Explaining Decisions of Neural Networks with Layer-wise Relevance Propagation

Talk for Deep Learning Class@SUTD, 2019.

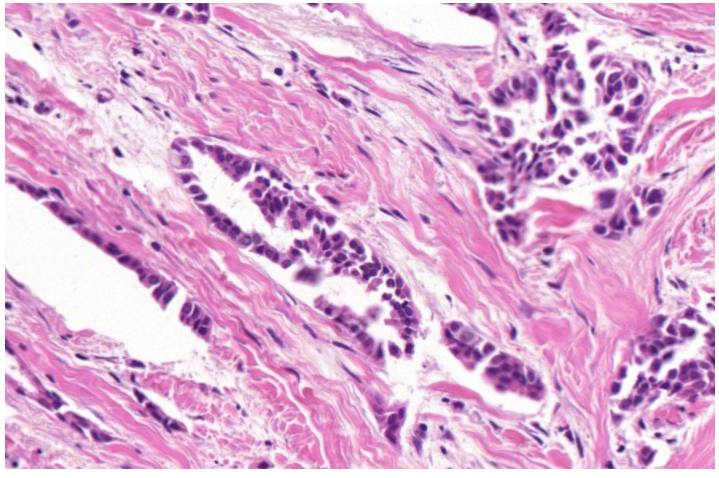
Alexander Binder
Joint work with G. Montavon, S. Lapuschkin (Bach), K.-R. Müller, W. Samek, P. Chong, Y. Elovici

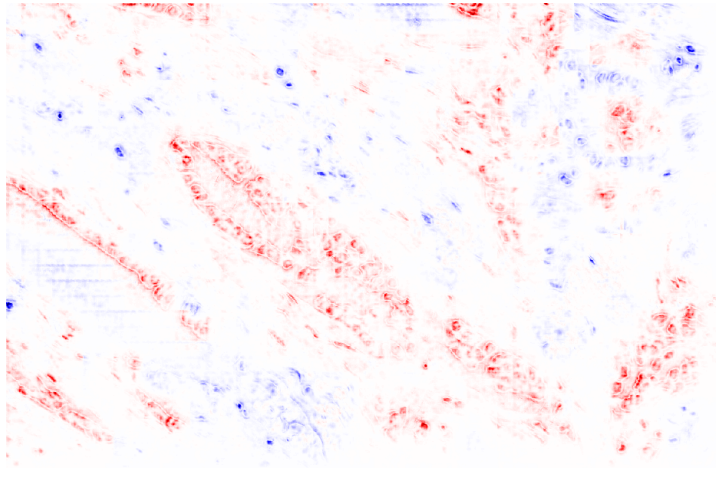
ISTD Pillar, Singapore University of Technology and Design (SUTD)

March 30, 2019

What can an explanation do?

For an image: compute a score for every pixel

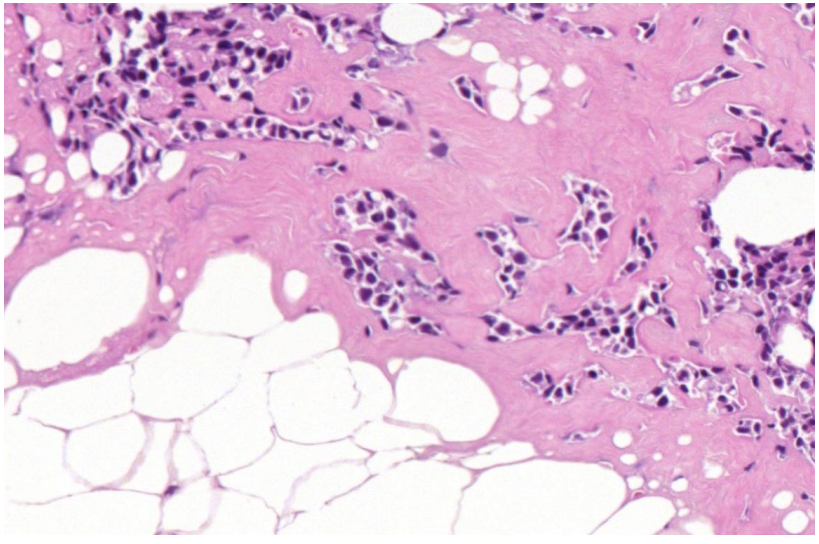


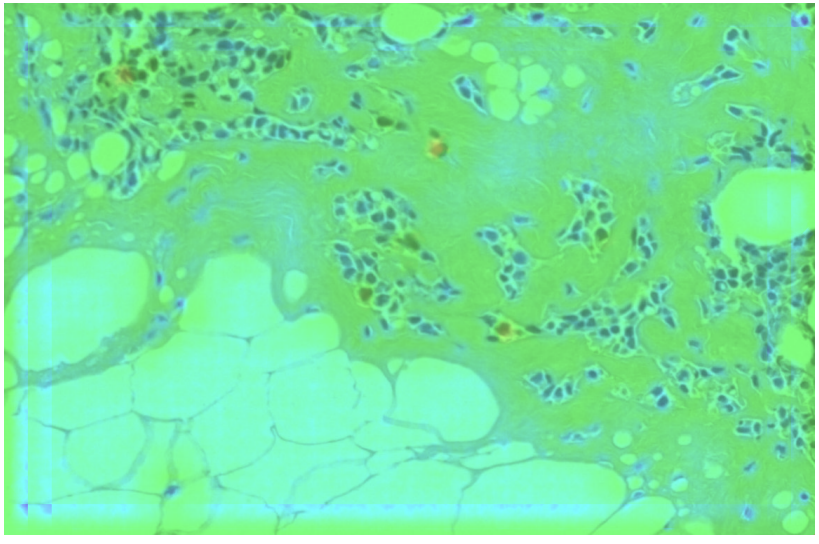


The value of explanations

- Design/Prototyping/Research Phase:
  A. Identify Fail Cases $\longrightarrow$ Iterative Dataset Design
  B. evaluate Impact of data augmentation on unlabeled datasets
  C. Identify Biases in Train+Test data
- Application Phase:
  - Provide confidence in either machine-made or human-made prediction in case of disagreement

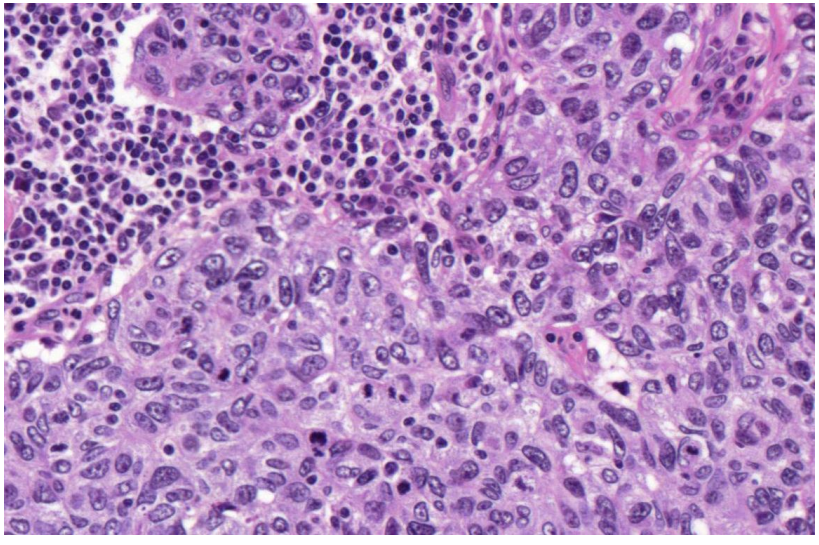
A. (Explanation beyond mere curiousity:) Identify Fail Cases

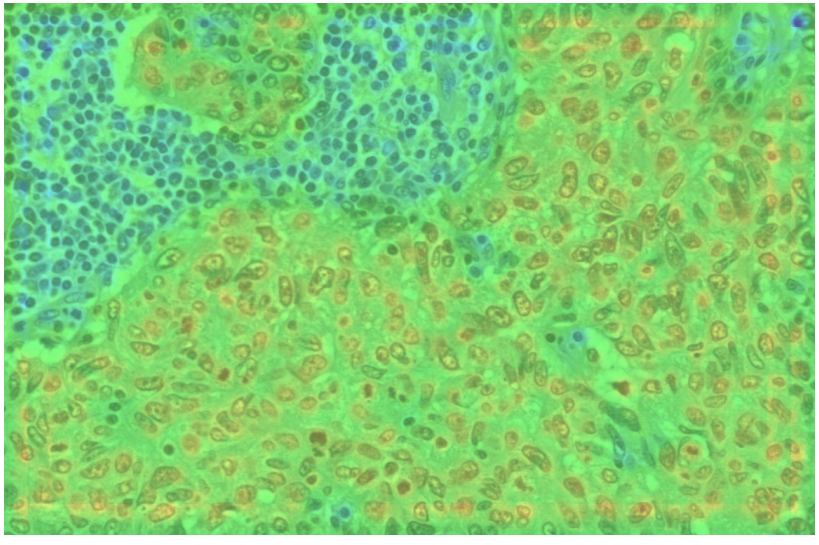




Too similar to lymphocytes (see next slide)

A. (Explanation beyond mere curiousity:) Identify Fail Cases



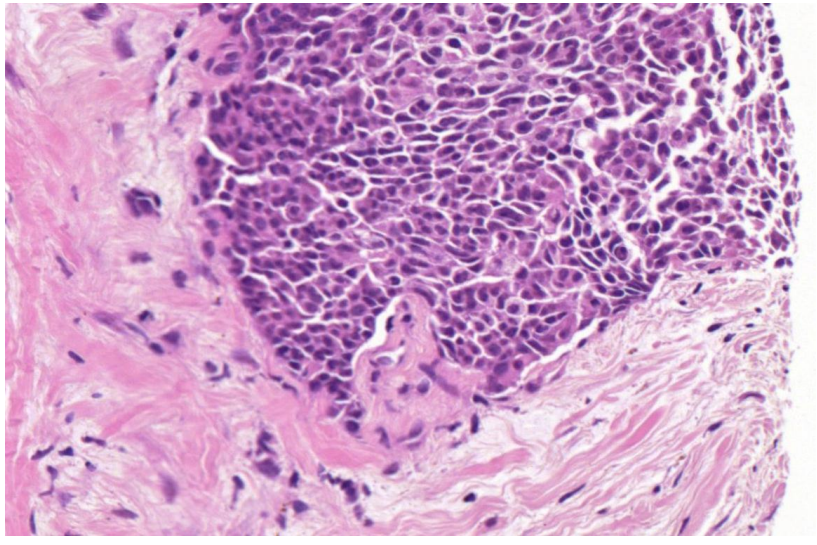


see the TiLs here

A. (Explanation beyond mere curiousity:) Identify Fail Cases

Why not just using test error ?

1. some problems: labels very costly, unlabeled data abundant



A. (Explanation beyond mere curiousity:) Identify Fail Cases

Why not just using test error ?

1. some problems: labels very costly, unlabeled data abundant
2. if some fail case not in labeled testset, test error unable to detect it.
   - some medical problems have high variability within a class, one initial dataset will miss some subclasses
3. if "subclass" was not a priori identified and labeled as such, test error cannot show consistent failure on it.

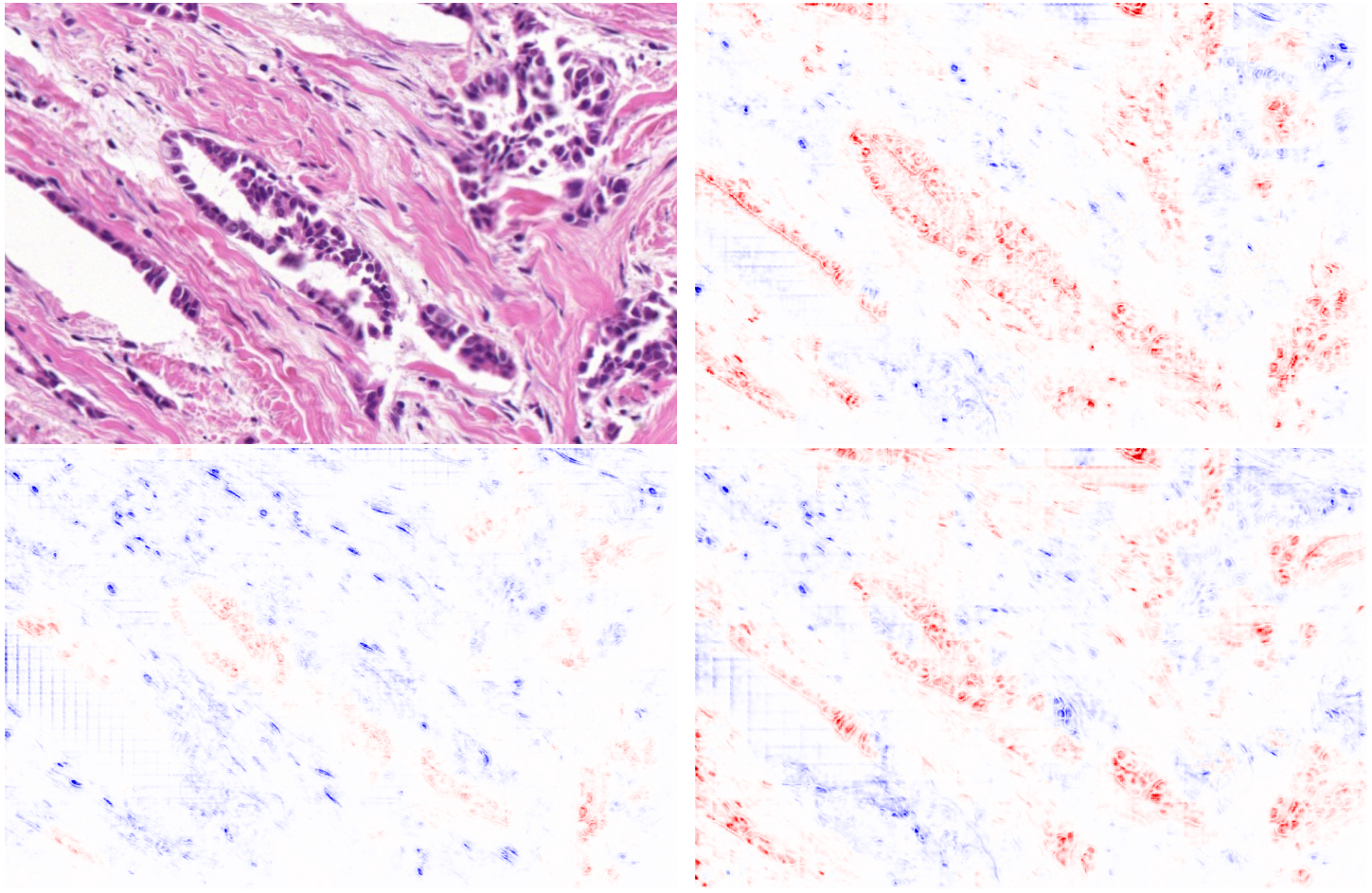
$\Longrightarrow$ Tool for iterative Dataset design – identify relevant subclasses!

B. Evaluate Impact of data augmentation

Impact of Image scaling

orig

80%



100%

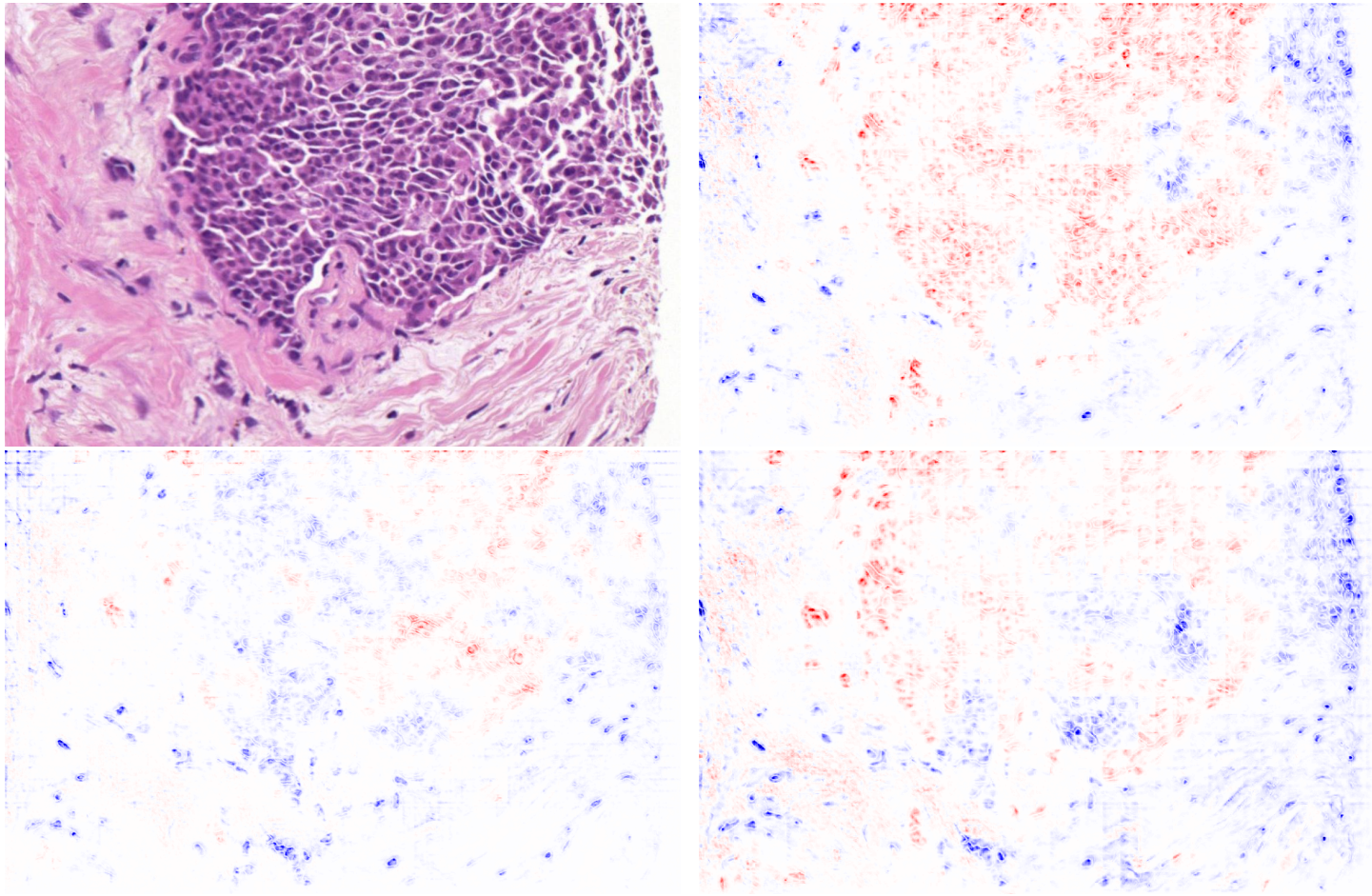
66%

B. Evaluate Impact of data augmentation

Impact of Image scaling

orig

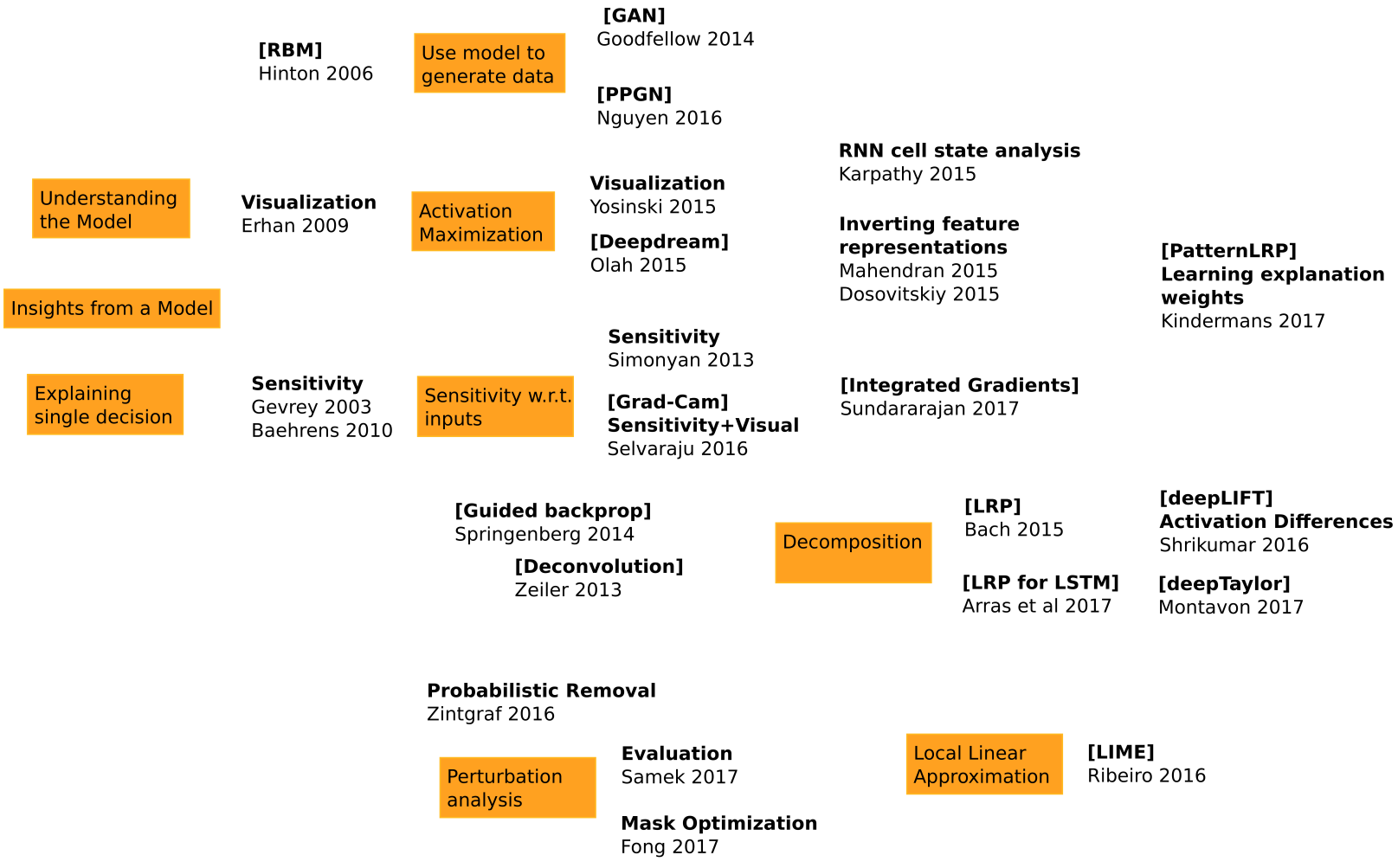
80%



100%

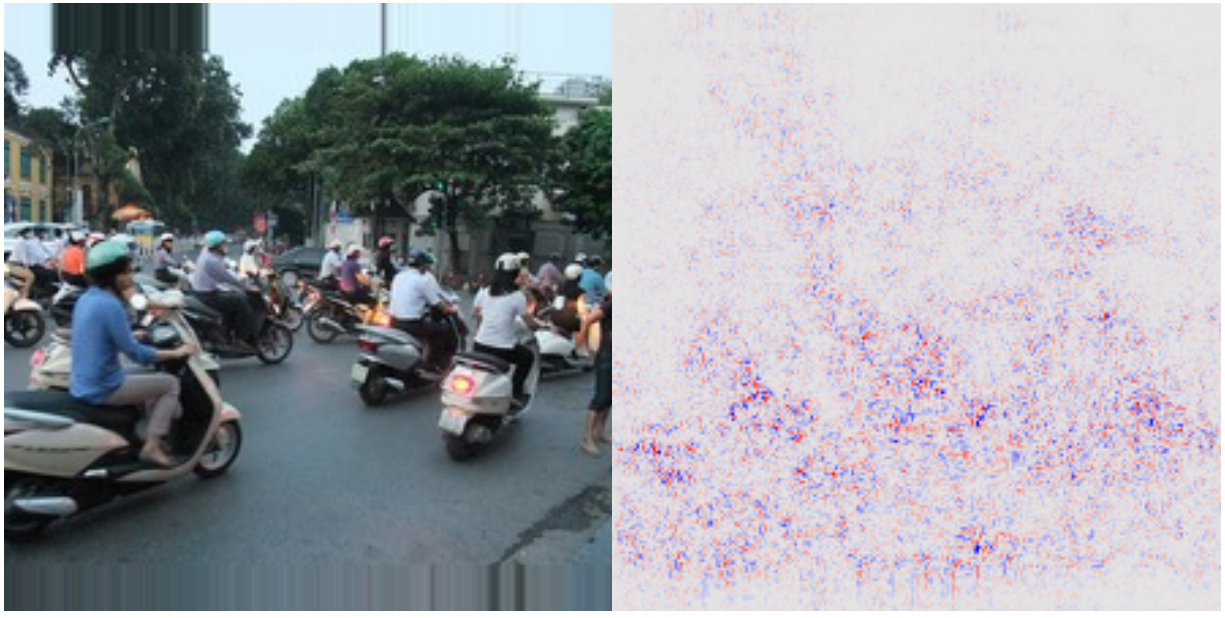
66%

Types of Explanations



Norms of Gradients

- PRO: easy to compute + understand
- CON: poor performance in off-the-data-manifold samples



Almost everything changes the prediction a lot.

Explanation by Decomposition

- Given: a prediction $f(x)$ over an input sample $x = (x_1, \ldots, x_d, \ldots, x_D)$,
- Goal: compute for every dimension d a relevance score $r_d(x)$ for its input x_d such that approximately:

$$f(x) = \sum_{d=1}^{D} r_d(x) \tag{1}$$

A motivational example I

A linear classifier

$$f(x) = \sum_{d=1}^{D} w_d x_d \tag{2}$$

$$f(x) = \sum_{d=1}^{D} r_d(x) \tag{3}$$

$$\Rightarrow \quad r_d(x) = ??? \tag{4}$$

A motivational example I

A linear classifier

$$f(x) = \sum_{d=1}^{D} w_d x_d \tag{2}$$

$$f(x) = \sum_{d=1}^{D} r_d(x) \tag{3}$$

$$\Rightarrow \quad r_d(x) = w_d x_d \tag{4}$$

Trivial rules

Given $f(x)$:

$$f(\mathbf{x}) = \sum_{d=1}^{D} r_d(\mathbf{x}) \tag{5}$$

$$r_d(\mathbf{x}) = f(\mathbf{x})/D \tag{6}$$

$$r_d(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & d = 1 \\ 0 & \text{else} \end{cases} \tag{7}$$

- non-plausible decompositions
- need additional constraints

Trivial rules

Given $f(x)$:

$$f(\mathbf{x}) = \sum_{d=1}^{D} r_d(\mathbf{x}) \tag{5}$$

$$r_d(\mathbf{x}) = f(\mathbf{x})/D \tag{6}$$

$$r_d(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & d = 1 \\ 0 & \text{else} \end{cases} \tag{7}$$

- non-plausible decompositions
- need additional constraints

Principled explanation - Taylor

- a decomposition $f(x) \approx \sum_d r_d(x)$ can be achieved by Taylor decomposition

$$f(\mathbf{x}) \approx f(\tilde{\mathbf{x}}) + (\nabla f)_{\tilde{\mathbf{x}}} \cdot (\mathbf{x} - \tilde{\mathbf{x}}) \tag{8}$$

- if $f(\tilde{\mathbf{x}}) = 0$,then

$$f(\mathbf{x}) \approx 0 + (\nabla f)_{\tilde{\mathbf{x}}} \cdot (\mathbf{x} - \tilde{\mathbf{x}}) \tag{9}$$

$$= \sum_d \frac{\partial f}{\partial x_d} (\mathbf{x} - \tilde{\mathbf{x}})_d = \sum_d r_d(x) \tag{10}$$

- one parameter of particular importance, the anchor point $\tilde{x}$

Principled explanation - Taylor

- a decomposition $f(x) \approx \sum_d r_d(x)$ can be achieved by Taylor decomposition

$$f(\mathbf{x}) \approx f(\tilde{\mathbf{x}}) + (\nabla f)_{\tilde{\mathbf{x}}} \cdot (\mathbf{x} - \tilde{\mathbf{x}}) \quad (11)$$

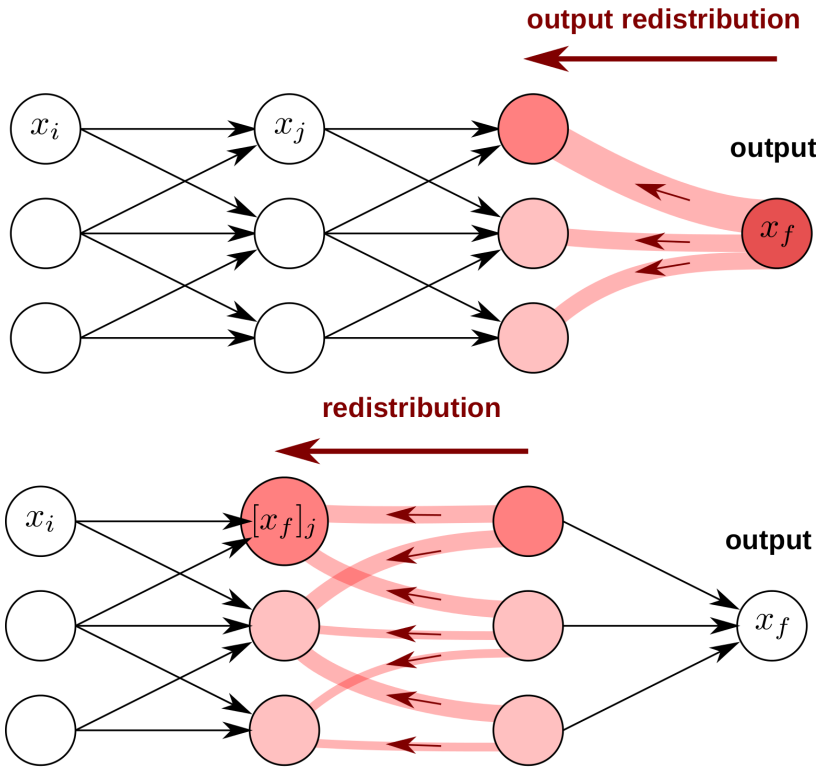
- Choice of anchor point $\tilde{x}$ implies **explanation relative to anchor point.** Choose which one?
- Idea: for classification: choose anchor point to be 1. a root, 2. which "is close".
- roots for classification problems: points of maximal uncertainty, where prediction labels are switching. Explain what makes prediction switch – Warning: classification specific assumption, can be modified.
- Can be extended to work with an average of Taylor decompositions over multiple anchor points.

Advertisement Warning

The next slides show authors own research. Views might be positively biased ;) . Nope I have not solved all interpretability problems with it.

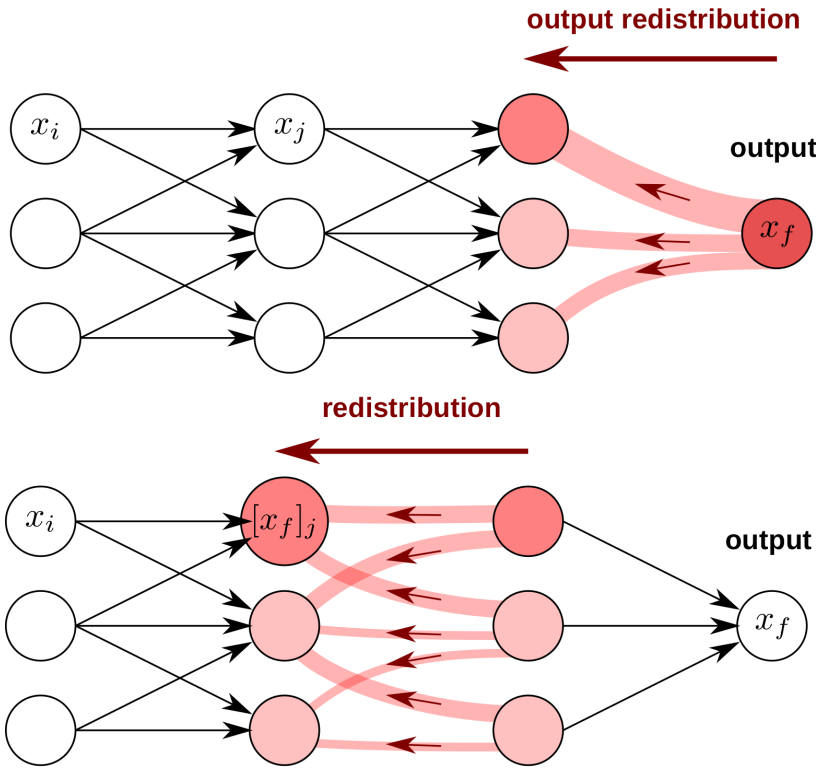
The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



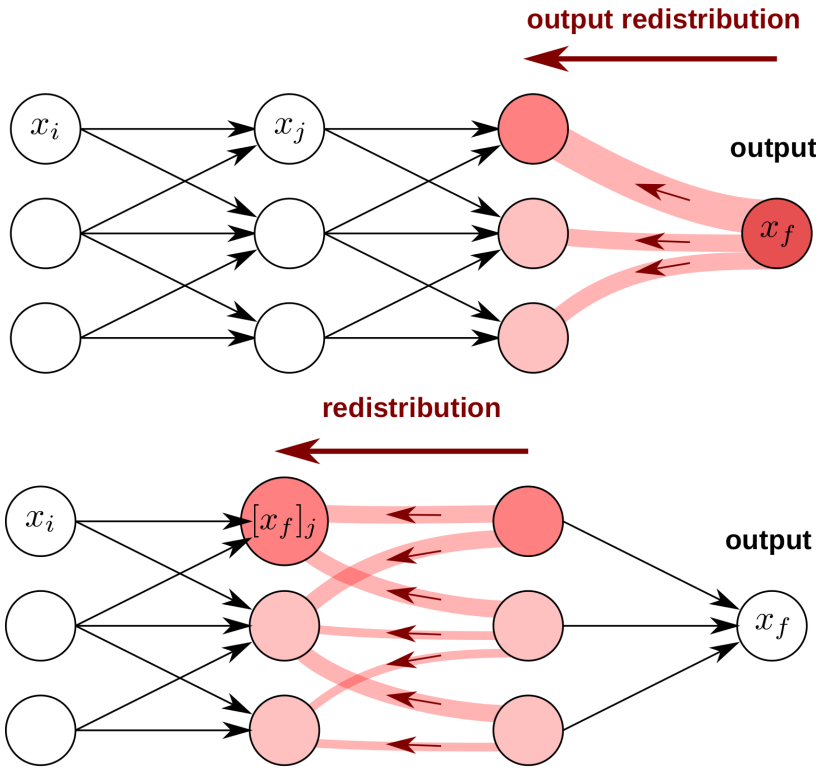
The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



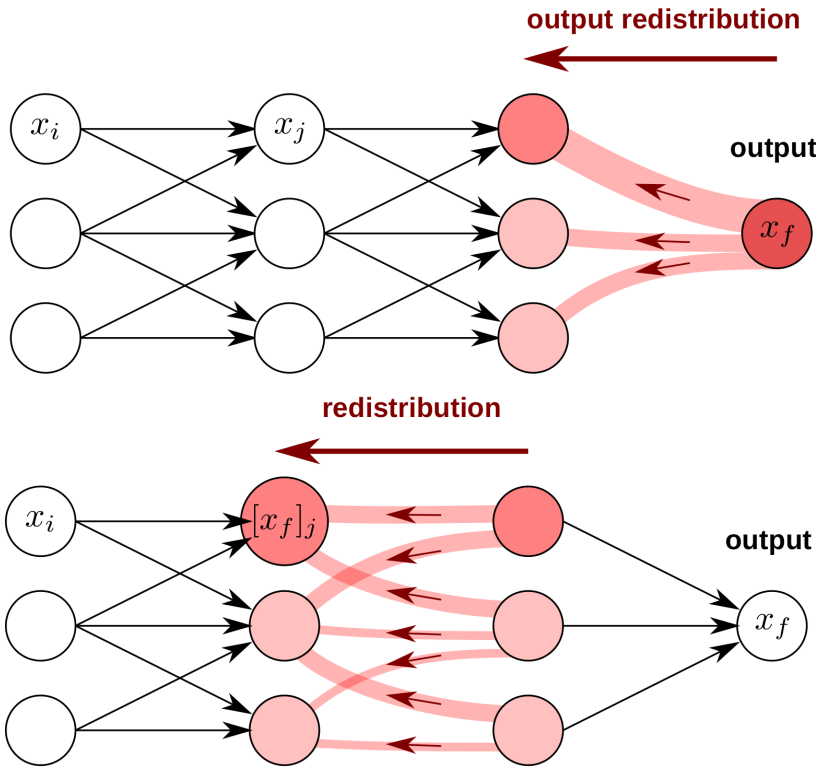
The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



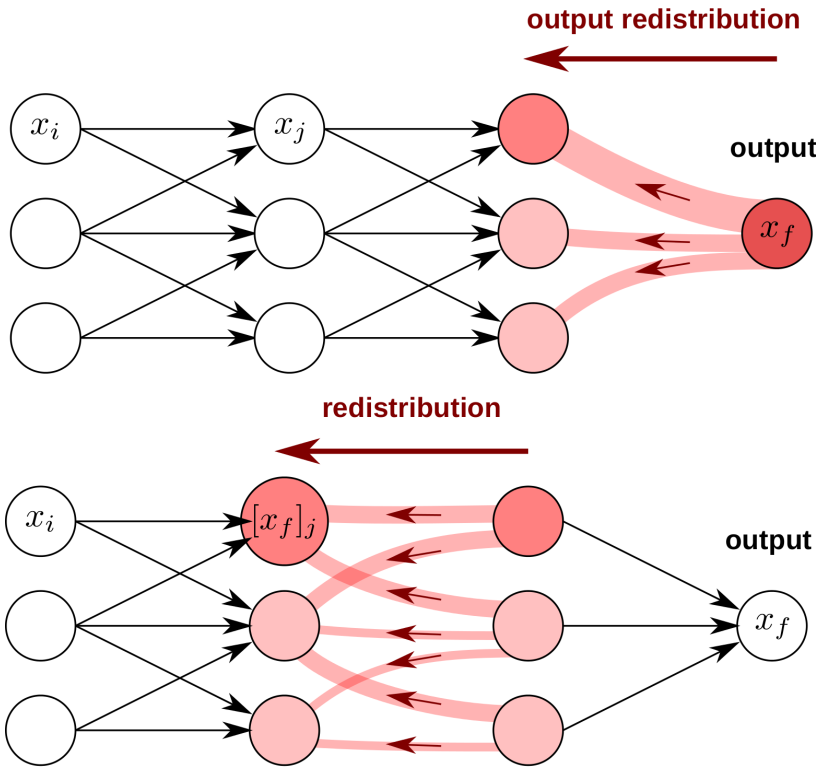
The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



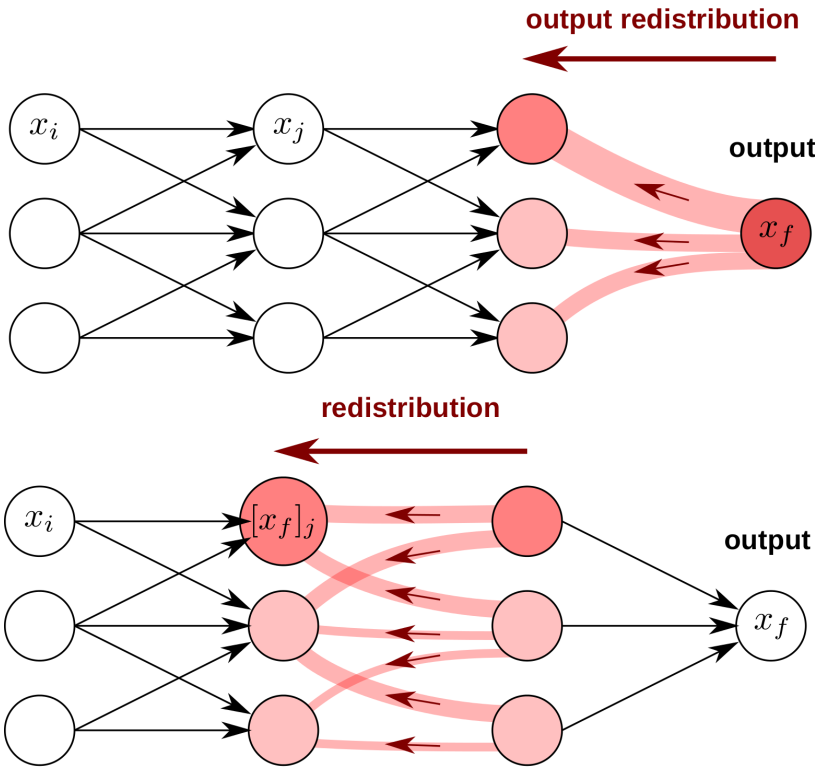
The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



The intuition behind it - I: Relevance Flow

- each neuron has a certain relevance
- redistribute relevance top-down: from neuron to its inputs
- relevance should be conserved when distributed (for comparability between relevances)
- think of a neuron:
  - inputs can stimulate it to fire – positive relevance!
  - inputs can inhibit it – ???
  - negative relevance ? close to zero relevance?



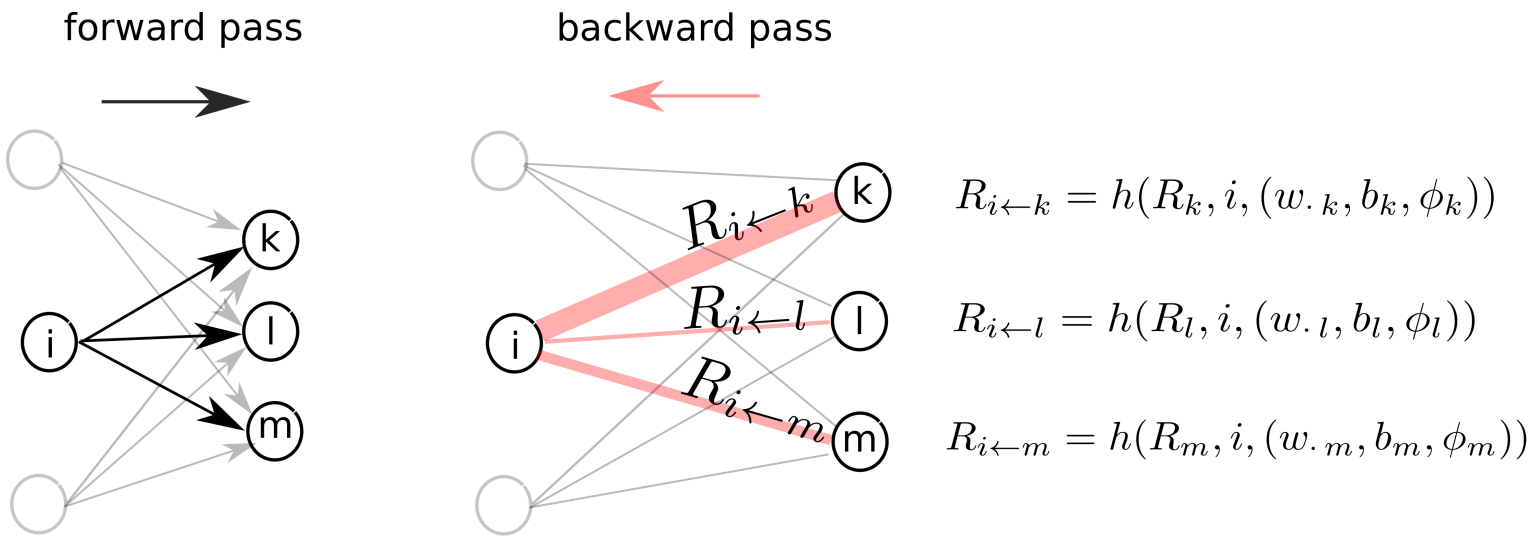
LRP as backward message passing

exist non-plausible decompositions!

(A) a decomposition of the real-valued prediction function into a sum of relevances as in $f(\mathbf{x}) = \sum_{d=1}^{D} r_d(\mathbf{x})$

(B) be able to recover as special case the result of interpretability for linear mappings $r_d(x) = w_d x_d$.

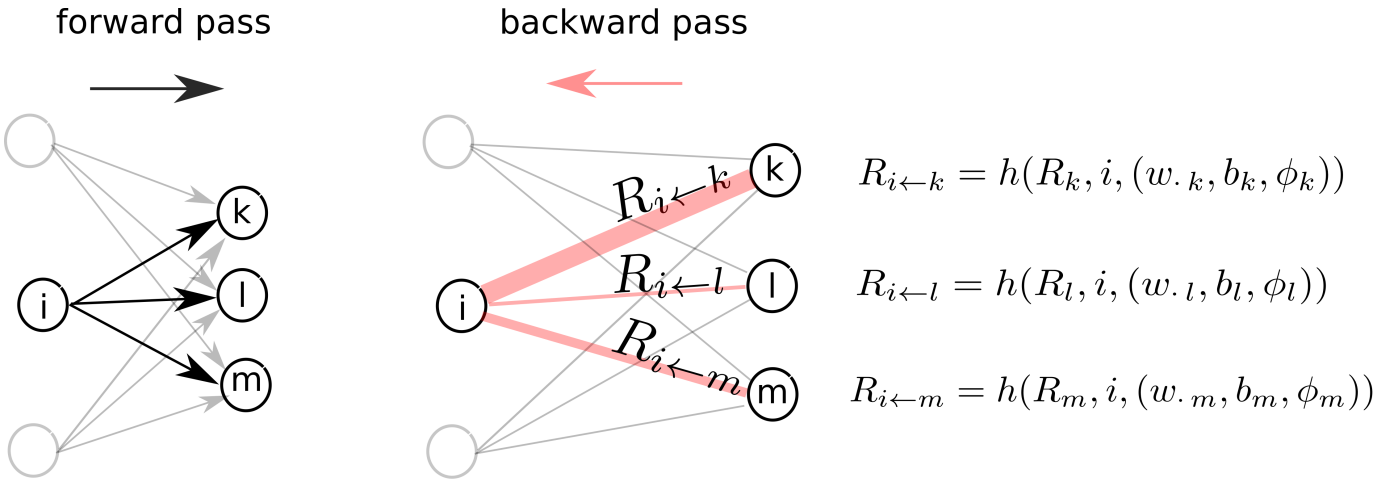
(C) The relevance is redistributed in a backward pass from neuron k to neuron i by a relevance message $R_{i \leftarrow k}$



$$R_i = R_{i\leftarrow k} + R_{i\leftarrow l} + R_{i\leftarrow m}$$

$$R_i := \sum_{k:\, i \text{ is input to } k} R_{i\leftarrow k}$$

LRP as backward message passing



$$R_i = R_{i\leftarrow k} + R_{i\leftarrow l} + R_{i\leftarrow m}$$

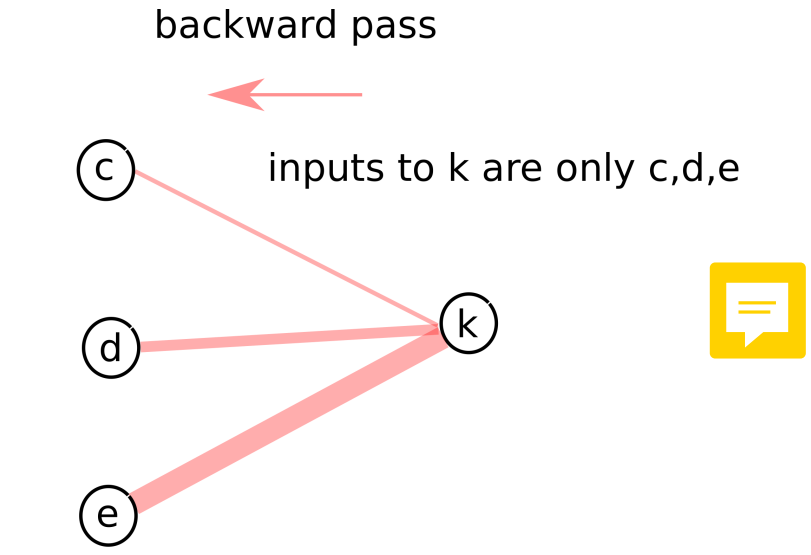
$$R_i := \sum_{k:\, i \text{ is input to } k} R_{i\leftarrow k}$$

(C) The relevance R_k of a neuron k is redistributed onto its inputs x_i by relevance messages $R_{i\leftarrow k}$. These messages are a function of R_k.

(D) The relevance of a neuron i is computed the sum of the relevance messages $R_{i\leftarrow j}$ which it has received from other neurons j, to which this neuron i contributes as input.

$R_i := \sum_{k:\, i \text{ is input to } k} R_{i\leftarrow k}$

LRP as backward message passing



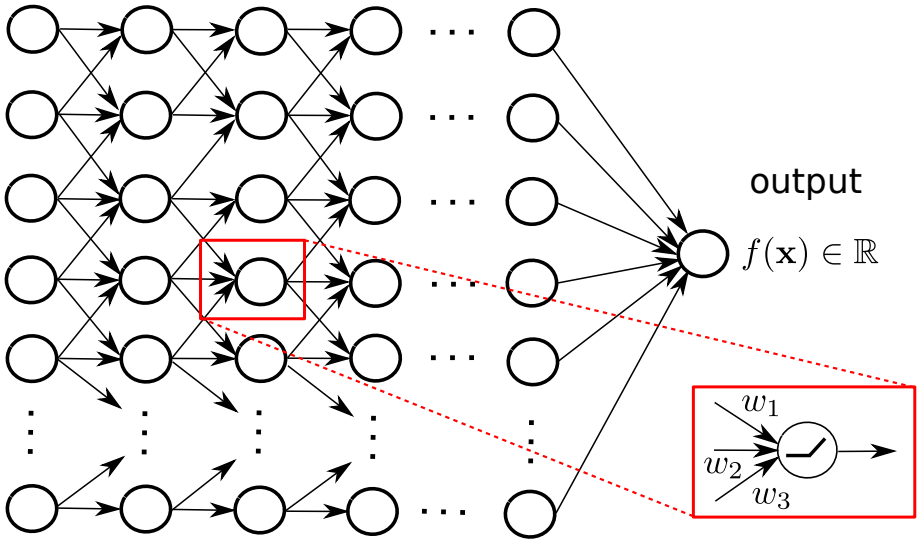
$$R_{c\leftarrow k} + R_{d\leftarrow k} + R_{e\leftarrow k} \approx R_k$$

$$\sum_{i:\, i \text{ is input to } k} R_{i\leftarrow k} \approx R_k$$

(E) This will be used to normalize redistribution rules. The difference between equations in (D) and (E) is the index which is summed over, i versus k. Requirement (E) has the purpose to ensure comparability of computed relevances, so that they are suitable to be ranked against each other.

$\sum_{k:\, i \text{ is input to } k} R_{i\leftarrow k} \approx R_k$

LRP for a whole layered neural network



- initialize relevance of output $R_{out} = f(x)$
- Backpropagate decomposition (start at output) - Retain structure of Chain Rule (but different messages!)
- decompose each neuron separately, send messages $R_{i \leftarrow k}$ backwards

need to define $R_{i \leftarrow k}$!

next steps

defined how LRP is used on a network as a whole: had LRP as backward message passing

now focus on defining the messages $R_{i \leftarrow k}$ for a single neuron

The intuition behind it - II

given one neuron k with relevance R_k:

$$x_k = g(\sum_i w_{ik} x_i + b_k) \Rightarrow R_k \approx \sum_i R_{i \leftarrow k} \tag{12}$$

- A larger input gets a larger share of the relevance:

- Assumption: A zero input does not do anything.
- idea: explain what makes a difference compared to the case of the input being zero: $g(0) = 0$.
- technically: $R_{i \leftarrow k} = R_k h(\{w_{\cdot k}, b_k, \phi\}, x)$, h is a non-decreasing function of $|w_{ik} x_i|$

The intuition behind it - II

given one neuron k with relevance R_k:

$$x_k = g(\sum_i w_{ik} x_i + b_k) \Rightarrow R_k \approx \sum_i R_{i \leftarrow k} \tag{12}$$

- A larger input gets a larger share of the relevance:
- Assumption: A zero input does not do anything.
- idea: explain what makes a difference compared to the case of the input being zero: $g(0) = 0$.
- technically: $R_{i \leftarrow k} = R_k h(\{w_{\cdot k}, b_k, \phi\}, x)$, h is a non-decreasing function of $|w_{ik} x_i|$

The intuition behind it - IIIa

given one neuron k with relevance R_k:

$$x_k = g(\sum_i w_{ik} x_i + b_k) \Rightarrow R_k \approx \sum_i R_{i \leftarrow k} \tag{13}$$

$R_{i \leftarrow k} = R_k h(\{w_{\cdot k}, b_k, \phi\}, x)$, h is a non-decreasing function of $|w_{ik} x_i|$
two basic ideas:

$$(A)\ R_{i \leftarrow k} \propto R_k\, w_{ik} x_i \tag{14}$$

$$(B)\ R_{i \leftarrow k} \propto R_k\, \alpha(w_{ik} x_i)_+ - R_k\, \beta(w_{ik} x_i)_- \tag{15}$$

$$z_+ = \max(0, z),\ \ z_- = \min(0, z) \tag{16}$$

$$r_i(x) = \sum_{k:\, x_i \text{ input to } x_k} r_{i \leftarrow k} \tag{17}$$

The intuition behind it - IIIb

A single neuron with relevance $\hat{R}(y)$ for the output y. Goal: assign relevance $r_d(x)$ to input x_d of the d-th dimension.

$$y = g(\sum_d w_d x_d + b) \tag{18}$$

two basic ideas:

$$(A)\ r_{d\leftarrow y} \propto \hat{R}(y)\, w_d x_d \tag{19}$$

$$(B)\ r_{d\leftarrow y} \propto \hat{R}(y)\, \alpha(w_d x_d)_+ - \hat{R}(y)\, \beta(w_d x_d)_- \tag{20}$$

$$z_+ = \max(0, z),\ \ z_- = \min(0, z) \tag{21}$$

$$r_d(x) = \sum_{y:\, x_d \text{ input to } y} r_{d\leftarrow y} \tag{22}$$

both ideas:

The intuition behind it - V

layer-wise relevance propagation as decomposition of a function $f(\mathbf{x})$ subject to constraints

$$f(\mathbf{x}) = g(\sum_d w_d x_d + b) \approx \sum_{d=1}^{D} r_d(\mathbf{x}) \tag{23}$$

from (B) β-rule:

$$r_{d \leftarrow y}(\mathbf{x}) = R_{out} \left((1+\beta) \frac{(w_d x_d)_+}{\sum_{d'} (w_{d'} x_{d'})_+ + b_+} - \beta \frac{(w_d x_d)_-}{\sum_{d'} (w_{d'} x_{d'})_- + b_-} \right) \tag{24}$$

- β – controls ratio of negative to positive evidence.
- $\beta = 0$ only positive evidence (analogous to e.g. guided backprop)
- suitable for conv layers (with modifications: batchnorm layers)

The intuition behind it - VI

layer-wise relevance propagation as decomposition of a function $f(\mathbf{x})$ subject to constraints

$$f(\mathbf{x}) = g(\sum_{d} w_d x_d + b) \approx \sum_{d=1}^{D} r_d(\mathbf{x}) \tag{25}$$

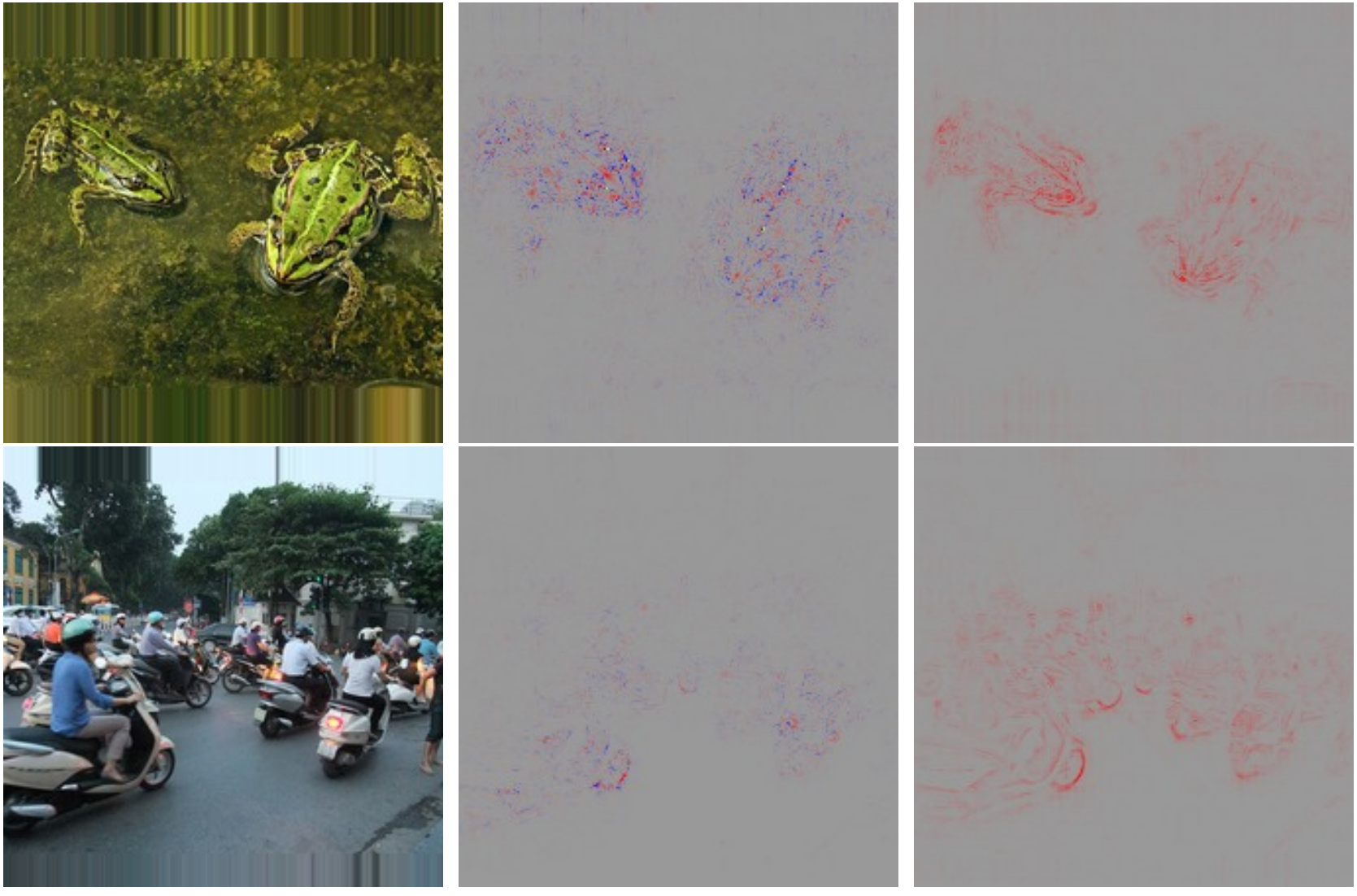
from (A) ϵ-rule:

$$r_{d \leftarrow y}(\mathbf{x}) = R_{out} \frac{w_d x_d}{\sum_{d'} w_{d'} x_{d'} + b + \epsilon} \tag{26}$$

- ϵ – dampening factor, numerical stabilization
- only suitable for fully connected layers

Examples (GoogleNet v1, caffe)

Examples (GoogleNet v1, caffe)



LRP from the Taylor perspective

LRP is related to Taylor decomposition
Explaining NonLinear Classification Decisions with Deep Taylor Decomposition, Montavon et al., Pattern Recognition 2017

- applies Taylor decomposition directly to the relevance function $R(x)$ for one neuron with inputs x, and backpropagate
- basic assumption:

$$r_d(x) = c_d x_d \tag{27}$$

  where c_d is locally a constant – locally linear model.
- different choices of roots yield different LRP-Rules (LRP helped to find these *exotic* roots ...)

von, 2017 yields a set of more general LRP-rules besides the above rules.

- see appendix for quick rule derivations

Treating oversparsity?

LRP Heatmaps tend to be very sparse. Sparsity can be adjusted with a simple trick, using compositionality of LRP scores.

Controlling Explanatory Heatmap Resolution and Semantics via Decomposition Depth, Bach et al., ICIP 2016

Ideas for Evaluation

Nice pictures, but is the decomposition meaningful at all ??

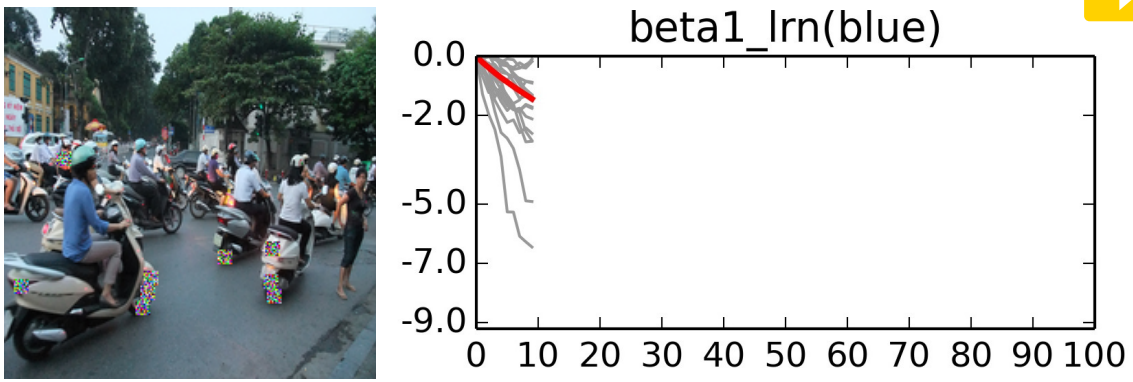
- Idea: use relevance scores r to order regions.
- *modify* regions in the implied order
- measure decrease of prediction $f(\mathbf{x})$ under modifications
- **key idea:** if most important pixels get the highest score, then prediction will decrease fast, as more and more pixels get modified

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



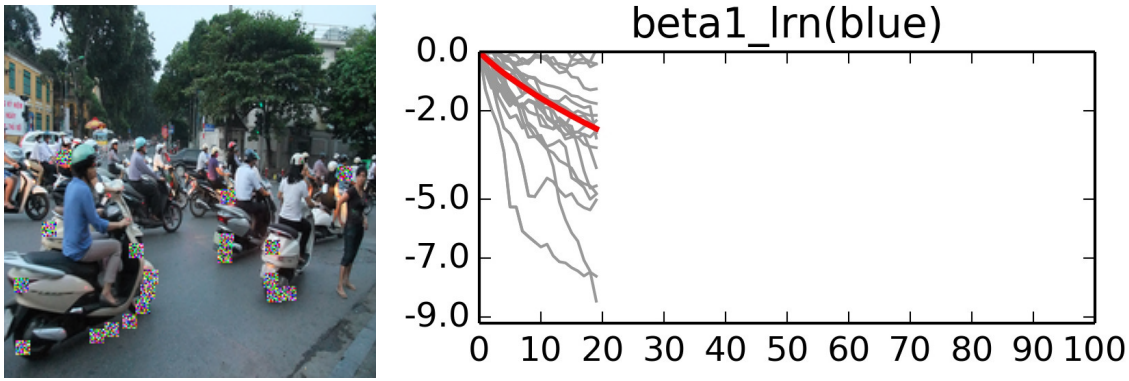
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



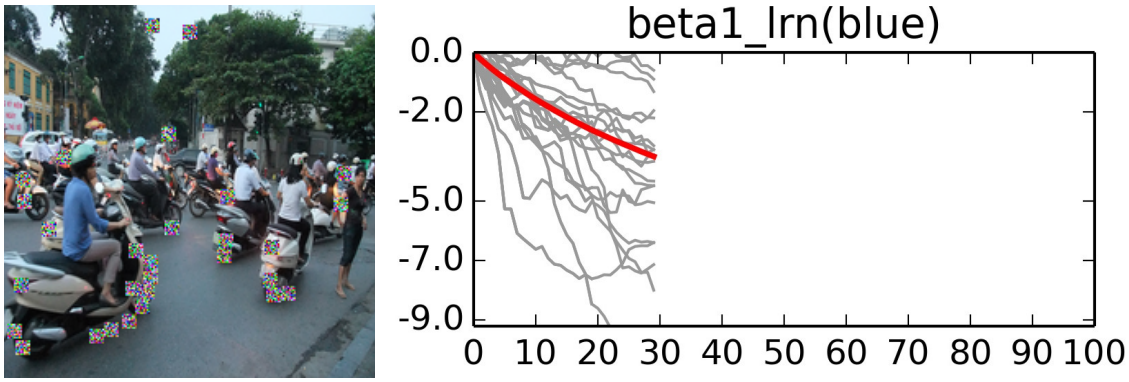
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



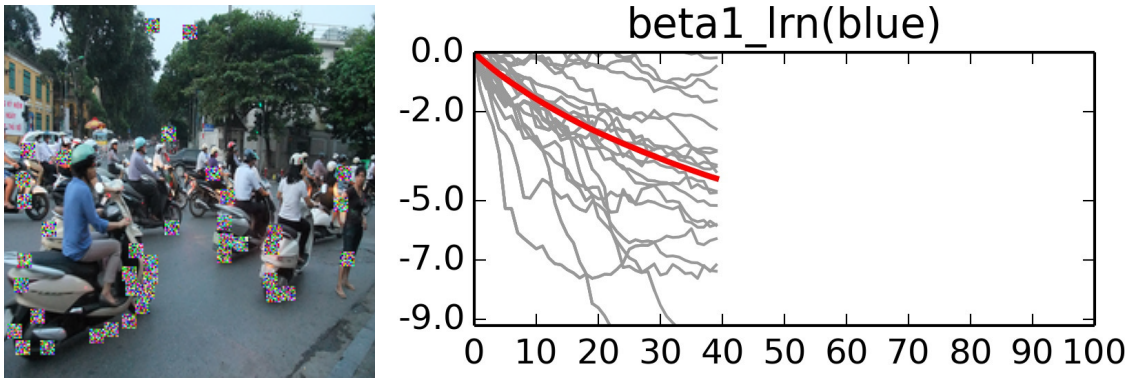
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



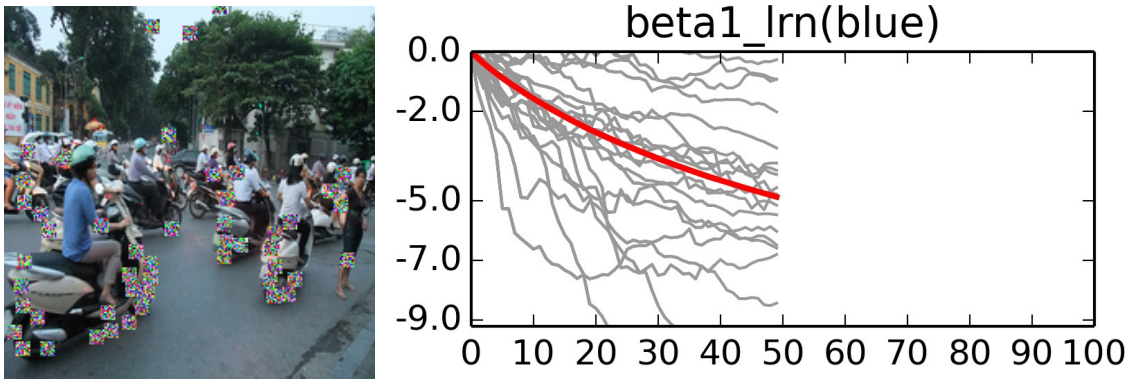
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



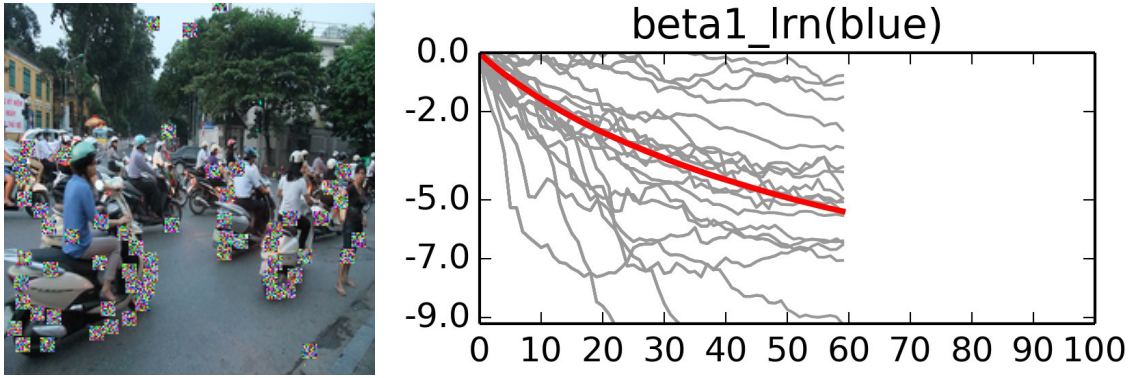
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



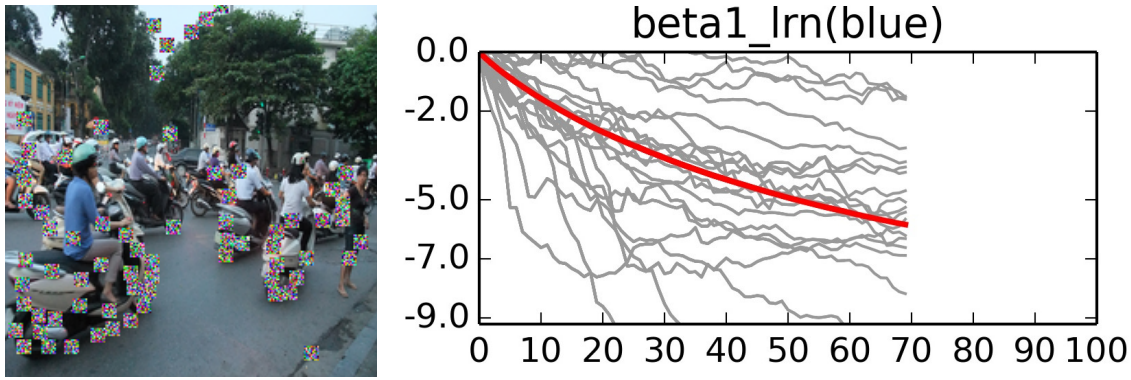
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



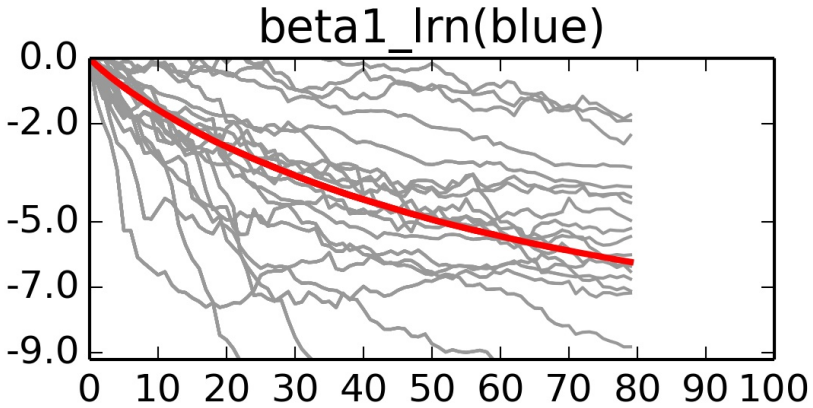
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



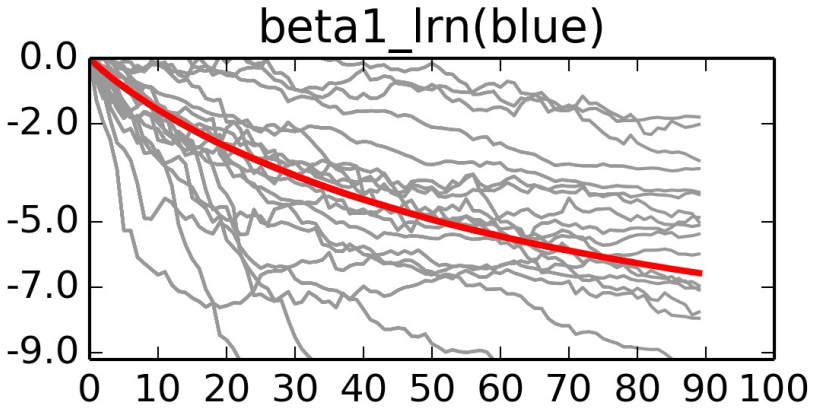
- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

Nice pictures, but is the decomposition meaningful at all ??
Idea:

- measure area under graph - averaged over many runs and images (red line)



- can compare against random orderings of pixels
- can use to compare methods (LRP, sensitivity, deconvolution)
- slow but independent of deduction of LRP

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

In practice,

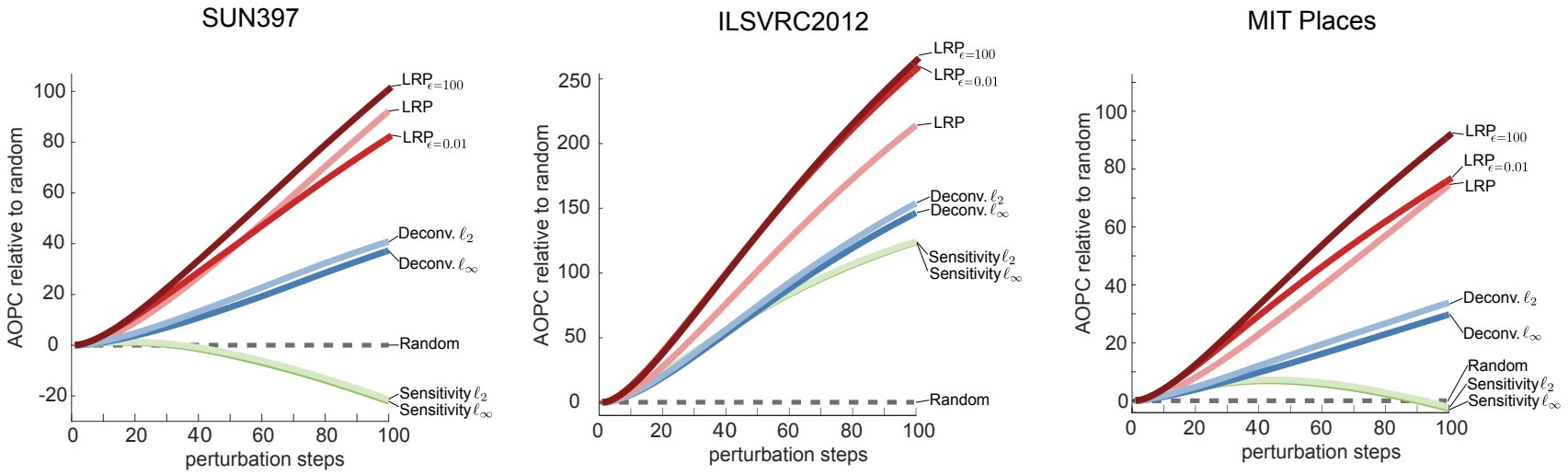
- modify a region of pixels (score for a region is the average over its pixels) – regions allow to capture local correlations
- compute average decrease for many different modifications of the same region
- evaluation does not need image labels

Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

Evaluation?

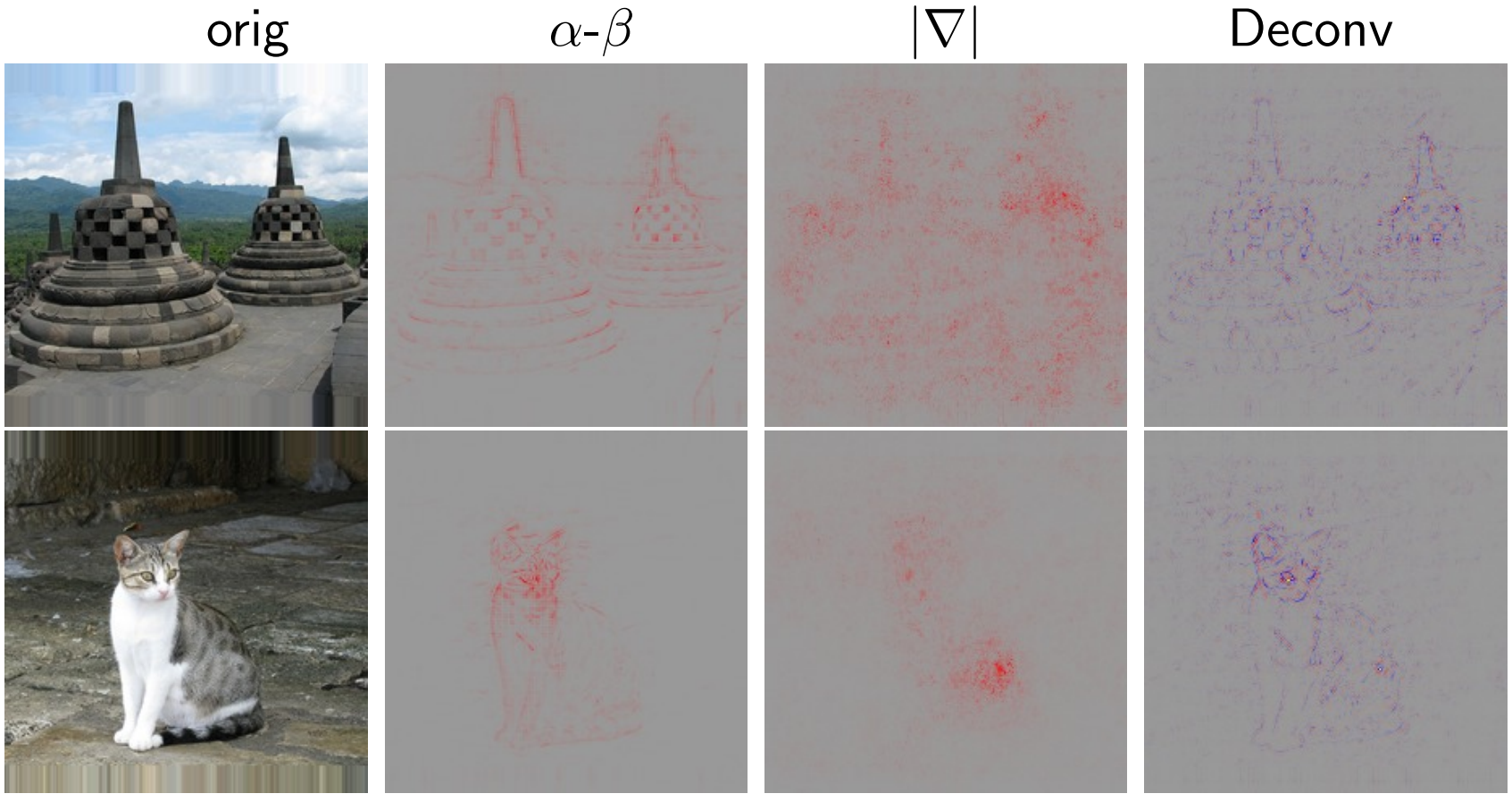
Evaluation for

- Imagenet classifier / Imagenet Test data
- MIT Places classifier / MIT Places Test data
- MIT Places classifier / SUN397 test data

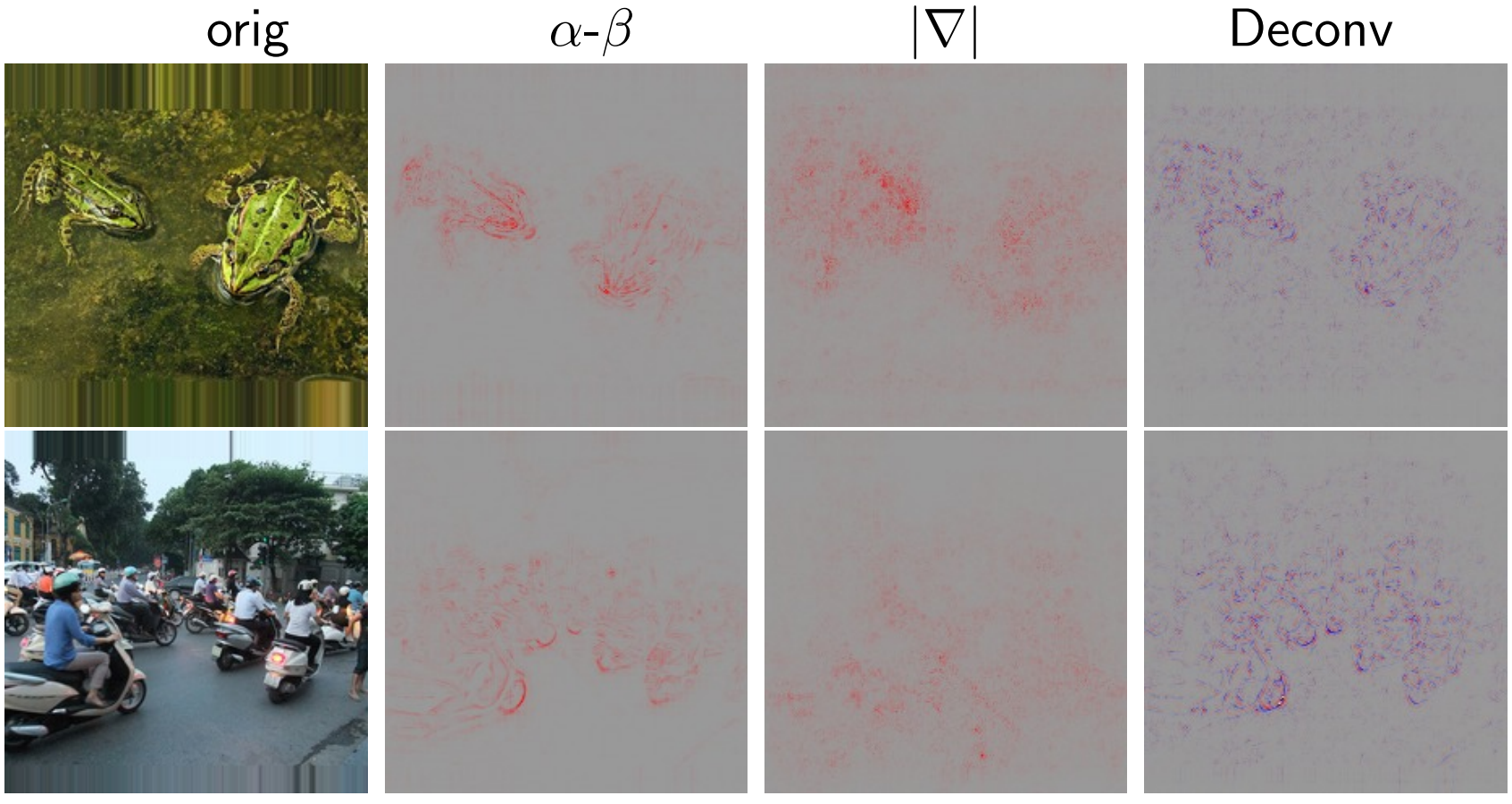


Evaluating the visualization of what a Deep Neural Network has learned, Samek et al., IEEE TNNLS 2017

See it by yourself



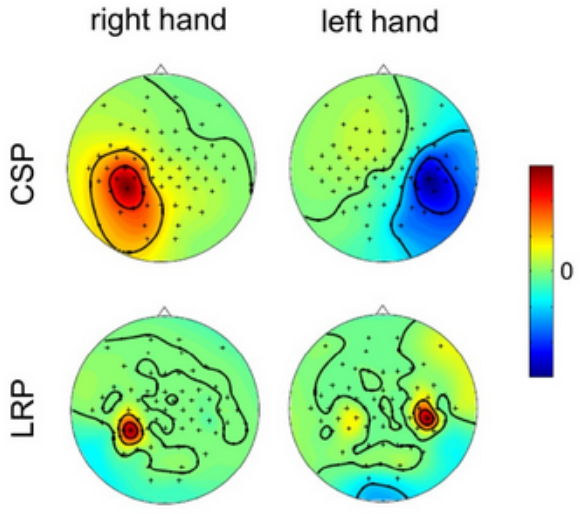
See it by yourself



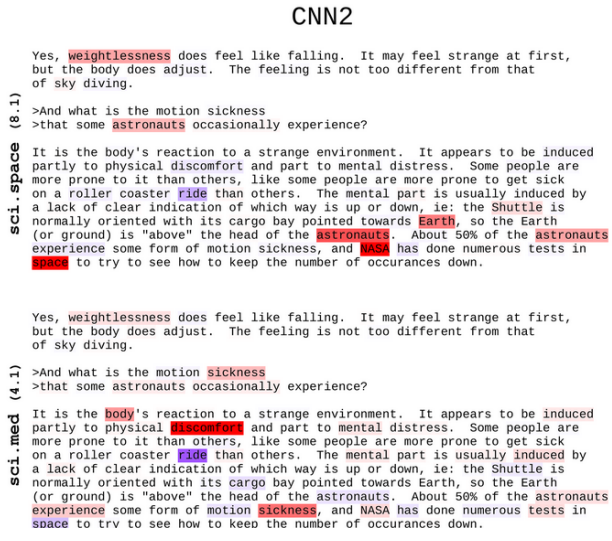
Worth reading

- deepLIFT (Shrikumar et al)
- LIME (Ribeiro et al.)
- patternLRP (Kindermans et al.)
- Leila Arras' papers on LRP for Text and LSTM

Applications I: Finding Evidence

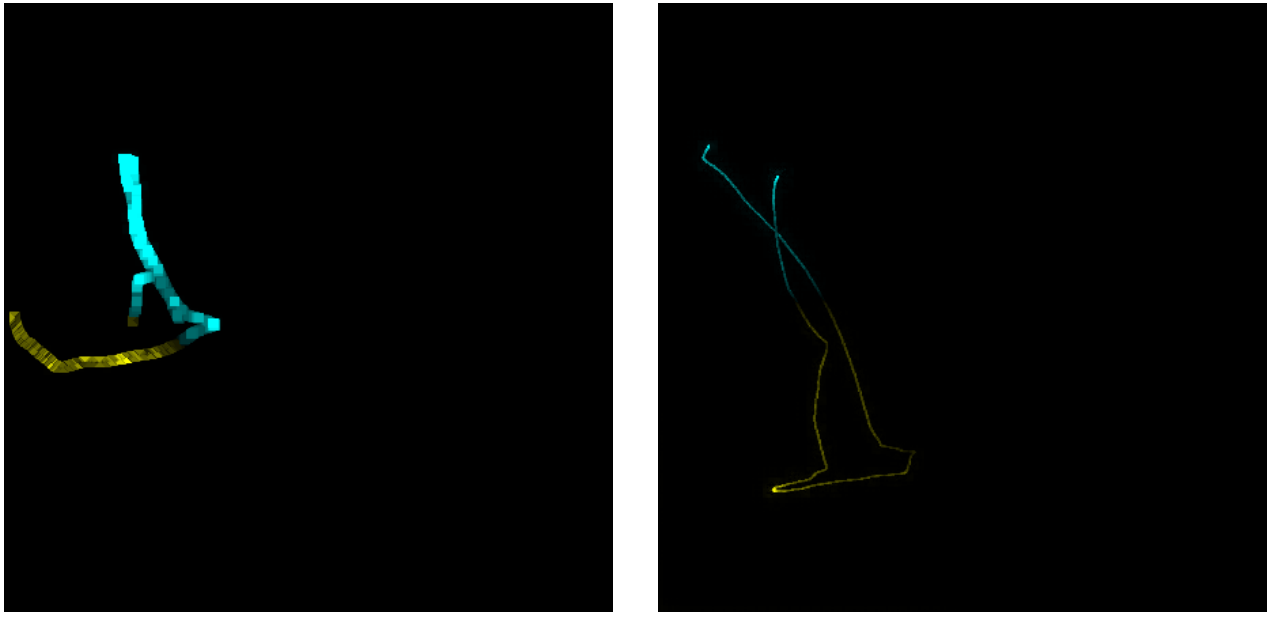


BCI: Sturm et al. Journal of Neuroscience Methods 274, 141-145, 2016



Text-CNN: Arras et al. ACL 2016 RepL4NLP Workshop

Applications I: Finding Evidence



Mouse authentication trajectories: Chong et al.
1^{st} Deep Learning and Security Workshop at S&P, 2018

Applications II: Finding Biases in your Training Data

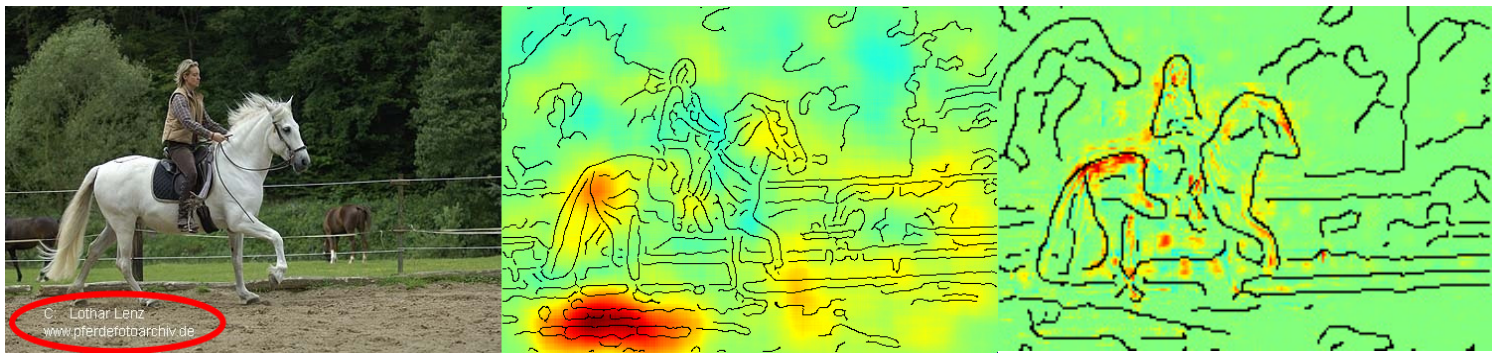
	aeroplane	bicycle	bird	boat	bottle	bus	car
Fisher	79.08%	66.44%	45.90%	70.88%	27.64%	69.67%	80.96%
DeepNet	88.08%	79.69%	80.77%	77.20%	35.48%	72.71%	86.30%
	cat	**chair**	**cow**	**diningtable**	**dog**	**horse**	**motorbike**
Fisher	59.92%	51.92%	47.60%	58.06%	42.28%	80.45%	69.34%
DeepNet	81.10%	51.04%	61.10%	64.62%	76.17%	81.60%	79.33%
	person	**pottedplant**	**sheep**	**sofa**	**train**	**tvmonitor**	**mAP**
Fisher	85.10%	28.62%	49.58%	49.31%	82.71%	54.33%	59.99%
DeepNet	92.43%	49.99%	74.04%	49.48%	87.07%	67.08%	72.12%

Analyzing Classifiers: Fisher Vectors and Deep Neural Networks, Lapuschkin et al., CVPR 2016

Applications II: Finding Biases in your Training Data

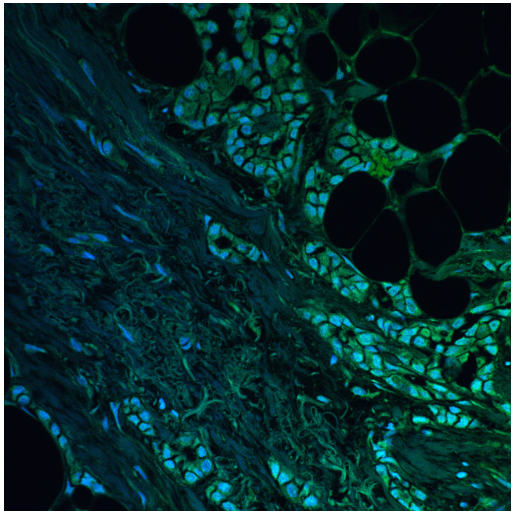
	aeroplane	bicycle	bird	boat	bottle	bus	car
Fisher	79.08%	66.44%	45.90%	70.88%	27.64%	69.67%	80.96%
DeepNet	88.08%	79.69%	80.77%	77.20%	35.48%	72.71%	86.30%
	cat	chair	cow	diningtable	dog	horse	motorbike
Fisher	59.92%	51.92%	47.60%	58.06%	42.28%	80.45%	69.34%
DeepNet	81.10%	51.04%	61.10%	64.62%	76.17%	81.60%	79.33%
	person	pottedplant	sheep	sofa	train	tvmonitor	mAP
Fisher	85.10%	28.62%	49.58%	49.31%	82.71%	54.33%	59.99%
DeepNet	92.43%	49.99%	74.04%	49.48%	87.07%	67.08%	72.12%

Image Fisher Vector Deep Neural Net

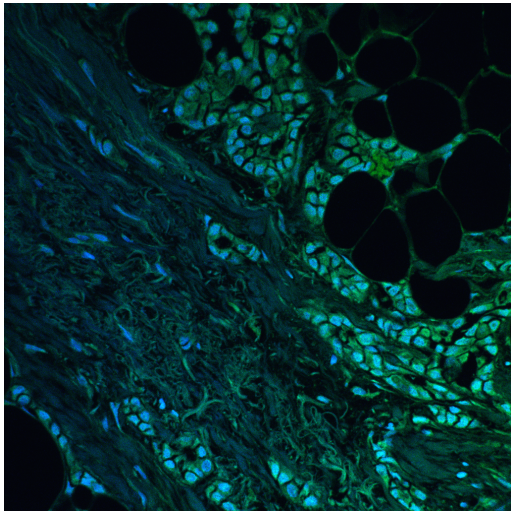
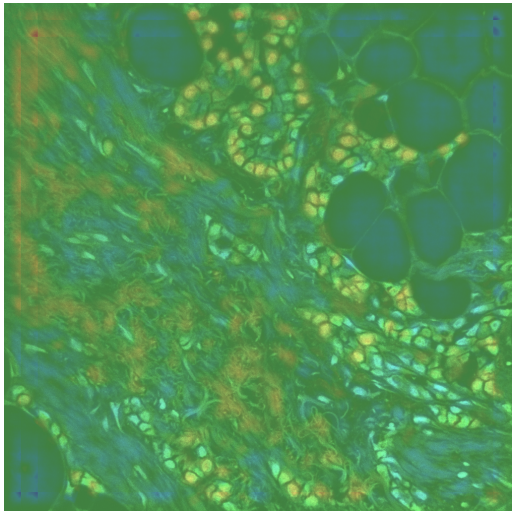
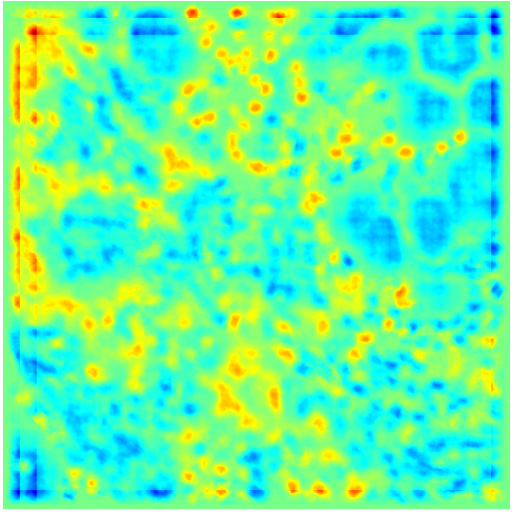


Analyzing Classifiers: Fisher Vectors and Deep Neural Networks, Lapuschkin et al., CVPR 2016

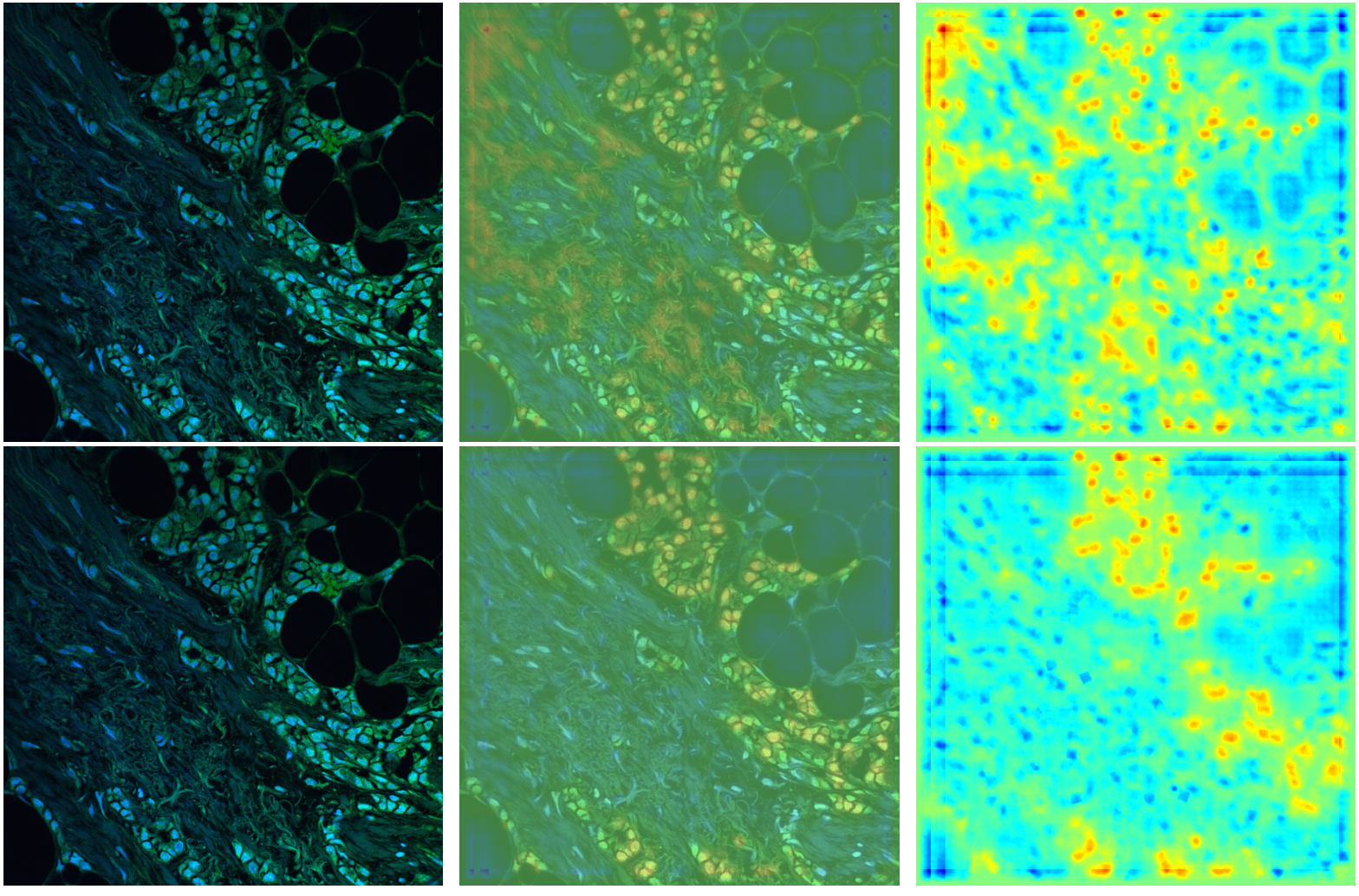
Applications III: Iterative Dataset Design



Applications III: Iterative Dataset Design

Applications III: Iterative Data Design

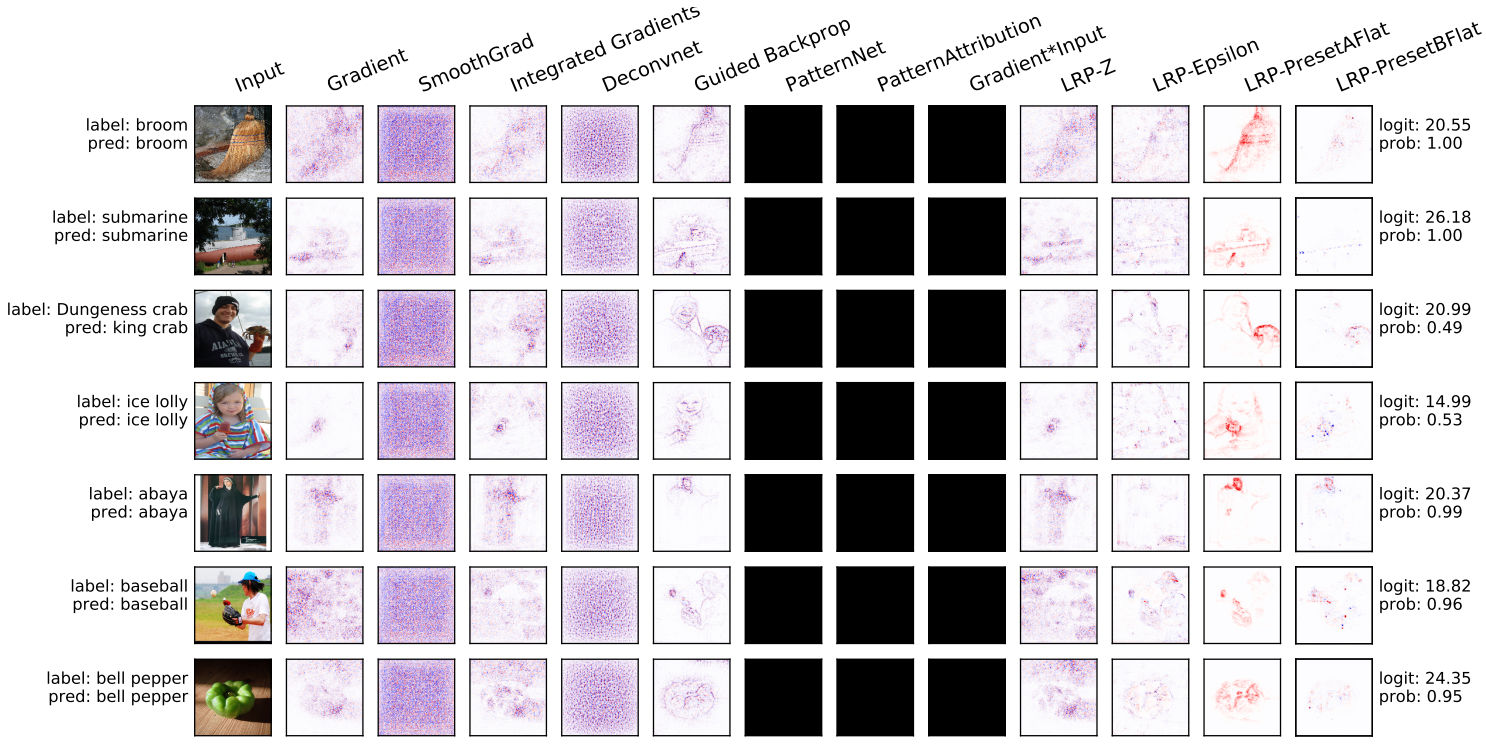


Applications III: How well does LRP scale?

DenseNet-121

Made with Keras:

`https://github.com/albermax/innvestigate`

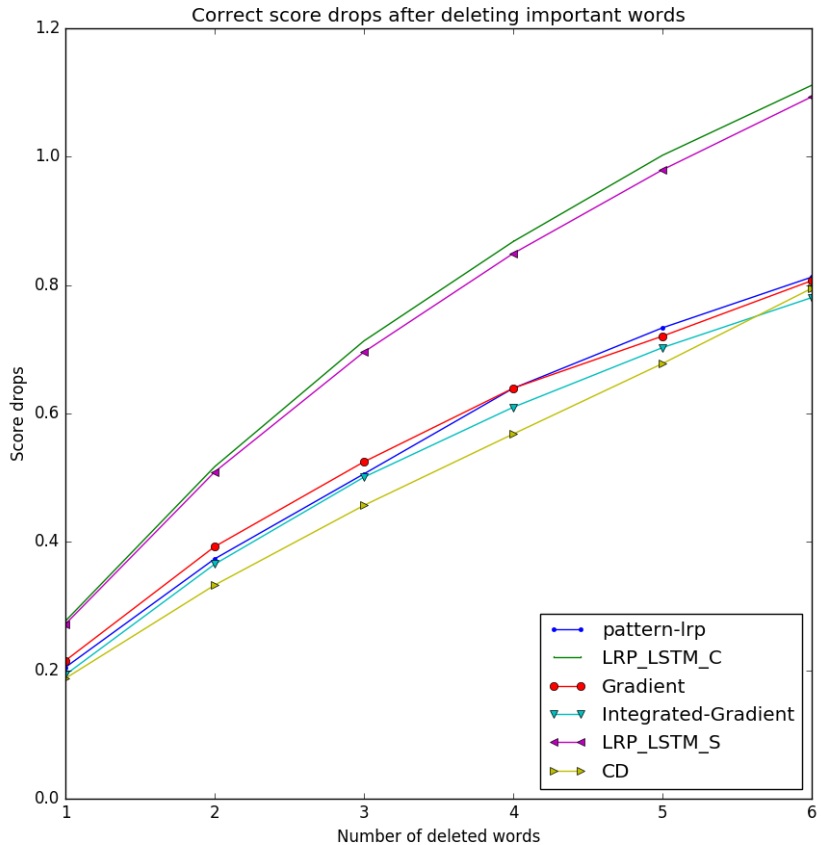


Applications III: How well does LRP scale?

LSTM

see works by Leila Arras et al.

Below is a perturbation test result.

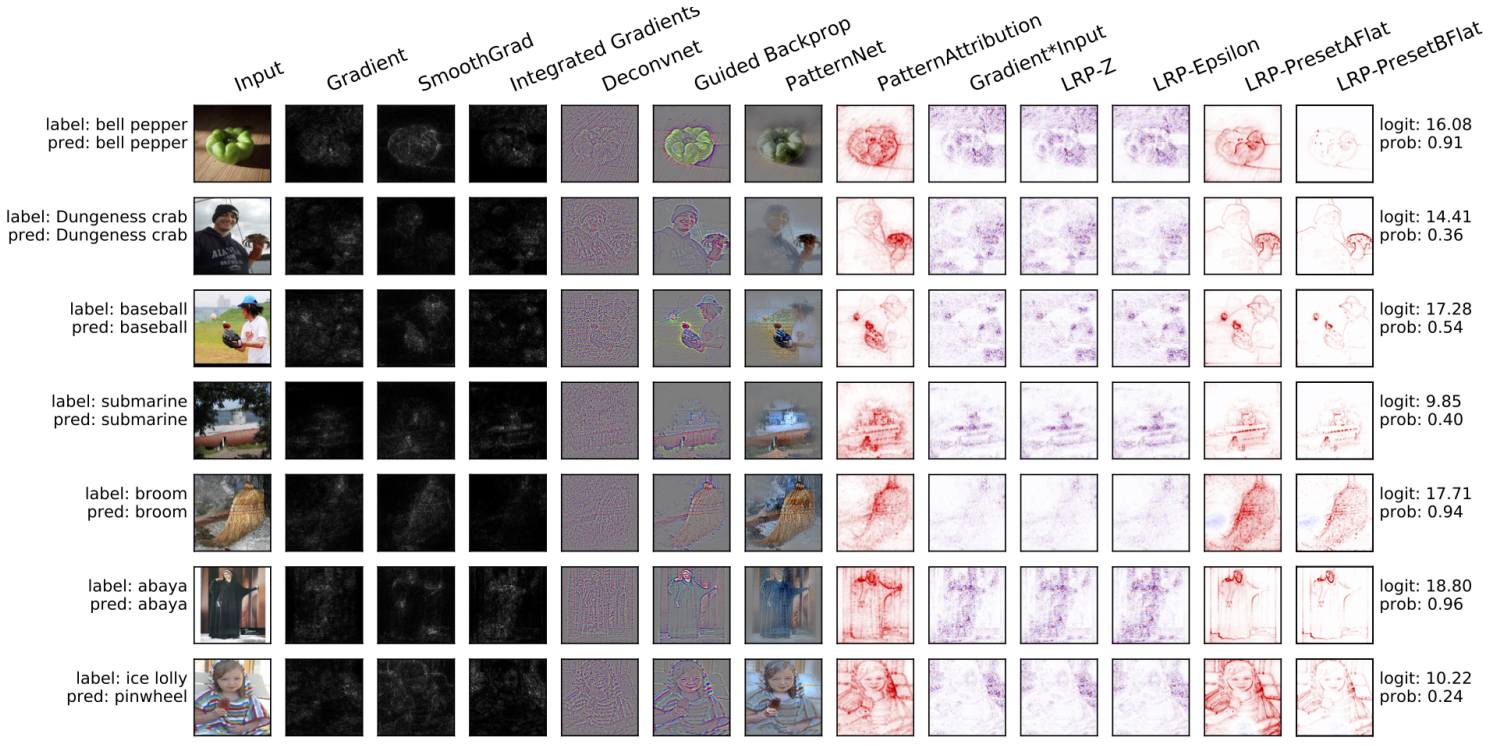


Applications III: How well does LRP scale?

VGG

Made with Keras:

`https://github.com/albermax/innvestigate`



The value of explanations – during the Application Phase

- Use case: Algorithm makes a prediction that serves as second opinion. Grading of cell morphology. Explanation required to resolve confliction opinions.

The value of interpretability during the application phase:

- Helps to clarify in case of diverging opinion: has the human overlooked something?
- Understanding the reasons for a prediction helps to verify/accept proposal made by the machine

Thank you!

Links (LRP for LSTM for example):
http://www.heatmapping.org/
Tutorial: http://www.heatmapping.org/tutorial/
LRP for Keras: https://github.com/albermax/innvestigate
LRP Toolbox:
https://github.com/sebastian-lapuschkin/lrp_toolbox
Demos: https://lrpserver.hhi.fraunhofer.de/