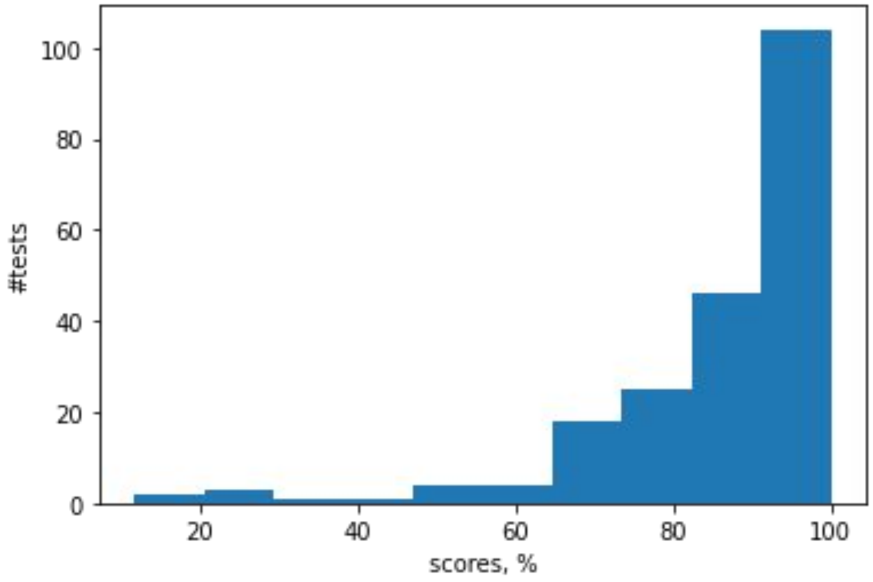
Test results

Total number of test results : 208

Average score : 87.19 %

Score distribution :



Preliminary results. Dynamics

Taking the currently maximum scores, we have the following results:

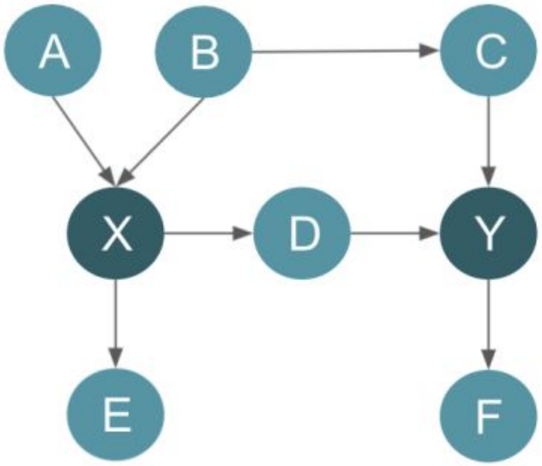
“5” (from 80%) :	159	76%
“4” (between 60 and 80%) :	37	18%
“3” (between 40 and 60%) :	6	3%
“failed” (less 40%) :	6	3%

Some common mistakes

В двухступенчатом MDL (а именно, $L(D, H) = L(D|H) + L(H)$) отсутствует компонент, учитывающий сложность модели H

Some common mistakes

Примените backdoor критерий для оценки влияния X на Y. Выберите наименьшее по размеру множество вершин, необходимых для блокирования всех backdoor-путей

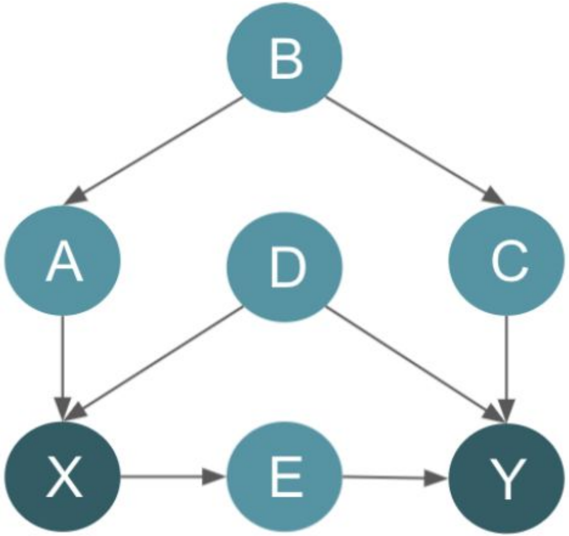


А) {B}
Б) {A, C}
В) {B, C}
Г) {A, B, C}
Д) {D}
Е) {B, D}

Правильные ответы: А

Some common mistakes

Примените backdoor критерий для оценки влияния X на Y. Выберите наименьшие по размеру множество вершин, необходимых для блокирования всех backdoor-путей



А) {B}
Б) {D}
В) {B, D}
Г) {C, D}
Д) {A, C}
Е) {E}
Ж) {A, E}

Правильные ответы: В, Г

Lecture 7. Layer-wise Relevance Propagation and Deep Taylor Decomposition

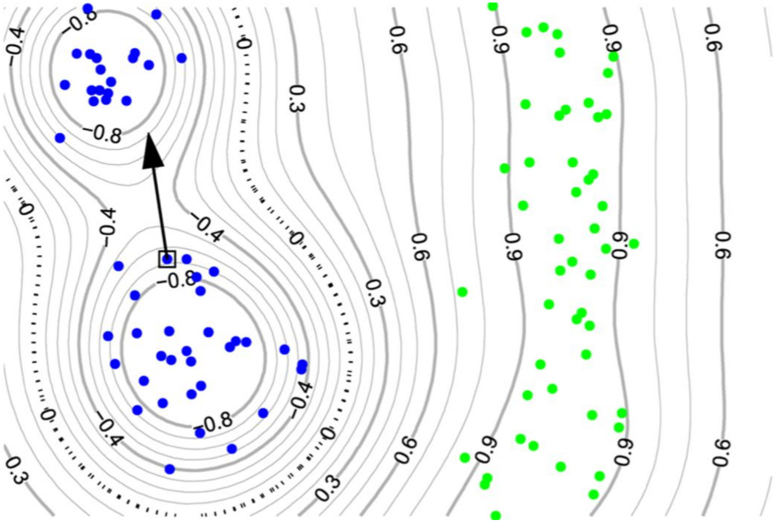
Tatiana Makhalova
MADE, 30.10.2021

Part 1. Layer-wise Relevance Propagation

First order Taylor approximation

The first order Taylor approximation $f(x) \approx f(x_0) + \sum_{d=1}^{V} f'(x_0)(x_{(d)} - x_{0(d)})$

Saliency (sensitivity) maps: an image x is explained by $f'(x)$



The **blue** dots are labeled **negatively**, the **green** dots are labeled **positively**

Local gradient of the classification function at the prediction point

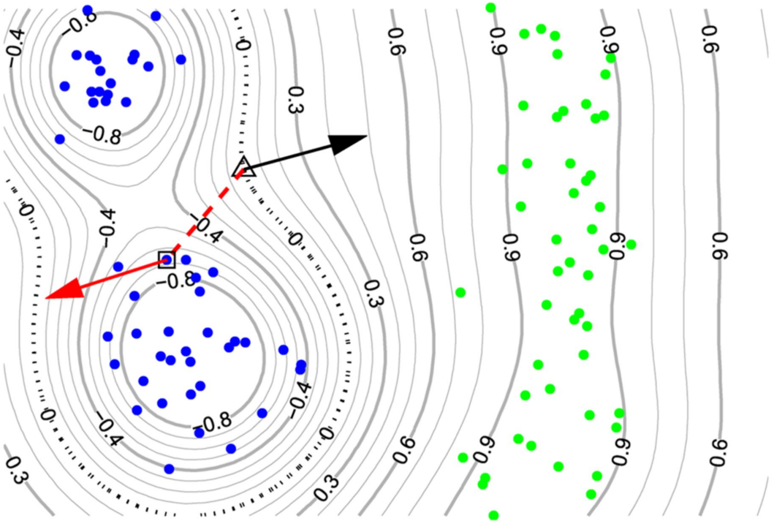
The closest neighbors of the other class can be found at a very different angle. Thus, the local gradient at the prediction point x may not be a good explanation for the contributions of single dimensions to the function value $f(x)$

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140

First order Taylor approximation

The first order Taylor approximation $f(x) \approx f(x_0) + \sum_{d=1}^{V} f'(x_0)(x_{(d)} - x_{0(d)})$

We are interested to find out the contribution of each pixel relative to the state of maximal uncertainty of the prediction, i.e., $f(x_0) = 0$, i.e., $f(x) \approx \sum_{d=1}^{V} f'(x_0)(x_{(d)} - x_{0(d)})$



Δ the nearest root point x_0 on the decision boundary

→ $f'(x_0)$

- - - $x - x_0$

→ the approximation of $f(x)$ by Taylor expansion around x_0 (equivalent to the diagonal of the outer product between $f'(x_0)$ and $x - x_0$)

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

Layer-wise relevance propagation

Goal: to find out the contribution of each input pixel to a particular prediction

Idea: In case of classification, to find out the contribution of each pixel relative to the state of maximal uncertainty of the prediction which is given by the set of points $\boldsymbol{f(x_0) = 0}$, since $\boldsymbol{f(x) > 0}$ denotes presence and $\boldsymbol{f(x) < 0}$ absence of the learned structure, x_0 is called a reference point

Remark: $\boldsymbol{f(x) < 0}$ is less desirable values, since it is difficult to interpret negative evidence for a class

Basic hypothesis (the conservation property): relevance is constant throughout the layers, i.e.,

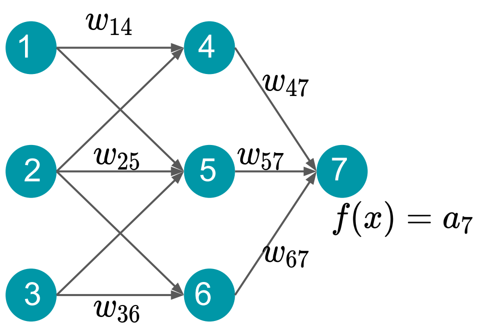
$$f(x) = \cdots = \sum_{d \in l+1} R_d^{(l+1)} = \sum_{d \in l} R_d^{(l)} = \ldots = \sum_{d \in 1} R_d^{(1)}$$

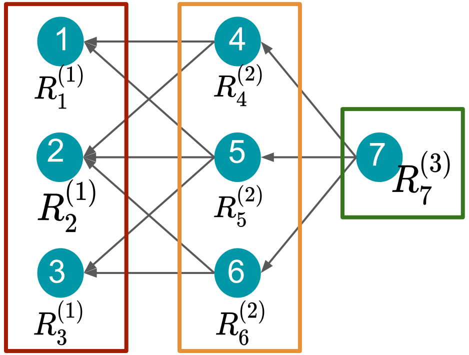
where $R_d^{(l)}$ a relevance score for each dimension d at the lavel l

Remark: the decomposition satisfying the conservation property is not unique and there is not guarantee that it yields a meaningful interpretation

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

Relevance propagation: global conservation

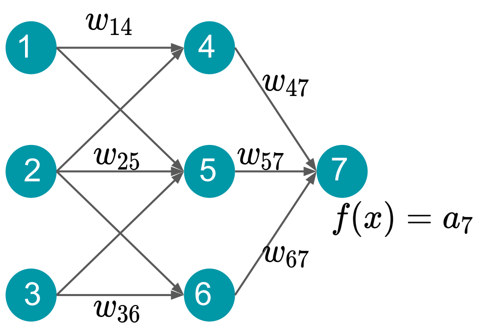


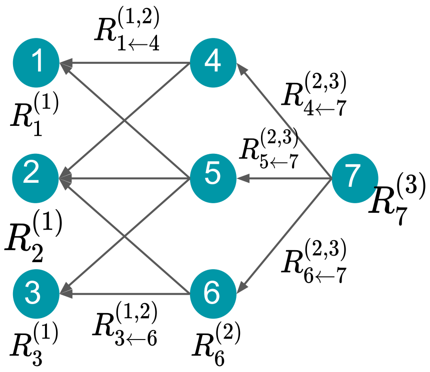


$$f(x) = \boxed{R_7^{(3)}} = \boxed{R_4^{(2)} + R_5^{(2)} + R_6^{(2)}} = \boxed{R_1^{(1)} + R_2^{(1)} + R_3^{(1)}}$$

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

Relevance propagation: local conservation





$$R_7^{(3)} = R_{4\leftarrow 7}^{(2,3)} + R_{5\leftarrow 7}^{(2,3)} + R_{6\leftarrow 7}^{(2,3)}$$

$$R_k^{(l+1)} = \sum_{i\,:\,i \text{ is input for neuron } k} R_{i\leftarrow k}^{(l,l+1)}$$

$$R_1^{(1)} = R_{1\leftarrow 4}^{(1,2)} + R_{1\leftarrow 5}^{(1,2)}$$

$$R_i^{(l)} = \sum_{k:i \text{ is input for neuron } k} R_{i\leftarrow k}^{(l,l+1)}$$

Connection between global and local relevance:

$$\sum_k R_k^{(l+1)} = \sum_k \sum_{i:i \text{ is input for neuron } k} R_{i\leftarrow k}^{(l,l+1)} = \sum_i \sum_{k:i \text{ is input for neuron } k} R_{i\leftarrow k}^{(l,l+1)} = \sum_i R_i^{(l)}$$

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

Relevance propagation. main properties

Global conservation:

$$f(x) = \cdots = \sum_{d \in l+1} R_d^{(l+1)} = \sum_{d \in l} R_d^{(l)} = \ldots = \sum_{d \in 1} R_d^{(1)}$$

Local conservation:

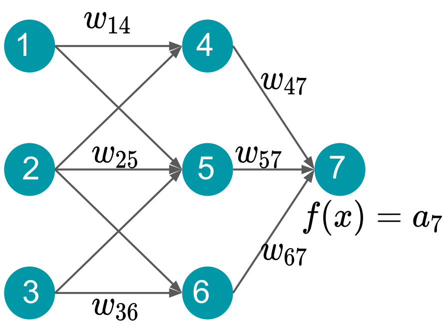
$$R_k^{(l+1)} = \sum_{i\,:\,i \text{ is input for neuron } k} R_{i \leftarrow k}^{(l,l+1)} \qquad R_i^{(l)} = \sum_{k:i \text{ is input for neuron } k} R_{i \leftarrow k}^{(l,l+1)}$$

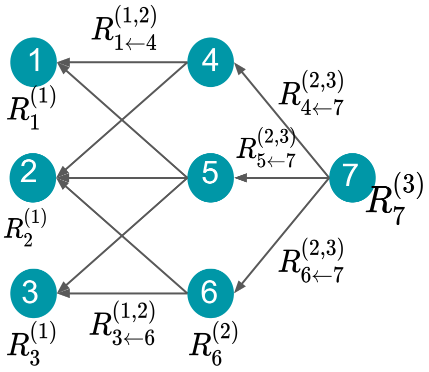
Example of relevance propagation

Let a_j be an activation of the j-th neuron

Then the relevance can be distributed as follows: $R_j = \sum_k \frac{a_j w_{jk}}{\sum_j a_j w_{jk}} R_k$

$$R_{j \leftarrow k}^{(l,l+1)} = \frac{a_j w_{jk}}{\sum_j a_j w_{jk}}$$





Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

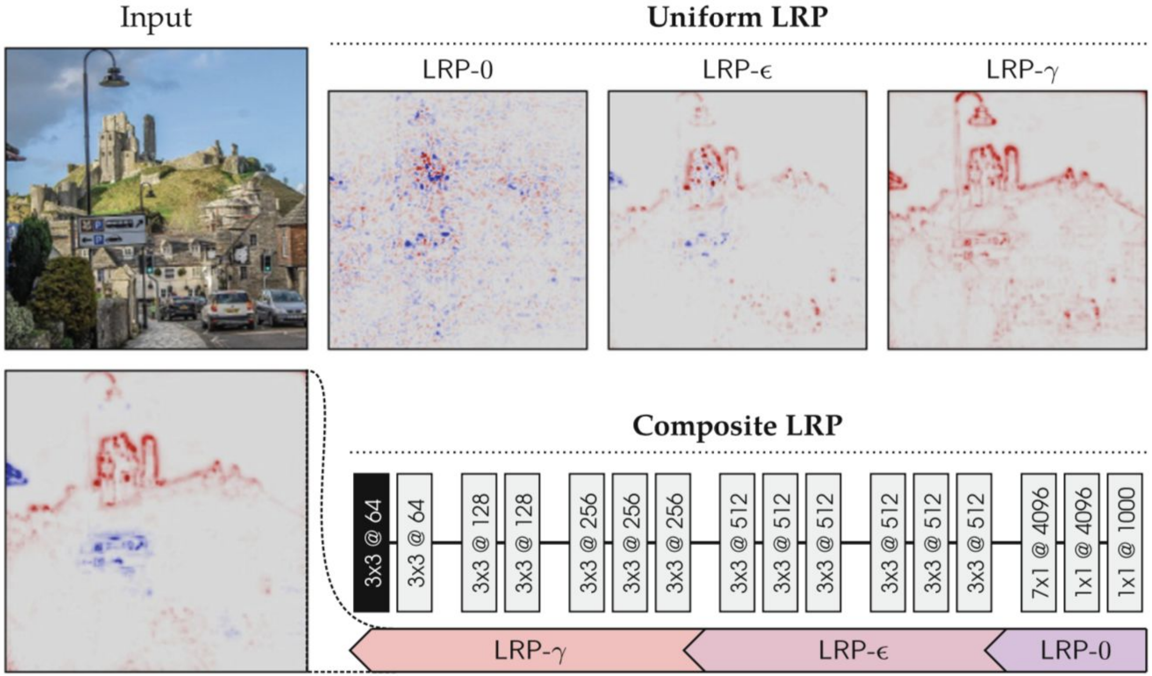
LRP for Deep Rectifier Networks

DRN is composed of neurons $a_j = \max(0, \Sigma_i a_i w_{ij} + b_i)$

	propagation property	approximation of the conservation property
LRP-0	$R_j = \sum_k \frac{a_j w_{jk}}{\sum_j a_j w_{jk} + b_j} R_k$	$\sum_j R_{j \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \left(1 - \frac{b_k}{\sum_j a_j w_{jk} + b_k}\right)$
LRP-ε	$R_j = \sum_k \frac{a_j w_{jk}}{\sum_j \varepsilon + a_j w_{jk} + b_j} R_k$	$\sum_j R_{j \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \left(1 - \frac{b_k + \varepsilon}{\sum_j a_j w_{jk} + b_k + \varepsilon}\right)$
LRP-γ	$R_j = \sum_k \frac{a_j (w_{jk} + \gamma w_{jk}^+)}{\sum_j \varepsilon + a_j (w_{jk} + \gamma w_{jk}^+) + b_j} R_k$	$\sum_j R_{j \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \left(1 - \gamma \frac{b_k + b_k^+}{\sum_j a_j w_{jk} + b_k + \sum_j (a_j w_{jk})^+ + b_k^+}\right)$
LRP-$\alpha\beta$	$R_j = \sum_k \left(\alpha \frac{(a_j w_{jk})^+}{\sum_j (a_j w_{jk}^+) + b_j} + \beta \frac{(a_j w_{jk})^-}{\sum_j (a_j w_{jk}^-) + b_j}\right) R_k$ $\alpha + \beta = 1$	$\sum_j R_{j \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \left(1 - \alpha \frac{(a_j w_{jk})^+}{\sum_j (a_j w_{jk})^+ + b_k^+}\right.$ $\left. - \beta \frac{(a_j w_{jk})^+}{\sum_j (a_j w_{jk})^- + b_k^-}\right)$

Montavon, et al. "Layer-wise relevance propagation: an overview." Explainable AI: interpreting, explaining and visualizing deep learning (2019): 193-209

An example of a combination of rules



Input image and pixel-wise explanations of the output neuron ‘castle’ obtained with various LRP procedures.

Parameters are $\varepsilon = 0.25$ std and $\gamma = 0.25$.

Montavon et al. "Layer-wise relevance propagation: an overview." Explainable AI: interpreting, explaining and visualizing deep learning (2019): 193-209.

When and which rules should be used?

LRP-0 picks many local artifacts of the function. Thus, the explanation is **overly complex** and **does not focus sufficiently** on the “actual explanation”, the explanation is neither faithful nor understandable

LRP-ε removes **noise elements** in the explanation to keep only a limited number features for the explanation. It provides a **faithful** explanation, but **too sparse to be easily understandable**

LRP-γ is easier for a human to understand because **features are more densely highlighted**, but it also picks unrelated concepts for the explanation, thus it is rather unfaithful

Composite LRP overcomes the disadvantages of the approaches above

Montavon, Grégoire, et al. "Layer-wise relevance propagation: an overview." Explainable AI: interpreting, explaining and visualizing deep learning (2019): 193-209.

Deep Taylor Decomposition. Motivation

How to justify LRP rules theoretically?

In LRP we express relevance of a neuron k using the relevance of the neurons from the upper layer

How to interpret negative values of relevance?

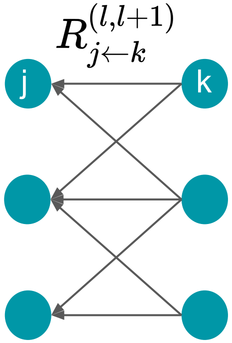
Classifying a ball, a dark ball on a bright background would have negative gradient, while white ball on darker background would have a positive gradient*.

* Smilkov, Daniel, et al. "SmoothGrad: removing noise by adding noise." arXiv preprint arXiv:1706.03825 (2017).

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Mu¨ller, K.R.: Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recogn. 65, 211–222 (2017)

Part 2. Deep Taylor Decomposition

Positive relevance propagation



Let a_j be a **non-negative** activation of the j-th neuron

We do not use negative values since it is difficult to interpret negative evidence for a class

Then the relevance can be distributed as follows:

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_j a_j w_{jk}} R_k \qquad R_j = \sum_k \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} R_k$$

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{a_i w_{ij}}{\sum_i a_i w_{ij}} \qquad R_{j\leftarrow k}^{(l,l+1)} = \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} R_k$$

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." PloS one 10.7 (2015): e0130140.

Positive relevance propagation

$$R_j = \sum_k \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} \boxed{R_k}$$

assume $R_k = a_k c_k$, where c_k is a positive constant

$$R_j = a_j \boxed{\sum_k w_{jk}^+ \frac{max(0, \sum_j a_j w_{jk})}{\sum_j a_j w_{jk}^+} c_k}$$

$R_j = a_j c_j$ where c_j is positive and approximately constant

Deep Taylor Decomposition

In DTD we suppose that the relevance of the current neuron k is a function of the lower-level neuron activations $\{a_i\}$, i.e., $R_j(\{a_i\})$, where {} denotes a vector.

$$R_j = a_j c_j$$

$$\begin{aligned} R_j(\{a_i\}) = a_j(\{a_i\}) \cdot c_j &= max(0, \textstyle\sum_i a_i w_{ij} + b_j) \cdot c_j \\ &= max(0, \textstyle\sum_i a_i w_{ij} c_j + b_j c_j) \\ &= max(0, \textstyle\sum_i a_i w'_{ij} + b'_j) \end{aligned}$$

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Mu ̈ller, K.R.: Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recogn. 65, 211–222 (2017)

Deep Taylor Decomposition

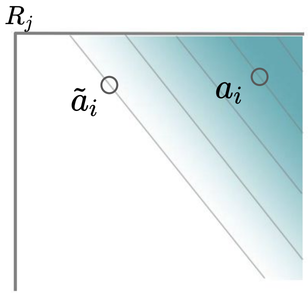
$$R_j(\{a_i\}) = max(0, \sum_i a_i w'_{ij} + b'_j)$$

First-order Taylor expansion $R_j(\{a_i\}) = R_j(\{\tilde{a}_i^{(j)}\}) + \sum_i \frac{\partial R_j}{\partial a_i}\Big|_{\tilde{a}_i^{(j)}} \cdot (a_i - \tilde{a}_i^{(j)}) + \varepsilon$

Conservation property:

$$\sum_j R_j = \left(\frac{\partial(\sum_j R_j)}{\partial\{x_i\}}\Big|_{\{\widetilde{x}_i\}}\right)^\top \cdot (\{x_i\} - \{\widetilde{x}_i\}) + \varepsilon$$

$$= \sum_i \underbrace{\sum_j \frac{\partial R_j}{\partial x_i}\Big|_{\{\widetilde{x}_i\}} \cdot (x_i - \widetilde{x}_i)}_{R_i} + \varepsilon,$$

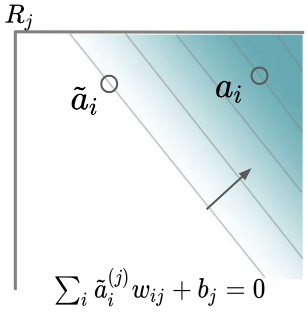


Remark: due to the potentially complex relation between a_j and R_k, finding an appropriate reference point and computing the gradient locally is difficult

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Mu ̈ller, K.R.: Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recogn. 65, 211–222 (2017)

Deep Taylor Decomposition

First-order Taylor expansion $R_j(\{a_i\}) = R_j(\{\tilde{a}_i^{(j)}\}) + \sum_i \frac{\partial R_j}{\partial a_i}\Big|_{\tilde{a}_i^{(j)}} \cdot (a_i - \tilde{a}_i^{(j)}) + \varepsilon$

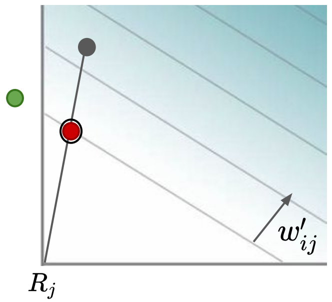


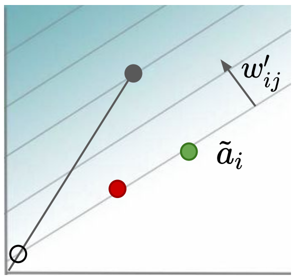
We search for a root point such that $R_j(\{a_i\}) = 0 + \sum_i \underbrace{\boxed{\frac{\partial R_j}{\partial a_i}\Big|_{\tilde{a}_i^{(j)}} \cdot (a_i - \tilde{a}_i^{(j)})}}_{R_{i \leftarrow j}} + 0$

Closed form solution: $R_{i \leftarrow j} = \frac{v_{ij} w_{ij}}{\sum_i v_{ij} w_{ij}} R_j$ where $v_{ij} \propto a_i - \tilde{a}_i^{(j)}$ is the root search direction

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Müller, K.R.: Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recogn. 65, 211–222 (2017)

Choosing search directions for a root point





root point	search direction
nearest root	$v_{ij} = w_{ij}$
origin	$v_{ij} = a_i$
LRP equivalent	$v_{ij} = a_i 1_{w'_{ij}>0}$

Remark: Only the last one guarantees that the root point 1) belongs to the input domain, 2) has non-negative relevance scores

Commonly used LRP rules

Name	Formula	Usage	DTD
LRP-0 [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$	Upper layers	✓
LRP-ϵ [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k$	Middle layers	✓
LRP-γ	$R_j = \sum_k \frac{a_j (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j (w_{jk} + \gamma w_{jk}^+)} R_k$	Lower layers	✓
LRP-$\alpha\beta$ [7]	$R_j = \sum_k \left(\alpha \frac{(a_j w_{jk})^+}{\sum_{0,j} (a_j w_{jk})^+} - \beta \frac{(a_j w_{jk})^-}{\sum_{0,j} (a_j w_{jk})^-} \right) R_k$	Lower layers	$\times^a$
flat [30]	$R_j = \sum_k \frac{1}{\sum_j 1} R_k$	Lower layers	$\times$
w^2-rule [36]	$R_i = \sum_j \frac{w_{ij}^2}{\sum_i w_{ij}^2} R_j$	First layer ($\mathbb{R}^d$)	✓
$z^{\mathcal{B}}$-rule [36]	$R_i = \sum_j \frac{x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_i x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-} R_j$	First layer (pixels)	✓

([a]DTD interpretation only for the case $\alpha = 1, \beta = 0$.)

Montavon, Grégoire, et al. "Layer-wise relevance propagation: an overview." Explainable AI: interpreting, explaining and visualizing deep learning (2019): 193-209.

Further reading: Layer-Wise Relevance Propagation

Site with plenty of sources: http://heatmapping.org/

Tutorial on implementation: https://git.tu-berlin.de/gmontavon/lrp-tutorial

Example of implementation: https://github.com/atulshanbhag/Layerwise-Relevance-Propagation/blob/master/vgg/lrp.py

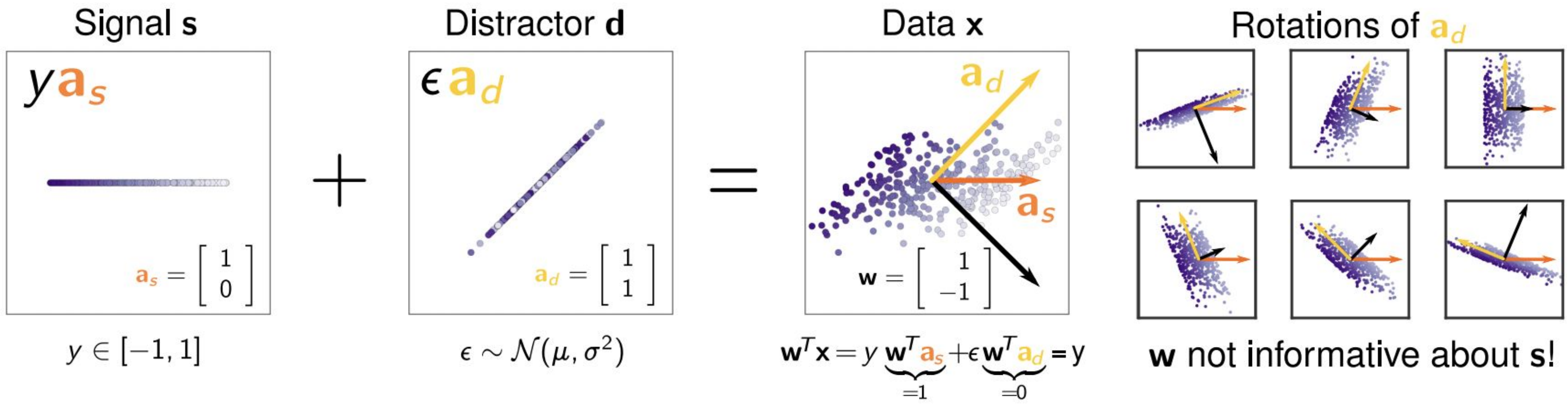
Part 3. PatternNet and PatternAttribution

PatternNet and PatternAttribution

Reasoning line

- the linear model is the simplest neural network
- the explanation methods should work correctly in the limit of simplicity
- data contain the signal and some distortion
- the explanation method should work well at least for the simplest cases

Understanding a linear model. Example



The weight vector is not aligned with the signal, its primary is to **cancel the distractor**

The weight vercor is able to filter out the distractor, that is why it is called a filter

Understanding a linear model

For a given data

$$\boldsymbol{x} = \underbrace{y\boldsymbol{a}_s}_{signal} + \underbrace{\varepsilon\boldsymbol{a}_d}_{distractor} = \boldsymbol{s} + \boldsymbol{d}$$

$$\boldsymbol{w}^T\boldsymbol{x} = y\underbrace{\boldsymbol{w}^T\boldsymbol{a}_s}_{=1} + \varepsilon\underbrace{\boldsymbol{w}^T\boldsymbol{a}_d}_{=0} = y$$

The filter w tells us how to extract y optimally from data x. The pattern as is the direction in the data along which the desired output y varies

Going back to the studied models

- **Functions** (gradients, saliency maps): $\partial y / \partial \boldsymbol{x} = \boldsymbol{w}$
- **Signals** (DeconvNet, Guided Backpropagaton): propagating back an "assumed signal" $\boldsymbol{s} = \boldsymbol{a}_s y$ toward the signal direction of each neuron
- **Attribution** (LRP, DeepTaylor): assessing how much the signal dimensions contribute the the output through the layers

$$\boldsymbol{r}_i^{l-1} = \frac{\boldsymbol{w} \odot (\boldsymbol{x} - \boldsymbol{x}_0)}{\boldsymbol{w}^T \boldsymbol{x}} \boldsymbol{r}_i^l \qquad \boldsymbol{r}_i^{output} = y$$

Remark: selecting a root point for the DeepTaylorDecomposition corresponds to estimating the distractor $\boldsymbol{d} = \boldsymbol{x}_0$ and, by that, the signal $\boldsymbol{s}^* = \boldsymbol{x} - \boldsymbol{x}_0$

Training a model

Goal: to train the filter $\boldsymbol{w}$ to extract y such that $\boldsymbol{w}^\mathsf{T}\boldsymbol{x} = y$, $\boldsymbol{w}^\mathsf{T}\boldsymbol{s} = y$, $\boldsymbol{w}^\mathsf{T}\boldsymbol{d} = 0$

It is an **ill-posed problem**. We could limit ourselves to the linear estimators if the **following form**: $\hat{\mathbf{s}} = \boldsymbol{u}(\boldsymbol{w}^\mathsf{T}\boldsymbol{u})^{-1}y$ with a random vector $\boldsymbol{u}$ such that $\boldsymbol{w}^\mathsf{T}\boldsymbol{u} \neq 0$

For such an estimator $\hat{\mathbf{s}}$ it is always true that $\boldsymbol{w}^\mathsf{T}\hat{\mathbf{s}} = y$

There exist **infinitely many solutions** (as many as the propagation rules for the previously considered methods)

Quality measure for signal estimates

Let the signal estimator be $S(x)$ = $\hat{\mathbf{s}}$, the distractor be $\hat{\boldsymbol{d}} = \boldsymbol{x} - S(\boldsymbol{x})$and $w^{\mathsf{T}}x$ = y:

$$\rho(S) = 1 - \max_{\boldsymbol{v}} corr\left(\boldsymbol{w}^T \boldsymbol{x}, \boldsymbol{v}^T \left(\boldsymbol{x} - S(\boldsymbol{x})\right)\right)$$

This criterion introduces an additional constraint by measuring how much information about y can be reconstructed from the residuals $x - S(x)$ using a linear projection

The **best signal estimators remove most of the information in the residuals** and thus yield large ϱ

Thus, we want that the distractor correlate a lot with the weight vector w (filter) (since the primary objective of the filter is to cancel the distortion)

Quality measure for signal estimates

Let the signal estimator be $S(x)$ = $\hat{\mathbf{s}}$ is optimal w.r.t. the following equation

$$\rho(S) = 1 - \max_{\boldsymbol{v}} corr\left(\boldsymbol{w}^T\boldsymbol{x}, \boldsymbol{v}^T\left(\boldsymbol{x} - S(\boldsymbol{x})\right)\right) = 1 - \max_{\boldsymbol{v}} \frac{\boldsymbol{v}^T \mathrm{cov}[\hat{\boldsymbol{d}}, y]}{\sqrt{\sigma^2_{\boldsymbol{v}^T\hat{\boldsymbol{d}}}\sigma^2_y}}$$

if the correlation is 0 for all possible v, i.e., $\forall \boldsymbol{v}, \mathrm{cov}[y, \hat{\boldsymbol{d}}]\boldsymbol{v} = \boldsymbol{0}$

From the linearity of the covariance and $\hat{\boldsymbol{d}} = \boldsymbol{x} - S(\boldsymbol{x})$ it follows

$$\mathrm{cov}[y, \hat{\boldsymbol{d}}] = \boldsymbol{0} \Rightarrow \mathrm{cov}[\boldsymbol{x}, y] = \mathrm{cov}[S(\boldsymbol{x}), y]$$

Signal estimates for the linear neurons

A linear neuron can extract only a linear signal from x, thus we may assume the following dependency:

$$S_{\boldsymbol{a}}(\boldsymbol{x}) = \boldsymbol{a}\boldsymbol{w}^T\boldsymbol{x}.$$

Plugging this into $\operatorname{cov}[\boldsymbol{x}, y] = \operatorname{cov}[S(\boldsymbol{x}), y]$ we get

$$\operatorname{cov}[\boldsymbol{x}, y] = \operatorname{cov}[\boldsymbol{a}\boldsymbol{w}^T\boldsymbol{x}, y] = \boldsymbol{a}\operatorname{cov}[y, y] \Rightarrow \boldsymbol{a} = \frac{\operatorname{cov}[\boldsymbol{x}, y]}{\sigma_y^2}$$

Since the correlation is invariant to scaling, we constrain $\boldsymbol{v}^T\hat{\boldsymbol{d}}$ to have $\sigma^2_{\boldsymbol{v}^T\hat{\boldsymbol{d}}} = \sigma_y^2$

Signal estimates for the ReLU neurons

ReLU does not propagate the negative activations, thus we distinguish the following regimes:

$$\boldsymbol{x} = \begin{cases} \boldsymbol{s}_+ + \boldsymbol{d}_+ & \text{if } y > 0 \\ \boldsymbol{s}_- + \boldsymbol{d}_- & \text{otherwise} \end{cases}$$

The signal can be estimated as follows:

$$S_{\boldsymbol{a}+-}(\boldsymbol{x}) = \begin{cases} \boldsymbol{a}_+ \boldsymbol{w}^T \boldsymbol{x}, & \text{if } \boldsymbol{w}^T \boldsymbol{x} > 0 \\ \boldsymbol{a}_- \boldsymbol{w}^T \boldsymbol{x}, & \text{otherwise} \end{cases}$$

Signal estimates for the ReLU neurons

Let $\mathbb{E}[x]_+$ and $\mathbb{E}[x]_-$ be expectations over x within positive and negative regimes, respectively, and π_+ be the expected ratio of input $\boldsymbol{x}$ with $\boldsymbol{w}^\mathsf{T}\boldsymbol{x}$ then

$$\mathrm{cov}[\boldsymbol{x}, y] = \quad \pi_+ \left(\mathbb{E}_+[\boldsymbol{x}y] - \mathbb{E}_+[\boldsymbol{x}]\,\mathbb{E}[y]\right) \; + \quad (1-\pi_+)\left(\mathbb{E}_-[\boldsymbol{x}y] - \mathbb{E}_-[\boldsymbol{x}]\,\mathbb{E}[y]\right)$$
$$\mathrm{cov}[\boldsymbol{s}, y] = \quad \pi_+ \left(\mathbb{E}_+[\boldsymbol{s}y] - \mathbb{E}_+[\boldsymbol{s}]\,\mathbb{E}[y]\right) \; + \quad (1-\pi_+)\left(\mathbb{E}_-[\boldsymbol{s}y] - \mathbb{E}_-[\boldsymbol{s}]\,\mathbb{E}[y]\right)$$

Plugging it into $\mathrm{cov}[\boldsymbol{x}, y] = \mathrm{cov}[S(\boldsymbol{x}), y]$

$$\mathbb{E}_+[\boldsymbol{x}y] - \mathbb{E}_+[\boldsymbol{x}]\,\mathbb{E}[y] \quad = \quad \mathbb{E}_+[\boldsymbol{s}y] - \mathbb{E}_+[\boldsymbol{s}]\,\mathbb{E}[y]$$

Since $S_{\boldsymbol{a}+-}(\boldsymbol{x}) = \begin{cases} \boldsymbol{a}_+\boldsymbol{w}^T\boldsymbol{x}, & \text{if } \boldsymbol{w}^T\boldsymbol{x} > 0 \\ \boldsymbol{a}_-\boldsymbol{w}^T\boldsymbol{x}, & \text{otherwise} \end{cases}$ $\qquad \boldsymbol{a}_+ \quad = \quad \dfrac{\mathbb{E}_+[\boldsymbol{x}y] - \mathbb{E}_+[\boldsymbol{x}]\,\mathbb{E}[y]}{\boldsymbol{w}^T\,\mathbb{E}_+[\boldsymbol{x}y] - \boldsymbol{w}^T\,\mathbb{E}_+[\boldsymbol{x}]\,\mathbb{E}[y]}$

PatternNet

It is a **layer-wise back-projection** of the estimated signal to input space

The signal estimator is approximated as a superposition of neuron-wise, non-linear signal estimator $S_{\boldsymbol{a}+-}$ at each layer

It is equal to the **gradient** where during the backward pass the **weights** of the network are **replaced** by **the informative directions**

Initialization: $\boldsymbol{s}_i^{output} = \boldsymbol{y}$

Linear or convolutional layers: $\boldsymbol{s}^{l-1,i} = \boldsymbol{a}_+ s_i^l$

Conservation property: $\boldsymbol{s}_i^{l-1} = \sum_j \boldsymbol{s}_i^{l-1,j}$

ReLU: $\boldsymbol{s}_i^{l-1} = \begin{cases} \boldsymbol{s}_i^l, & x_i^l > 0 \\ 0, & \text{otherwise} \end{cases}$

PatternAttribution

It exposes the attribution $\boldsymbol{w} \odot \boldsymbol{a}_{+}$
It can be considered as a root point for DeepTaylorDecomposition

Let $\boldsymbol{r}^{l-1,i}$ be the relevance of the l-th layer. The distribution has the following form

$$\boldsymbol{r}^{l-1,i} = \frac{\boldsymbol{w} \odot (\boldsymbol{x} - \boldsymbol{x}_0)}{\boldsymbol{w}^T \boldsymbol{x}} r_i^l$$

For the original LRP $\boldsymbol{x}_0 = 0$
For PatternAttribution $\boldsymbol{x}_0 = \boldsymbol{x} - \boldsymbol{a}_{+} \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}$, that gives $\boldsymbol{r}^{l-1,i} = \boldsymbol{w} \odot \boldsymbol{a}_{+} \; \boldsymbol{r}^{l}_{,i}$

Method comparison