_{\{\tilde{x}_i\}^{(j)}} \cdot (x_i - \tilde{x}_i^{(j)})}_{R_{ij}} + \varepsilon_j,$$

where ε_j denotes the Taylor residual, and where $|_{\{\tilde{x}_i\}^{(j)}}$ indicates that the derivative has been evaluated at the root point $\{\tilde{x}_i\}^{(j)}$. The identified term R_{ij} is the redistributed relevance from neuron x_j to neuron x_i in the lower layer. To determine the total relevance of neuron x_i, one needs to pool relevance coming from all neurons $\{x_j\}$ to which the neuron x_i contributes:

$$R_i = \sum_j R_{ij}.$$

Combining the last two equations, we get

$$R_i = \sum_j \frac{\partial R_j}{\partial x_i}\Big|_{\{\tilde{x}_i\}^{(j)}} \cdot (x_i - \tilde{x}_i^{(j)}). \tag{6}$$

This last equation will be central for computing explicit relevance redistribution formulas based on specific choices of root points $\{\tilde{x}_i\}^{(j)}$.

It can be verified from the equations above that if $\forall_j : \sum_i R_{ij} = R_j$, in particular, when all residuals ε_j are zero, then $\sum_i R_i = \sum_j R_j$, i.e. the propagation from one layer to another is conservative in the sense of Definition 1. Moreover, if each layer-wise Taylor decomposition in the network is conservative, then, the chain of equalities $R_f = \ldots = \sum_j R_j = \sum_i R_i = \ldots = \sum_p R_p$ holds, and the global pixel-wise decomposition is thus also conservative. This chain of equalities is referred by [25] as layer-wise relevance conservation. Similarly, if Definition 2 holds for each local Taylor decomposition, the positivity of relevance scores at each layer $R_f, \ldots, \{R_j\}, \{R_i\}, \ldots, \{R_p\} \geq 0$ is also ensured. If all

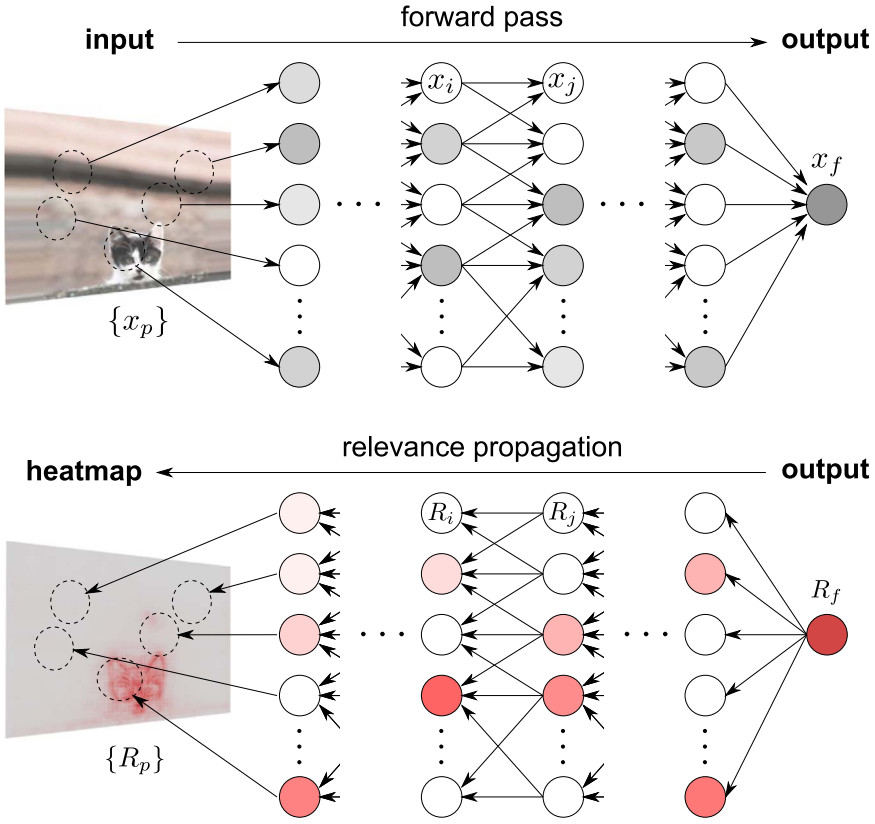

Fig. 2. Computational flow of deep Taylor decomposition. A prediction for the class "cat" is obtained by forward-propagation of the pixel values $\{x_p\}$, and is encoded by the output neuron x_f. The output neuron is assigned a relevance score $R_f = x_f$ representing the total evidence for the class "cat". Relevance is then backpropagated from the top layer down to the input, where $\{R_p\}$ denotes the pixel-wise relevance scores, that can be visualized as a heatmap.

local Taylor decompositions are consistent in the sense of Definition 3, then, the whole decomposition is consistent in the same sense.

Fig. 2 illustrates the procedure of layer-wise relevance propagation on a cartoon example where an image of a cat is presented to a deep network. If the neural network has been designed and trained successfully for the detection task, it is likely to have a structure, where neurons are modeling specific features at distinct locations. In such network, relevance redistribution is not only easier in the top layer where it has to be decided which neurons, and not pixels, are relevant for the object "cat". It is also easier in the lower layers where the relevance has already been redistributed to the relevant neurons, and where the final redistribution step only involves a few neighboring pixels.

4. Application to one-layer networks

As a starting point for better understanding deep Taylor decomposition, in particular, how it leads to practical propagation rules, we work through a simple example, with advantageous analytical properties. We consider a detection-pooling network made of one layer of nonlinearity. The network is defined as

$$x_j = \max\left(0, \sum_i x_i w_{ij} + b_j\right) \quad \text{and} \quad x_k = \sum_j x_j \tag{7}$$

where $\{x_i\}$ is a d-dimensional input, $\{x_j\}$ is a detection layer, x_k is the output, and $\theta = \{w_{ij}, b_j\}$ are the weight and bias parameters of the network. The one-layer network is depicted in Fig. 3.

The mapping $\{x_i\} \to x_k$ defines a function $g \in \mathcal{G}$, where $\mathcal{G}$ denotes the set of functions representable by this one-layer network. We will set an additional constraint on biases, where we force $b_j \leq 0$ for all j. Imposing this constraint guarantees the existence of a root point $\{\tilde{x}_i\}$ of the function g (located at the origin), and thus also ensures the applicability of standard Taylor decomposition, for which a root point is needed. We now perform the deep Taylor decomposition of this function. We start by equating the predicted output to the amount of total relevance that must be backpropagated, i.e., $R_k = x_k$. The relevance for the top layer can now be expressed in terms of lower-layer neurons as:

$$R_k = \sum_j x_j \tag{8}$$

Having established the mapping between $\{x_j\}$ and R_k, we would like to redistribute R_k onto neurons $\{x_j\}$. Using Taylor decomposition (Eq. (4)), redistributed relevances R_j can be written as:

$$R_j = \frac{\partial R_k}{\partial x_j}\Big|_{\{\tilde{x}_j\}} \cdot (x_j - \tilde{x}_j). \tag{9}$$

We still need to choose a root point $\{\tilde{x}_j\}$. The list of all root points of this function is given by the plane equation $\sum_j \tilde{x}_j = 0$. However, for the root to play its role of reference point, it should be admissible. Here, because of the application of the function $\max(0, \cdot)$ in the preceding layer, the root point must be positive. The only point that is both a root ($\sum_j \tilde{x}_j = 0$) and admissible ($\forall\, j\colon \tilde{x}_j \geq 0$) is $\{\tilde{x}_j\} = \mathbf{0}$. Choosing this root point in Eq. (9), and

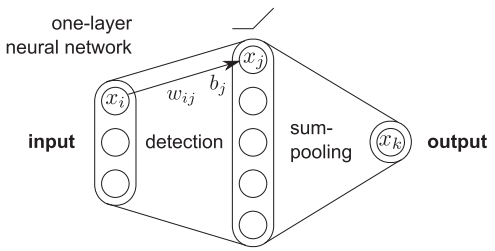

Fig. 3. Detection-pooling network that implements Eqs. (7): the first layer detects features in the input space, the second layer pools the detected features into an output score.

observing that the derivative $\frac{\partial R_k}{\partial x_j} = 1$, we obtain the first rule for relevance redistribution:

$$R_j = x_j \tag{10}$$

In other words, the relevance must be redistributed on the neurons of the detection layer in same proportion as their activation value. Trivially, we can also verify that the relevance is conserved during the redistribution process ($\sum_j R_j = \sum_j x_j = R_k$) and positive ($R_j = x_j \geq 0$). Let us now express the relevance R_j as a function of the input neurons $\{x_i\}$. Because R_j=x_j as a result of applying the propagation rule of Eq. (10), we can write

$$R_j = \max\left(0, \sum_i x_i w_{ij} + b_j\right), \tag{11}$$

that establishes a mapping between $\{x_i\}$ and R_j. To obtain redistributed relevances $\{R_i\}$, we will apply Taylor decomposition again on this new function. The identification of the redistributed total relevance $\sum_j R_j$ onto the preceding layer was identified in Eq. (6) as:

$$R_i = \sum_j \frac{\partial R_j}{\partial x_i}\Big|_{\{\tilde{x}_i\}^{(j)}} \cdot (x_i - \tilde{x}_i^{(j)}). \tag{12}$$

Relevances $\{R_i\}$ can therefore be obtained by performing as many Taylor decompositions as there are neurons in the hidden layer. We will introduce below various methods for choosing a root $\{\tilde{x}_i\}^{(j)}$ that consider the diversity of possible input domains $\mathcal{X} \subseteq \mathbb{R}^d$ to which the data belongs. Each choice of input domain and associated method to find a root will lead to a different rule for propagating relevance $\{R_j\}$ to $\{R_i\}$.

4.1. Unconstrained input space and the w^2-rule

We first consider the simplest case where any real-valued input is admissible ($\mathcal{X} = \mathbb{R}^d$). In that case, we can always choose the root point $\{\tilde{x}_i\}^{(j)}$ that is nearest in the Euclidean sense to the actual data point $\{x_i\}$. When $R_j > 0$, the nearest root of R_j as defined in Eq. (11) is the intersection of the plane equation $\sum_i \tilde{x}_i^{(j)} w_{ij} + b_j = 0$, and the line of maximum descent $\{\tilde{x}_i\}^{(j)} = \{x_i\} + t \cdot \boldsymbol{w}_j$, where $\boldsymbol{w}_j$ is the vector of weight parameters that connects the input to neuron x_j and $t \in \mathbb{R}$. The intersection of these two subspaces is the nearest root point. It is given by $\{\tilde{x}_i\}^{(j)} = \{x_i - \frac{w_{ij}}{\sum_i w_{ij}^2}(\sum_i x_i w_{ij} + b_j)\}$. Injecting this root into Eq. (12), the redistributed relevance becomes:

$$R_i = \sum_j \frac{w_{ij}^2}{\sum_{i'} w_{i'j}^2} R_j \tag{13}$$

The propagation rule consists of redistributing relevance according to the square magnitude of the weights, and pooling relevance across all neurons j. This rule is also valid for R_j=0, where the actual point $\{x_i\}$ is already a root, and for which no relevance needs to be propagated.

Proposition 1. *For all $g \in \mathcal{G}$, the deep Taylor decomposition with the w^2-rule is consistent in the sense of Definition 3.*

The w^2-rule resembles the rule by [46,44] for determining the importance of input variables in neural networks, where absolute values of w_{ij} are used in place of squared values. It is important to note that the decomposition that we propose here is modulated by the upper layer data-dependent R_js, which leads to an individual explanation for each data point.

4.2. Constrained input space and the z-rules

When the input domain is restricted to a subset $\mathcal{X} \subset \mathbb{R}^d$, the nearest root of R_j in the Euclidean sense might fall outside of $\mathcal{X}$. Finding the nearest root in this constrained input space can be difficult. An alternative is to further restrict the search domain to a subset of $\mathcal{X}$ where nearest root search becomes feasible again. We first study the case $\mathcal{X} = \mathbb{R}^d_+$, which arises, for example in feature spaces that follow the application of rectified linear units. In that case, we restrict the search domain to the segment $(\{x_i 1_{w_{ij}<0}\}, \{x_i\}) \subset \mathbb{R}^d_+$, that we know contains at least one root. The relevance propagation rule then becomes:

$$R_i = \sum_j \frac{z_{ij}^+}{\sum_{i'} z_{i'j}^+} R_j$$

(called z^+-rule), where $z_{ij}^+ = x_i w_{ij}^+$, and where w_{ij}^+ denotes the positive part of w_{ij}. This rule corresponds for positive input spaces to the $\alpha\beta$-rule proposed by [25] with $\alpha = 1$ and $\beta = 0$. The z^+-rule will be used in Section 5 to propagate relevances in higher layers of a neural network where neuron activations are positive.

Proposition 2. *For all $g \in \mathcal{G}$ and data points $\{x_i\} \in \mathbb{R}^d_+$, the deep Taylor decomposition with the z^+ -rule is consistent in the sense of Definition 3.*

For image classification tasks, pixel spaces are typically subjects to box-constraints, where an image has to be in the domain $\mathcal{B} = \{\{x_i\} : \forall_{i=1}^d\, l_i \leq x_i \leq h_i\}$, where $l_i \leq 0$ and $h_i \geq 0$ are the smallest and largest admissible pixel values for each dimension. In that new constrained setting, we can restrict the search for a root on the segment $(\{l_i 1_{w_{ij}>0} + h_i 1_{w_{ij}<0}\}, \{x_i\}) \subset \mathcal{B}$, where we know that there is at least one root at its first extremity. Finding the nearest root on that segment and injecting it into Eq. (12), we obtain the relevance propagation rule:

$$R_i = \sum_j \frac{z_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_{i'} z_{i'j} - l_i w_{i'j}^+ - h_i w_{i'j}^-} R_j$$

(called $z^{\mathcal{B}}$-rule), where $z_{ij} = x_i w_{ij}$, and where we note the presence of data-independent additive terms in the numerator and denominator. The idea of using an additive term in the denominator was formerly proposed by [25] and called ϵ-stabilized rule. However, the objective of [25] was to make the denominator non-zero to avoid numerical instability, while in our case, the additive terms serve to enforce positivity.

Proposition 3. *For all $g \in \mathcal{G}$ and data points $\{x_i\} \in \mathcal{B}$, the deep Taylor decomposition with the $z^{\mathcal{B}}$ -rule is consistent in the sense of Definition 3.*

Detailed derivations of the proposed rules, proofs of Propositions 1–3, and algorithms that implement these rules efficiently are given in the supplement.

5. Application to deep networks

In order to represent efficiently complex hierarchical problems, one needs deeper architectures. These architectures are typically made of several layers of nonlinearity, where each layer extracts features at different scale. An example of deep architecture is shown in Fig. 4 (left). In this example, the input is first processed by feature extractors localized in the pixel space. The resulting features are combined into more complex mid-level features that cover more pixels. Finally, these more complex features are combined in a final stage of nonlinear mapping, that produces a score determining whether the object to detect is present in the input image or not. A practical example of deep network with similar hierarchical architecture, and that is frequently used for image recognition tasks, is the convolutional neural network [47]. In Section 3, we have assumed the existence and knowledge of a functional mapping between the neuron activities at a given layer and relevances in the higher layer. This was the case for the small network of Section 4. However, in deeper architectures, the mapping may be unknown (although it may still exist). In order to redistribute the relevance from higher to lower layers, one needs to make this mapping explicit. For this purpose, we introduce the concept of relevance model.

A relevance model is a function that maps a set of neuron activations at a given layer to the relevance of a neuron in a higher layer, and whose output can be redistributed onto its input variables,

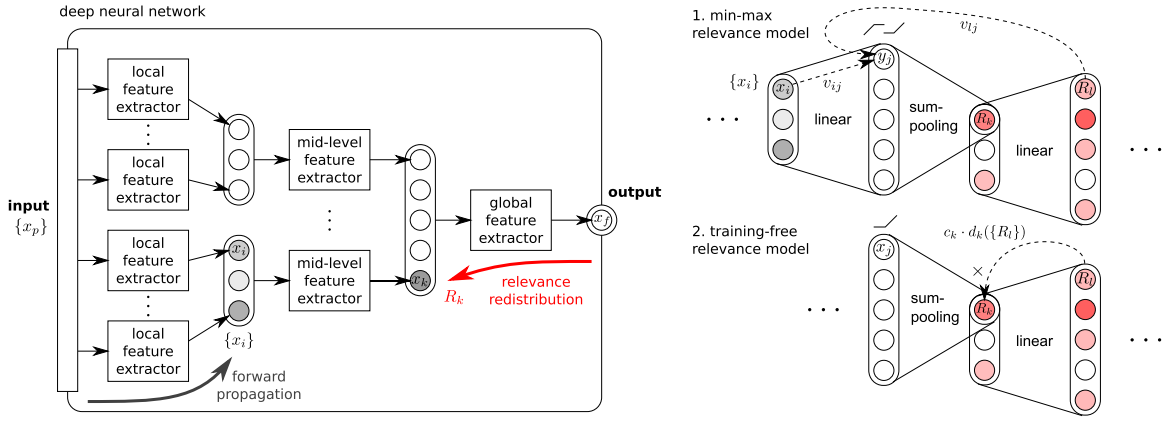

Fig. 4. Left: Example of a 3-layer deep network, composed of increasingly high-level feature extractors. Right: Diagram of the two proposed relevance models for redistributing relevance onto lower layers.

for the purpose of propagating relevance backwards in the network. For the deep network of Fig. 4 (left), on can for example, try to predict R_k from $\{x_i\}$, which then allows us to decompose the predicted relevance R_k into lower-layer relevances $\{R_i\}$. The relevance models we will consider borrow the structure of the one-layer network studied in Section 4, and for which we have already derived a deep Taylor decomposition.

Upper-layer relevance is not only determined by input neuron activations of the considered layer, but also by high-level information (i.e. abstractions) that have been formed in the top layers of the network. These high-level abstractions are necessary to ensure a global cohesion between low-level parts of the heatmap.

5.1. Min–max relevance model

We first consider a trainable relevance model of R_k. This relevance model is illustrated in Fig. 4 (top right) and is designed to incorporate both bottom-up and top-down information, in a way that the relevance can still be fully decomposed in terms of input neurons. It is defined as

$$y_j = \max\left(0, \sum_i x_i v_{ij} + a_j\right) \quad \text{and} \quad \widehat{R}_k = \sum_j y_j.$$

where $a_j = \min(0, \sum_l R_l v_{lj} + d_j)$ is a negative bias that depends on upper-layer relevances, and where $\sum_l$ runs over the detection neurons of that upper-layer. This negative bias plays the role of an inhibitor, in particular, it prevents the activation of the detection unit y_j of the relevance model in the case where no upper-level abstraction in $\{R_l\}$ matches the feature detected in $\{x_i\}$.

The parameters $\{v_{ij}, v_{lj}, d_j\}$ of the relevance model are learned by minimization of the mean square error objective

$$\min \langle (\widehat{R}_k - R_k)^2 \rangle,$$

where R_k is the true relevance, $\widehat{R}_k$ is the predicted relevance, and $\langle \cdot \rangle$ is the expectation with respect to the data distribution. Because the relevance model has the same structure as the one-layer network described in Section 4, in particular, because a_j is negative and only weakly dependent on the set of neurons $\{x_i\}$, one can apply the same set of rules for relevance propagation. We compute

$$R_j = y_j \tag{14}$$

for the pooling layer and

$$R_i = \sum_j \frac{q_{ij}}{\sum_{i'} q_{i'j}} R_j \tag{15}$$

for the detection layer, where $q_{ij} = v_{ij}^2$, $q_{ij} = x_i v_{ij}^+$, or $q_{ij} = x_i v_{ij} - l_i v_{ij}^+ - h_i v_{ij}^-$ if choosing the w^2-, z^+-, $z^{\mathcal{B}}$-rules respectively. This set of equations used to backpropagate relevance from R_k to $\{R_i\}$, is approximately conservative, with an approximation error that is determined by how much on average the output of the relevance model $\widehat{R}_k$ differs from the true relevance R_k.

5.2. Training-free relevance model

A large deep neural network may have taken weeks or months to train, and we should be able to explain it without having to train a relevance model for each neuron. We consider the original feature extractor

$$x_j = \max\left(0, \sum_i x_i w_{ij} + b_j\right) \quad \text{and} \quad x_k = \|\{x_j\}\|_q$$

where the L^q-norm can represent a variety of pooling operations such as sum-pooling or max-pooling. Assuming that the upper-layer has been explained by the z^+-rule, and indexing by l the detection neurons of that upper-layer, we can write the relevance R_k as

$$R_k = \sum_l \frac{x_k w_{kl}^+}{\sum_{k'} x_{k'} w_{k'l}^+} R_l.$$

Taking x_k out of the sum, and using the identity $\sum_j x_j = \|\{x_j\}\|_1$ for $x_j \geq 0$, we can rewrite the relevance as

$$R_k = \left(\sum_j x_j\right) \cdot c_k \cdot d_k$$

where $c_k = \frac{\|\{x_j\}\|_q}{\|\{x_j\}\|_1}$ is a L^q/L^1 pooling ratio, and $d_k = \sum_l \frac{w_{kl}^+ R_l}{\sum_{k'} x_{k'} w_{k'l}^+}$ is a top-down contextualization term. Modeling the terms c_k and d_k as constant under a perturbation of the activities $\{x_j\}$, we obtain the "training-free" relevance model, that we illustrate in Fig. 4 (bottom right). We give below some arguments that support the modeling of c_k and d_k as constants.

First, we can observe that c_k is indeed constant under certain transformations such as a homogeneous rescaling of the activations $\{x_j\}$, or any permutation of neurons activations within the pool. More generally, if we consider a sufficiently large pool of neurons $\{x_j\}$, independent variations of individual neuron activations within the pool can be viewed as swapping activations between neurons without changing the actual value of these activations. These repeated swaps also keep the norms and their ratio constant. For the top-down term d_k, we remark that the most direct way it is influenced by $\{x_j\}$ is through the variable $x_{k'}$ of the sum in the denominator of d_k, when $k' = k$. As the sum combines many neuron activations, the effect of $\{x_j\}$ on d_k can also be expected to be very limited. Modeling c_k and d_k as constants enables us to backpropagate the relevance on the lower layers: Because the relevance model R_k above has the same structure as the network of Section 4 (up to a constant factor $c_k d_k$), it is easy to

derive its Taylor decomposition, in particular, we obtain the rules

$$R_j = \frac{x_j}{\sum_{j'} x_{j'}} R_k,$$

where relevance is redistributed in proportion to activations in the detection layer, and

$$R_i = \sum_j \frac{q_{ij}}{\sum_{i'} q_{i'j}} R_j,$$

where $q_{ij} = w_{ij}^2$, $q_{ij} = x_i w_{ij}^+$, or $q_{ij} = x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-$, corresponding to the w^2-, z^+-, $z^{\mathcal{B}}$-rules respectively. If we choose the z^+-rule for that layer again, the same training-free decomposition technique can be applied to the layer below, and the process can be repeated until the input layer. Thus, when using the training-free relevance model, all layers of the network must be decomposed using the z^+-rule, except for the first layer to which other rules can be applied such as the w^2-rule or the $z^{\mathcal{B}}$-rule.

6. Experiments

In this section, we would like to test how well deep Taylor decomposition performs empirically, in particular, if the resulting heatmaps are able to pinpoint the relevant information in the input data. We first consider a neural network composed of two detection-pooling layers applied on a simple MNIST-based task. Then, we test our method on large convolutional neural networks for general image classification. Table 1 lists the main technical properties of the various methods used in the experiments.

6.1. Experiment on MNIST

The MNIST dataset consists of 60 000 training and 10 000 test images of size 28×28 representing handwritten digits, along with their label (from 0 to 9). We consider an artificial problem consisting of detecting the presence of a digit with label 0–3 in an image of size 28×56 built as a concatenation of two MNIST digits. There is a virtually infinite number of possible combinations.

A neural network with 28×56 input neurons and one output neuron is trained on this task. The input values are coded between −0.5 (black) and +1.5 (white). The neural network is composed of a first detection-pooling layer with 400 detection neurons sum-pooled into 100 units (i.e. we sum-pool non-overlapping groups of 4 detection units). A second detection-pooling layer with 400 detection neurons is applied to the 100-dimensional output of the previous layer, and activities are sum-pooled onto a single unit representing the deep network output. Positive examples are assigned target value 100 and negative examples are assigned target value 0. The neural network is trained to minimize the mean-square error between the target values and its output x_f. Treating the supervised task as a regression problem forces the network to assign approximately the same amount of relevance to all positive examples, and as little relevance as possible to the negative ones. Weights of the network are initialized using a normal distribution of mean 0 and standard deviation 0.05. Biases are initialized to zero and constrained to be negative or zero throughout training. Training data is extended with translated versions of MNIST digits. The deep network is trained using stochastic gradient descent with minibatch size 20, for 300 000 iterations, and using a small learning rate.

Table 1
Summary of the technical properties of neural network heatmapping methods described in this paper.

	Sensitivity	Taylor	Deep Taylor (min–max)	Deep Taylor (training-free)
Conservative	*No*	*No*	*Yes*[a]	*Yes*
Positive	*Yes*	*Yes*[b]	*Yes*	*Yes*
Consistent	No	No	**Yes**[a]	**Yes**
Unique solution	**Yes**	No[c]	No[c]	**Yes**
Training-free	**Yes**	**Yes**	No	**Yes**
Fast computation	**Yes**	No	**Yes**	**Yes**

[a] up to a fitting error between the redistributed relevance and the relevance model output.
[b] using the differentiable approximation $\max(0, x) = \lim_{t\to\infty} t^{-1}\log(0.5 + 0.5\exp(tx))$.
[c] root search and relevance model training are potentially nonconvex.

We compare four heatmapping techniques: sensitivity analysis, standard Taylor decomposition, and the min-max and training-free variants of deep Taylor decomposition. Sensitivity analysis is straightforward to apply. For standard Taylor decomposition, the root $\tilde{\boldsymbol{x}}$ is chosen to be the nearest point such that $f(\tilde{\boldsymbol{x}}) < 0.1 f(\boldsymbol{x})$. For the deep Taylor decomposition models, we apply the z^+-rule in the top layer and the $z^{\mathcal{B}}$-rule in the first layer. The $z^{\mathcal{B}}$-rule is computed using as lower- and upper-bounds $l_p = -0.5$ and h_p=1.5. For the min-max variant, the relevance model in the first layer is trained to minimize the mean-square error between the relevance model output and the true relevance (obtained by application of the z^+-rule in the top layer). It is trained in parallel to the actual neural network, using similar training parameters.

Fig. 5 shows the analysis for 12 positive examples generated from the MNIST test set and processed by the deep neural network. Heatmaps are shown below their corresponding example for each heatmapping method. In all cases, we can observe that the heatmapping procedure correctly assigns most of the relevance to pixels where the digit to detect is located, and ignores the distracting digit.

Sensitivity analysis produces unbalanced and incomplete heatmaps, with some examples reacting strongly, and others weakly. There is also a non-negligible amount of relevance allocated to the border of the image (where there is no information), or placed on the distractor digit. Nearest root Taylor decomposition ignores irrelevant pixels in the background but is still producing spurious relevance on the distractor digit. On the other hand, deep Taylor decomposition produces relevance maps that are less affected by the distractor digit and that are also better balanced spatially. The heatmaps obtained by the trained min-max model and the training-free method are of similar quality, suggesting that the approximations made in Section 5.2 are also valid empirically.

Fig. 6 quantitatively evaluates the heatmapping techniques of Fig. 5. The scatter plots compare the total output relevance with the sum of pixel-wise relevances. Each point in the scatter plot is a different example drawn independently from the input distribution. These scatter plots test empirically for each heatmapping method whether it is conservative in the sense of Definition 1. In particular, if all points lie on the diagonal line of the scatter plot, then $\sum_p R_p = R_f$, and the heatmapping is conservative. The histograms just below test empirically whether the studied heatmapping methods satisfy positivity in the sense of Definition 2, by counting the number of times (shown on a log-scale) pixel-wise contributions R_p take a certain value. Red color in the histogram indicates positive relevance assignments, and blue color indicates negative relevance assignments. Therefore, an absence of blue bars in the histogram indicates that the heatmap is positive (the desired behavior). Overall, the scatter plots and the histograms produce a complete description of the degree of consistency of the heatmapping techniques in the sense of Definition 3.

Sensitivity analysis only measures a local effect and therefore does not conceptually redistribute relevance onto the input. However, we can still measure the relative strength of computed sensitivities between examples or pixels. The nearest root Taylor decomposition is positive, but dissipates relevance. The deep Taylor decomposition with the min-max relevance model produces near-conservative heatmaps, and the training-free deep Taylor decomposition produces heatmaps that are fully conservative. Deep Taylor decomposition spreads relevance onto more pixels than competing methods, as shown by the shorter tail of its relevance histogram. Both deep Taylor decomposition variants shown here also ensures positivity, due to the application of

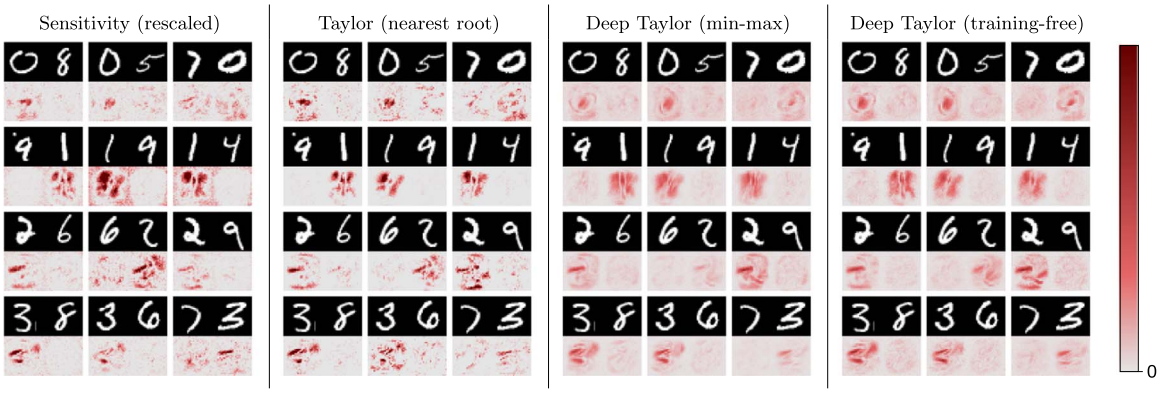

Fig. 5. Comparison of heatmaps produced by various decompositions and relevance models. Each input image is presented with its associated heatmap.

the $z^{\mathcal{B}}$- and z^{+}-rule in the respective layers.

6.2. Experiment on ILSVRC

We now apply the fast training-free deep Taylor decomposition to explain decisions made by large neural networks (BVLC Reference CaffeNet [48] and GoogleNet [12]) trained on the dataset of the ImageNet large scale visual recognition challenges ILSVRC 2012 [49] and ILSVRC 2014 [50] respectively. We keep the neural networks unchanged. We compare our decomposition method to sensitivity analysis. Both methods perform a single backward pass in the network and are therefore suitable for analyzing the predictions of these highly complex models.

The methods are tested on a number of images from Pixabay.com and Wikimedia Commons. The $z^{\mathcal{B}}$-rule is applied to the first convolution layer. For all higher convolution and fully-connected layers, the z^{+}-rule is applied. Positive biases (that are not allowed in our deep Taylor framework) are treated as neurons, on which relevance can be redistributed (i.e. we add $\max(0, b_j)$ in the denominator of $z^{\mathcal{B}}$- and z^{+}-rules). Normalization layers are bypassed in the relevance propagation pass. In order to visualize the heatmaps in the pixel space, we sum the relevances of the three color channels, leading to single-channel heatmaps, where the red color designates relevant regions.

Fig. 7 shows the heatmaps resulting from deep Taylor decomposition for four different images. For example, heatmaps identify as relevant the dorsal fin of the shark and the head of the cat. The heatmaps can detect two instances of the same object within a single image, here, the two frogs. The heatmaps also ignore most of the distracting structure, such as the horizontal lines above the cat's head. Sometimes, the object to detect is shown in a less stereotypical pose or is hard to separate from the background. For example, the sheeps are overlapping and are superposed to a background of same color, leading to a more diffuse heatmap.

Sensitivity analysis ignores or overrepresents some of the relevant regions. For example, the leftmost frog in the first image is assigned more importance than the second frog. Some of the contour of the shark in the second image is ignored. On the other hand, deep Taylor decomposition produces heatmaps that cover the explanatory features in a more comprehensive manner and also better match the saliency of the objects to detect in the input image. See [27] for a quantitative comparison of sensitivity analysis and relevance propagation methods similar to deep Taylor decomposition on this data.

Decompositions for CaffeNet and GoogleNet predictions have a high level of similarity. It demonstrates a certain level of transparency of the method to the choice of deep network architecture supporting the prediction. We can however observe that GoogleNet heatmaps are of higher quality, in particular, with better spatial resolution, and the ability to detect relevant features even in cluttered scenes such as the last image, where the characteristic v-shaped nose of the sheep is still identified as relevant. Instead, AlexNet is more reliant on context for its predictions, and uses more pixels to detect contours of the relevant objects. The observation that more accurate predictions are supported by better resolved input patterns is also in line with other studies [51,52].

Fig. 8 studies the special case of an image of class “volcano”, and a zoomed portion of it. On a global scale, the heatmapping method recognizes the characteristic outline of the volcano. On a local scale, the relevance is present on both sides of the edge of the volcano, which is

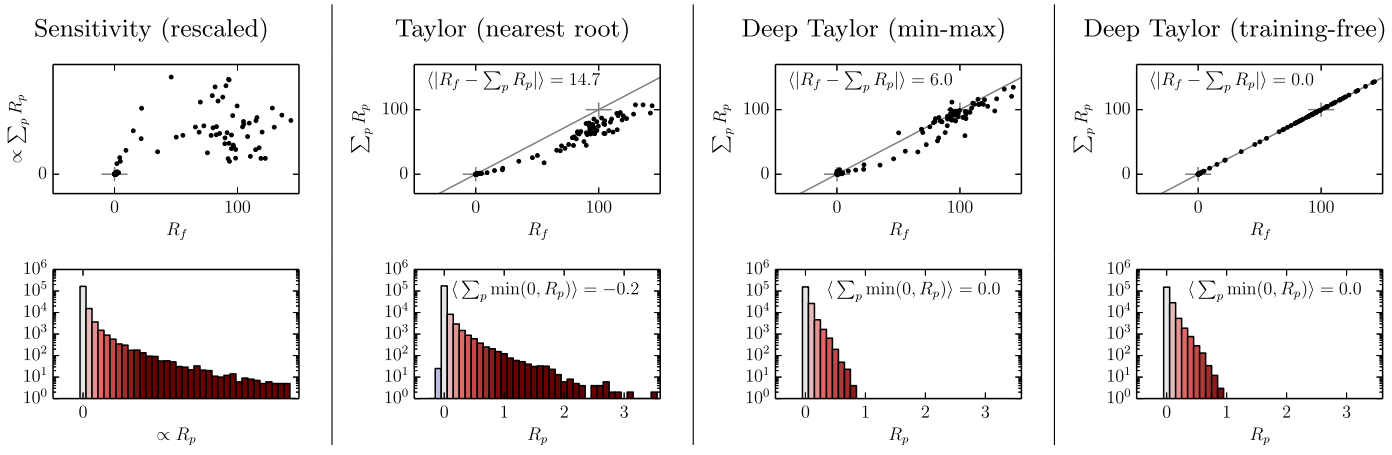

Fig. 6. *Top*: Scatter plots showing for each type of decomposition and data points the predicted class score (*x*-axis), and the sum-of-relevance in the input layer (*y*-axis). *Bottom:* Histograms showing the number of times (on a log-scale) a particular pixel-wise relevance score occurs.

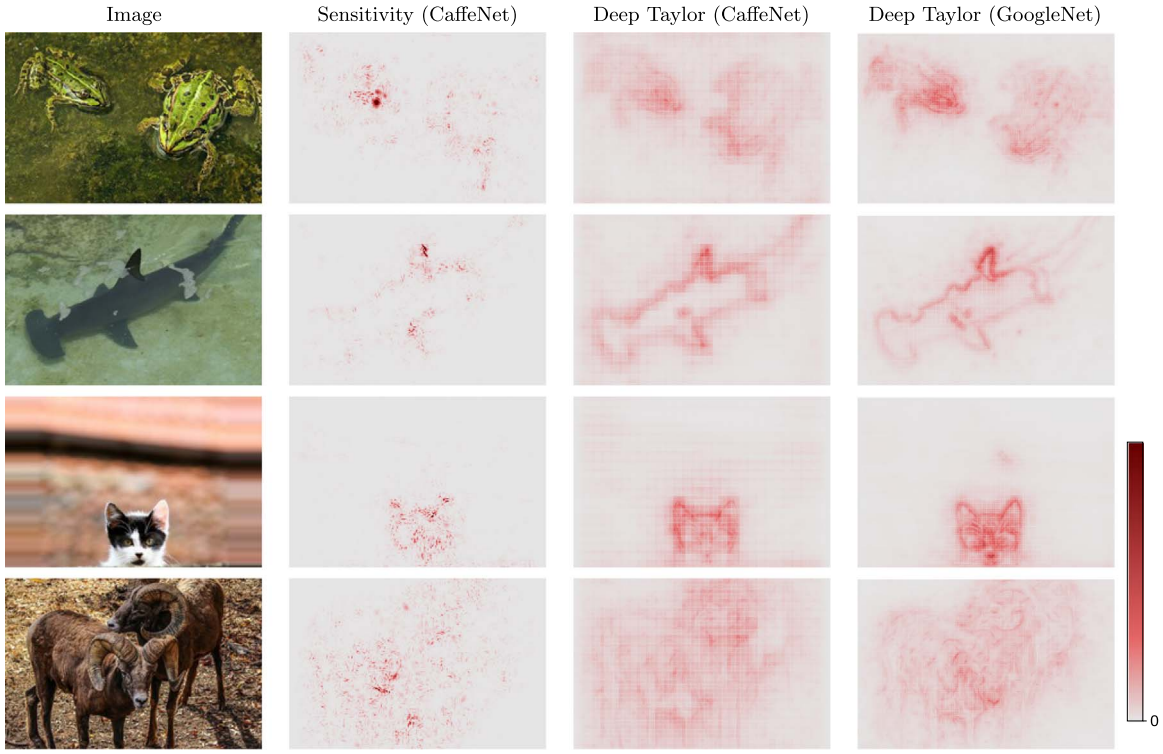

Fig. 7. Images of different ILSVRC classes ("frog", "shark", "cat", and "sheep") given as input to a deep network, and displayed next to the corresponding heatmaps. Heatmap scores are summed over all color channels of the image.

consistent with the fact that the two sides of the edge are necessary to detect it. The zoomed portion of the image also reveals different stride sizes in the first convolution layer between CaffeNet (stride 4) and GoogleNet (stride 2). Observation of these global and local characteristics of the heatmap provides a visual feedback of the way relevance flows in the deep network.

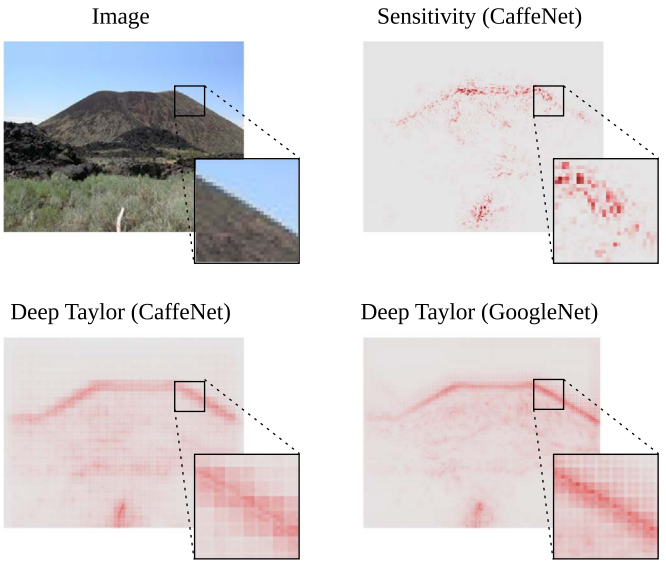

Fig. 8. Image with ILSVRC class "volcano", displayed next to its associated heatmaps and a zoom on a region of interest.

7. Conclusion

Nonlinear machine learning models have become standard tools in science and industry due to their excellent performance even for large, complex and high-dimensional problems. However, in practice it becomes more and more important to understand the underlying nonlinear model, i.e., to achieve transparency of *what* aspect of the input makes the model decide. To achieve this, we have contributed by novel conceptual ideas to deconstruct nonlinear models. Specifically, we have proposed a novel approach to relevance propagation called deep Taylor decomposition, and used it to assess the importance of single pixels in image classification tasks. We were able to compute *heatmaps* that clearly and intuitively allow to better understand the role of input pixels when classifying an unseen data point. We have shed light on theoretical connections between the Taylor decomposition of a function, and rule-based relevance propagation techniques, showing a clear relationship between the two approaches for a particular class of neural networks. We have introduced the concept of relevance model as a mean to scale the analysis to networks with many layers. Our method is stable under different architectures and datasets, and does not require hyperparameter tuning. We would like to stress, that we are free to use as a starting point of our framework either an own trained and carefully tuned neural network model or we may also download existing pre-trained deep network models (e.g. the BVLC CaffeNet [48]) that have already been shown to achieve excellent performance on benchmarks. In both cases, our method provides explanation. In other words our approach is orthogonal to the quest for enhanced results on benchmarks, in fact, we can use any benchmark winner and then enhance its transparency to the user.

Acknowledgments

This work was supported by the Brain Korea 21 Plus Program through the National Research Foundation of Korea; the Deutsche Forschungsgemeinschaft (DFG) [grant MU 987/17-1]; a SUTD Start-

Up Grant; and the German Ministry for Education and Research as Berlin Big Data Center (BBDC) [01IS14013A]. This publication only reflects the authors views. Funding agencies are not liable for any use that may be made of the information contained herein.

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.patcog.2016.11.008.

References

[1] M.I. Jordan, Learning in Graphical Models, MIT Press, Cambridge, MA, USA, 1998.
[2] B. Schölkopf, A.J. Smola, Learning with Kernels Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, Cambridge, MA, USA, 2002.
[3] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based learning algorithms, IEEE Trans. Neural Netw. 12 (2) (2001) 181–201.
[4] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, MIT Press, Cambridge, MA, USA, 2006.
[5] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Inc., New York, NY, USA, 1995.
[6] G. Montavon, G.B. Orr, K.-R. Müller (eds.), Neural Networks: Tricks of the Trade, Lecture Notes in Computer Science, vol. 7700. Springer, Berlin Heidelberg, 2012.
[7] Y. LeCun, L. Bottou, G.B. Orr, K.-R. Müller, Efficient backprop, in: Neural Networks: Tricks of the Trade, 2nd ed., Springer, Berlin Heidelberg, 2012, pp. 9–48.
[8] R.E. Schapire, Y. Freund, Boosting, MIT Press, Cambridge, MA, USA, 2012.
[9] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[10] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, vol. 25, 2012, pp. 1106–1114.
[11] D.C. Ciresan, A. Giusti, L.M. Gambardella, J. Schmidhuber, Deep neural networks segment neuronal membranes in electron microscopy images, in: Advances in Neural Information Processing Systems, vol. 25, 2012, pp. 2852–2860.
[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S.E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7–12, 2015, 2015, pp. 1–9.
[13] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P.P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (2011) 2493–2537.
[14] R. Socher, A. Perelygin, J. Wu, J. Chuang, C.D. Manning, A. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, October 2013, pp. 1631–1642.
[15] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21–24, 2010, Haifa, Israel, 2010, pp. 495–502.
[16] Q.V. Le, W.Y. Zou, S.Y. Yeung, A.Y. Ng, Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in: The 24th IEEE Conference on Computer Vision and Pattern Recognition, pp. 3361–3368, 2011.
[17] E.P. Ijjina, K.M. Chalavadi, Human action recognition using genetic algorithms and convolutional neural networks, Pattern Recognit. (2016).
[18] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, O.A. von Lilienfeld, Machine learning of molecular electronic properties in chemical compound space, New J. Phys. 15 (9) (2013) 095003.
[19] P. Baldi, P. Sadowski, D. Whiteson, Searching for exotic particles in high-energy physics with deep learning, Nat. Commun. 5 (4308) (2014).
[20] S. Haufe, F.C. Meinecke, K. Görgen, S. Dähne, J. Haynes, B. Blankertz, F. Bießmann, On the interpretation of weight vectors of linear models in multivariate neuroimaging, NeuroImage 87 (2014) 96–110.
[21] R. Oaxaca, Male-female wage differentials in urban labor markets, Int. Econ. Rev. 14 (3) (1973) 693–709.
[22] B. Poulin, R. Eisner, D. Szafron, P. Lu, R. Greiner, D.S. Wishart, A. Fyshe, B. Pearcy, C. Macdonell, J. Anvik, Visual explanation of evidence with additive classifiers, in: Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, 2006, pp. 1822–1829.
[23] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: visualising image classification models and saliency maps, CoRR (2013) vol. abs/1312.6034.
[24] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: Computer Vision – ECCV 2014 – 13th European Conference, 2014, pp. 818–833.
[25] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLoS One 10 (7) (2015) e0130140.
[26] D. Rumelhart, G. Hinton, R. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536.
[27] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, Evaluating the visualization of what a deep neural network has learned, IEEE Trans. Neural Netw. Learn. Syst. 99 (2016) 1–14.
[28] M.L. Braun, J.M. Buhmann, K.-R. Müller, On relevant dimensions in kernel feature spaces, J. Mach. Learn. Res. 9 (2008) 1875–1908.
[29] G. Montavon, M.L. Braun, T. Krueger, K.-R. Müller, Analyzing local structure in kernel-based learning: explanation, complexity, and reliability assessment, IEEE Signal Process. Mag. 30 (4) (2013) 62–74.
[30] G. Montavon, M.L. Braun, K.-R. Müller, Kernel analysis of deep networks, J. Mach. Learn. Res. 12 (2011) 2563–2581.
[31] I.J. Goodfellow, Q.V. Le, A.M. Saxe, H. Lee, A.Y. Ng, Measuring invariances in deep networks, in: Advances in Neural Information Processing Systems, vol. 22, pp. 646–654, 2009.
[32] D. Erhan, A. Courville, Y. Bengio, Understanding Representations Learned in Deep Architectures, Technical Report 1355, University of Montreal, 2010.
[33] R. Krishnan, G. Sivakumar, P. Bhattacharya, Extracting decision trees from trained neural networks, Pattern Recognit. 32 (12) (1999) 1999–2009.
[34] R. Krishnan, G. Sivakumar, P. Bhattacharya, A search technique for rule extraction from trained neural networks, Pattern Recognit. Lett. 20 (3) (1999) 273–280.
[35] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, K.-R. Müller, How to explain individual classification decisions, J. Mach. Learn. Res. 11 (2010) 1803–1831.
[36] W. Landecker, M.D. Thomure, L.M.A. Bettencourt, M. Mitchell, G.T. Kenyon, S.P. Brumby, Interpreting individual classifications of hierarchical networks, in: IEEE Symposium on Computational Intelligence and Data Mining, 2013, pp. 32–38.
[37] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, R. Fergus, Intriguing properties of neural networks, CoRR (2013) vol. abs/1312.6199.
[38] S. Bazen, X. Joutard, The Taylor decomposition: a unified generalization of the Oaxaca method to nonlinear models, Technical Report 2013-32, Aix-Marseille University, 2013.
[39] H. Fang, S. Gupta, F.N. Iandola, R.K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J.C. Platt, C.L. Zitnick, G. Zweig, From captions to visual concepts and back, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7–12, 2015, 2015, pp. 1473–1482.
[40] H. Larochelle, G.E. Hinton, Learning to combine foveal glimpses with a third-order Boltzmann machine, in: Advances in Neural Information Processing Systems 23, 2010, pp. 1243–1251.
[41] K. Xu, J. Ba, R. Kiros, K. Cho, A.C. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 2048–2057.
[42] L. Arras, F. Horn, G. Montavon, K.-R. Müller, W. Samek, Explaining predictions of non-linear classifiers in NLP, in: Proceedings of the Workshop on Representation Learning for NLP at Association for Computational Linguistics (ACL), 2016.
[43] I. Sturm, S. Lapuschkin, W. Samek, K.-R. Müller, Interpretable deep neural networks for single-trial EEG classification, J. Neurosci. Methods 274 (2016) 141–145.
[44] M. Gevrey, I. Dimopoulos, S. Lek, Review and comparison of methods to study the contribution of variables in artificial neural network models, Ecol. Model. 160 (3) (2003) 249–264.
[45] S. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: a simple and accurate method to fool deep neural networks, CoRR (2015) vol. abs/1511.04599.
[46] D.G. Garson, Interpreting neural-network connection weights, AI Expert 6 (4) (1991) 46–51.
[47] Y. LeCun, Generalization and network design strategies, in: Connectionism in Perspective, Elsevier, Zurich, Switzerland, 1989.
[48] Y. Jia, Caffe: an open source convolutional architecture for fast feature embedding, 2016, ⟨http://caffe.berkeleyvision.org⟩.
[49] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, F.-F. Li, The ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) ⟨http://www.image-net.org/challenges/LSVRC/2012⟩.
[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M.S. Bernstein, A.C. Berg, F. Li, Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[51] W. Yu, K. Yang, Y. Bai, H. Yao, Y. Rui, Visualizing and comparing convolutional neural networks, CoRR (2014) vol. abs/1412.6631.
[52] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, W. Samek, Analyzing classifiers: Fisher vectors and deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2912–2920.

Grégoire Montavon received a Masters degree in Communication Systems from École Polytechnique Fédérale de Lausanne, in 2009 and a Ph.D. degree in Machine Learning from the Technische Universität Berlin, in 2013. He is currently a Research Associate in the Machine Learning Group at TU Berlin.

Sebastian Lapuschkin received a Masters degree in Computer Science from Technische Universität Berlin, in 2013. He currently is a Research Associate in the Machine Learning Group at the Fraunhofer Heinrich-Hertz-Institute while pursuing is Ph.D. at TU Berlin. His research interests are computer vision, machine learning and data analysis.

Alexander Binder is Assistant Professor at the Singapore University of Technology and Design. He received a Ph.D. in Machine Learning from Technische Universität Berlin, in 2013. He participated in Pascal VOC and ImageCLEF competitions before. His research interests include neural networks, image analysis and medical imaging.

Wojciech Samek received a Diploma degree in Computer Science from Humboldt University Berlin in 2010 and the Ph.D. degree in Machine Learning from Technische Universität Berlin, in 2014. Currently, he directs the Machine Learning Group at Fraunhofer Heinrich Hertz Institute. His research interests include neural networks and signal processing.

Klaus-Robert Müller (Ph.D. 92) has been a Professor of computer science at TU Berlin since 2006; co-director Berlin Big Data Center. He won the 1999 Olympus Prize of German Pattern Recognition Society, the 2006 SEL Alcatel Communication Award, and the 2014 Science Prize of Berlin. Since 2012, he is an elected member of the German National Academy of Sciences – Leopoldina.