Aspects of explainable AI

Alexander Binder

University of Oslo (UiO)

May 4, 2023



UiO : Department of Informatics
University of Oslo

In this lecture we can cover only a few aspects (a lecture into coding LRP for pytorch takes me alone 1 hour!).

- model interpretation (two approaches)
  - t-SNE and UMAP embeddings (pytorch + scikit learn): show similarities for features from a dataset
  - compute which inputs activate a feature most
- decision interpretation (one unlabeled test sample as compared to a dataset)
  - Shapley values
  - Linearization-based ideas: Gradient, sensitivity, gradient times input, Grad-CAM, guided backprop, LRP and Deep Taylor
  - applications to finding biases in your data, to identify systematic failcases

Learning goals I

- an overview of different questions in explainability
- some selected aspects:
  - low-dimensional embeddings of a set of samples: t-SNE
  - samples maximizing activations in a feature layer
  - nearest examples with respect to a metric defined by a feature layer
  - applications to finding biases in your data, to identify systematic failcases

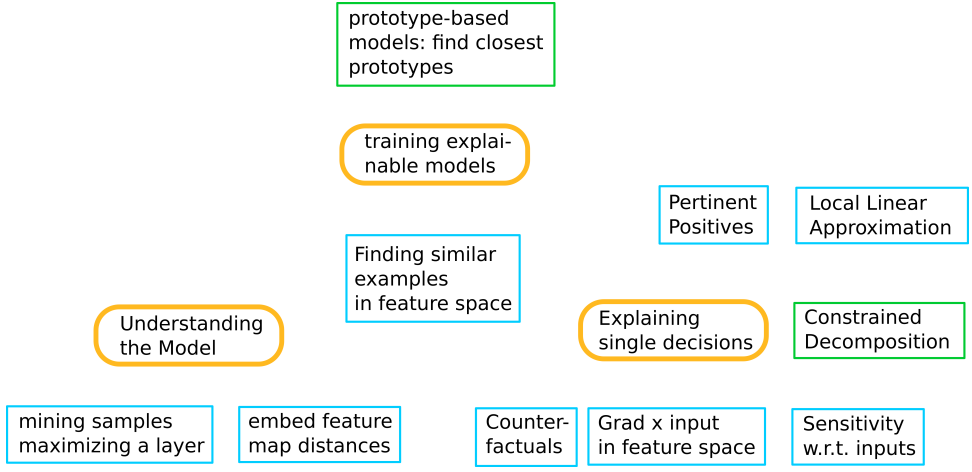
Learning goals II

- an overview of different questions in explainability
- some selected aspects:
  - explanations of a single decision
    - norm of gradients, gradient times input, integrated gradient, Grad-CAM, Occlusion, Guidedbackprop, LRP

You do not need to memorize complex LRP or t-SNE formulas for the exam, only simple formulas like gradient times input

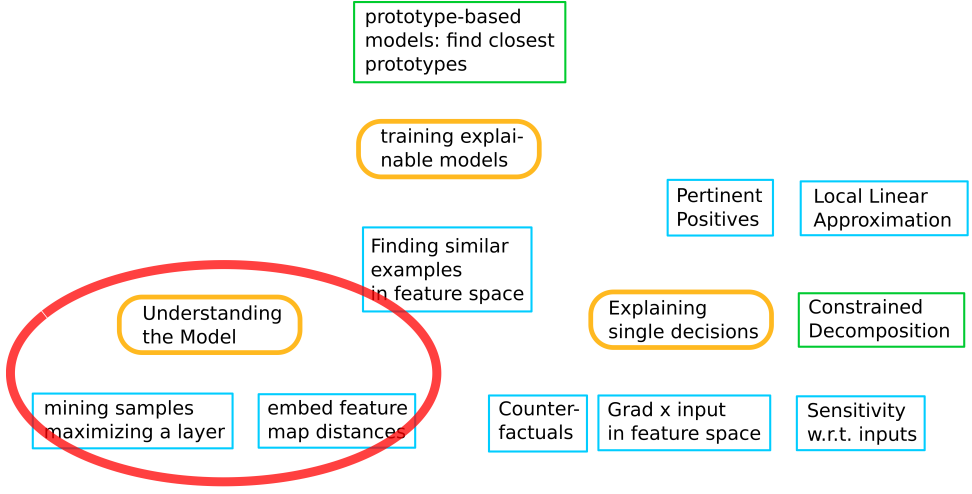
1. The definition of an explanation depends on the question

1. As many definitions of explanation as there are different questions



Authors opinion: no method rules them all

1. As many definitions of explanation as there are different questions

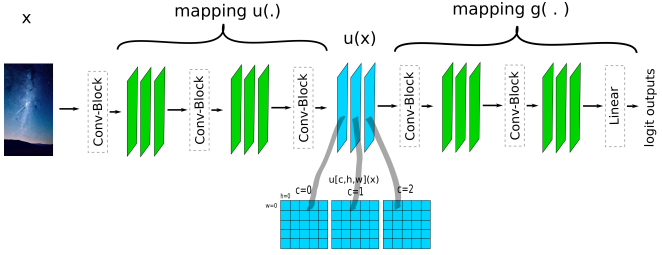


Authors opinion: no method rules them all

1. Model interpretation
2. Decision interpretation

Define low-dimensional embeddings which approximate high-dimensional distances of a feature map

Popular in deep learning: t-SNE
van der Maaten et al.: https://lvdmaaten.github.io/tsne/examples/caltech101_tsne.jpg



- take any feature map of a neural network $u[0, c, h, w](x)$ which depends on an input sample x. Compute distances between different samples x_i, x_j:

$$d_u(x_i, x_j) = \|u[0, c, h, w](x_i) - u[0, c, h, w](x_j)\| = \|u_i - u_j\|$$

- Idea: visualize distances $d_u(x_i, x_j)$ over a set of samples $x_i, x_j, x_k, x_l, x_m, \ldots$ by low-dimensional approximation

Suppose you have feature maps of 200 images, how to visualize the similarities between these feature maps ?

- Technical problem: how to represent a number of d-dimensional features in 2-dims such that the distances from the original samples in d dimensions are meaningfully preserved?

- One way is t-SNE (L. van der Maaten):
  http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf

  https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html#sphx-glr-auto-examples-manifold-plot-lle-digits-py
- Working principle: Given high dimensional data points $D_n = (u_1, \ldots, u_n)$, map each $u_i \in D_n$ onto a 2-dimensional data point y_i such that y_i, y_j which have similar distances to each other as the samples u_i, u_j from the set D_n.

other methods for visualizing similarities between features?

- t-SNE http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
- alternative: UMAP https://arxiv.org/abs/1802.03426

Properties and limitations of such methods in general:

- https://distill.pub/2016/misread-tsne/ – what one gets out from t-SNE (and this class), depends a lot on the perplexity parameter choice, and it may be very different from the original distances.
- It is a visualization, not a kind of truth. Results need to be validated by another way of analysis.

- $\odot$ IDEA: visualize distances by low-dimensional approximation
- (-) how to choose a low-dimensional approximation? Is there the one?
- (-) parameter sensitivity https://distill.pub/2016/misread-tsne/
- (+) makes things visualizable
- (!) alternatives: UMAP, Isomap (Tenenbaum et al. graph defined by k-nearest neighbors and euclidean distances along edges), PCA projections (K. Pearson), many others
- (!) good for exploration but requires a follow up confirmation of findings

Next: another way of Understanding the model

DeepDream as an example of Activation Maximization



Credit: https://github.com/gordicaleksa/pytorch-deepdream

(!) In what ways can one enhance it with more than esthetic value?

(+) visually pleasing

simple idea:

- choose a feature map channel $f[0, c, h, w]$ and a statistic of a feature map, for example

$$s_c(x) = \sum_{h,w} \max(0, f[0, c, h, w])$$

- the feature map statistic depends on the input image x
- do gradient-based optimization of x to maximize s_c - similar as for adversarial attacks
- in code: use $L = -s_c$ as Loss function, but minimize over the input sample instead of model parameters, as done for adversarial attacks

Next: another way of Understanding the model as a whole

- select a layer/feature map $h(\cdot)$
- given a sample x and its feature map $h(x)$, find the closest examples in a test set, given a norm $\|\cdot\|$ or metric $d(h(x), h(x_i))$

sort similar samples x_i according to $\|h(x_i) - h(x)\|$

Understanding the model: Mining samples which maximize a channel activation

The top-3 images which maximally activate a particular channel of layer `layer4.2.conv2` of a ResNet-50 after fine-tuning on Pascal VOC. Selection within the Pascal VOC validation set. The picture also shows an explanation which region in the image is contributing to the activation of the channel (LRP explanation from the maximally activated spatial element).



Channel has learnt to detect bus views.

- ⊙ select a layer/feature map $h(\cdot)$
- ⊙ given a sample x and its feature map $h(x)$, find the closest examples in a test set, given a norm $\|\cdot\|$ or metric $d(h(x), h(x_i))$
- (+) uses model-based learned similarities around x. $h(\cdot) = h_w(\cdot)$. w are network parameters
- (-) if $\|h(x_i) - h(x)\|$ is small in a layer, it does not guarantee that the final prediction will be similar for x_i and x or that the internal reasoning will be similar for x_i and x. Could diverge in higher layers!
  - Extreme case of divergence: $h(\cdot) =$ input layer and adversarials.
- (!) difference to t-SNE: this tries to explain similarities to a single sample x as defined by a chosen feature map, not visualize a set of samples or what activates a feature map most

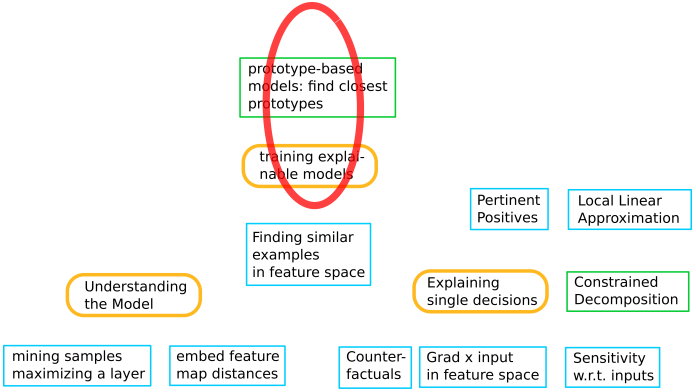
1 Model interpretation

2 Decision interpretation

What is the difference between Model understanding and Decision interpretation ?

- Have a single sample $x = (x_1, \ldots, x_d)$ and a prediction $f(x)$.
- Decision interpretation: focus on that one concrete sample
- Define algorithms/measures which state:
  - how much are parts of x are important to arrive at the prediction $f(x)$.

1. The definition of an explanation depends on the question



Authors opinion: no method rules them all

Two ideas:

- Find the most similar samples to the sample which is used for prediction.
- make sure to use only similar samples which were involved in the prediction

Has some known examples:

- k-nearest neighbors
- What classifier can be expressed like this ?

$$f(x) = \sum_i \alpha_i y_i k(x_i, x)$$

Two ideas:

- Find the most similar samples to the sample which is used for prediction.
- make sure to use only similar samples which were involved in the prediction

This Looks Like *That*: Deep Learning for Interpretable Image Recognition

Chaofan Chen*
Duke University
cfchen@cs.duke.edu

Oscar Li*
Duke University
oscarli@alumni.duke.edu

Chaofan Tao
Duke University
chaofan.tao@duke.edu

Alina Jade Barnett
Duke University
abarnett@cs.duke.edu

Jonathan Su
MIT Lincoln Laboratory†
su@ll.mit.edu

Cynthia Rudin
Duke University
cynthia@cs.duke.edu

Credit: Chen et al. https://proceedings.neurips.cc/paper/2019/file/adf7ee2dcf142b0e11888e72b43fcb75-Paper.pdf

Two ideas:

- Find the most similar samples to the sample which is used for prediction.
- make sure to use only similar samples which were involved in the prediction

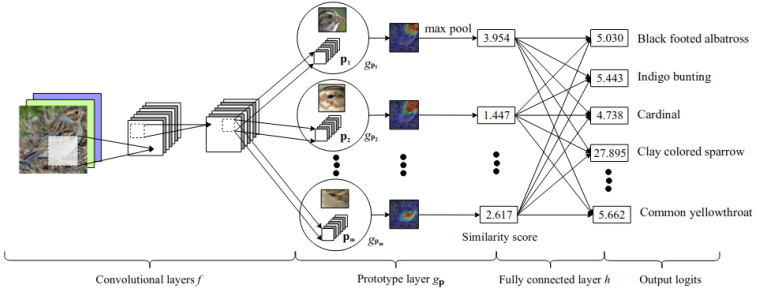


Figure 2: ProtoPNet architecture.

Credit: Chen et al. https://proceedings.neurips.cc/paper/2019/file/adf7ee2dcf142b0e11888e72b43fcb75-Paper.pdf

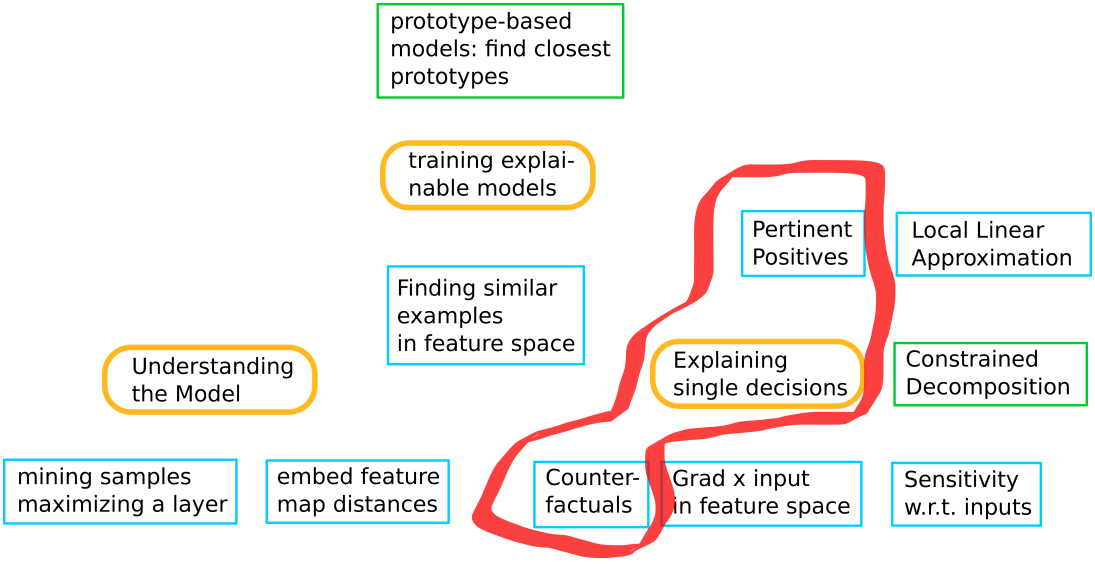
Two ideas:

- ⊙ Find the most similar samples to the sample which is used for prediction.
- ⊙ make sure to use only similar samples which were involved in the prediction

Properties ?

- (!) how to choose prototypes without combinatorial issues?
- (!) Prototypes learned or from training data?
- (!) possible semantic gap: input space versus feature space similarities
  https://arxiv.org/abs/2105.02968 – sensitivity to JPEG artefacts
- (!) scalability (intra-class and inter-class)?

1. The definition of an explanation depends on the question



Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives

Amit Dhurandhar*
IBM Research
Yorktown Heights, NY 10598
adhuran@us.ibm.com

Pin-Yu Chen*
IBM Research
Yorktown Heights, NY 10598
pin-yu.chen@ibm.com

Ronny Luss
IBM Research
Yorktown Heights, NY 10598
rluss@us.ibm.com

Chun-Chen Tu
University of Michigan
Ann Arbor, MI 48109
timtu@umich.edu

Paishun Ting
University of Michigan
Ann Arbor, MI 48109
paishun@umich.edu

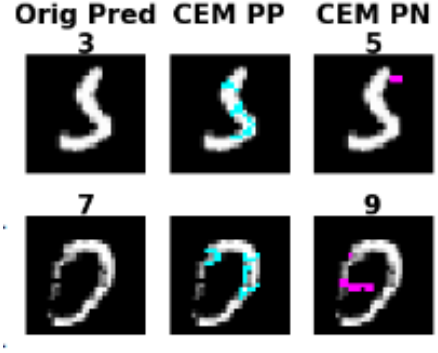
Karthikeyan Shanmugam
IBM Research
Yorktown Heights, NY 10598
karthikeyan.shanmugam2@ibm.com

Payel Das
IBM Research
Yorktown Heights, NY 10598
daspa@us.ibm.com

Credit: https://proceedings.neurips.cc/paper/2018/file/c5ff2543b53f4cc0ad3819a36752467b-Paper.pdf

Pertinent positive: what to retain from a sample?

Pertinent negative: what to change so that prediction switches?



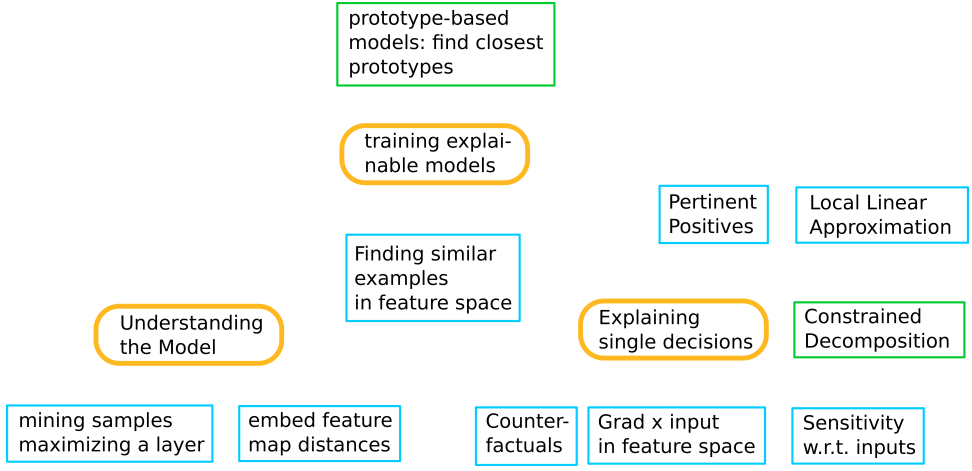
Pertinent positive: cyan, pertinent negative: pink

Credit: https://proceedings.neurips.cc/paper/2018/file/c5ff2543b53f4cc0ad3819a36752467b-Paper.pdf

- CHAL: how to delete/replace information? Result is plausible/outlier?
- many different PP/PN – how to integrate them?

Is this a complete picture?

Many ways to define an explanation of a prediction or a model.




衆聲
探象之圖

One can take it further in a human-machine interaction context in collaboration with HCI experts:

A dialogue between the human and the system.

Possibly open-ended question answering?

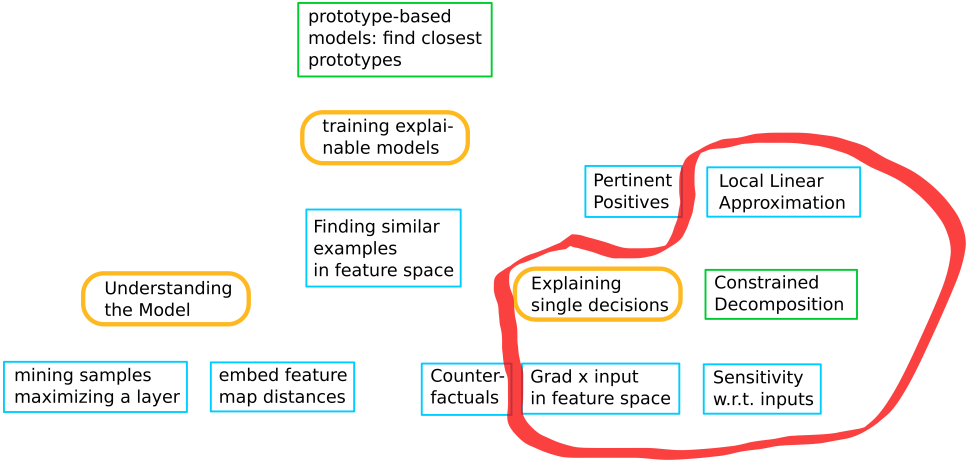
A system which can query on uncertain parts with counter questions.

System brings evidence when providing an answer – Explanations as a reasoning problem with self-awareness of ones internals.

- A path from explanation to reasoning about oneself.
- One possible goal of building general intelligent systems.

A more narrow scope: explaining predictions on a single sample by decomposition

Many ways to define an explanation of a prediction or a model.

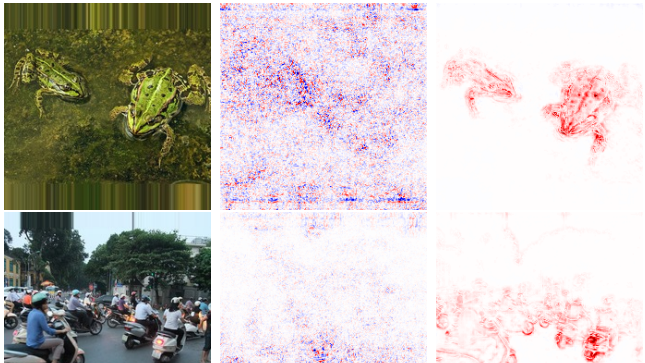


A more narrow scope: explaining predictions on a single sample by decomposition

- case of images: compute a score $r_d(x)$ for every input dimension d of the input sample $x = (x_1, \ldots, x_d, \ldots, x_D)$

$$\underbrace{f(x) \approx \sum_{d=1}^{D} r_d(x)}_{\textit{decomposition}}$$

- objective function is left open



What is a good explanation within the set of decomposition approaches?

What is a good explanation within the set of decomposition approaches?

- Is there a theoretically optimal approach?

Is there a theoretically optimal approach?

2. Shapley values

The Setup:

- Have function f, and a point to be explained $x = (x_1, \ldots, x_d)$.
- We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x.

Shapley value

then the Shapley value is defined as:

$$\phi_j(f, x) = \frac{1}{d} \sum_{S \subseteq \{1,\ldots,d\} \setminus \{j\}} \frac{f(x_{S \cup \{j\}}) - f(x_S)}{\binom{d-1}{|S|}}$$

Next slide:

- understand that Shapley values of a feature j can be interpreted as an average/expectation over the difference of two predictions
  - prediction on a set with feature j added
  - prediction on a set without feature j added

Its interpretation?

- Have function f, and a point to be explained $x = (x_1, \ldots, x_d)$.
- We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x.
- then the Shapley value is defined as:

$$\phi_j(f, x) = \frac{1}{d} \sum_{S \subseteq \{1,\ldots,d\} \setminus \{j\}} \frac{f(x_{S \cup \{j\}}) - f(x_S)}{\binom{d-1}{|S|}}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1 \text{ sets } S \text{ without } j, |S|=k} \sum \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|}$$

Game Theory: S is set of players playing a game with outcome $f(x_S)$.
j a member which can join the set with outcome $f(x_{S \cup \{j\}})$

Favourable theoretical properties:

- does not make a difference, zero Shapley-value

$$\forall S: \; f(x_{S\cup\{j\}}) = f(x_S) \Rightarrow \phi_j(f,x) = 0$$

- value-equal pair of features j, k, identical Shapley-value:

$$\forall S: \; f(x_{S\cup\{j\}}) = f(x_{S\cup\{k\}}) \Rightarrow \phi_j(f,x) = \phi_k(f,x)$$

- Efficiency:

$$\sum_{j=1}^{d} \phi_j(f,x) = f(x) - E_X[f(X)]$$

  - decomposition of $f(x)$
  - distributes the difference between function value $f(x)$ and its expectation $E_X[f(X)]$ onto all dimensions in an equal way (not shown here[1])

[1] Grabisch https://www.worldscientific.com/doi/abs/10.1142/S0218488597000440

We have interpreted SHAP.

How to categorize this approach? – This is slide provides an outlook, not exam topic.

$$\phi_j(f,x) = \frac{1}{d} \sum_{S \subseteq \{1,\dots,d\} \setminus \{j\}} \frac{f(x_{S\cup\{j\}}) - f(x_S)}{\binom{d-1}{|S|}}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1 \text{ sets } S \text{ without } j, |S|=k} \sum \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|}$$

We have interpreted SHAP.
How to categorize this approach? – This is slide provides an outlook, not exam topic.

$$\phi_j(f,x) = \frac{1}{d} \sum_{S \subseteq \{1,\dots,d\}\setminus\{j\}} \frac{f(x_{S\cup\{j\}}) - f(x_S)}{\binom{d-1}{|S|}}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1 \text{ sets } S \text{ without } j, |S|=k} \sum \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|}$$

Related to ... Combinatorially exhaustive occlusion differences for j, to occlusion methods
differential values: sets with player j, sets without j
(cf. Petliuk et al. RISE https://arxiv.org/abs/1806.07421)

This is slide provides an outlook, not exam topic.

- combinatorially exhaustive summing: can approximate by Monte-Carlo. Strumbelj, Kononenko (2014): https://link.springer.com/article/10.1007/s10115-013-0679-x

$$\phi_j(f, x) \approx \frac{1}{n} \sum_{l=1}^{n} f(x_{S_l \cup \{j\}}) - f(x_{S_l \setminus \{j\}})$$

We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x. – Outside of tabular data types this is a very strong assumption.

- How to remove a region in an image, a language sentence ?
- How to remove an interval of a data point being a time series ?
- is S, $S \cup \{j\}$ plausible or outlier (bonds and molecules) ?

Dimensions of a data sample $x \leftrightarrow$ Players in a game ?

No optimality guarantee when

- applicability assumptions do not hold for your data type
- one has to use approximations (e.g. MC)

- Shapley is very suitable for Tree predictors on low-dimensional tabular data

What other methods exist for data types where Shapley assumptions do not hold well?

3. Linearizations

- have input sample $x = (x_1, \ldots, x_D)$, a prediction $f(x)$ (classification or any other setting)
- question: which parts of x are important for the prediction $f(x)$?

This is slide provides an outlook, not exam topic.

Starting point was: $\underbrace{f(x) \approx \sum_{i=1}^{d} r_d(x)}_{\text{Decomposition}}$

if linear model: $e(x) = w \cdot x = \sum_d w_d x_d$

$$f(x) = \text{sign}(e(x))$$

then use this: $r_d(x) = w_d x_d$

or this: Shapley for $e(x)$: $\phi_d(e, x) = w_d x_d - w_d E[x_d]$

- Starting point was: $f(x) = \sum_{i=1}^{d} r_d(x)$ f is non-linear now
- Taylor Expansion up to first order:

$$\begin{aligned} f(x) &\approx f(x_0) + \nabla f(x_0) \cdot (x - x_0) \\ &= f(x_0) + \sum_d \frac{\partial f}{\partial x_d}|_{x_0}(x_d - x_{0,d}) \end{aligned}$$

- use as explanation:

$$r_d(x) = \frac{\partial f}{\partial x_d}|_{x_0}(x_d - x_{0,d})$$

- Gradient $\times$ Input
- Integrated Gradient
- LIME
- Grad-CAM
- LRP

Use as explanation:

$$r_d(x) = \frac{\partial f}{x_d}(x)\, x_d, \tag{1}$$

$$\mathbf{R} = (r_d, d = 1, \ldots, D) = \nabla f(x) \cdot x \tag{2}$$

- ⊙ theoretical motivation: derivation via global Taylor decomposition
- (–) can be noisy for deep ReLU-networks due to gradient shattering

This is slide provides an outlook, not exam topic.
Use as explanation:

$$r_d(x) = \frac{\partial f}{x_d}(x)\, x_d, \tag{3}$$

$$\mathbf{R} = (r_d, d = 1, \ldots, D) = \nabla f(x) \cdot x \tag{4}$$

⊙ derivation via global Taylor decomposition for a point x_0 orthogonal to the gradient ($\nabla f(x) \cdot x_0 = 0$) in the point x to be explained.

$$f(x_0) = f(x) + \nabla f(x) \cdot (x_0 - x) + \mathcal{O}(\|x - x_0\|^2) \tag{5}$$

$$\Rightarrow f(x) \approx f(x_0) + \nabla f(x) \cdot (x - x_0) \tag{6}$$

$$= f(x_0) + \nabla f(x) \cdot x - \underbrace{\nabla f(x) \cdot x_0}_{=0} \tag{7}$$

$f(x_0)$ is a scalar term, independent of any dimension

(–) can be noisy for deep ReLU-networks due to gradient shattering

Gradient shattering

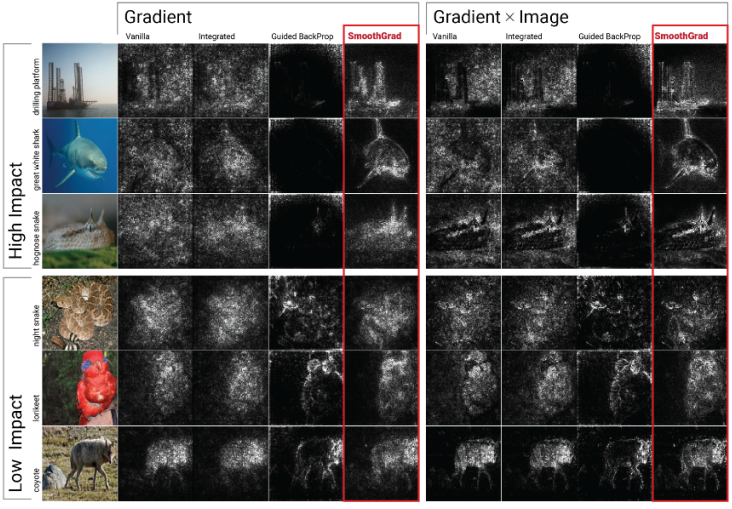


Figure 5. Qualitative evaluation of different methods. First three (last three) rows show examples where applying SMOOTHGRAD had high (low) impact on the quality of sensitivity map.

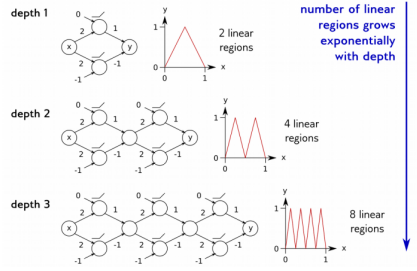
Credit: https://arxiv.org/pdf/1706.03825.pdf

This is slide provides an outlook, not exam topic.

The noisyness of gradient × input and related methods for deep ReLU-networks: Gradient Shattering Effect

- Montufar et al. 2014 https://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of-deep-neural-networks.pdf.
- Balduzzi et al. 2017 http://proceedings.mlr.press/v70/balduzzi17b/balduzzi17b.pdf.

The Shattered gradients problem [Montufar'14, Balduzzi'17]



Worth reading: works of Marco Ancona

For the next slides on methods you do not need to remember the formulas.
What you are required to understand is that

- Integrated Gradient and Grad-CAM use linearizations and averaging to remove noise
- LRP is also based on the idea to use linearizations

Integrated Gradient: *Axiomatic attribution for deep networks*
Sundararajan et al., *ICML 2017*
a heuristic very similar to the gradient times input idea:

$$R_d(x) = (x_d - x_d^{(0)}) \frac{1}{m} \sum_{k=1}^{m} \frac{\partial f}{\partial x_d}|_{z=x^{(0)}+\frac{k}{m}(x-x^{(0)})}$$

- $\odot$ Averages over partial derivatives along multiple points $x^{(0)} + \frac{r}{R}(x - x^{(0)})$ along a path from $x^{(0)}$ to x.
- $(+)$ Why that can be better than gradient $\times$ input? Averaging gradients smoothes out the gradient shattering.
- $(+)(-)$ IG gets better and slower when hundreds of points are used ($+$ slows down).

Smilkov et al. https://arxiv.org/pdf/1706.03825.pdf
Take the averaging idea further

- if you have any attribution map method $M(x)$ which depends on an input image x, then:
- compute an average over many slightly noised $x + \mathcal{N}(0, \sigma^2)$ input images

$$\widehat{M}(x) = \frac{1}{K} \sum_{i=1}^{K} M(x + n_i), n_i \sim \mathcal{N}(0, \sigma^2)$$

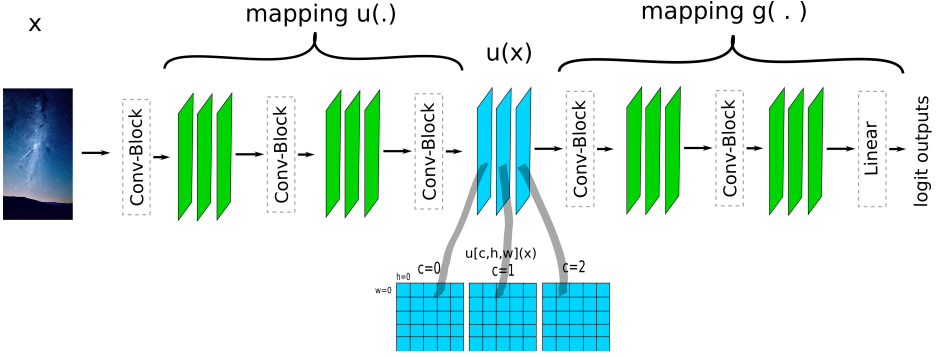
Compare to integrated gradient:

$$R_d(x) = \frac{1}{K} \sum_{i=1}^{K} (x_d - x_d^{(0)}) \frac{\partial f}{\partial x_d}|_{z=x^{(0)}+\frac{i}{K}(x-x^{(0)})}$$

IG samples along a line from $x^{(0)}$ to x (and uses a special $M(x)$). Smoothgrad samples from a normal distribution around x.

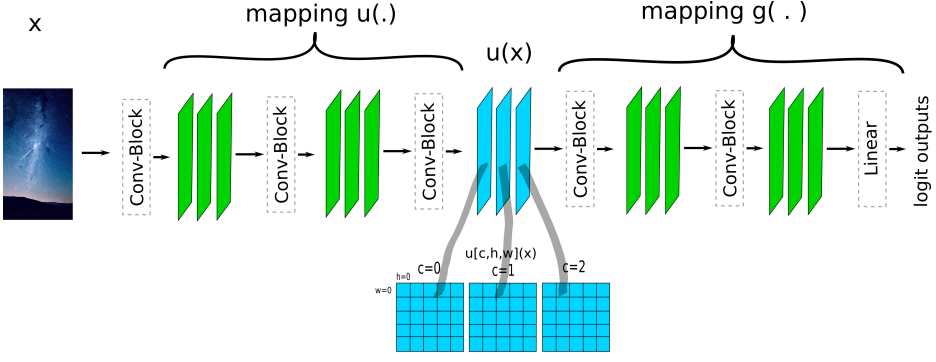
A heuristic variation on the gradient times input idea - applied in feature map space.
https://arxiv.org/abs/1610.02391

- given a feature map $u(x)$ with components $u[c, h, w](x)$ having C channels, width W, height H, and a predictor $f(x) = g(u(x))$ where $g(\cdot)$ are some layers on top of u.



Gradient times input in the space of u would be:

$$R[c, h, w] = \nabla^u \, g|_{u(x)} \odot u(x) = \left. \frac{\partial g}{\partial u[c, h, w]} \right|_{u(x)} u[c, h, w](x)$$

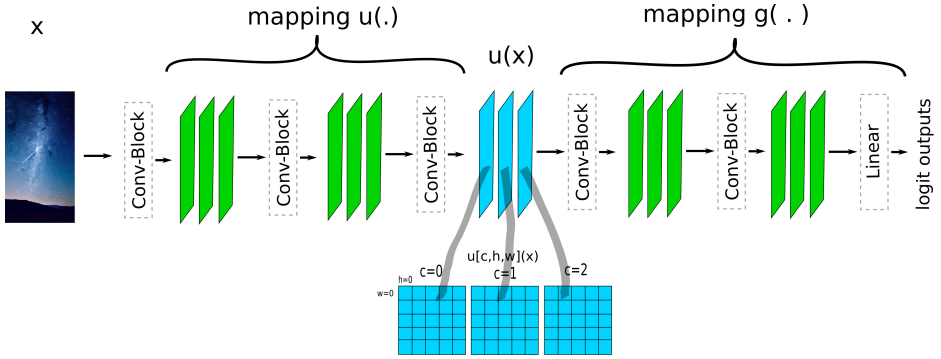


Gradient times input in the space of u would be:

$$R[c,h,w] = \nabla^u \, g|_{u(x)} \odot u(x) = \left.\frac{\partial g}{\partial u[c,h,w]}\right|_{u(x)} u[c,h,w](x)$$

Grad-Cam replaces the partial derivative $\frac{\partial g}{\partial u[c,h,w]}$ by a spatially averaged version

$$\alpha_c = \frac{1}{HW} \sum_{h,w} \left.\frac{\partial g}{\partial u[c,h,w]}\right|_{u(x)}$$



Grad-Cam replaces the partial derivative $\frac{\partial g}{\partial u[c,h,w]}$ by a spatially averaged version

$$\alpha_c = \frac{1}{HW} \sum_{h,w} \frac{\partial g}{\partial u[c,h,w]}\Big|_{u(x)}$$

$$R[h,w] = ReLU(\sum_{c=1}^{C} \alpha_c \; u[c,h,w])$$

and applies a weighted sum $\sum_c$ over all channels in this layer+ a *ReLU*

Grad-Cam replaces the partial derivative $\frac{\partial g}{\partial u[c,h,w]}$ by a spatially averaged version

$$\alpha_c = \frac{1}{HW} \sum_{h,w} \frac{\partial g}{\partial u[c,h,w]}\Big|_{u(x)}$$

$$R[h,w] = ReLU(\sum_{c=1}^{C} \alpha_c \; u[c,h,w])$$

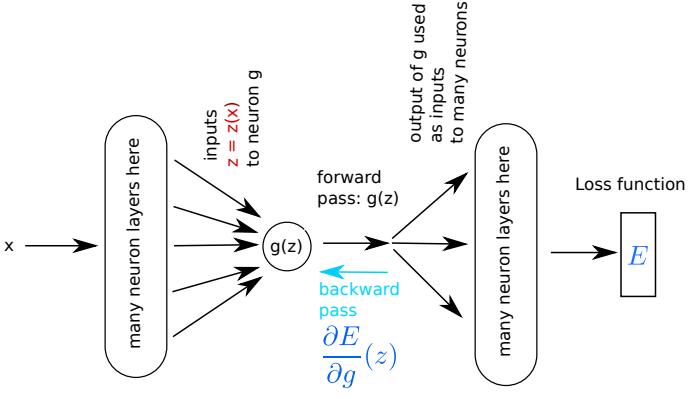
high level view:

- gradient times input in feature space of a chosen layer
- with spatial averaging of gradients

- ⊙ the spatial averaging helps to smooth with gradient shattering
- ⊙ the relu suppresses spatial locations with approximately irrelevant contributions to the score
- ⊙ need to choose a feature map as free parameter, in principle extendible to an overlay of responses of multiple layers (why choose only a single layer?)
- (+) simple to implement
- (−) low resolution unless one does pixel-wise multiplication with guided backprop (but then why not using guided BP right away?)

Backpropagation with a heuristic to cancel out parts of the backpropagated scores:
Consider a relu activation neuron $g(z) = relu(w \cdot z + b)$:

- this receives in the forward pass the value vector $z = z(x)$ from the layers below.
- In the backward pass it received the derivative with respect to itself $\frac{\partial E}{\partial g}(z)$ from the layers above.



Guided backprop says: do backprop but **zero out incoming gradient** $\frac{\partial E}{\partial g}(z)$ at an activation layer ... when passing to through $g()$ if:

- the input to the activation of the neuron are negative $z < 0$
- the gradient arriving at this neuron is negative $\frac{\partial E}{\partial g}(z) < 0$

Why? Look only on the bright side of life

- if the input to the activation of the neuron is negative $z < 0$, then z is a suppressing neuron. rule: Ignore gradients from suppressing neurons, pass through only gradient signal from activating neurons
- if the gradient arriving at this neuron is negative. rule: ignore gradients which decrease the function value, look only at gradients which increase the prediction
- its a heuristic to look only at one end of effects: activating signals and gradients

Need to take the absolute value to get something useful. Sign has no meaning in there.

(+) gives often clean heatmaps, high resolution heatmaps

(+) very easy to implement using a backward hooks at ReLUs

(−) its a heuristic, no theoretical underpinning. Not really sure what it does.

(−) open what to do with non-relu activations

(−) high precision but lack of sensitivity to explanation for different classes

- take an input $x = (x_1, \ldots, x_D)$. Occlude a subset $S \subset \{x_1, \ldots, x_D\}$ of dimensions by some value $x \mapsto \tilde{\tilde{x}}$.
- measure $r_S(x) = f(x) - f(\tilde{\tilde{x}})$ as sensitivity of the subset S
- order subsets S according $r_S(x)$ to find the most sensitive subsets

Choice of subset scale matters! Example

- simplest case: subset consists of single dimension $S = \{x_d\}$
- using subsets of more than one element allows to capture correlations in f, example

$$f(x) = \max(|x_1|, |x_2|) + |x_3|$$

  only changing x_1 to a value $x_1 = 0$ wont change $f(x)$, thus would mask sensitivity of f to inputs in x_1, x_2.

- images: what pixel size to occlude ? 5×5 ? or superpixels from a superpixel segmentation algorithm? or extended superpixels to cover boundaries?

need to have a clear idea how to **occlude a region**

- can be simple for certain financial tasks. example: replace an input dimension value by the median, or measure for (median, 25%-,75%-quantile) and choose the largest sensitivity over quantiles
- images – can get more complicated ... black square ? from another image ? gray square?
- uncertainty? $\longrightarrow$ compute statistic e.g. median over multiple occlusions of the same region if there is no good unique occlusion
- images: can use GAN inpainting (deepfill) to generate occlusions which do not result in outliers https://openaccess.thecvf.com/content/ACCV2020/html/Agarwal_Explaining_image_classifiers_by_removing_input_features_using_generative_models_ACCV_2020_paper.html

(+) direct measurement of local sensitivity

(+) super simple to implement: create n copies of an image, do mod in each copy, run forward pass

(+) forward pass only, black-box compatible

(−) need to decide on occlusion region size and how to occlude

(−) slow, really

Starting point of many methods:

- given: A. trained model f, B. a prediction $f(x)$ for input $x = (x_1, \ldots, x_d, \ldots, x_D)$.
- general case: To compute a relevance score $r_d(x)$ for every input dimension x_d of input x explaining the prediction $f(x)$, such that approximately:

$$f(x) \approx \sum_{d=1}^{D} r_d(x) \quad \leftarrow \text{ decomposition with constraints} \tag{8}$$

- one way to achieve this, is first order Taylor decomposition around some point $x^{(0)}$

$$f(x) \approx f(x^{(0)}) + \nabla f|_{x_0} \cdot (x - x^{(0)}) = \sum_d \left.\frac{\partial f}{\partial x_d}\right|_{x_0} (x_d - x_d^{(0)}) = \sum_d r_d(x)$$

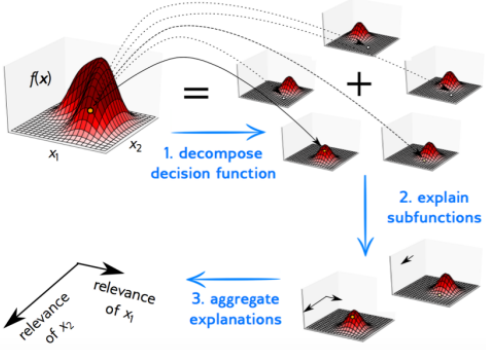
$$r_d(x) := \left.\frac{\partial f}{\partial x_d}\right|_{x_0} (x_d - x_d^{(0)})$$

- linearizes relatively to $x^{(0)}$. Explains the difference of prediction in x relative to the prediction made in $x^{(0)}$. Note: a relative explanation!
- the next method applies such an idea for every layer, and stacks decompositions

image LRP-α-β

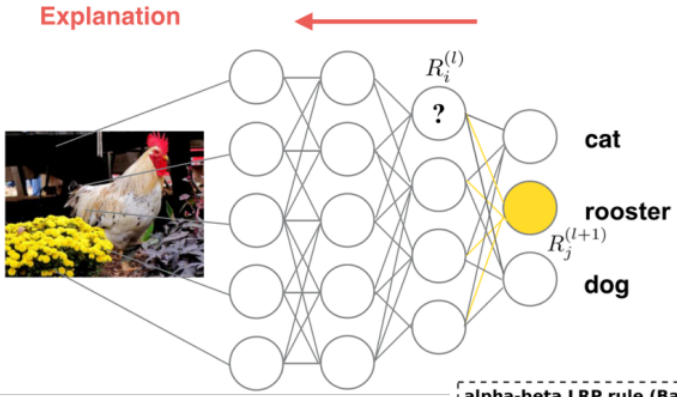


- Divide and conquer: decompose network in layers



credit: W. Samek

- Taylor approximation per layer/neuron
- easier to find roots for one layer
- robustness to gradient shattering with β-rule or adaptive β-rule



Theoretical interpretation
Deep Taylor Decomposition
(Montavon et al., 2017)

alpha-beta LRP rule (Bach et al. 2015)

$$R_i^{(l)} = \sum_j \left(\alpha \cdot \frac{(x_i \cdot w_{ij})^+}{\sum_{i'}(x_{i'} \cdot w_{i'j})^+} + \beta \cdot \frac{(x_i \cdot w_{ij})^-}{\sum_{i'}(x_{i'} \cdot w_{i'j})^-}\right) R_j^{(l+1)}$$

where $\alpha + \beta = 1$

credit: W. Samek

(−) Technically one root per layer. Need to validate choices by measuring explanation quality.

(−) using everywhere LRP-ϵ reduces to $\nabla f(x) \odot x$

(−) poor recall due to sparsity, gradient based methods often have better recall.

(+) high precision, gradient based methods often have lower precision

(+) works well for images with established presets (ignore biases, use LRP-β, γ), LSTMs, high-dimensional data

(+) does reasonably well when properly implemented, e.g.

Evaluating Explanation Methods for Deep Learning in Security,
Warnecke, Arp, Wressnegger, Rieck, *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*

- correlation to human intuition
- correlation to bounding boxes
- correlation to other ground truth? Problem with that: a neural network might learn other correlations present in your data(!!!) than your ground truth!!!
- important precision versus recall measures can differ a lot among methods
  - gradient-based: lower precision, higher recall (due to noise)
  - lrp/guided BP/deep LIFT: higher precision, lower recall (due to sparsity)
- authors opinion: there is not the one ...

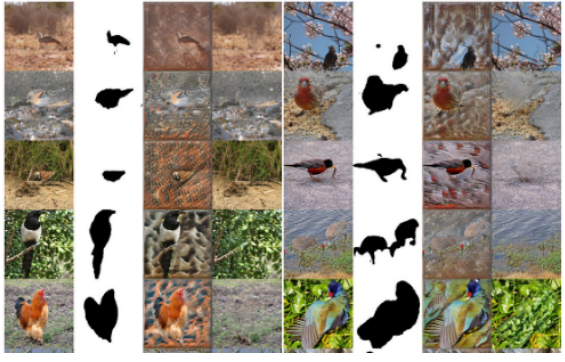
For the next slides you do not need to remember the math.
takeaway points:

- There is no single explanation quality measure. Use a multiple of them
- Explanations can be used to screen unlabeled data for selecting data to be labeled.
- Explanations can be used to find parts of models which are relevant for the prediction
- Explanations can be used to identify biases

- Measuring explanation quality is difficult. No one measure fits all.

  Explaining image classifiers by removing input features using generative models, Agarwal & Nguyen, *ACCV 2020*
  https://openaccess.thecvf.com/content/ACCV2020/papers/Agarwal_Explaining_image_classifiers_by_removing_input_features_using_generative_models_ACCV_2020_paper.pdf

(a) Real (b) Mask (c) Preserve (d) Delete (e) Real (f) Mask (g) Preserve (h) Delete

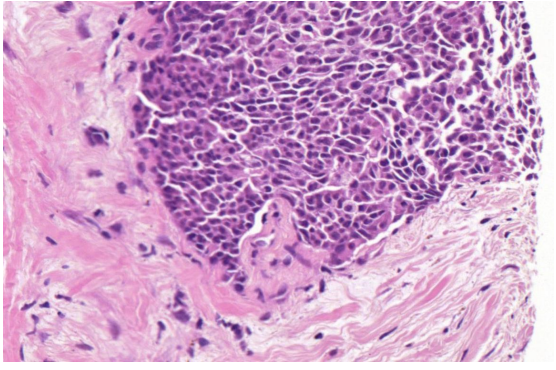


A. Iterative Dataset Design (medical imaging): Identify what you need to label for the next round

The most undervalued, but possibly least technical application:
XAI for Identifying what to label for the next round of training

Resolving challenges in deep learning-based analyses of histopathological images using explanation methods
M Hägele, P Seegerer, S Lapuschkin, M Bockmayr, W Samek, F Klauschen, KR Müller, A Binder,
Nature Scientific Reports 2020

- some problems: labels very costly, unlabeled data abundant



Approach for solution development

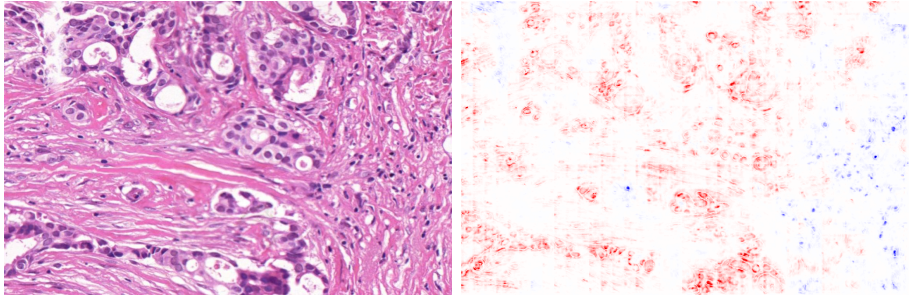
(1) *label* an initial dataset

(2) train a model

(3) identify shortcomings

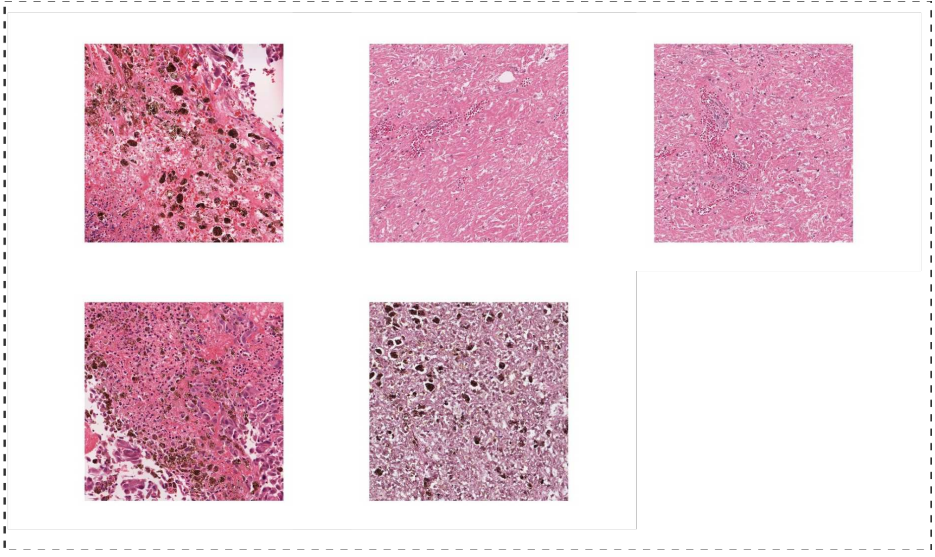
(4) *repeat cycle* by expanding the dataset, go back to step (2) and retrain

- task: decide which unlabeled data to add into next iteration of train and test set

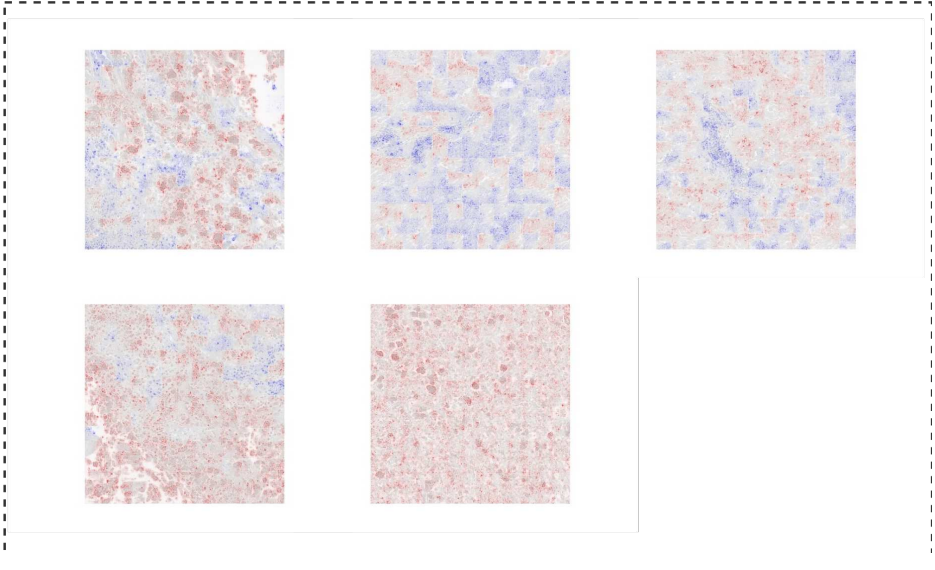


- Interpretability for efficiency in the selection step for labeling

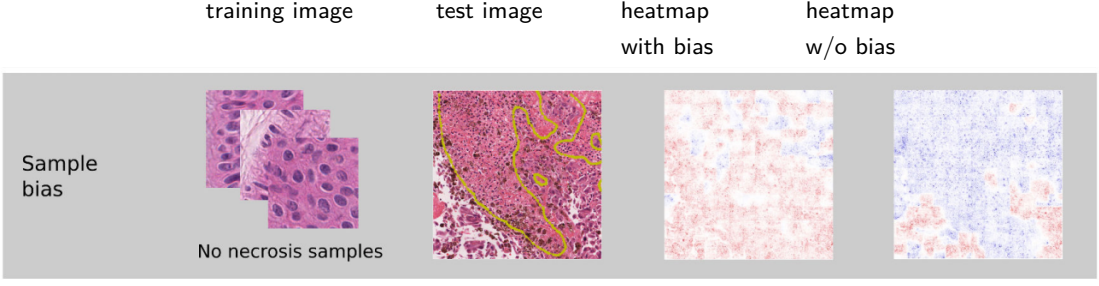
Original HE images with necrosis regions:



Training **without** labeled necrosis samples.

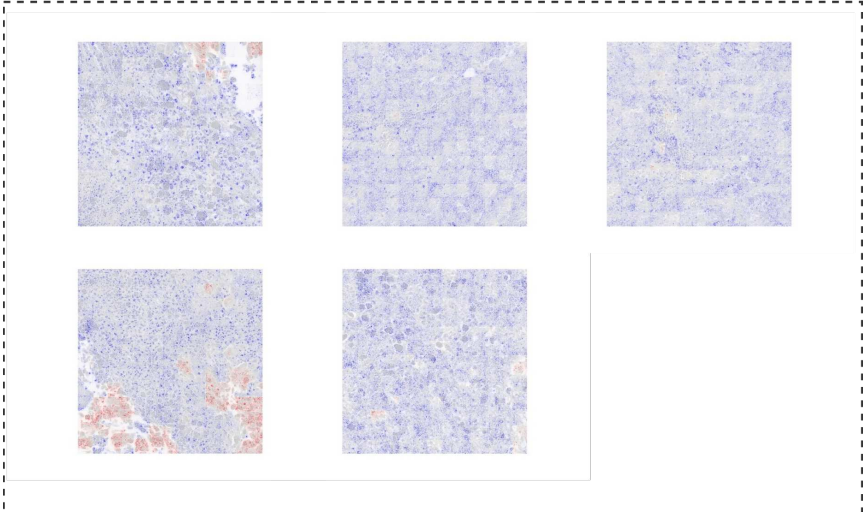


- left heatmap: false positive scores on unlabeled subclass.
- right heatmap: after augmenting training dataset with necrosis samples (labeled as negative)



- test error unable to detect bad performance on missing subclasses

Training **with** necrosis samples.



your version1 labels and test set error cannot discover it

- ⊙ Similar cases exist for rare pathological findings.
- ⊙ Requires an expert to map explanations on facts from the domain.
- (+) Works well for cases such that ground truth can be visually ascertained.

- ⊙ Requires an expert to map explanations on facts from the domain.
- (-) There are cases when this is challenging:

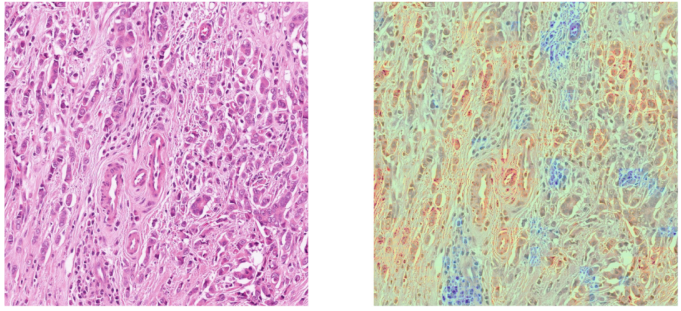
Interpretable Deep Neural Network to Predict Estrogen Receptor Status from Haematoxylin-Eosin Images

Philipp Seegerer[1], Alexander Binder[2(✉)], René Saitenmacher[1], Michael Bockmayr[3,6], Maximilian Alber[3], Philipp Jurmeister[3], Frederick Klauschen[3], and Klaus-Robert Müller[1,4,5]

[1] Machine Learning Group, Technical University Berlin, Berlin, Germany
[2] Singapore University of Technology and Design (SUTD), Singapore, Singapore
alexander_binder@stud.edu.sg
[3] Institute of Pathology, Charité University Hospital, Berlin, Germany
[4] Department of Brain and Cognitive Engineering, Korea University, Seoul, Korea
[5] Max-Planck-Institute for Informatics, Campus E1 4, Saarbrücken, Germany
[6] Department of Pediatric Hematology and Oncology, University Medical Center Hamburg-Eppendorf, Hamburg, Germany

P. Seegerer et al, Artificial Intelligence and Machine Learning for Digital Pathology: State-of-the-Art and Future Challenges, 2020

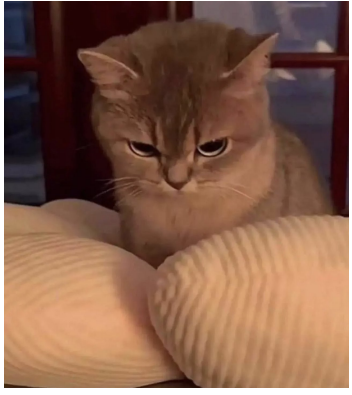
- ⊙ Requires an expert to map explanations on facts from the domain.
- (-) There are cases when this is challenging:



(a) TCGA-A2-A0CR

P. Seegerer et al, Artificial Intelligence and Machine Learning for Digital Pathology: State-of-the-Art and Future Challenges, 2020

(-) There are cases when this is challenging:



No Domain Knowledge, no use for it??

Identifying what is used in feature spaces when one cannot visually ascertain the ground truth II:

XAI for Neural net pruning[2]

[2] *Pruning by explaining: A novel criterion for deep neural network pruning*
SK Yeom, P Seegerer, S Lapuschkin, A Binder, S Wiedemann, K-R Müller, W Samek
Pattern Recognition 2021

Motivation to prune neural networks:

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Downsampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

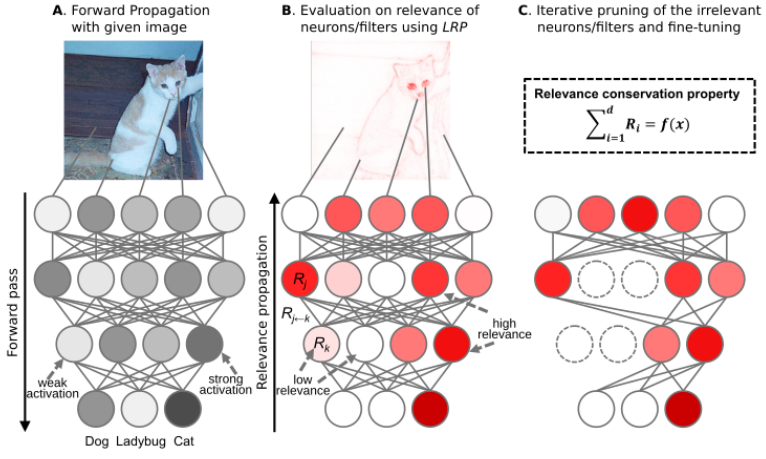
credit: He et al, https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

- Many channels. Each channel is the output of a convolutional pattern detector.
- Not all channels contribute after finetuning on a (smaller) target dataset.
- Pruning of channels for improved compute efficiency

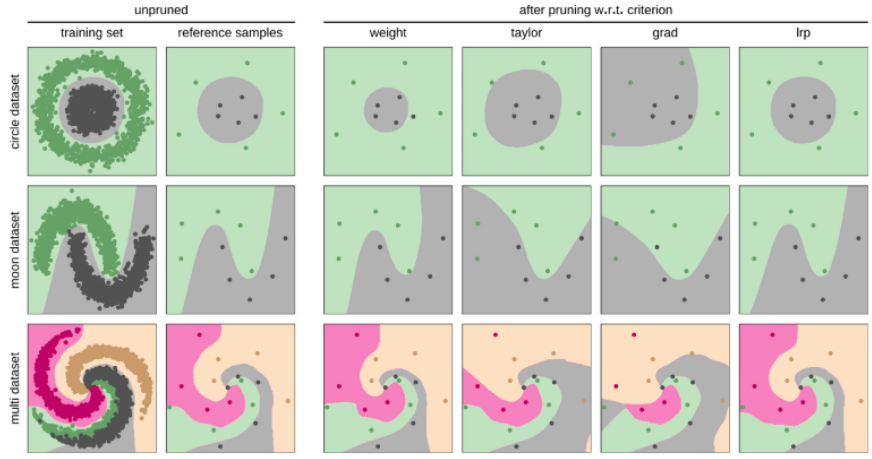


a channel from layer4.2.conv2

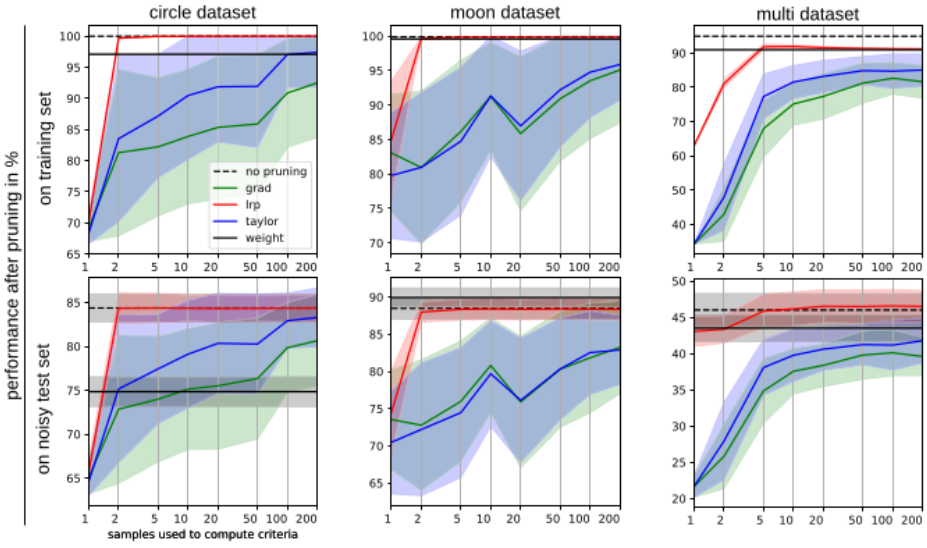
We might not want to prune this channel ! So we need a criterion.



- compute relevance scores over set of samples
- prune neurons with smallest absolute relevance scores

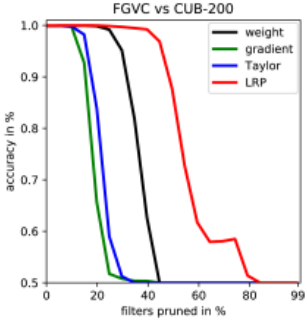


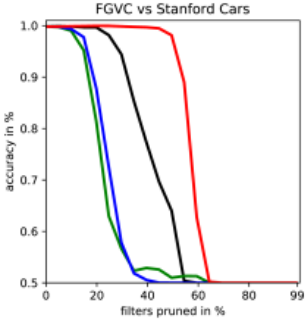
- next: on toy data first
- in order to demonstrate the raw potential of neuron selection: pruning without fine-tuning for 1000 neurons, 4 dense layers

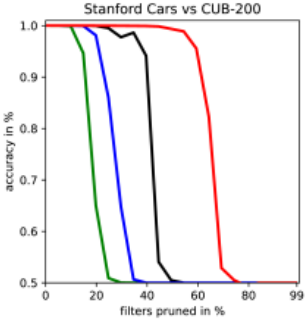


- more than 5 samples: LRP performs well

- next: on Imagenet-pretrained networks
- in order to demonstrate the raw potential of neuron selection methods: pruning without fine-tuning

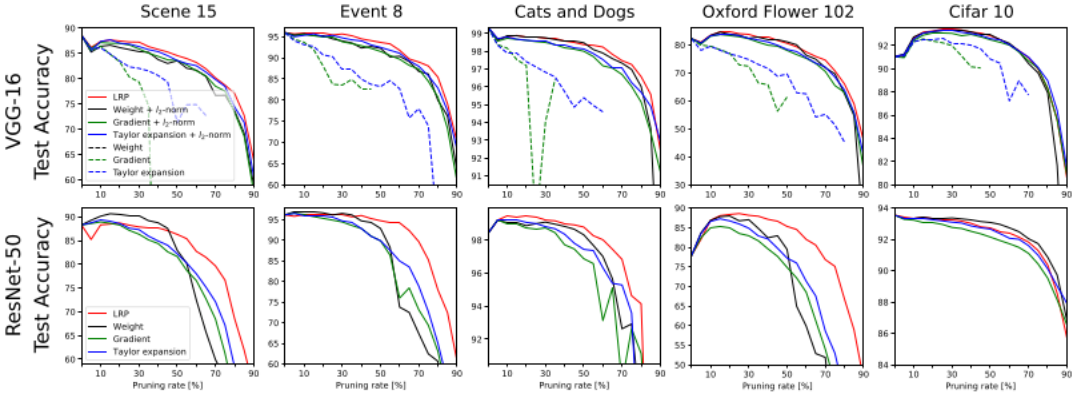




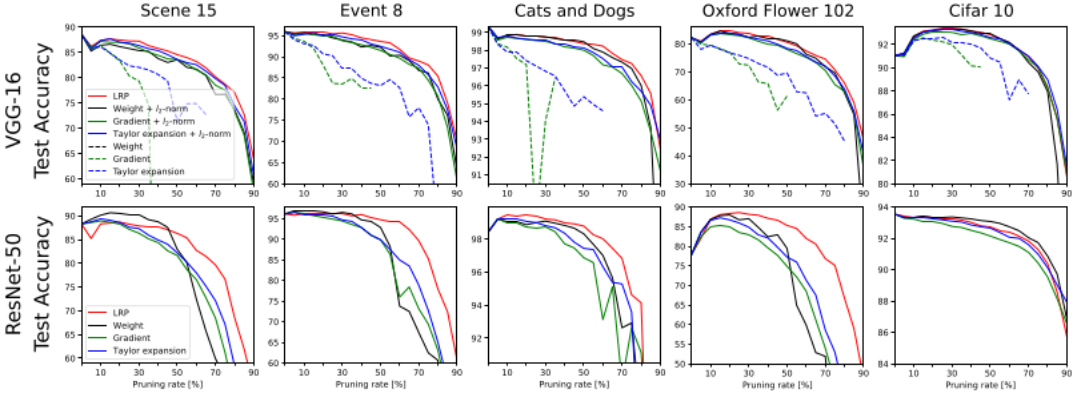


- pruning computed on 2×10 samples from target domain classes. Performed without fine-tuning retraining on the target domain (case with very small sample size available)
- Dataset is a binary classification problem created by combining two datasets taken from FGVC Aircraft, CUB-200-2011 birds and Stanford Cars. Target domain datasets have similar concepts in the Imagenet source. ResNet-50.
- different network, coder, lrp implementation. Too good to be true?

- next: on Imagenet-pretrained networks, fine-tuned on target domain
- with fine-tuning after pruning to heal damage caused by pruning methods (requires more data!)



- application case: with fine-tuning retraining on the target domain
- LRP based pruning often with better performance



- Observation: XAI is useful also in absence of domain knowledge.

Finding biases in data:
https://www.nature.com/articles/s41467-019-08987-4
See also works by S.Lapuschkin, G.Montavon, W.Samek (my collaborators :))



There is no one optimal explanation. LRP works in practice, though.

Credit: Hanabusa Itcho