

# Survey: Visual Approaches Towards Interpretability Of Neural Networks

Malte Zietlow, Anna-Lena Büßelmann

28th December 2019

## Abstract

In this paper we give a theoretical overview of methods for understanding non-linear predictors (primarily Neural Networks) visually. Our focus lies on heat-mapping methods that try to assign to each input-variable the relevance of this variable towards the model-output. From our literature-review we derive a taxonomy of gradient and perturbation-based approaches. We provide an overview of commonly used metrics for evaluating the efficiency of such heatmapping-methods. Further, we describe a set of sanity-checks for assessing the reliability for a given metric.

## 1 Introduction

More and more Neural Networks (NNs) are used in the day to day life. NNs learn something and then replicate it, but how exactly the internals of the network work is not as easy to find out. Finding a way to communicate in an explainable way how NNs compute their output has been a problem ever since.

Because of that, many different approaches have been made over the last years. Today there are multiple models,

which allow a human to understand a NN, like heatmaps or diagrams, which show the most important parts of the input.

### 1.1 Structure of this paper

This paper aims to give a brief overview on the current state of research of visualizing a NN.

Section 1 describes the methodology of the paper, by defining the target audience and finding criteria for inclusion of research. Thereafter definitions for a heatmap, interpretability and explanation are made in section 2. Next in section 3 the Taxonomy for the following chapters is introduced. Then general concepts for understanding the later explained methods are established.

The methods for visualization are categorized in Gradient-Based Methods, Perturbation-Based Methods and Meta-Methods. Gradient-Based Methods observe the infinitesimal change in model prediction with respect to a root point  $x_0$  of the model  $f$ . Perturbation-Based Methods are based on actually varying the input (by either perturbation or occlusion) and interpreting the differences in the output. Meta-Methods use approaches of the other categories and refine them further

by combining them oder adding an extra step to the explanation.

Thereafter known issues are addressed. Finally a conclusion is found and future research is presented.

## 1.2 Target audience

This paper is targeted at an audience that is looking for a theoretical introduction to the field of heatmapping. Knowledge of basic building blocks (such as perceptron, convolutional-, recurrent-cells), activation-functions (such as ReLU, tanh), common datasets (such as MNIST, ImageNet), and well known models (such as the VGG-family, AlexNet) is assumed.

## 1.3 Criteria for inclusion

To limit the amount of included research, criteria for inclusion was found. Research which was accepted by a renowned conference or published by a researcher, who is well known for other publications, will be included in this paper. In addition the amount of citations of publications must be significant. Furthermore new research will be mentioned, which was published in the last few months, but not accepted by a conference until now or with just a small number of citations, but shows fundamental new findings.

# 2 Background

In this section the general motivation for this paper is described and definitions for interpretability, explanations and heatmaps are found.

## 2.1 What is interpretability?

According to Gilpin et al. [12] "The goal of interpretability is to describe the internals of a system in a way that is understandable to humans". For this an explanation must be found, that is simple enough and uses meaningful vocabulary for a person. In addition the bias and knowledge must be considered. Doshi-Velez and Kim [10] use a similar definition of Interpretability that is adapted to machine learning: Interpretability is "the ability to explain or to present in understandable terms to a human". Doshi-Velez and Kim [10] also mention, that not all ML systems need to be interpretable, because not all systems require human-machine-interaction. /par But especially in today's society, where more and more ML is used, humans want to understand the decisions for safety and ethical reasons. They will not use something they do not understand or trust. Doshi-Velez and Kim [10] also mention, that for knowledge gain a scientific understanding is indispensable.

## 2.2 What is an explanation?

There are multiple views on what an explanation actually is. For this paper a scientific definition is chosen. An explanation focuses on explaining the process inside a neural network and answers the question "Why does this particular input lead to that particular output?" [12, p. 2]

## 2.3 What is a heatmap?

For visualizing the decisions of a neural network most researchers are using heatmaps. Heatmaps describe, which parts of the input data are most important for the output. A heatmap can exist in vari-

ous forms, e.g. as a list of objects, which is sorted by relevance or for visual images a image, which shows the most important pixels by color. [4] also mention, that most often the task of one neuron is researched. Such maps are called attribution maps, in which red and blue markers indicate if the point of data (pixel) contributes positively or negatively to the feature.

### 3 Taxonomy

Visual interpretability of NN as discussed in this survey finds itself as a subset of the *explainable artificial intelligence*-research sector. Alternative approaches are rule-extraction or intrinsic methods [28]. The field of visual interpretability itself can be broken into multiple subsets, for example Maaten and Hinton [24] propose to build a mapping of latent-representations of input-samples via dimensionality reduction onto a 2D-Plane. However, in this work we are mainly concerned with generating heatmaps (section 2.3) for input-samples. We distinguish between Gradient-Based and Perturbation-Based methods, which we found to dominate the literature in this area of research.

#### 3.1 Gradient-Based

Gradient-Based approaches generate heatmaps by observing the change in model-output  $f(x)$  with respect to an input-variable  $x_i$ . In this subsection, the set of gradient-based approaches will be further split into Function, Signal, and Attribution approaches.

##### 3.1.1 Functions, Signal and Attribution

In the past years, a large variety of gradient-based methods have been pro-

posed (e.g. [5, 39, 35, 16, 22, 13, 34, 7, 36, 18, 40, 33, 32]). Broadly speaking, each method explains either Function, Signal, or Attribution [16, 17]. Usually, this is done by *projecting* the model-output  $f(x)$  back into the input-space  $\mathbb{R}^{|x|}$  of input-sample  $x$ , with  $x \in \mathbb{R}^{|x|}$ .

**Function** approximates the amount by which an infinitesimal change in each dimension (in the following: *variable*)  $i$  in the input-sample  $x$  influences the model-output  $f(x)$ . The strength and direction are then simply the partial derivative with respect to the input-variable  $x_i$ , i.e.  $\frac{\partial f(x)}{\partial x_i}$ . However, as function-approaches are usually considered to be only poorly informative [16] and are only used as a baseline, they will not be detailed in this work. For details, please refer to, for example, Simonyan, Vedaldi, and Zisserman [34].

**Signal** tries to reveal which input-variables caused the prediction [16, 39] Signal-approaches are, to the best of our knowledge, constrained to Convolutional Neural Networks (CNNs). Signal-approaches include for example DeConvNet [39] and Guided Backpropagation section 5.4, which have been shown to only approximate patterns that cause increased neuron-activity in higher layers [17] (i.e. layers close to the output-layer). A more recent approach to this is PatternNet section 5.5

**Attribution** tries to approximate the contribution of each input-value  $x_i \in x$ . This is usually called either *attribution* or *relevance*. Approaches to this are SHAP [22], ProtoDash [13], Layer-wise Relevance Propagation (LRP)

section 5.1, Taylor-Decomposition (TD) section 5.2, Deep-Taylor-Decomposition (DTD) section 5.3, and PatternAttribution section 5.5.

### 3.2 Perturbation-Based

Not only Gradient-Based, but many Perturbation-Based methods were developed in the last years ([3, 30, 23, 15, 41]). They visualize how parts of the input data correspond with certain features and which influence they have on the given output. This is done by altering the input in certain ways and measuring the differences. Results are visualized in either a diagram (Prediction Difference Analysis: section 6.4, Anchors: section 6.5) or a heatmap (LIME: section 6.1). Some methods like LIME (section 6.1) are even able, to work with text as well as pictures and can present their results in both, diagrams and heatmaps.

The Contrastive Explanations Method (section 6.3) uses an own approach and visualizes Pertinent Positives (PPs) and Pertinent Negatives (PNs). What these are is explained later. Activation Maximization (section 6.6) computes an input for a certain feature, that activates the feature the most, here the output is an image.

## 4 General Concepts

This section provides some detail about mathematical notation and concepts that might be used throughout this paper.

### 4.1 Mathematical Notation

Among all works in this survey, only few stick to a unified mathematical notation

— and if so, this is usually due to it being the same authors as in the coinciding work. In this survey, for Gradient-Based methods, the mathematical notation of Bach et al. [5] is used for its simplicity. Usually when writing about datasets, input-samples, weights, and biases vector and matrix notation is required. Bach et al. [5] omit all this. For a counterexample of a more matrix-centered notation, kindly refer to Kindermans et al. [16]. For mathematical notation this work builds upon the shoulders of [16]

### 4.2 General Concepts of Gradient-Based methods

Most gradient-based methods use similar concepts as was shown for example by Ancona et al. [4]. In this subsection, some general concepts will be introduced and referred to later, to ease notation and prevent accidental repetition.

#### 4.2.1 Pixel-wise Decomposition

One of the core assumptions in gradient-based methods is the decomposability of prediction  $f(x)$  into relevances  $R_i \in \mathbb{R}$ , where  $x$  is an input-sample.

$R_i < 0$  is interpreted as evidence against the prediction  $f(x)$

$R_i > 0$  is interpreted as evidence for the prediction  $f(x)$

$R_i = 0$  is a neutral relevance, the according input-variable  $x_i$  is usually called a ‘root-point’ [5],  $x_0$ , for model  $f$ .

In the LRP-framework, proposed by Bach et al. [5], each prediction is approximately the sum of relevances of its input-variables

$x_i$

$$f(x) \approx \sum_{i=1}^{|x|} R_i. \quad (1)$$

The rationale behind eq. (1) is to allow human subjects an intuitive judgement of the decision-making process for model  $f$ . Given a dataset of medical imagery, a human subject might verify that classifier  $f$  assigns high relevance to pixels of e.g. cancerous cells when classifying the input-sample as cancerous.

#### 4.2.2 Message-Notation

Messages are a mathematical notation introduced by Bach et al. [5] as a language for describing *inverted* NNs. Given two layers,  $l$  and  $l + 1$ , and two neurons,  $i \in l$  and  $j \in l + 1$ , the message  $R_{i \leftarrow j}^{(l, l+1)}$  is sent from neuron  $j$  to neuron  $i$  iff there exists a path from neuron  $i$  to neuron  $j$ . This relationship is visualized in fig. 2.

#### 4.3 Axioms

Following the terminology of [36], axioms describe desirable properties which a method should satisfy.

**Conservativity** A heatmapping  $R^{(1)}$  is *conservative* if the sum of assigned relevances in the pixel space corresponds to the total relevance detected by the model.[26] Formally, this is expressed by eq. (1).

**Positivity** A heatmapping  $R^{(1)}$  is *positive* if all values forming the heatmap are greater or equal to zero[26], formally

$$\forall i : R_i \geq 0.$$

**Consistency** A heatmapping  $R^{(1)}$  is *consistent*, if it is following both conservativity and positivity.[26] By this,

if there is no detection in the input-sample, the heatmap is also forced to be zero (instead of positive and negative values that cancel each other)[26].

#### 4.4 Sanity Checks for Heatmapping Methods

Sanity checks are simple rules that evaluate whether a proposed method can possibly work. Two such sanity checks for heatmapping methods have been proposed by Adebayo et al. [1].

##### Model Parameter Randomization the

output of a heatmapping method for the trained model should differ significantly from the output of the same heatmapping method for an untrained, randomly initialized model. Otherwise, the heatmapping method does not depend on model parameters and cannot be used to interpret the model.

**Data Randomization** given a dataset of labeled input-samples, a subsidiary dataset of randomly-labeled input-samples is built. The output of a heatmapping method for a model trained on the correctly-labeled dataset should differ significantly from the output of the same heatmapping method for a model trained on the randomly-labeled dataset. Otherwise, the heatmapping method does not rely on the latent concepts present in input-samples, i.e. that the model trained on correctly-labeled dataset has actually learned to distinguish between cancerous/non-cancerous input-samples while the model trained on a randomly-labeled dataset has overfit and memorized input-samples.

for details on similarity metrics for Data Randomization and Model Parameter Randomization refer to [1]

## 4.5 Metrics

Although the field of heatmapping has received growing attention over the past decade, only little work has gone into the development of metrics. This is an issue, because absence of ground truth makes the search for metrics a non-trivial task — while the lack of (reliable) metrics leads to flaws in the comparison of methods.

This subsection will provide a theoretical overview of commonly used metrics, sections 4.5.1 to 4.5.3. Tools for verifying the reliability of metrics are given in 4.5.4 Sanity Checks for Metrics. A final overview and discussion are presented in 4.5.5 Discussion

### 4.5.1 Area Over the Perturbation Curve

Proposed by Wojciech Samek et al. [38], Area Over the Perturbation Curve (AOPC) is a generalization of the *pixel flipping method* presented in [5, pp. 34 sqq.]. AOPC measures the degradation of model-score for a certain input-sample, as input-variables are disabled in an iterative manner. Following either Most Relevant First (MoRF) or Least Relevant First (LeRF) order. MoRF and LeRF are extracted from the set of locations

$$\mathcal{O} = (\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_L),$$

which is ordered by decreasing relevance, i.e. informally:

$$(i > j) \implies \text{relevance}(\mathbf{r}_i) < \text{relevance}(\mathbf{r}_j)$$

Here, a location  $\mathbf{r}_p, p \in [1, L]$  must not be a single input, but could represent any

constant multidimensional array of inputs (e.g. a  $9 \times 9$ -grid of pixels, as seen in the original paper).

**Disabling Input Variables** Disabling input variables comes with the risk of violating dataset statistics such as mean and variance. A sample-perturbation function that disables the region  $\mathbf{r}_k$  in the input-sample  $\mathbf{x}$  is formally introduced as  $g(\mathbf{x}, \mathbf{r}_k)$ . Wojciech Samek et al. propose to four sample-perturbation functions

- replacing each value in  $\mathbf{r}_p$  by sampling a uniform distribution  $\mathcal{U}$
- replacing each value in  $\mathbf{r}_p$  by sampling a Dirichlet distribution  $\mathcal{D}$
- replacing each value in  $\mathbf{r}_p$  by the average value of all samples for the current input-variable
- applying a Gaussian filter with  $\sigma = 3$  to  $\mathbf{r}_p$

Note that the latter two, Gaussian filter and average value for the current input-variable, remove significantly less information from the input-sample than sampling from either  $\mathcal{U}$  or  $\mathcal{D}$ . Therefore (as removing information from the input-sample is the goal of disabling input variables) Wojciech Samek et al. favor the first two, i.e. sampling  $\mathcal{U}$  or  $\mathcal{D}$ .

**Most Relevant and Least Relevant First** MoRF (eqs. (2) and (3)) and LeRF (eqs. (4) and (5)) are both defined

recursively

$$\mathbf{x}_{\text{MoRF}}^{(0)} = \mathbf{x} \quad (2)$$

$$\forall 1 \leq k \leq L : \mathbf{x}_{\text{MoRF}}^{(k)} = g(\mathbf{x}_{\text{MoRF}}^{(k-1)}, \mathbf{r}_k) \quad (3)$$

$$\mathbf{x}_{\text{LeRF}}^{(0)} = \mathbf{x} \quad (4)$$

$$\forall 1 \leq k \leq L : \mathbf{x}_{\text{LeRF}}^{(k)} = g(\mathbf{x}_{\text{LeRF}}^{(k-1)}, \mathbf{r}_{L+1-k}) \quad (5)$$

where  $\mathbf{x}_{\text{MoRF} / \text{LeRF}}^{(n)}$ ,  $n \in [0, k]$  is the perturbed input-sample at iteration  $n$  and by the respective order MoRF / LeRF.

**Area Over the Perturbation Curve** The AOPC is then defined as

$$\text{AOPC}_M = \frac{\langle \sum_{k=0}^L f(\mathbf{x}_M^{(0)}) - f(\mathbf{x}_M^{(k)}) \rangle}{L+1},$$

where  $\langle \cdot \rangle$  denotes the average over all input-samples in the data set and  $M \in \{\text{MoRF}, \text{LeRF}\}$ . Informally, AOPC calculates the arithmetic mean of the average decrease in model-accuracy caused by disabling a range from 0 to  $L$  input variables.

#### 4.5.2 Faithfulness

For faithfulness [2, 37], as opposed to Area Over the Perturbation Curve, model-outputs are computed with only a *single* input-variable disabled at a time. Faithfulness is computed by taking Pearson correlation between the drop in model-output (when disabling the input-variable) and the relevance assigned to the input-variable by the heatmapping-method. Formally, given a sample  $\mathbf{x}$  with  $|\mathbf{x}| = n$  input-variables, an input-variable  $i \in \mathbf{x}$ , the sample  $\mathbf{x}_i$  where  $i$  is disabled, a drop in model-output  $\Delta_i$  and a relevance for the input-variable  $R_i$ :

$$\Delta_i = f(\mathbf{x}) - f(\mathbf{x}_i)$$

$$\text{Faithfulness}_{\mathbf{x}} = \frac{1}{n} \rho(R_i, \Delta_i)$$

gives the Faithfulness for a single image  $\mathbf{x}$ . The Faithfulness over the full dataset is computed as

$$\text{Faithfulness} = \langle \rho(R, \Delta) \rangle$$

where, again,  $\langle \cdot \rangle$  denotes the average over all input-samples in the data set

#### 4.5.3 Remove And Retrain

Hooker et al. point out that previously mentioned metrics, sections 4.5.1 and 4.5.2, introduce new data distributions by artificially disabling input-variables. Therefore, violating one of the key assumptions in machine learning: ‘training and prediction data should arise from the same distribution’ [25]. Further, Hooker et al. show that model-performance degrades significantly slower when retraining in comparison to simply disabling input-variables, as can be seen in fig. 1.

For each input-sample, a heatmap  $\mathcal{H}$  is produced. Where again,  $\mathcal{H}$  is the set of feature importance estimates  $\{e_i^o\}_{i=1}^N$ . New training and test datasets are generated at different degradation levels  $t = [10, 30, 50, 70, 90]$ , with each  $t_i$  a percentage of input-variables per sample. To protect against random performance fluctuations, Hooker et al. retrain five models  $f_0^M(\mathbf{x}_i), f_1^M(\mathbf{x}_i), \dots, f_4^M(\mathbf{x}_i)$  for each metric  $M$  and each degradation method.

**Keep And Retrain** Similar to MoRF and LeRF, Hooker et al. propose Keep And Retrain (KAR). In KAR, input-variables are disabled in LeRF order.

#### 4.5.4 Sanity Checks for Metrics

Just as for heatmaps, sanity checks were proposed for metrics [37]. For determining whether a metric *measures the intended*

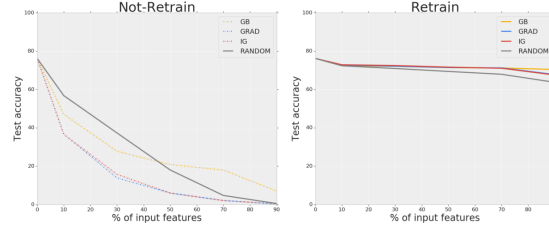


Figure 1: The figure shows the drastic decrease in model accuracy when disabling input-variables without retraining, but significantly smoothed when retraining is undertaken. Evaluated over Guided Backprop (GP), the unweighted gradient (GRAD), Integrated Gradients (IG) and random baseline. **Left:** evaluation of ROAR without retraining. **Right:** evaluation of ROAR with retraining. Adopted from [14]

property and provides consistent results, Tomsett et al. build upon three psychometric reliability tests (notice that both Inter-rater reliability and Inter-method reliability assume a *fixed* metric)

**Inter-rater reliability** measures if heatmapping-methods are ranked consistently over the set of all input-samples. For calculation, Tomsett et al. refer to Krippendorff’s  $\alpha$ .

$\alpha = 1$  implies that a metric assigns the same ranking-order for the heatmapping-methods (i.e. if a heatmapping method is superior, it is consistently superior).

$\alpha = 0$  implies that a metric assigns a random score for each heatmapping-method. Therefore, the fidelity of a heatmapping method cannot be predicted using the observed metric.

**Inter-method reliability** measures the correlation of metric-scores for different heatmapping methods over the set of all input-samples. For calculation, Tomsett et al. refer to Spearman’s  $\rho$ . A large  $\rho$  implies high inter-method reliability.

**Internal consistency** measures if different metrics observe the same property (e.g. fidelity). For calculation, Tomsett et al. take the correlation (again, Spearman’s  $\rho$ ) of metric-scores from different metrics over the same heatmap.

#### 4.5.5 Discussion

Unfortunately, the discussed metrics either require significant computational effort (ROAR, section 4.5.3) or have been shown unreliable (AOPC and Faithfulness, sections 4.5.1 and 4.5.2) [14, 37]. Further, an evaluation of ROAR in the like of Sanity Checks for Metrics has not been performed due to its computational complexity.

This makes reliable comparison of methods almost impossible. For future research, evaluating the reliability of ROAR and finding reliable, less computational complex metrics must be a priority. Proposing new methods is of subordinate importance, given that researchers remain unable to evaluate against preexisting approaches.



## 5 Gradient-Based Methods

This section provides a theoretical introduction to gradient based methods. First, Attribution-approaches (sections 5.1 to 5.3) are presented, then Signal-approaches section 5.4. Finally, section 5.5 are discussed, where PatternNet is a Signal-approach and PatternAttribution a Attribution-approach.

### 5.1 Layer-Wise Relevance Decomposition

LRP attempts to provide a measure for the Pixel-wise Decomposition, discussed earlier in section 4.2.1. The relevances are computed *backwards* from the output of the model ( $f(x)$ ) to its input ( $x$ ) after prediction time. Relevance is denoted as  $R_i^l$ , the relevance of neuron  $i$  in layer  $l$ , where  $l = 1$  is the input layer. LRP is formulated by Bach et al. [5] as a set of constraints:

1. LRP must follow 4.2.1 completely.
2. each layer of the neural network can be represented by relevances, such that

$$\begin{aligned} f(x) &= \dots \\ &= \sum_{j \in l+1} R_j^{(l+1)} = \sum_{i \in l} R_i^{(l)} \\ &= \dots = \sum_{h \in x} R_h^{(1)}. \end{aligned} \quad (6)$$

Informally, eq. (6) implies that relevances can be propagated from the model-output  $f(x)$  to the input-variables in  $x$ .

3. the relevance of the *output neuron* is defined as the model-output  $f(x)$ .
4. the relevance of *every other neuron* is the sum of its incoming messages,

formally

$$R_i^{(l)} = \sum_{\{j | \text{Path } P_{i \rightarrow j} \text{ exists}\}} R_{i \leftarrow j}^{(l, l+1)}. \quad (7)$$

5. the total relevance that a neuron sends out as messages is equal to its own relevance, formally

$$R_j^{(l+1)} = \sum_{\{i | \text{Path } P_{i \rightarrow j} \text{ exists}\}} R_{i \leftarrow j}^{(l, l+1)}. \quad (8)$$

6. each message  $R_{i \leftarrow j}^{(l, l+1)}$  is the product of the relevance of neuron  $j$ ,  $R_j^{(l+1)}$ , weighted by the *relative strength* of the path  $P_{i \rightarrow j}$ , formally

$$R_{i \leftarrow j}^{(l, l+1)} = R_j^{(l+1)} \frac{a_i w_{ij}}{\sum_{h \in (l)} a_h w_{hj} + b_j}, \quad (9)$$

where  $a_i$  is the output of neuron  $i$  (precisely the output of its activation-function),  $w_{ij}$  are the weights that connect neurons  $i$  and  $j$  and  $b_j$  is the bias for layer  $j$ .

Equation (9) shows the simplest case of a *weighting*-term. Different approaches for stabilizing the propagation, such as  $\alpha\beta$ -LRP and  $\epsilon$ -LRP, are further discussed in [5, pp. 20–22].

In summary, LRP requires a conservation of relevance between layers (constraint 2), the relevance of a neuron to be both the sum of its incoming weighted messages (constraints 4 and 6) and the sum of its outgoing weighted messages (constraints 5 and 6).

### 5.2 Taylor-Decomposition

As an approximation to LRP, TD was proposed in [5]. It directly defines relevances  $R_h^{(1)}$  at the (slightly modified) input-sample  $x'$  and therefore avoids the process

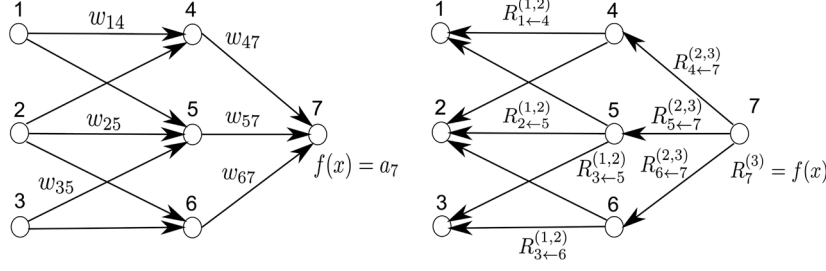


Figure 2: **Left:** A neural network at prediction time. **Right:** a neural network at relevance-propagation time, adopted from [5]

of decomposing relevances for each layer. Instead, relevance  $R_h^{(1)}$  is defined by developing a first order Taylor series of the input  $x$  around a root point  $x_0$ , such that  $f(x_0) = 0$ . Informally, TD attributes relevance by weighting the difference of the image and the root point  $(x - x_0)$  with the partial derivative of the model with respect to the input  $x$ . Formally

$$R_h^{(1)} \approx (x - x_0)_{(h)} * \frac{\partial f}{\partial x_{(h)}}(x_0). \quad (10)$$

Here,  $x'$  as mentioned above is  $x' := x - x_0$ .

Again, a set of constraints is formulated to find the root-point  $x_0$

1.  $x_0$  must be a root point of  $f$ , such that  $f(x_0) = 0$
2.  $x_0$  must be in the neighborhood of  $x$  under some distance metric, e.g. the Euclidean L2-norm.

A major issue with TD is satisfying constraint 2. Bach et al. [5] do not provide a solution for sparsely populated data-domains, e.g. datasets of natural images. Yet, when the root point is ill-defined, Kerdmians et al. [17] were able to show that TD produces bad results.

### 5.3 Deep-Taylor-Decomposition

DTD as proposed by Montavon et al. [26] tries to overcome the difficulties that the search for root-points according to constraint 2 posed to simple TD. For this, they introduce axioms positivity and consistency which DTD must satisfy. Also, they propose a method for finding root-points  $x_0(i) := x_i + t * v_i$ , where  $v_i$  is called *search direction*. Montavon et al. use this definition to define a set of weighting terms, similar to those in LRP and TD. Formally, they introduce a generalization of the weighting parameter in eq. (11)[see 26, Supplementary Material]

$$R_{i \leftarrow j}^{(l, l+1)} = \sum_j \frac{v_i w_{ij}}{\sum_i v_i w_{ij}} R_j^{(l+1)} \quad (11)$$

and then derive specific weighting parameters by defining  $v_i$ .

**$w^2$ -Weighting** which is similar to eq. (9) but does only rely on the weights  $w_{ij}$  that connect  $i$  and  $j$ .  $v_i$  is chosen to be  $v_i := w_{ij}$ .

$$R_{i \leftarrow j}^{(l, l+1)} = R_j^{(l+1)} \frac{w_{ij}^2}{\sum_{h \in (l)} w_{hj}^2}. \quad (12)$$

$w^2$ -weighting is used, when the input-domain is unrestricted, i.e.  $x \in \mathbb{R}^{|x|}$ .

**$z^+$ -Weighting** is equal to the  $\alpha\beta$ -Rule of LRP, with  $\beta = 0, \alpha = 1$ . Here,  $v_i$  is the set of all  $x_i$  for which the corresponding  $w_{ij}$  is positive, formally  $w_{ij}^+ = \{w_{ij} | w_{ij} \geq 0\}$  and  $v_i := \{x_i | w_{ij} \in w_{ij}^+\}$ . The weighting therefore is

$$R_{i \leftarrow j}^{(l, l+1)} = R_j^{(l+1)} \frac{x_i w_{ij}^+}{\sum_{h \in (l)} x(h) w_{hj}^+}. \quad (13)$$

In [26], eq. (13) is modeled with  $x_i w_{ij}^+ =: z_{ij}^+$ , from which the name  $z^+$ -weighting is derived.  $z^+$ -weighting is used, when the input-domain is restricted to positive values, for example in a setting with ReLU-activations, i.e.  $x \in \mathbb{R}_+^{|x|}$ . A generalization of the  $z^+$ -weighting, as discussed in [26], is equal to  $\alpha\beta$ -LRP.

**$z^B$ -Weighting** is used, when the input-domain is restricted to an upper- and lower-bound, for example in image-classification tasks. For details, please refer to [26, pp. 215 sq.]

In contrast to TD, DTD defines not immediately the relevance at the input-sample  $R_h^{(1)}$ , but only the relevance of messages. It therefore incorporates ideas of both LRP (relevance-propagation, even partial equality with the generalization of  $z^+$ -weighting) and TD (approximation of function value with respect to a root-point  $x_0$ ). The relevance at the input-sample is received by a relevance model that then propagates the messages from  $f(x)$  to  $R_h^{(1)}$ .

**Relevance-Models** Montavon et al. [26] propose two relevance models (*Min-Max Relevance Model* and *Training-free Relevance Model*), for which details are provided

in [26]. They are presented as an inversion of the original NN, incorporating their structure. However, no explicit constraints on this property are formulated. The flow in a relevance model is visually depicted in fig. 3. First a prediction  $f(x) =: x_f$  is made (left side) which is then fed through the Relevance Model which extends  $R_f := x_f$  back to the input sample. While the *Training-free Relevance Model* must, as its name suggests, not be trained, the *Min-Max Relevance Model* has to be trained in a supervised fashion. Montavon et al. [26] show that both, the training-free and min-max relevance models, lead to very similar results.

## 5.4 Guided Backprop

Guided Backpropagation as proposed by Springenberg et al. [35] is a method for reproducing the Signal that a NN extracted from an input-sample  $x$ . It builds upon the deconvolutional-method, first proposed by Zeiler and Fergus [39].

## 5.5 PatternNet and PatterAttribution

Using either a simple linear model [16] or a constant shift in mean [17], Kindermans et al. were able to show the vulnerability of gradient methods (such as LRP, TD, DTD, Gradient x Input, and Integrated Gradients) to random changes in the input-sample (see fig. 4). To overcome this vulnerability of preexisting methods, Kindermans et al. [16] introduce *signal estimators*  $S(x)$  that will be explained in this subsection. Signal estimators root in the idea that each input-sample  $x$  of a dataset is a combination of underlying signal  $s$  (contains all information about e.g. the correct class) and distractor  $d$  (does not contain any information about e.g. the cor-

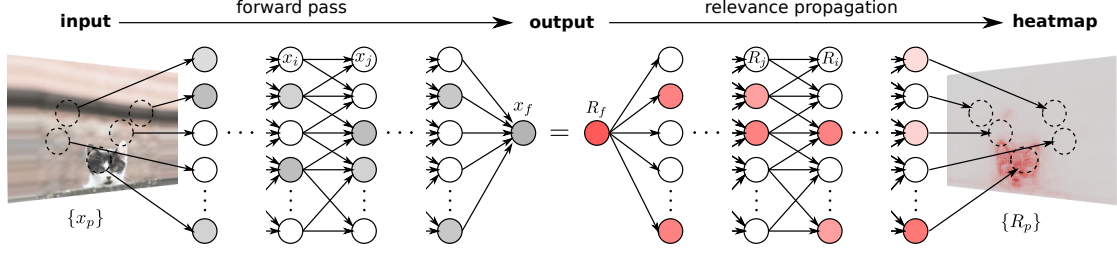


Figure 3: **Left:** A neural network at prediction time. **Right:** exemplary DTD Relevance Model, adopted from [26]. Note, that in this work, usually  $R_h$  is used instead of  $R_p$ .

rect class) such that  $x = s \circ d$ , with  $\circ$  being some operation, for simplicity let  $\circ$  denote summation  $\circ := +$  [16]. A signal estimator  $S(x)$  is then used to extract  $d$  from  $x$ , such that for an optimal signal estimator:  $s = x - S(x)$ .

**The Filter-Based Estimator  $S_w$**  is equal to the  $w^2$ -weighting discussed in section 5.3.

$$S_w = \frac{w_{ij}^2}{\sum_{h \in (l)} w_{hj}^2}. \quad (14)$$

Although simple, Kindermans et al. [16] show that  $S_w(x)$  neither holds theoretically, being unable to separate  $s$  and  $d$ , nor empirically as shown in fig. 5.

**The Linear Estimator  $S_a$**  is based on the assumption of a linear perceptron. Although this assumptions does not even hold for simple ReLU's, it provides a significant improvement over  $S_w$  as shown in fig. 5.

$$S_{a(i)}(x) = x_{(i)}(a_{(i)}w_{ij}) \quad (15)$$

please note that, while the weights  $w_{ij}$ , which are taken from a trained model, and the input-variable of the input-sample  $x_{(i)}$  are fixed parameters,  $a_{(i)}$  must be trained on a given

dataset. Details on the training are provided in [16]. Equation (15) is an exemplary formulation of  $S_a$  for the input layer.

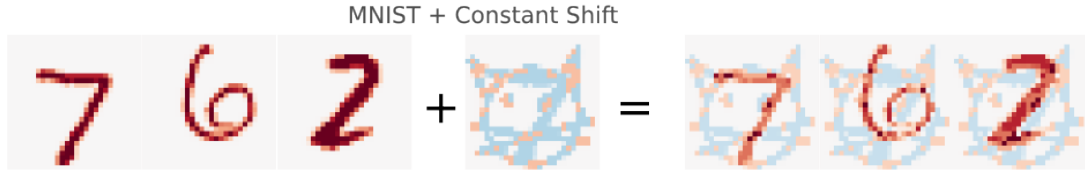
**The Two-Component Estimator  $S_{a+-}$**  is tailored to ReLU-activation functions. Kindermans et al. [16] notice that due to applying ReLU's, the weights used in previous signal estimators  $S_w$ ,  $S_a$  are only trained on positive signal  $s_+$  and distractor  $d_+$  values. To account also for negative  $s_-$ ,  $d_-$ , they split the estimator accordingly

$$S_{a+-(-i)} = \begin{cases} x_{(i)}(a_{+(i)}w_{ij}), & \text{if } (x_{(i)}w_{ij}) > 0 \\ x_{(i)}(a_{-(i)}w_{ij}), & \text{otherwise} \end{cases} \quad (16)$$

again note that, while the weights  $w_{ij}$  and the input-variable  $x_{(i)}$  are fixed parameters,  $a_{+(i)}$ ,  $a_{-(i)}$  must be trained on a given dataset. Details on the training are provided in [16].

In summary, Kindermans et al. propose i) the concept of signal estimators in order to overcome the vulnerability of previous methods to a constant-shift in the input and ii) two new signal estimators,  $S_a$  and  $S_{a+-}$ , which significantly improve upon the previously (and naively) used estimator  $S_w$ . A visual example for the three signal estimators is provided in fig. 6.

## "Cat"astrophic Attribution Failure



## Attribution Methods

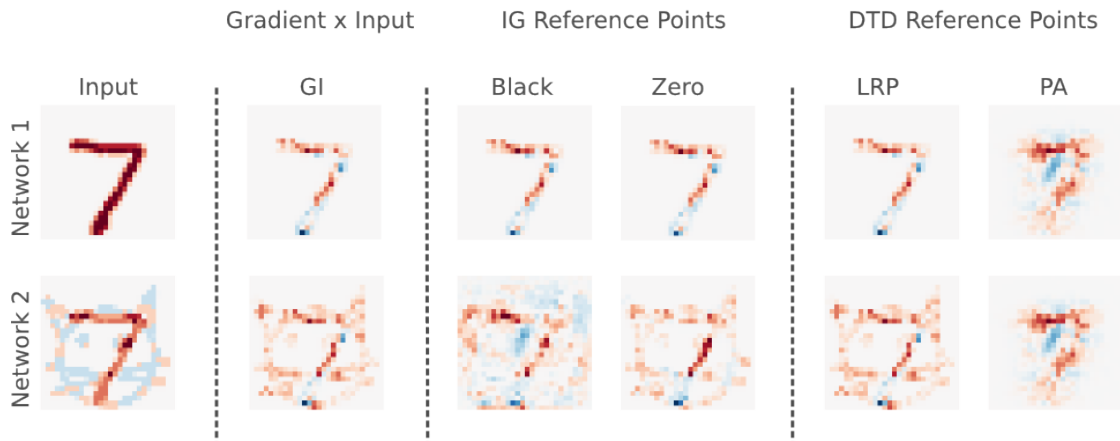


Figure 4: Evaluation of attribution methods on two models (Network 1 and Network 2). Network 1 is trained on the well known MNIST-Dataset, while Network 2 is trained on a manipulated Version of MNIST with a constant shift (i.e. a hand-drawn image of a cat). Because the constant shift does not add any information to an image, a consistent attribution method should provide a similar explanation for both Network 1 and Network 2, when applied to the same input image with only the constant shift. However, Gradient x Input, Integrated Gradients and LRP show a difference between Network 1 and Network 2. For further details, refer to [17] from where this figure was adopted.

For *PatternAttribution*, Kindermans et al. [16] then use the derived signal estimator as a weighting parameter in the DTD-framework, which can be interpreted as an informed choice of root-point  $x_0$ . For *PatternNet*, only the signal itself is reconstructed.

## 6 Perturbation-Based Methods

Perturbation based methods for Visualizing NNs are methods, which compute “the attribution of an input feature (or set of features) by removing, masking or altering them, and running a forward pass on the new input, measuring the difference with the original output” [Acona.2018]. These methods “allow a direct estimation of the marginal effect of a feature” ([Acona.2018]), but do not achieve the needed performance when it comes to a higher number of features.

Because of this, there are many new approaches in this field. Various methods are based on perturbation of input data. In this meta study the focus will be on Prediction Difference Analysis, but multiple other methods will be mentioned and the current state of research will be explained.

### 6.1 LIME

LIME is another Algorithm that aims to explain the predictions of a NN and is short for “Local Interpretable Model-agnostic Explanations”. It was first introduced by Ribeiro, Singh, and Guestrin [29] and is freely accessible at <https://github.com/marcotcr/lime>.

It is applicable to different types of data, like pictures and text and computes the most relevant features of this input. For

a picture LIME would output the relevant parts of a picture, for a text the decisive words in some form of diagram. The explainers correspond with the actual explanation at 90 to 97%. Multiple examples will be seen in this paper and compared to other methods [29].

### 6.2 Occlusion Methods

Occlusion Methods, otherwise known as Input Masking or Representation Erasure, alter the input by concealing certain parts of the input data and measuring the difference in the output compared to the original output.

Li, Monroe, and Jurafsky [21] use this method for word embeddings. They mask the Pre- or Suffix or other Dimensions of a word and calculate the difference in the output. Like this, the most important dimensions can be found. Interestingly erasure of certain words returns a negative importance score, this means, it improves the decision of the network [21].

### 6.3 Contrastive Explanations Method CEM

The Contrastive Explanations Methods (CEMs) was first published by Dhurandhar et al. [8] in 2018 in their paper “Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives”. The CEM approach aims to identify the minimal amount of pixels to justify a classification, called PPs. On the other hand side it identifies the pixels as well, that need to be “turned off” in order to change the classification of the image. These pixels are called PNs. Like this, the most important pixels can be found [23].

“For example, when justifying the classification of a handwritten image of a 3,

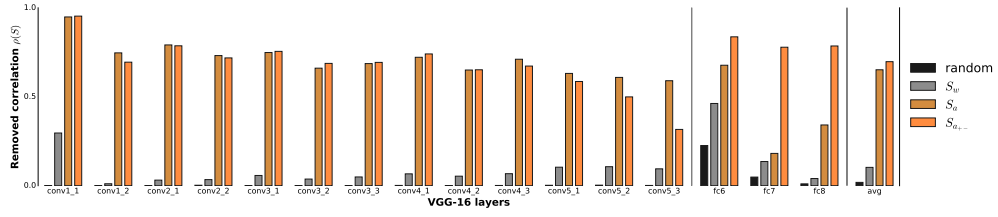


Figure 5: Evaluation of Signal Estimators for VGG-16 on different layers. Higher values are better. A random Signal Estimator is used as baseline. For details on the quality measure  $\rho$  refer to [16] from where this figure was adopted.

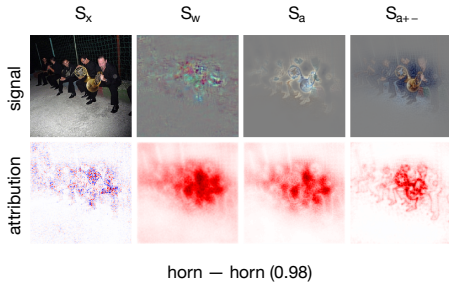


Figure 6: Visual example for the three signal estimators.  $S_x$  is the identity estimator with  $S_x(x) = x$ . Figure adopted from [16]

the method will identify a subset of non-zero or on-pixels within the 3 which by themselves are sufficient for the image to be predicted as a 3 even if all other pixels are turned off (that is, made zero to match background). Moreover, it will identify a minimal set of off-pixels which if turned on (viz. a horizontal line of pixels at the right top making the 3 look like a 5) will alter the classification” [23, p. 2]. This example is visualized in fig. 7. For the algorithm Dhurandhar et al. [8] found formulas to calculate such PPs and PNs. They applied this method to a dataset of handwritten numbers, like seen in the example fig. 7. They trained a NN, which finally reached an accuracy of 99.4% and then used CEM to visualize these decisions.

In addition they used a a convolutional autoencoder for some results, which improves this method even further. In fig. 8 the output of CEM is visualized.

To be able to better see the improvements made by this method, LIME and LRP were applied to the same celebrity dataset. It can clearly be seen, that CEM’s results are better understandable for humans.

Luss et al. [23] improved this method in 2019 and made it applicable to RGB images. They found, that “For colored images, PPs offer better direction as to what

is important for the classification versus too much direction of LIME (shows too many features) or too little direction by Grad-CAM (only focuses on smiles), while for gray-scale images, neither PPs, LIME, or Grad-CAM are particularly informative versus PNs.” Luss et al. [23, p. 7]



Figure 7: CEM, LRP and LIME applied to the MNIST dataset. [8]

## 6.4 Prediction Difference Analysis

The Prediction Difference Analysis was developed 2008 by Robnik-Sikonja and Kononenko [31] and published in the paper “Explaining Classifications for Individual Instances”. They address 3 different types of explanation: instance explanation, model explanation and domain explanation. Instance explanation aims to explain a “classification of a single instance at model level” [31, p. 2]. Model explanation measures the averages of explanations over multiple instances and can provide a more general explanation of features and the importance of features values.

Domain explanation is still unknown, but “if the accuracy of the model is high, it should be quite similar to the model explanation.” [31, p. 2] Firstly they define a model as a function  $f : x \rightarrow f(x)$  which maps instances to numerical values. To calculate the prediction Difference some definitions must be made: An instance  $x$  has a value for each attribute  $A_i$ . To now calculate the effect of  $A_i$  on  $x$  they observe the models prediction for  $f(x \setminus A_i)$ .

If there is just a minor difference the influence of  $A_i$  is small, if there is a major difference the influence is large. The defined formula is:  $predDiff_i(x) = f(x) - f(x \setminus A_i)$ .

The difference can be evaluated as either























Original Class Pred	yng, ml, smlg	yng, fml, smlg	yng, fml, not smlg	yng, fml, not smlg	old, ml, not smlg
Original					
Pert. Neg. Class Pred	old, ml, smlg	old, fml, smlg	yng, ml, not smlg	yng, fml, smlg	old, ml, smlg
Pertinent Negative					
Pert. Neg. Explanations	+gray hair	+oval face	+single hair color, -bangs	+makeup +oval face	+cheekbones
Pertinent Positive					
LIME					

Figure 8: CEM and LIME applied to a celebrity dataset [23]

#### information difference

$$\inf Diff_i(y|x) = \log_2 p(y|x) - \log_2 p(y|x \setminus A_i) \quad (17)$$

#### weight of evidence

$$odds(z) = \frac{p(z)}{p(\bar{z})} = \frac{p(z)}{1 - p(z)} \quad (18)$$

$$WE_i(y|x) = \log_2(odds(y|x)) - \log_2(odds(y|x \setminus A_i)) \quad (19)$$

#### difference of probabilities (output classes)

$$probDiff_i(y|x) = p(y|x) - p(y|x \setminus A_i) \quad (20)$$

The simplest way of computing  $p(y|x \setminus A_i)$  is to replace the value of the attribute  $A_i$

with a special but unknown value, knowing, that this approach can lead to incorrectness if the modeling technique does not handle unknown values naturally (naive Bayesian classifier). To not depend on the model's implementation they chose an approach, that "simulates the lack of information about  $A_i$  with several predictions"[31, p. 4].

For nominal attributes they replace the actual value  $A_i = a_k$  with all possible values for  $A_i$  and weight the prediction by the

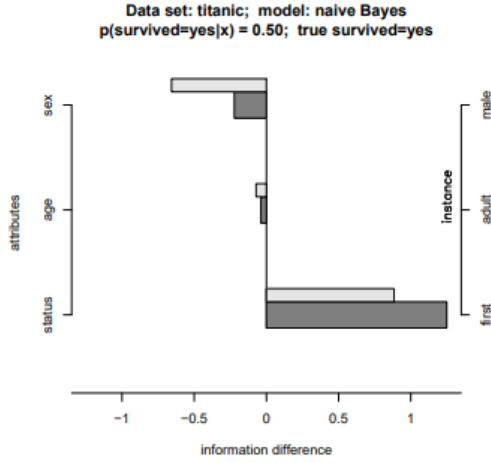


Figure 9: Prediction Difference Analysis on Titanic-dataset [31]

prior probability of the value:

$$p(y|x \setminus A_i) = \sum_{s=1}^{m_i} \left( \begin{array}{l} p(A_i = a_s | x \setminus A_i) \\ p(y|x \leftarrow A_i = a_s) \end{array} \right) \quad (21)$$

$$p(y|x \setminus A_i) = \sum_{s=1}^{m_i} \left( \begin{array}{l} p(A_i = a_s) \\ p(y|x \leftarrow A_i = a_s) \end{array} \right) \quad (22)$$

It should be kept in mind, that different instances predict different values, so the explanation differs from instance to instance. Additionally the explanation is dependent on the model, so if the model is incorrect, the explanation will reflect that. The same goes for class dependency, different classes will lead to different explanations. The results of the prediction difference can then be visualized in a diagram. This special method is called explainVis. Figure 9 is based on the Titanic dataset and

predicts the probability of survival of the person based on travelling class, age and gender.

The information difference is shown on the horizontal axis. Weight of evidence and difference of probabilities produce similar graphs and explanations but differ in scale. On the vertical axis the name of the attribute can be found on the left side and the values for chosen instances on the right side. The class probability  $p(y|x)$  which was calculated by the model is reported on the top. The length of the dark grey bar corresponds to the influence of the feature according to the information difference. Positive influence is given on the right-hand side and negative on the left-hand side. The light grey bar represents the influence across all instances for the corresponding attribute value. This shows the overall trend for the feature.

Zintgraf et al. [41] refined this approach even further. They found, that removing one feature at a time is not enough, because a complex neural network is robust to just one unknown feature, like a pixel in an image. Therefore multiple features should be removed at a time so see an impact. In [41], images are used to demonstrate this. Zintgraf et al. choose patches of connected pixels as their feature sets. The patches have a size of  $k \times k$ -pixel or for an rgb image  $k \times k \times 3$ . Because they use a sliding window style the patches overlap, which makes it possible to define the individual pixels impact. Depending on the size of the window different results can be found.

It is visible, that a small window does not show the intended results. On the other hand a large window leads to a blurry and not clearly understandable image as well. Therefore observing the results and varying the window size is neces-

sary, as can be seen in fig. 10 ([41]).

## 6.5 Anchors

Anchors were introduced in 2018 by Ribeiro, Singh, and Guestrin [30] and can also be referred as if-then rules. They are rules, that “sufficiently “anchors” the prediction locally – such that changes to the rest of the feature values of the instance do not matter” [30, p. 1]. This means, the values of other features are not important if an anchor exists, because the prediction will always be the same. In their paper they apply these anchors to tabular, text and image datasets.

Like in the prediction difference analyses the model is defined as  $f(x) \rightarrow y$ . An instance  $x$  is perturbed by a “perturbation distribution”  $D_x$  (from now on  $D$ ). The perturbations  $D$  should be in an interpretable form.  $A$  is a set of rules, which is applicable on  $x$ .  $A(x)$  returns true (1), if all feature predicates are true for instance  $x$ . If i)  $A(x) = 1$  and ii) “ $A$  is a sufficient condition for  $f(x)$  with high probability” [30, p. 2] (bigger than  $\tau$ ), then  $A$  is an anchor. Formally this can be described with

$$E_{D(z|A)}[\mathbb{1}_{f(x)=f(z)}] \geq \tau, A(x) = 1.$$

An example of anchors for texts can be seen in fig. 11. LIME is explained in section 6.1. In fig. 11 you can see possible Anchors: e.g. for anchor  $A = \text{“not”, “bad”}$  the model will predict “positive” with a probability of more than  $\tau$ . If one or more of these words are missing, there is no anchor and with that it is not certain, what the output will be. The paper presents two different ways for finding those anchors: a bottom-up construction or a Beam-Search.

With the bottom up search they try to find a rule with the highest estimated precision, and like this the shortest anchor.

This anchors tend have a high coverage and are easily understandable for humans. The downside of this approach is, that this greedy strategy is only able, to find one anchor at a time and the coverage is just negligibly. The beam search aims “to identify amongst many possible anchors the one that has the highest coverage” [30, p. 5]. This is done similar to the greedy approach. First all candidate rules are computed, then the best rules are selected.

## 6.6 Activation Maximization

Activation Maximization aims to find an input, which maximizes the output score for a certain class. It was first published in a technical report by Erhan et al. [11] in 2009. They restricted themselves to find the image of the dataset they had, that maximizes the feature.

Le et al. [20] refined this approach in 2012 and not only found the input images with the highest stimuli, but were able to compute an own picture (picture) [20].

There are multiple frameworks that can generate such pictures. A well-known method was developed by Zeiler and Fergus in 2014 called DeConvNe [39]. They mapped features to pixels, instead of the other way around: “To start, an input image is presented to the convnet and features computed throughout the layers. To examine a given convnet activation, we set all other activations in the layer to zero and pass the feature maps as input to the attached deconvnet layer. Then we successively (i) unpool, (ii) rectify and (iii) filter to reconstruct the activity in the layer beneath that gave rise to the chosen activation. This is then repeated until input pixel space is reached.” [39, p. 820].

Nguyen et al. [27] achieved this in 2016 by starting from a random image and cal-

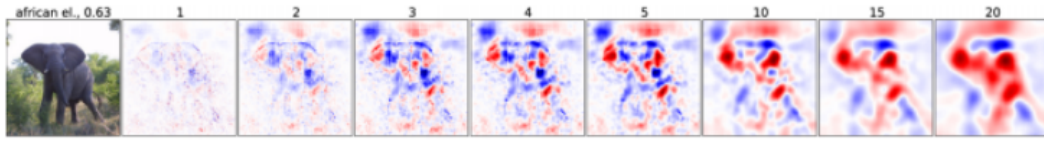


Figure 10: Prediction Difference Analysis with different window size [41]

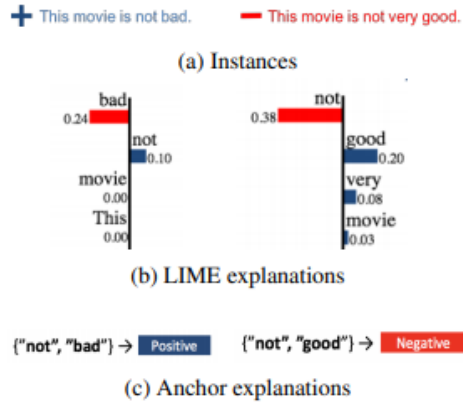


Figure 1: Sentiment predictions, LSTM

Figure 11: Possible Anchors for textual data [30]

culating via backpropagation how each pixel influences the output. Multiple studies have shown, that this can lead to unrealistic images. To improve in this point a general concept of a natural image must be learned by the NN. Therefore, Nguyen et al. are using image generator networks. They conclude, that they can understand better, which features a NN has learned exactly with this method [27].

In fig. 12 the learned feature “face” is demonstrated. On the upper half the actual input images are shown, which activate the neuron the most. On the bottom the generated image is shown.

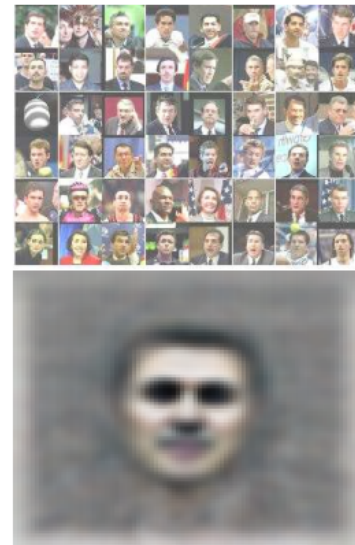


Figure 12: Inputs that activate the “face-neuron” the most. Top: actual input images Bottom: via back-propagation found input [20]

## 7 Meta Methods

Meta Methods are using other methods to calculate their results, e.g. they can use pre-generated heatmaps as an input.

### 7.1 Network Dissection

This method was introduced in 2017 by Bau et al. [6] and is only applicable on convolutional networks. Their method is divided into three steps: “

1. Identify a broad set of human-labeled visual concepts.
2. Gather hidden variables’ response to known concepts.
3. Quantify alignment of hidden variable–concept pairs.

” [6, p. 2] A human interpretable concept is often displayed by a combination of multiple variables and not only one. Nonetheless the paper measures the alignment between a single unit and a single interpretable concept. They did this, by finding pictures, that maximize the activation of a feature and then found the part of the picture, all of the pictures had in common (fig. 13).

Interestingly they found, that lower layers of a CN interpret concepts like color or texture and higher layers more complex concepts like part or object, much like the human way of interpretation [6].

### 7.2 Concept Activation Vectors

Concept Activation Vectors were published by Kim et al. [15] in 2018. It aims to improve heatmaps in the point, that is it is only applicable to one picture at a time. The problem with that is, that even if a picture is of the same object, different parts of

the object can be highlighted in the heatmap. This leads to confusion and mistrust.

Their method measures the activation of a layer  $l$  produced of certain inputs. They then define a “concept activation vector” (or CAV) as the normal to a hyperplane separating examples without a concept and examples with a concept in the model’s activations” [15, p. 3].

A concept can be a special feature of a photo, e.g. a striped texture. So to compute such CAV, analysts need a set of photos of striped objects and a set of random photos. “Then, a binary linear classifier can be trained to distinguish between the layer activations of the two sets [...]. This classifier [...] is a linear CAV for the concept” [15, p. 3]. Like this it can be found, which features exactly a NN learned. In the paper they used a maximization technique in addition to sort the photos. Like this, the underlying concept of a CAV can be visualized [15].

### 7.3 Spectral Relevance Analysis

The Spectral Relevance Analysis (SpRAy) was first published in 2019 by Lapuschkin et al. [19]. It aims to combine multiple explanations into one. It applies spectral clustering to pre-generated heatmaps and identifies recurrent patterns.

“The identified features may be truly meaningful representatives of the object class of interest, or they may be co-occurring features learned by the model but not intended to be part of the class, and ultimately of the model’s decision process. Since SpRAy can be efficiently applied to a whole large-scale dataset, it helps to obtain a more complete picture of the classifier behavior and reveal unexpected or ‘Clever Hans’ type decision making” [19, p. 8].

SpRAy consists of four steps:

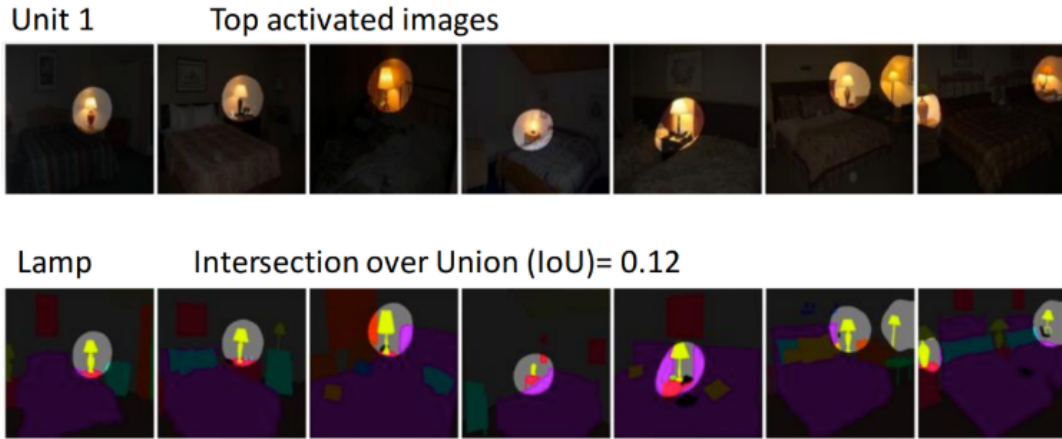


Figure 13: Network Dissection — concept “lamp”, [6]

1. The relevance maps for the dataset need to be calculated.
2. Thereafter the relevance maps need to be resized to be in a uniform form.
3. Spectral cluster analysis must be applied to the relevance maps. This groups classifier behaviors into clusters.
4. Then interesting clusters can then be found via eigengap analysis.

After that the results can be visualized [19].

## 8 Conclusion

Concluding this literature-review on visual-interpretability of non-linear predictors (with significant focus on NNs), this area of research feels more like a minefield, where every few months new work is published that either outperforms previous methods or renders them flat wrong [17, 9]. However, we find metrics to compare such methods to be either

computationally expensive or unreliable (section 4.5.4). With interpretability as a requirement for certain application domains (especially in the european union), cooling is improbable.

This paper summarized multiple methods, that aim to visualize these explanations. Since first methods were introduced in [31] many new approaches have evolved. Some of the most promising methods like CEM or PatternNet and PatternAttribution (section 5.5) have found new, but much more interpretable ways of explaining by standing upon the shoulders of previous research. Almost the complete corpus used for this survey is publicly available.

And even though, there is no favorable method at this time and for most approaches much more research is necessary, there are more ways than ever to visualize NNs in an understandable way.

## 9 Future Research

There have been many new approaches towards Perturbation-Based Methods in the last years. Methods like anchors or CEM are promising, but need more research in order to refine them and make them applicable to various types of data.

Besides, we find that new, reliable and less computational expensive metrics must be a priority for future research. This is a highly non-trivial task, because ground-truth is unavailable. As a first step we suggest that ROAR (section 4.5.3) as the most promising metric should be evaluated under the sanity-check-framework for metrics (section 4.5.4).

## References

- [1] Julius Adebayo et al. "Sanity Checks for Saliency Maps". In: *NIPS 2018*. NIPS'18. USA: Curran Associates Inc, 2018, pp. 9525–9536.
- [2] David Alvarez-Melis and Tommi S. Jaakkola. "Towards Robust Interpretability with Self-explaining Neural Networks". In: *NIPS 2018*. NIPS'18. USA: Curran Associates Inc, 2018, pp. 7786–7795. URL: <http://dl.acm.org/citation.cfm?id=3327757.3327875>.
- [3] Amit Dhurandhar et al. *Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives*. 2018.
- [4] Marco Ancona et al. "Towards Better Understanding of Gradient-Based Attribution Methods for Deep Neural Networks". In: *ICLR 2018*. 2018. URL: [https://www.researchgate.net/publication/321124808\\_A\\_unified\\_view\\_of\\_gradient-based\\_attribution\\_methods\\_for\\_Deep\\_Neural\\_Networks](https://www.researchgate.net/publication/321124808_A_unified_view_of_gradient-based_attribution_methods_for_Deep_Neural_Networks) (visited on 12/11/2019).
- [5] Sebastian Bach et al. "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation". In: *PLOS ONE* 10.7 (2015). ISSN: 1932-6203. DOI: 10.1371/journal.pone.0130140. (Visited on 12/08/2019).
- [6] David Bau et al. "Network Dissection: Quantifying Interpretability of Deep Visual Representations". In: *CVPR 2017*. 2017. URL: <https://arxiv.org/pdf/1704.05796.pdf> (visited on 12/23/2019).
- [7] Daniel Smilkov et al. *SmoothGrad: removing noise by adding noise*. 2017.
- [8] Amit Dhurandhar et al. "Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives". In: *32nd Conference on Neural Information Processing Systems (NIPS 2018)*. 2018. URL: <https://researcher.watson.ibm.com/researcher/files/us-adhuran/explanations-based-missing-final.pdf> (visited on 12/23/2019).
- [9] Ann-Kathrin Dombrowski et al. "Explanations can be manipulated and geometry is to blame". In: *CoRR* abs/1906.07983 (2019).
- [10] Finale Doshi-Velez and Been Kim. *Towards A Rigorous Science of Interpretable Machine Learning*. 2017. URL: <https://arxiv.org/pdf/1702.08608.pdf> (visited on 12/20/2019).
- [11] Dumitru Erhan et al. *Visualizing Higher-Layer Features of a Deep Network*. 2009. URL: <https://pdfs.semanticscholar.org/65d9/94fb778a8d9e0f632659fb33a082949a50d3.pdf> (visited on 12/22/2019).
- [12] Leilani H. Gilpin et al. "Explaining Explanations: An Overview of Interpretability of Machine Learning". In: *2018 IEEE 5th International Conference on Data Science and Advanced Analytics: DSAA 2018 : proceedings : 1-4 October 2018, Turin, Italy*. Ed. by Francesco Bonchi. Los Alamitos, California: Conference Publishing Services, IEEE Computer Society, 2018, pp. 80–89. ISBN: 978-1-5386-5090-5. DOI: 10.1109/DSAA.2018.00018.



- [13] Karthik S. Gurumoorthy et al. "Efficient Data Representation by Selecting Prototypes with Importance Weights". In: (2017).
- [14] Sara Hooker et al. "A Benchmark for Interpretability Methods in Deep Neural Networks". In: *NeurIPS 2019*. 2019.
- [15] Been Kim et al. "Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)". In: *ICML 2018*. 2018. URL: <https://arxiv.org/pdf/1711.11279.pdf> (visited on 12/22/2019).
- [16] Pieter-Jan Kindermans et al. "Learning how to explain neural networks: PatternNet and PatternAttribution". In: *ICLR 2018*. 2018. (Visited on 02/15/2018).
- [17] Pieter-Jan Kindermans et al. "The (Un)reliability of Saliency Methods". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Ed. by Wojciech Samek et al. Lecture Notes in Artificial Intelligence. Cham: Springer International Publishing and Springer, 2019, pp. 267–280. ISBN: 978-3-030-28954-6.
- [18] W. Landecker et al. "Interpreting individual classifications of hierarchical networks". In: *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. 2013, pp. 32–38.
- [19] Sebastian Lapuschkin et al. "Unmasking Clever Hans Predictors and Assessing What Machines Really Learn". In: *Nature Communications*. 2019. URL: <https://arxiv.org/pdf/1902.10178.pdf> (visited on 12/20/2019).
- [20] Quoc Le et al. "Building High-level Features Using Large Scale Unsupervised Learning". In: *29th International Conference on Machine Learning*. 2012. URL: <https://icml.cc/2012/papers/73.pdf> (visited on 12/23/2019).
- [21] Jiwei Li, Will Monroe, and Dan Jurafsky. "Understanding Neural Networks through Representation Erasure". In: *CoRR*. 2016. URL: <https://arxiv.org/pdf/1612.08220.pdf> (visited on 12/23/2019).
- [22] Scott M. Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". In: *NIPS 2017*. Ed. by I. Guyon et al. Curran Associates, Inc, 2017, pp. 4765–4774.
- [23] Ronny Luss et al. "Generating Contrastive Explanations with Monotonic Attribute Functions". In: *CoRR 2019*. URL: <https://arxiv.org/pdf/1905.12698.pdf> (visited on 12/23/2019).
- [24] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. URL: <http://www.jmlr.org/papers/v9/vandermaaten08a.html>.
- [25] Georgia McGaughey, W. Patrick Walters, and Brian Goldman. "Understanding covariate shift in model performance". In: *F1000Research* 5 (2016). ISSN: 2046-1402. doi: 10.12688/f1000research.8317.3.

- [26] Grégoire Montavon et al. “Explaining Nonlinear Classification Decisions with Deep Taylor Decomposition”. In: *Pattern Recognition* 65.C (2017), pp. 211–222. doi: 10.1016/j.patcog.2016.11.008. URL: <https://doi.org/10.1016/j.patcog.2016.11.008>.
- [27] Anh Nguyen et al. “Synthesizing the preferred inputs for neurons in neural networks via deep generator networks”. In: *30th International Conference on Neural Information Processing Systems*. 2016, pp. 3395–3403. URL: <https://arxiv.org/pdf/1605.09304.pdf> (visited on 12/22/2019).
- [28] Gabrielle Ras, Marcel van Gerven, and Pim Haselager. *Explanation Methods in Deep Learning: Users, Values, Concerns and Challenges*. 2018. arXiv: 1803.07517 [cs.AI].
- [29] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. ““Why Should I Trust You?” Explaining the Predictions of Any Classifier”. In: *KDD ’16 Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016. URL: <https://www.kdd.org/kdd2016/papers/files/rfp0573-ribeiroA.pdf> (visited on 12/23/2019).
- [30] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. “Anchors: High-Precision Model-Agnostic Explanations”. In: *32nd AAAI Conference on Artificial Intelligence*. 2018. URL: <https://homes.cs.washington.edu/~C2%A0marcotcr/aaai18.pdf> (visited on 12/23/2019).
- [31] Marko Robnik-Sikonja and Igor Kononenko. “Explaining Classifications for Individual Instances”. In: *IEEE Transactions on Knowledge and Data Engineering* 2008. 2008. URL: <http://lkm.fri.uni-lj.si/rmarko/papers/RobnikSikonjaKononenko08-TKDE.pdf> (visited on 12/23/2019).
- [32] Ramprasaath R. Selvaraju et al. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: *ICCV 2017*. 2017, pp. 618–626.
- [33] Avanti Shrikumar et al. *Not Just a Black Box: Learning Important Features Through Propagating Activation Differences*. 2016.
- [34] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”. In: *ICLR 2014*. 2014. URL: <https://arxiv.org/pdf/1312.6034.pdf> (visited on 12/20/2019).
- [35] Jost Tobias Springenberg et al. “Striving for Simplicity: The All Convolutional Net”. In: *ICLR 2015*. 2015.
- [36] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. “Axiomatic Attribution for Deep Networks”. In: *ICML 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 3319–3328.
- [37] Richard Tomsett et al. *Sanity Checks for Saliency Metrics*. 2019.

- [38] Wojciech Samek et al. “Evaluating the Visualization of What a Deep Neural Network Has Learned”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28 (2015), pp. 2660–2673.
- [39] Matthew D. Zeiler and Rob Fergus. “Visualizing and Understanding Convolutional Networks”. In: *ECCV 2014: Proceedings, Part I*. Ed. by David Fleet et al. Lecture notes in computer science. Cham: Springer, 2014. ISBN: 978-3-319-10590-1.
- [40] Jianming Zhang et al. “Top-Down Neural Attention by Excitation Backprop”. In: *Int. J. Comput. Vision* 126.10 (2018), pp. 1084–1102. ISSN: 0920-5691. URL: <https://doi.org/10.1007/s11263-017-1059-x>.
- [41] Luisa Zintgraf et al. “VISUALIZING DEEP NEURAL NETWORK DECISIONS: PREDICTION DIFFERENCE ANALYSIS”. In: *ICLR 2017*. 2017. URL: <https://arxiv.org/pdf/1702.04595.pdf> (visited on 12/23/2019).

## Glossary

**AOPC** Area Over the Perturbation Curve. 6–8

**CEM** Contrastive Explanations Method. 14, 15, 22, 23

**CNN** Convolutional Neural Network. 3

**DTD** Deep-Taylor-Decomposition. 4, 10–12, 14, 28

**KAR** Keep And Retrain. 7

**LeRF** Least Relevant First. 6, 7

**LRP** Layer-wise Relevance Propagation. 3, 4, 9–11, 13, 15, 28

**message** Messages are defined in 4.2.2 Message-Notation. 9, 11

**MoRF** Most Relevant First. 6, 7

**NN** Neural Network. 1, 3, 5, 11, 14, 20, 22

**path** A path  $P_{i,j}$  exists between two neurons  $i, j$  if the value of  $i$  is fed into  $j$ . 5

**PN** Pertinent Negative. 4, 14–16

**PP** Pertinent Positive. 4, 14–16

**ROAR** Remove And Retrain. 8, 23

**SpRAy** Spectral Relevance Analysis. 21

**TD** Taylor-Decomposition. 4, 9–11

## List of Figures

1	The effect of retraining on model degradation in metrics. . . . .	8
2	Single-Layer Neural Network; prediction and propagation via LRP . . .	10
3	Deeper Neural Network; prediction and propagation via DTD . . . . .	12
4	Evaluation of attribution methods via a constant shift on MNIST . . . . .	13
5	Evaluation of Signal Estimators for VGG-16 on different layers. . . . .	15
6	Visual example for the three signal estimators. . . . .	15
7	CEM, LRP and LIME applied to the MNIST dataset. [8] . . . . .	16
8	CEM and LIME applied to a celebrity dataset [23] . . . . .	17
9	Prediction Difference Analysis on Titanic-dataset [31] . . . . .	18
10	Prediction Difference Analysis with different window size [41] . . . . .	20
11	Possible Anchors for textual data [30] . . . . .	20
12	Inputs that activate the “face-neuron” the most. Top: actual input images Bottom: via backpropagation found input [20] . . . . .	20
13	Network Dissection — concept “lamp”, [6] . . . . .	22