Explaining Deep Nets

Grégoire Montavon (TU Berlin)

CompCancer Deep Learning Workshop

16 January 2020, Berlin

The deep learning ‘revolution’

AlphaGo beats Go human champ

Deep Net outperforms humans in image classification

IMAGENET

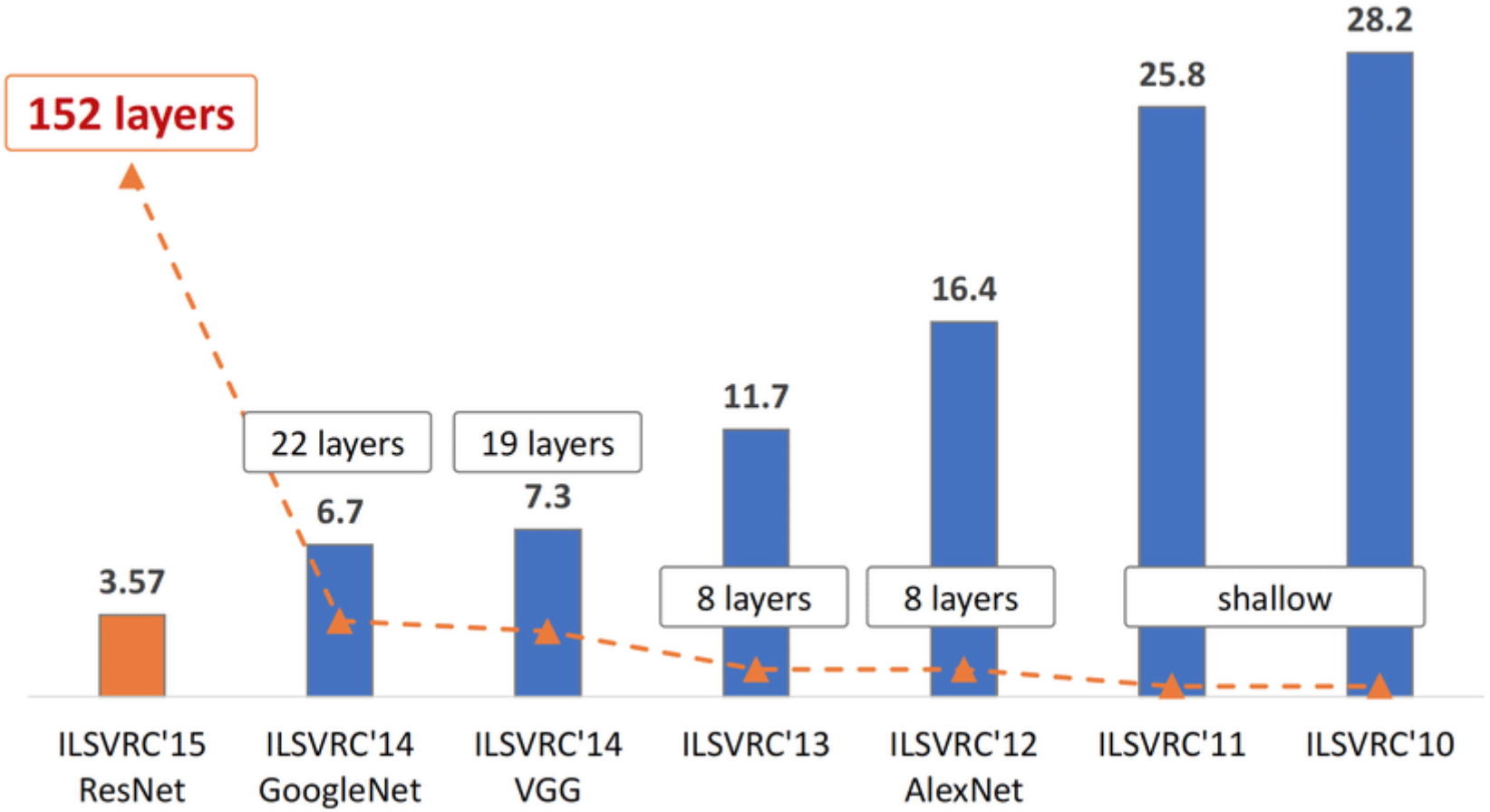
DeepStack beats professional poker players

Computer out-plays humans in "doom"

Visual Reasoning

What size is the cylinder that is left of the brown metal thing that is left of the big sphere?

The deep learning ‘revolution’

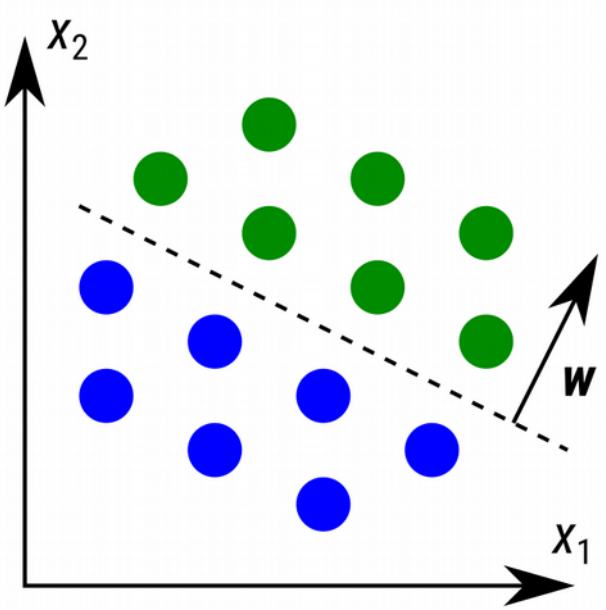


K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for imagerecognition,” CVPR 2016

Why Neural Networks ?

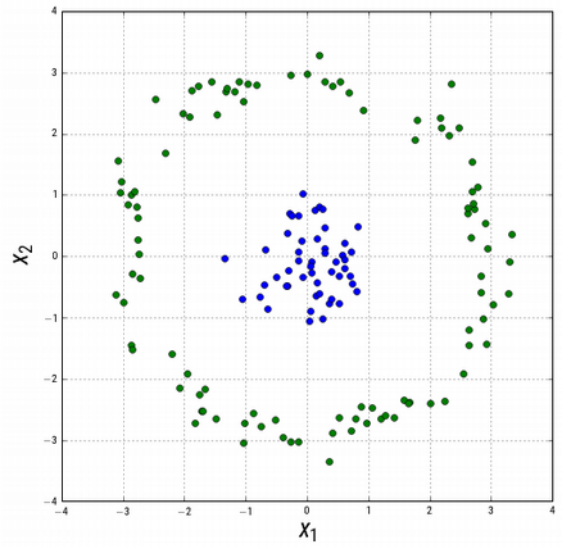
Linear Models

- \+ Closed form or convex formulation (simple to use and reproducible)
- − Lack representation power (most problems are nonlinear)

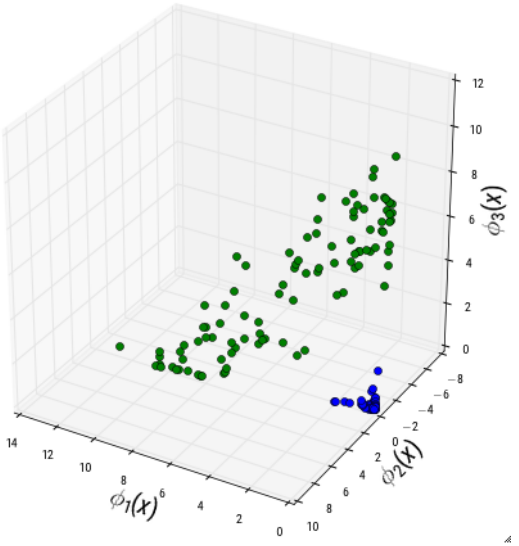


Towards Nonlinear Models

Feature-Expansions/Kernels: Nonlinearly expand the features in a systematic way so that problem becomes linear.



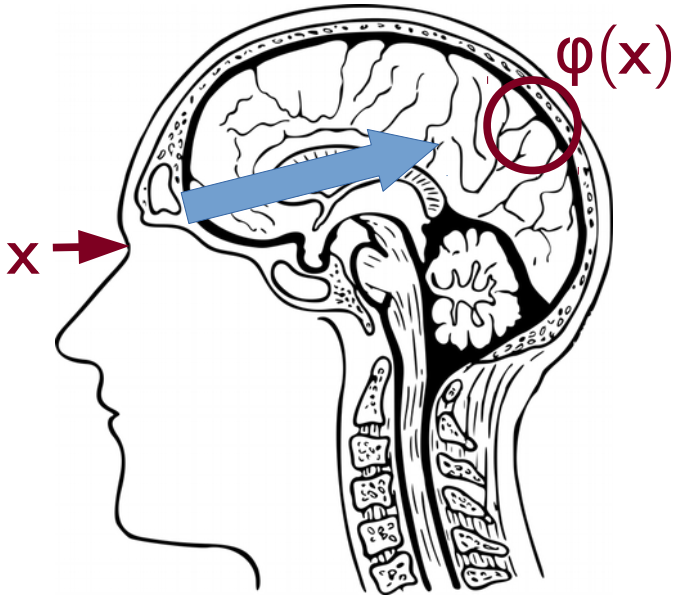
$$\phi(x_1, x_2) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2, x_1, x_2\right)$$



- \+ Approximates any function when the expansion is sufficiently large.
- − Many terms of the expansion won't be useful for the task ($\rightarrow$ computationally inefficient).

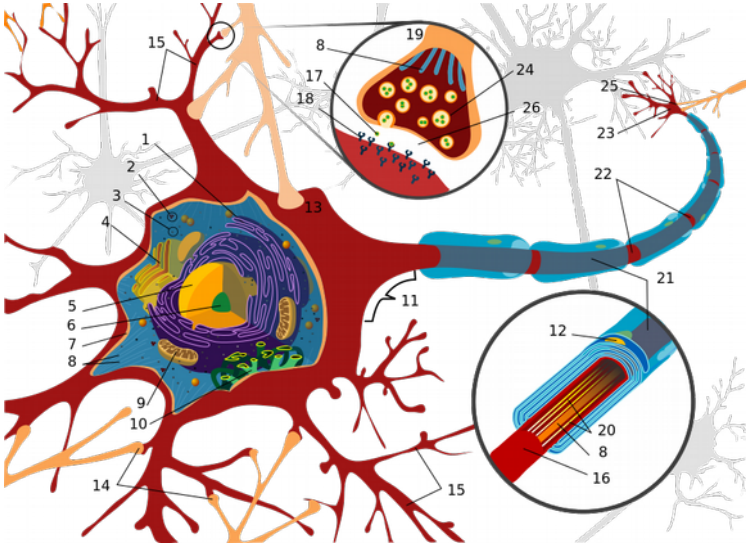
Towards Nonlinear Models

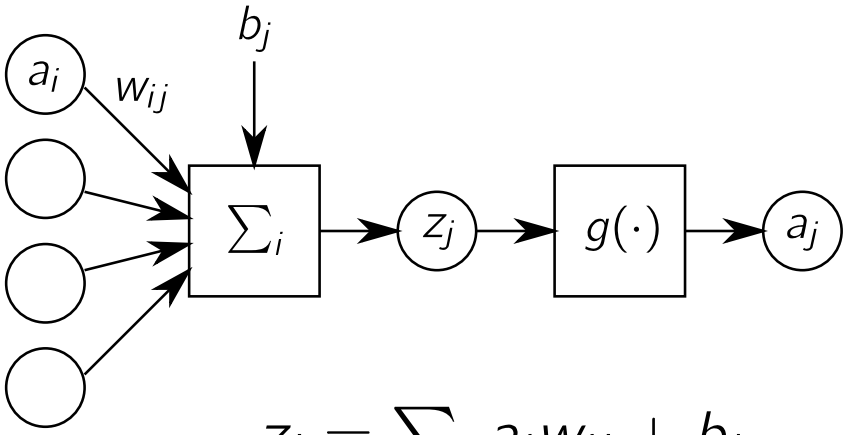
Observation: The brain learns task-relevant features from a limited 'budget' of neurons.



Question: Can we mimic this behavior with an artificial model ?

Biological vs. Artificial Neuron



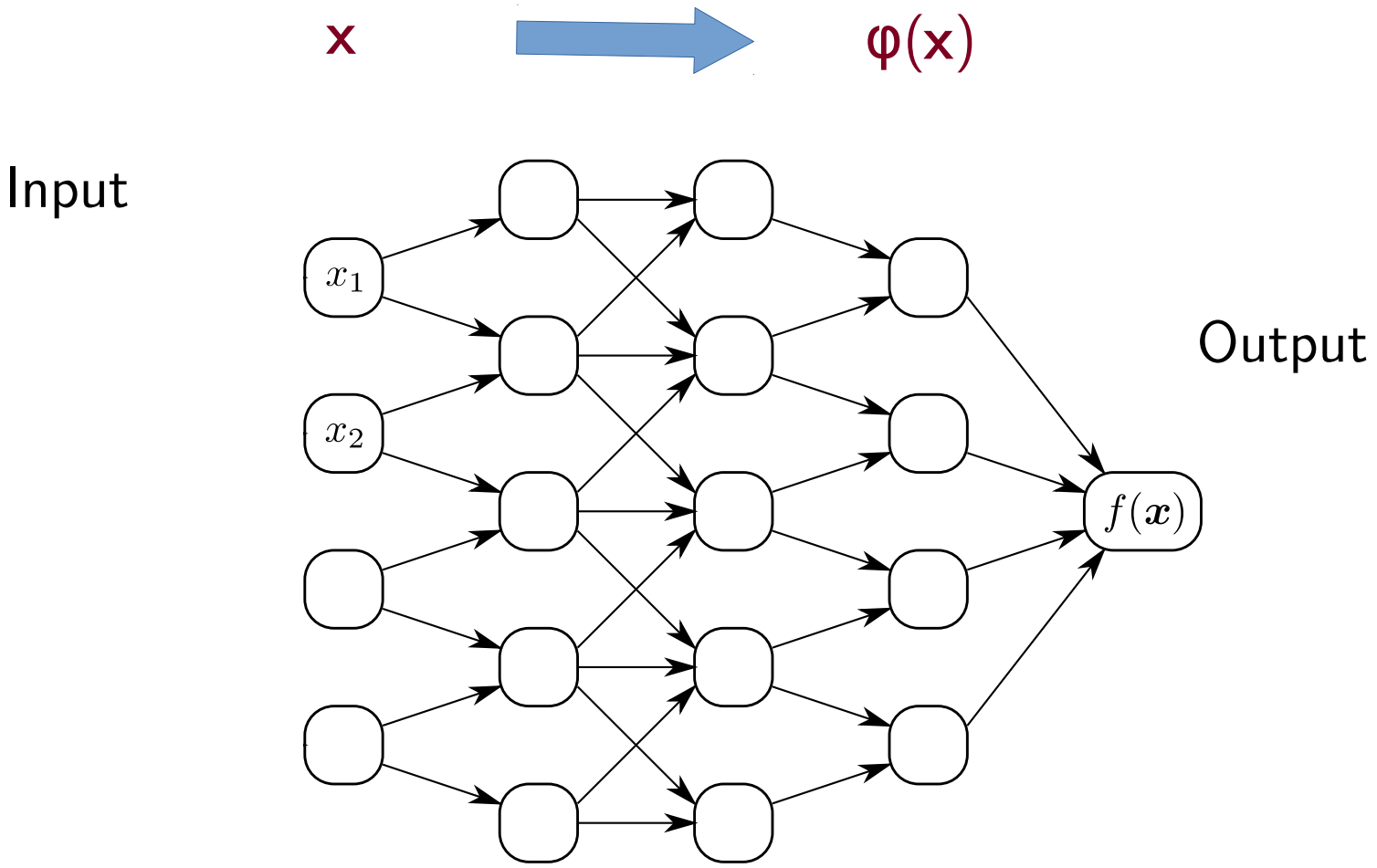


$$z_j = \sum_i a_i w_{ij} + b_j$$

$$a_j = g(z_j)$$

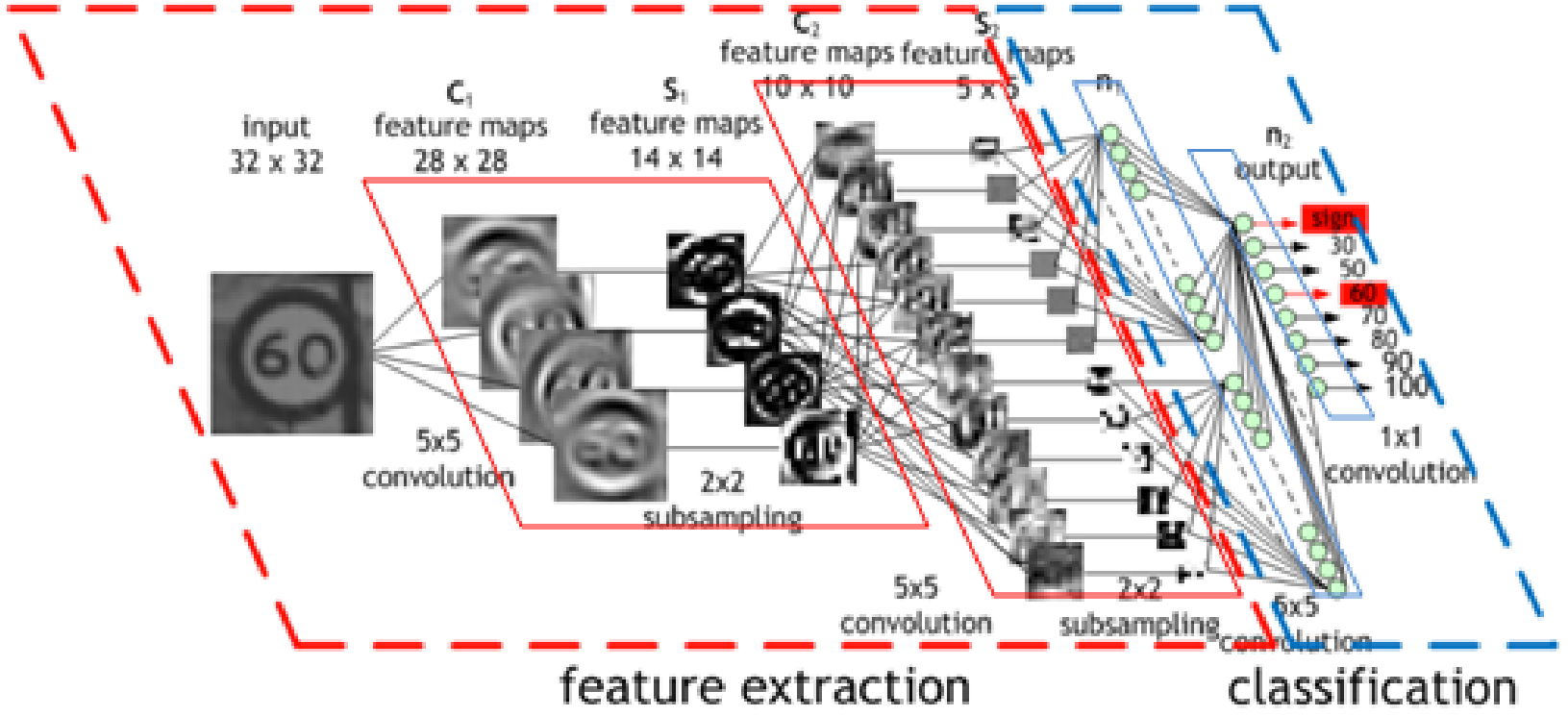
Common denominator: (1) They both have an adaptation mechanism. (2) Ability to represent abstractions derives from interconnecting a large number of them.

A Typical Artificial Neural Network

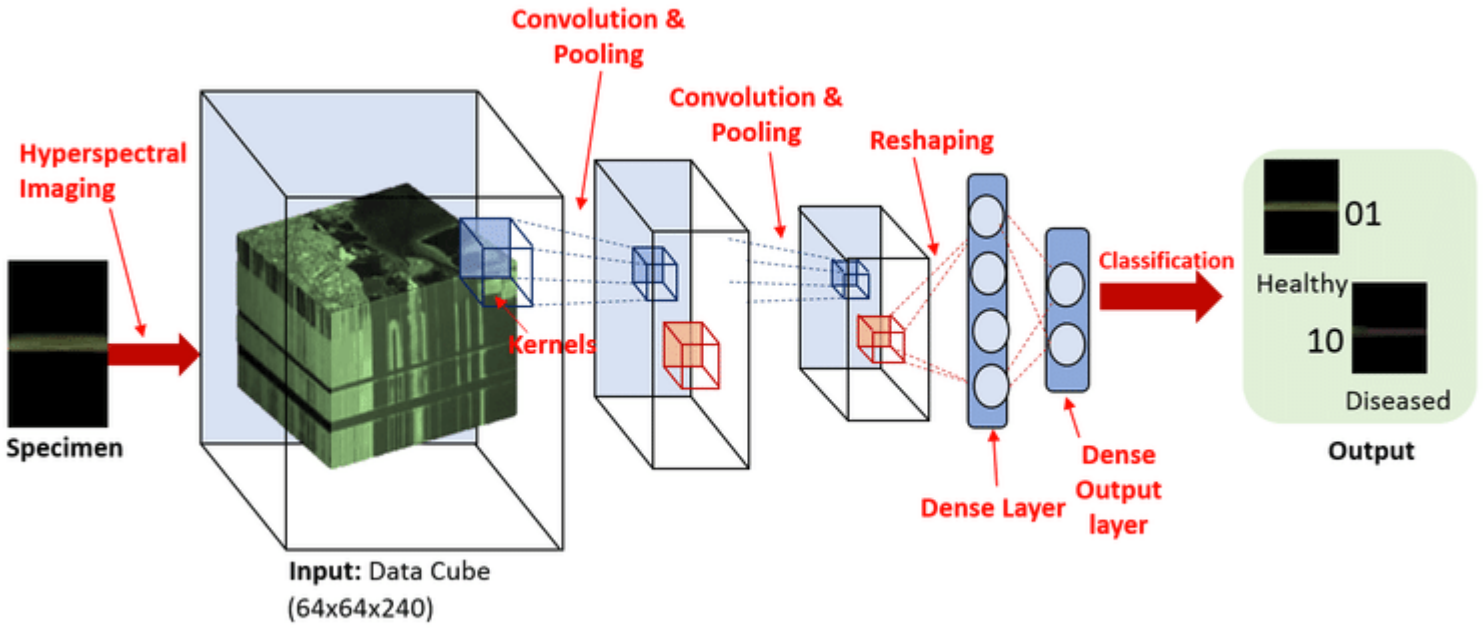


An Overview of Practical Deep Nets

Example 1: Convolutional Neural Network

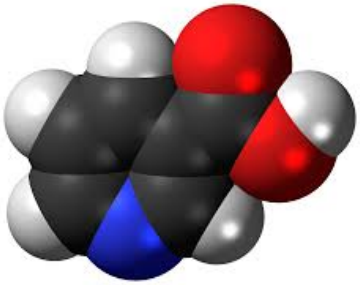


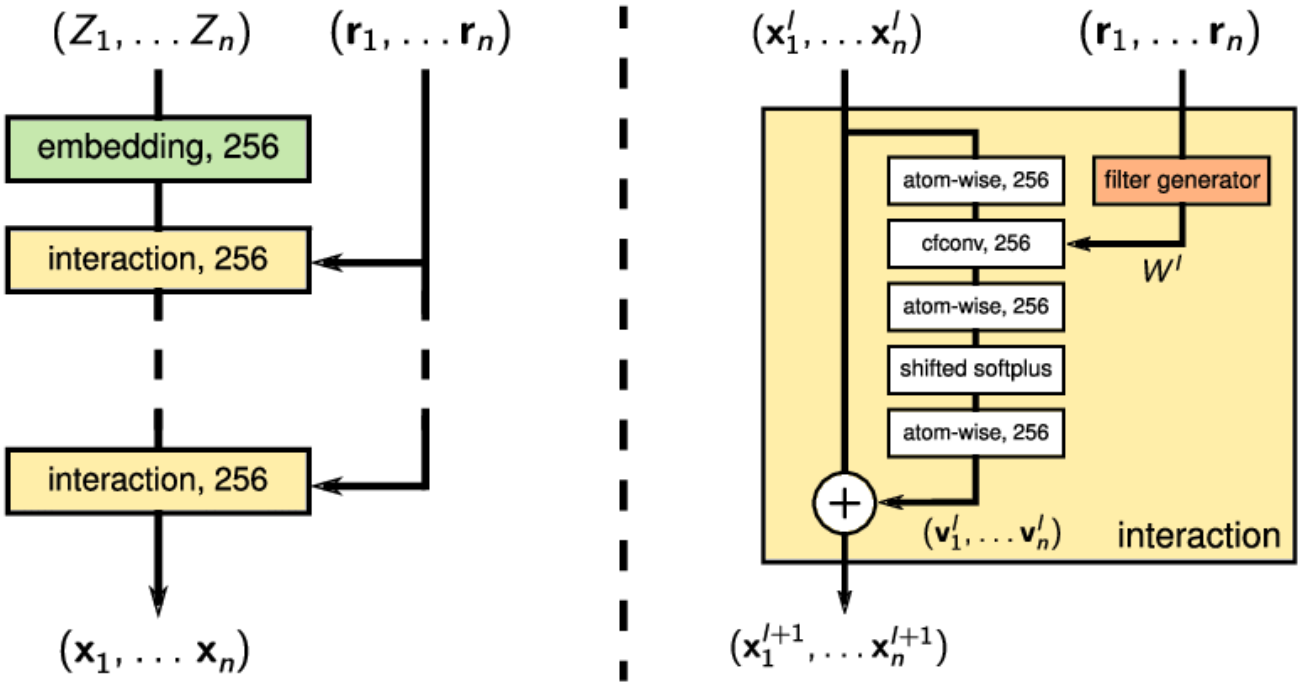
Example 2: 3D Convolutional Network



Source: Nagasubramanian et al. Explaining hyperspectral imaging based plant disease identification: 3D CNN and saliency maps

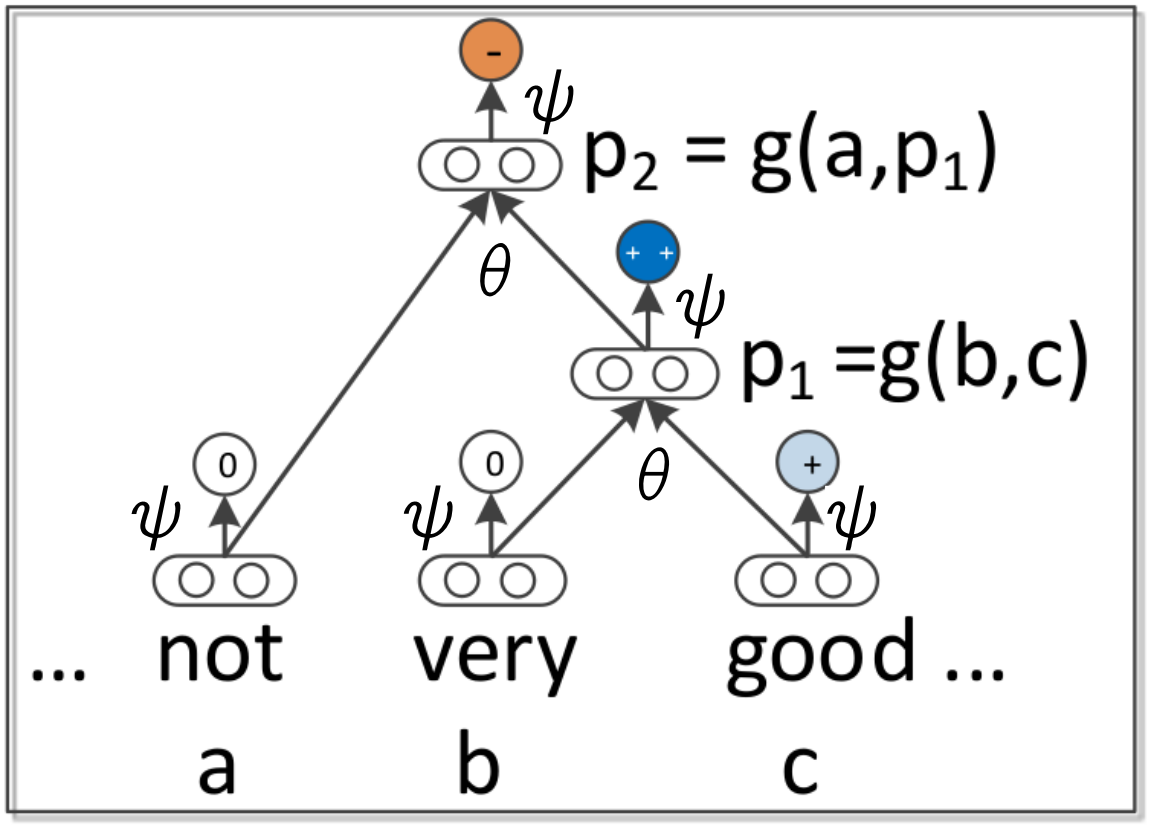
Example 3: Graph Neural Network





Source: Schütt et al.: SchNet: A continuous-filter convolutional neural network for modeling quantum interactions.

Example 4: Recursive Neural Networks



Parameters:

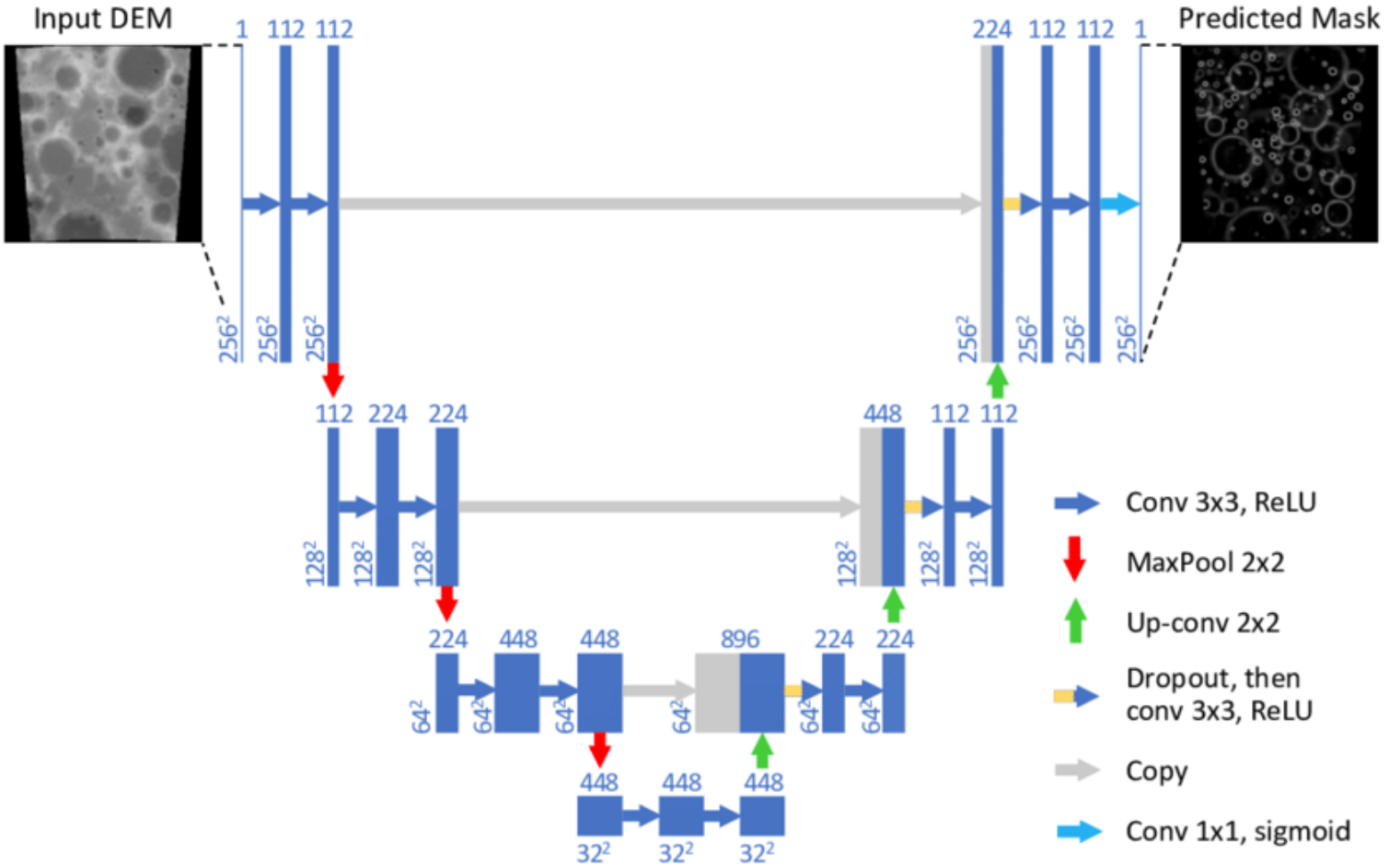
(θ, ψ)

merging text

read sentiment

Source: Socher et al. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank

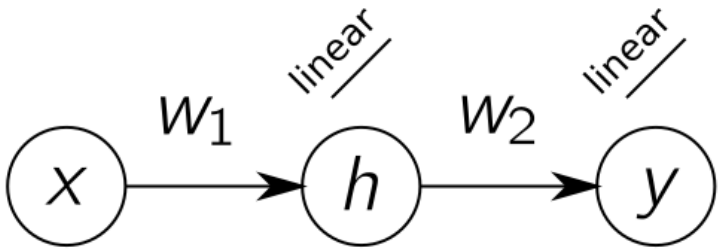
Example 5: U-Net for Segmentation



Source: Silburt et al. 2018 Lunar Crater Identification via Deep Learning

Fast DNN Training

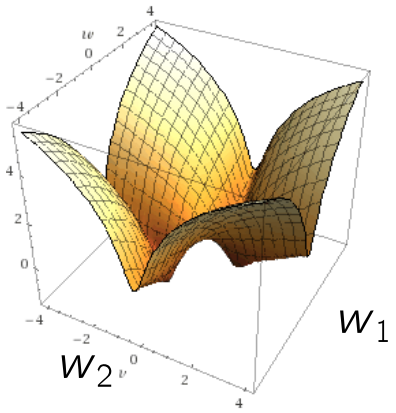
Characterizing the Error Function

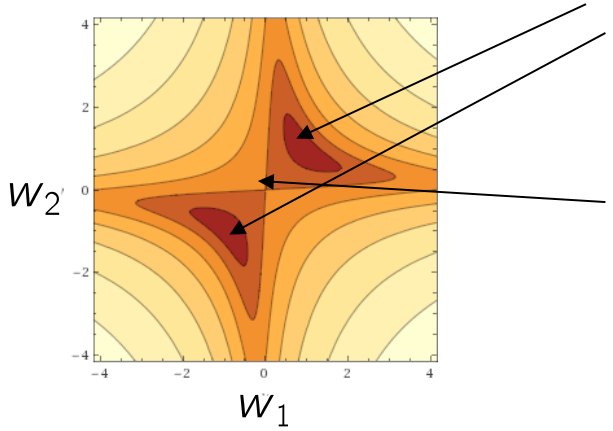


$$y = w_2 \cdot (w_1 \cdot x)$$

$$E = \frac{1}{N} \sum_n \| w_2 \cdot w_1 \cdot x_n - t_n \|^2 + \lambda(\| w_1 \|^2 + \| w_2 \|^2)$$

Example:
N=1, x_1=1, t_1=1, λ=0.1





two local minima

saddle point

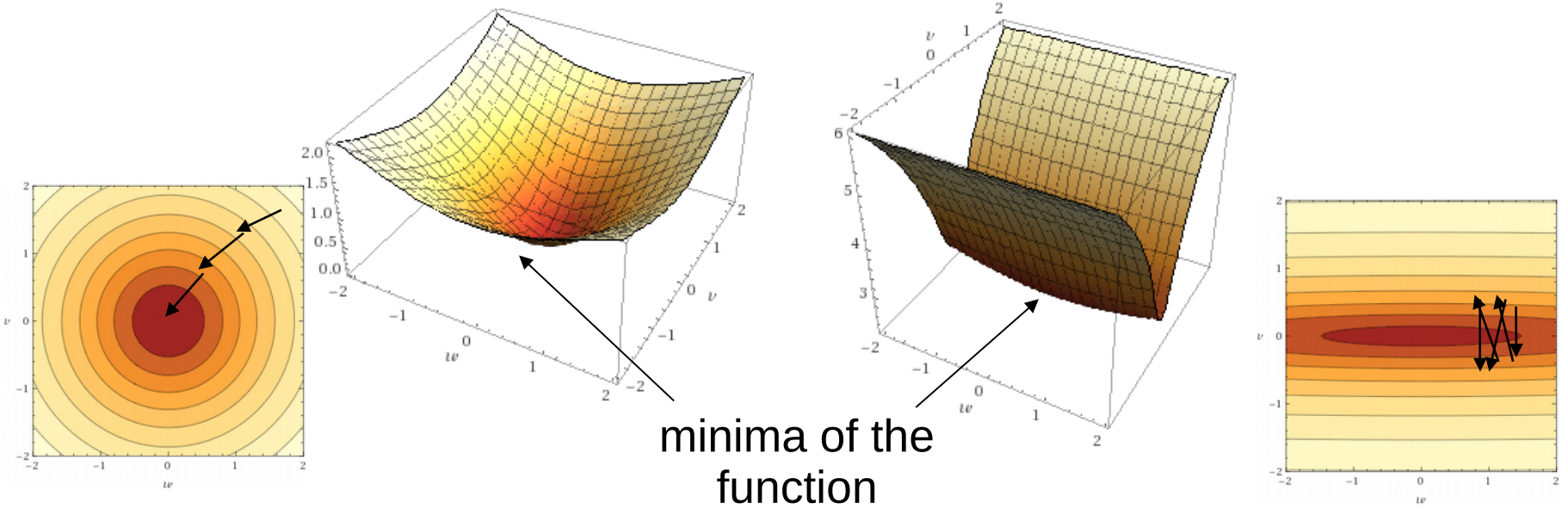
initialization matters

Characterizing the Error Function

Examples:

well-conditioned error function

poorly conditioned error function



Well-conditioned functions are easier to optimize.

Fast Training: Gradient Descent with Momentum

Despite all the good practices for structuring the neural network, the conditioning of the error function remains suboptimal.

This can be dealt with by improving the optimization procedure itself, for example, by choosing update directions as a weighted average of previous updates.

Image from Bishop'95

Momentum accelerates convergence along directions of low curvature.
Momentum can help to overcome a poorly conditioned neural network.

Fast Training: Avoid Redundant Computations

Objective to Minimize:

$$E(\theta) = \frac{1}{N} \sum_{n=1}^{N} E^{(n)}(\theta)$$

Batch GD:

while True:

$$\theta \leftarrow \theta - \gamma \frac{\partial}{\partial \theta} \frac{1}{N} \sum_{n=1}^{N} E^{(n)}$$

$O(N)$

Stochastic GD (SGD):

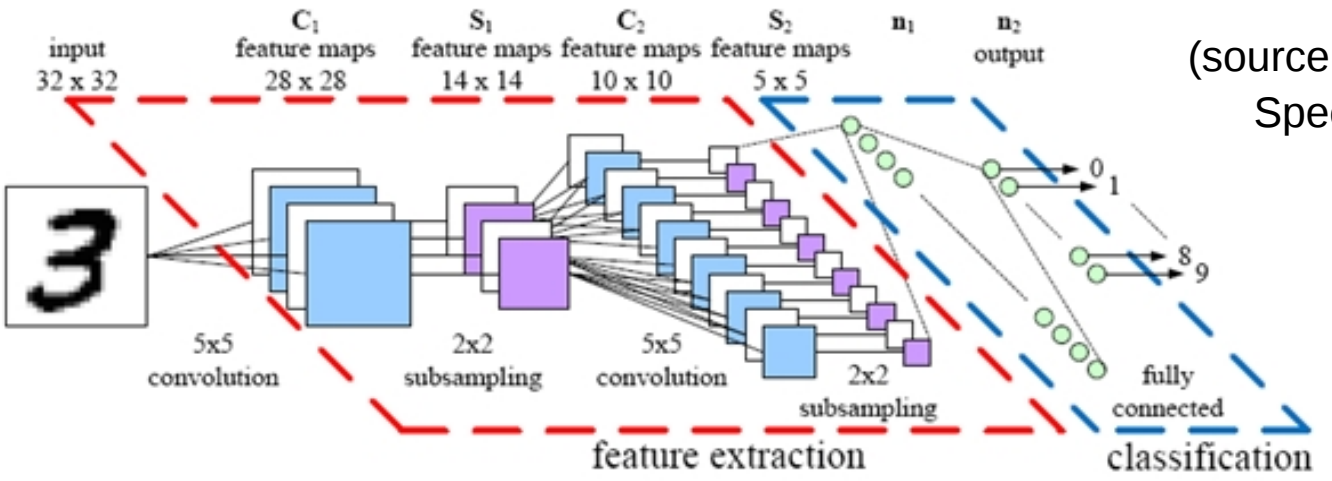
while True:

$$n \leftarrow \text{random}(1, N)$$

$$\theta \leftarrow \theta - \gamma \frac{\partial E^{(n)}}{\partial \theta}$$

$O(1)$

Fast Training: Avoid Computational Bottlenecks



(source: Peemen et al. 2011: Speed sign detection and recognition by convolutional neural networks).

- Lower layers detect simple features at exact locations.
- Higher layers detect complex features at approximate locations.
- Layers progressively replace spatial information with semantic information
  → keep dimensionality and number of connections low at each layer.

Fast Training: Avoid Computational Bottlenecks

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Table 1: GoogLeNet incarnation of the Inception architecture

Szegedy'14

DNNs and Generalization

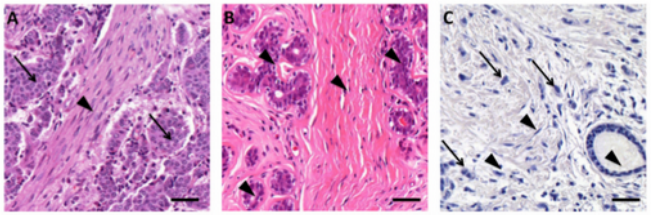
Towards Practical Applications

Deep Net outperforms humans in image classification

IMAGENET

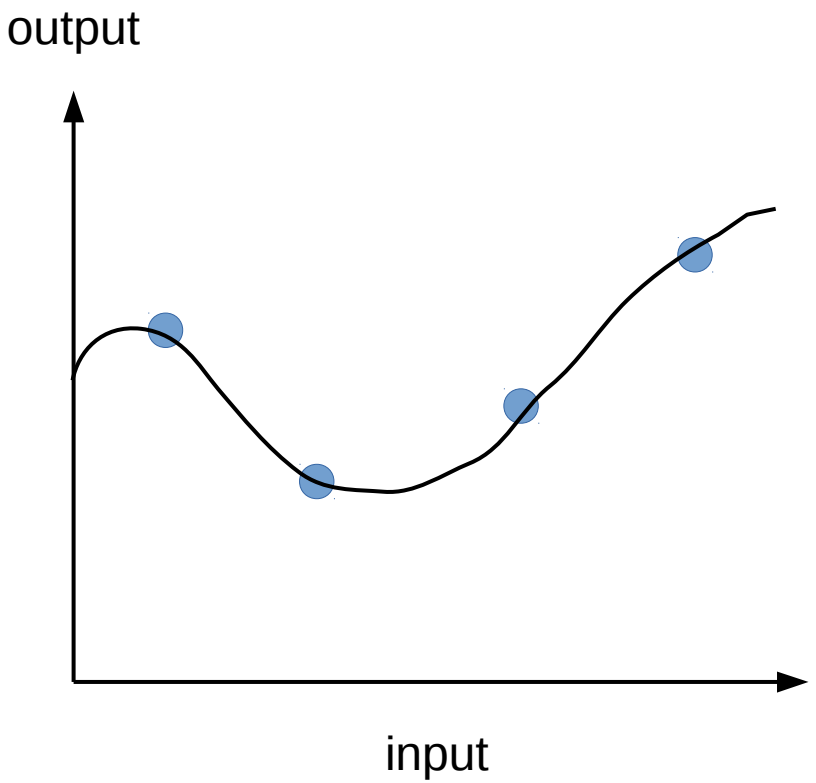
AlphaGo beats Go human champ

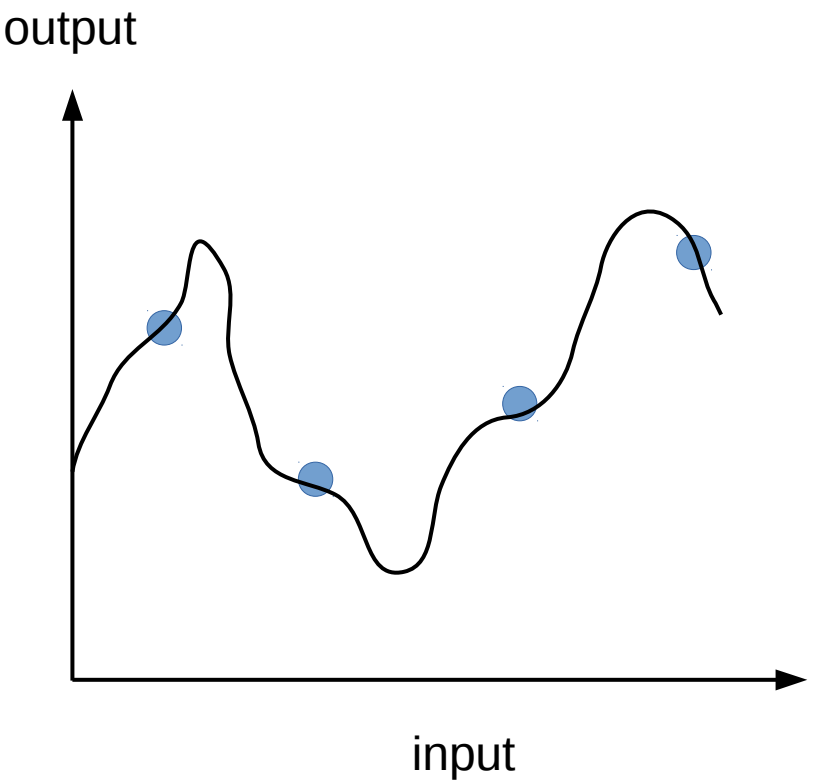
Medical Diagnosis



Autonomous Driving

Learning from Limited Data





Both models explain the few available training points perfectly. Which one to choose ?

Learning from Limited Data

Some milestones:

William of Ockham (1287–1347)

“Entia non sunt multiplicanda praeter necessitatem”

Vladimir Vapnik

Error bounds based on model complexity (VC dimension)

VC Dimension and Generalization Error

Generalization bound [Vapnik]:

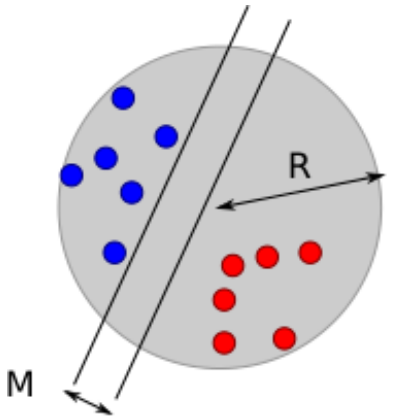
Let h denote the VC-dimension of $\mathcal{F}$. The true risk $R[f]$ (with $f \in \mathcal{F}$) is upper-bounded as:

$$R[f] \leq R_{\text{emp}}[f] + \sqrt{\frac{h\left(\log \frac{2N}{h} + 1\right) - \log(\eta/4)}{N}}$$

with probability $1 - \eta$.

VC-Dimension of Linear Models

$$f(\boldsymbol{x}, (\boldsymbol{w}, b)) = \text{sign}(\boldsymbol{w}^\top \boldsymbol{x} + b)$$



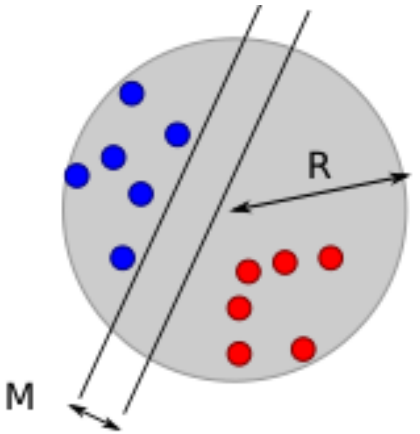
$$h = \min\left\{d + 1, 4\frac{R^2}{M^2} + 1\right\}$$

dimensions margin

Insight: to generalize well, we need to have a large margin between classes

Large Margin in Deep Networks

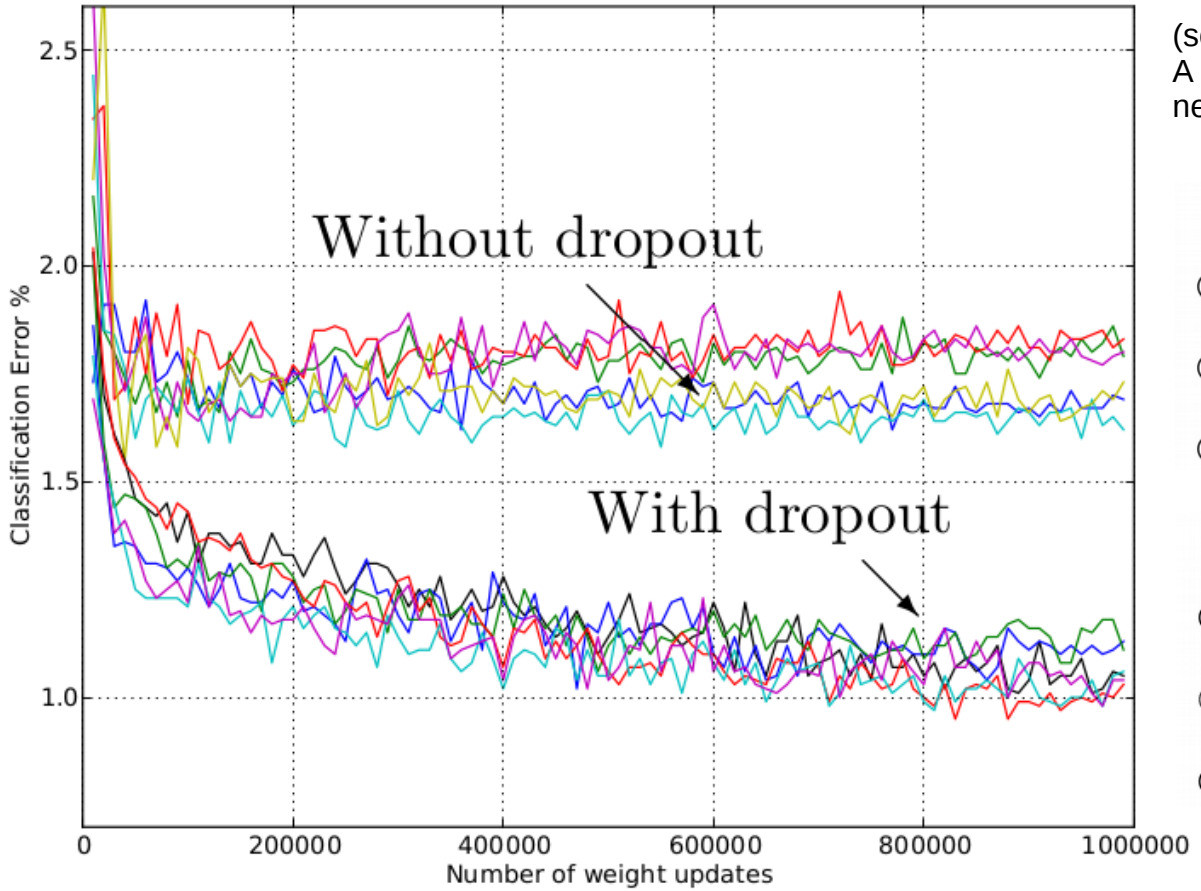
Cross-entropy loss (Bishop'95) used for classification induces a large-margin effect (points close to the margin receive stronger error gradient).



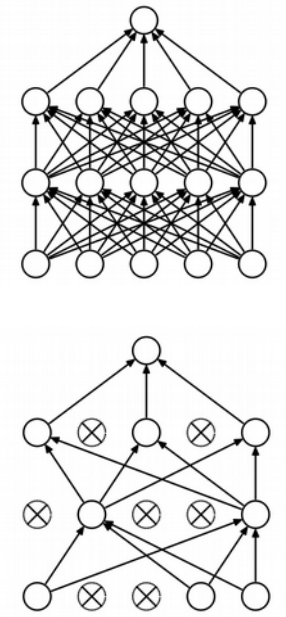
Training with noise makes the classification robust to small local perturbations of the data and neurons, e.g. dropout algorithm (Srivastava'14), stochastic gradient descent, etc.

The Dropout Algorithm (Srivastava'14)

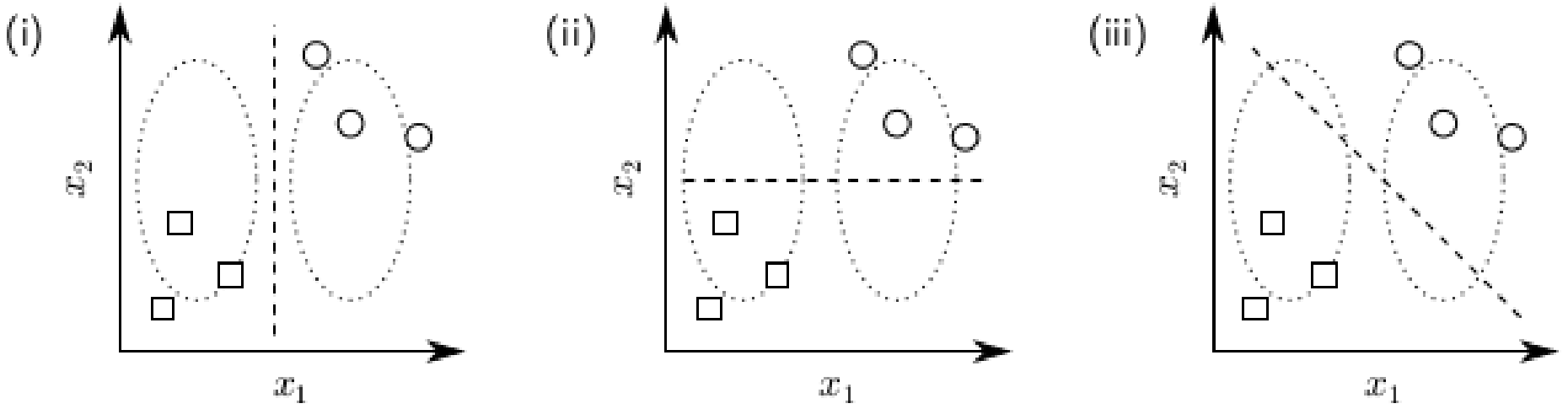
Results on MNIST dataset:



(source: Srivastava et al. 2014: Dropout: A simple way to prevent neural networks from overfitting).



Large Margin: What it Doesn't Solve

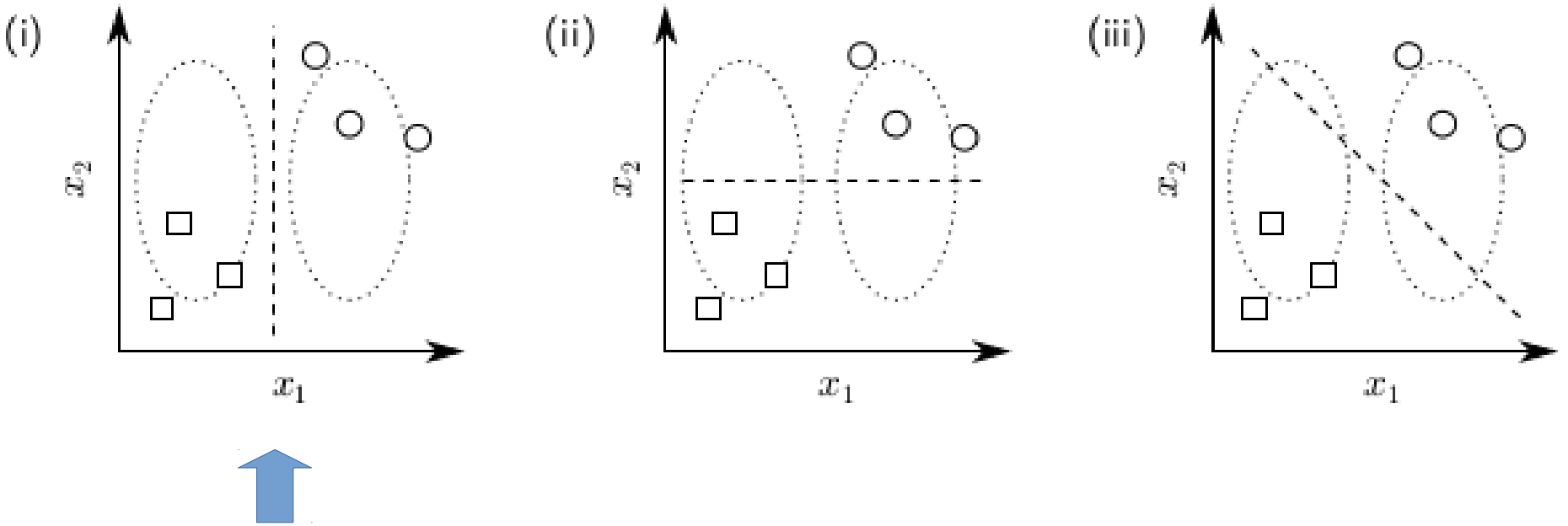


optimal $\neq$ large margin

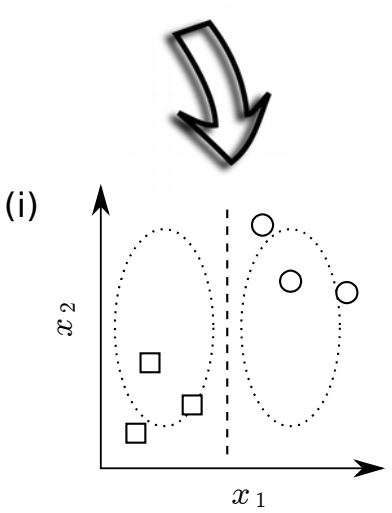
Observation: the large margin model clearly differs from the optimal model.

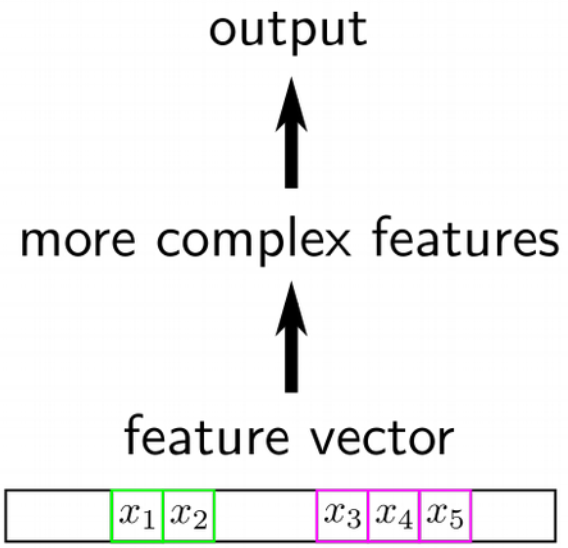
Reason: there are spurious correlations in the training data that do not exist in test data. The large margin model picks up these spurious correlations.

Introducing Prior Knowledge

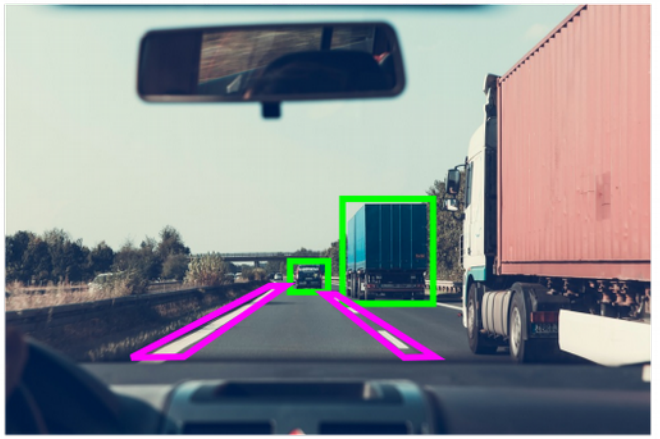


Example 1: Feature Selection



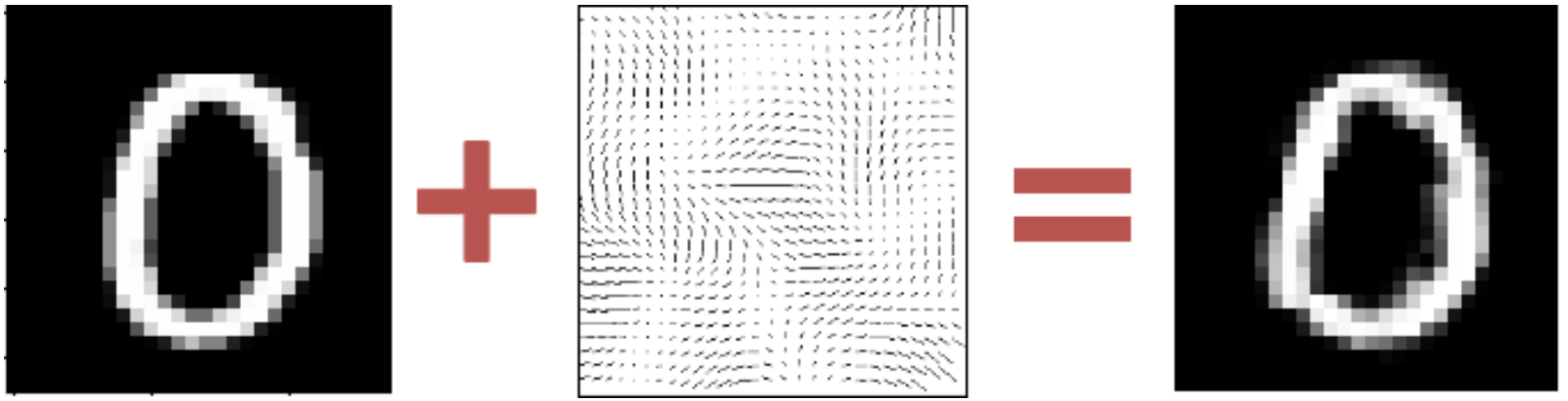


Idea: Tell explicitly to the neural network which features to look at in the data.



Example 2: Data Extension

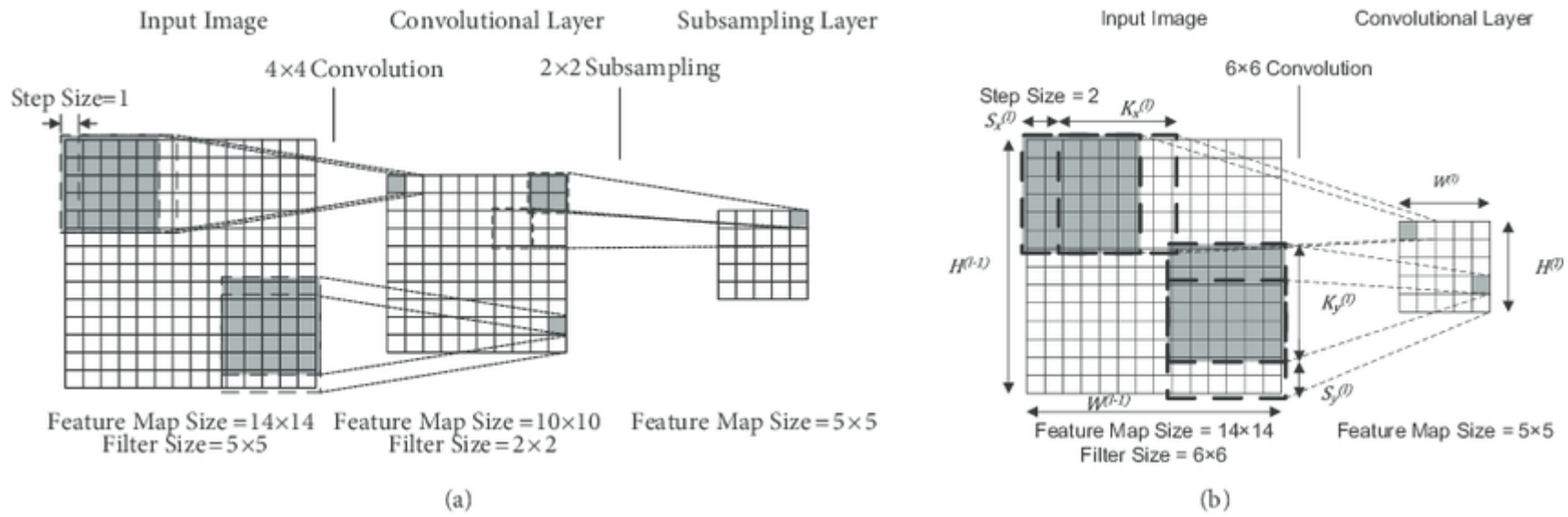
Idea: Tell explicitly what the neural network should be invariant to.



Source: M. Lerousseau “From Papers to Github #1: A practical guide to handwritten digits classifier & dataset preprocessing in Python and tensorflow”
https://marvinler.github.io/2017/03/11/from-papers-to-github-1.html

Example 3: Structured Models

Idea: Tell explicitly what the structure of the prediction function should look like.



Source: Liew et al. 2016 Gender Classification: A Convolutional Neural Network Approach

Example 4: Transfer Learning

Idea: Tell explicitly what representation (possibly learned from another dataset) is suitable for our task.

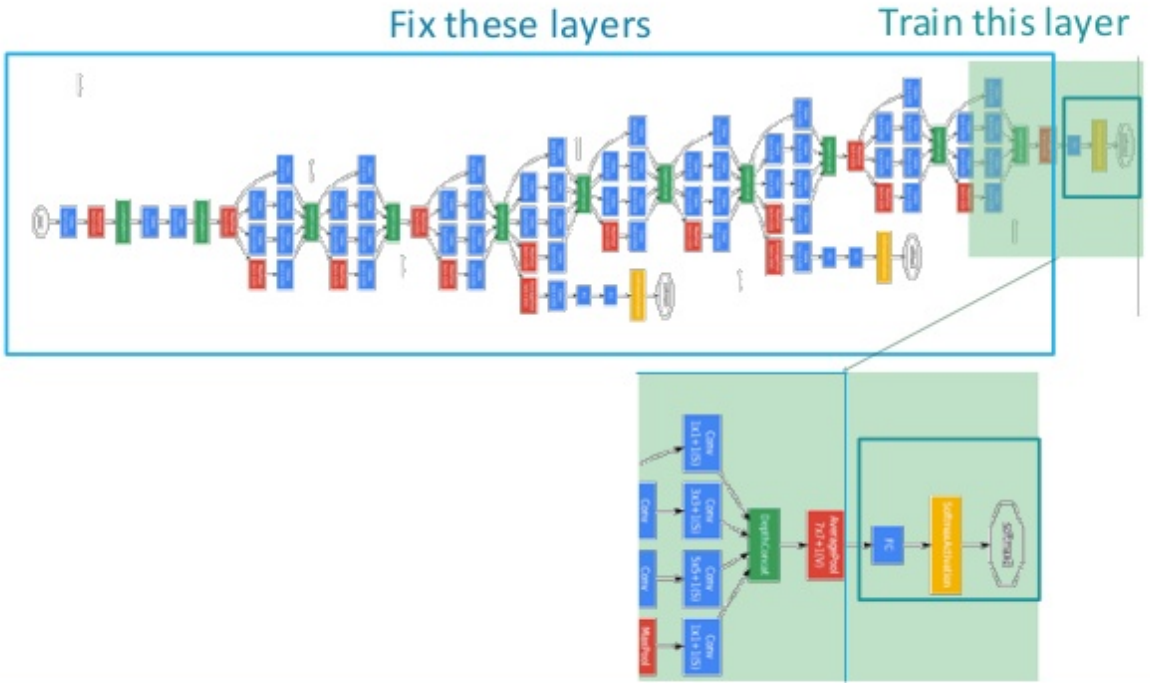


Image Source: Mark Chang, Applied Deep Learning 11/03 Convolutional Neural Networks

Verifying Learned Models

Pascal VOC 2007 Challenge

20 categories detection problem

Pascal VOC-2007 Results

Comparing Performance on Pascal VOC 2007
(Fisher Vector Classifier vs. ImageNet-Pretrained DeepNet)

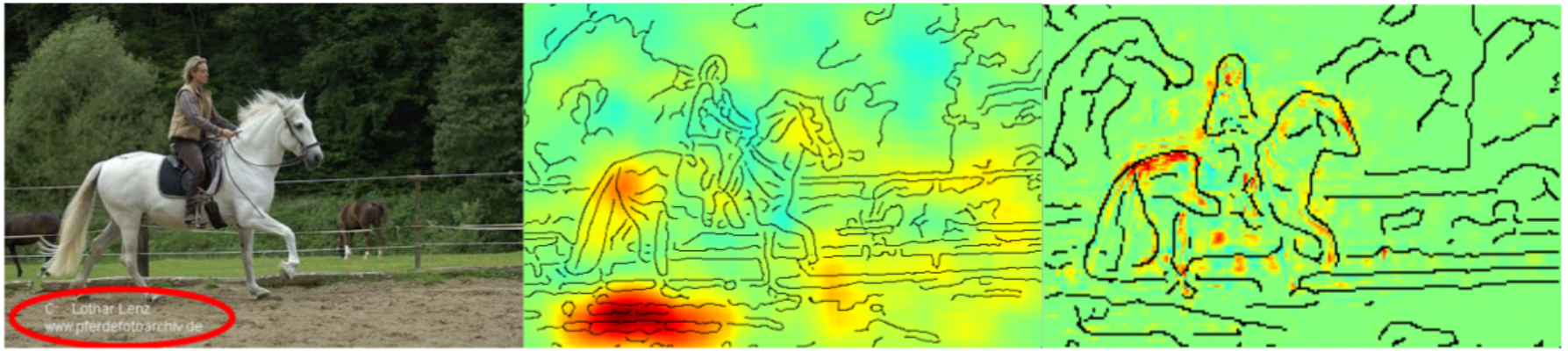
	aeroplane	bicycle	bird	boat	bottle	bus	car
Fisher	79.08%	66.44%	45.90%	70.88%	27.64%	69.67%	80.96%
DeepNet	88.08%	79.69%	80.77%	77.20%	35.48%	72.71%	86.30%
	cat	**chair**	**cow**	**diningtable**	**dog**	**horse**	**motorbike**
Fisher	59.92%	51.92%	47.60%	58.06%	42.28%	80.45%	69.34%
DeepNet	81.10%	51.04%	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%

Pascal VOC-2007 LRP Analysis

Comparing Performance on Pascal VOC 2007
(Fisher Vector Classifier vs. ImageNet-Pretrained DeepNet)

	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%

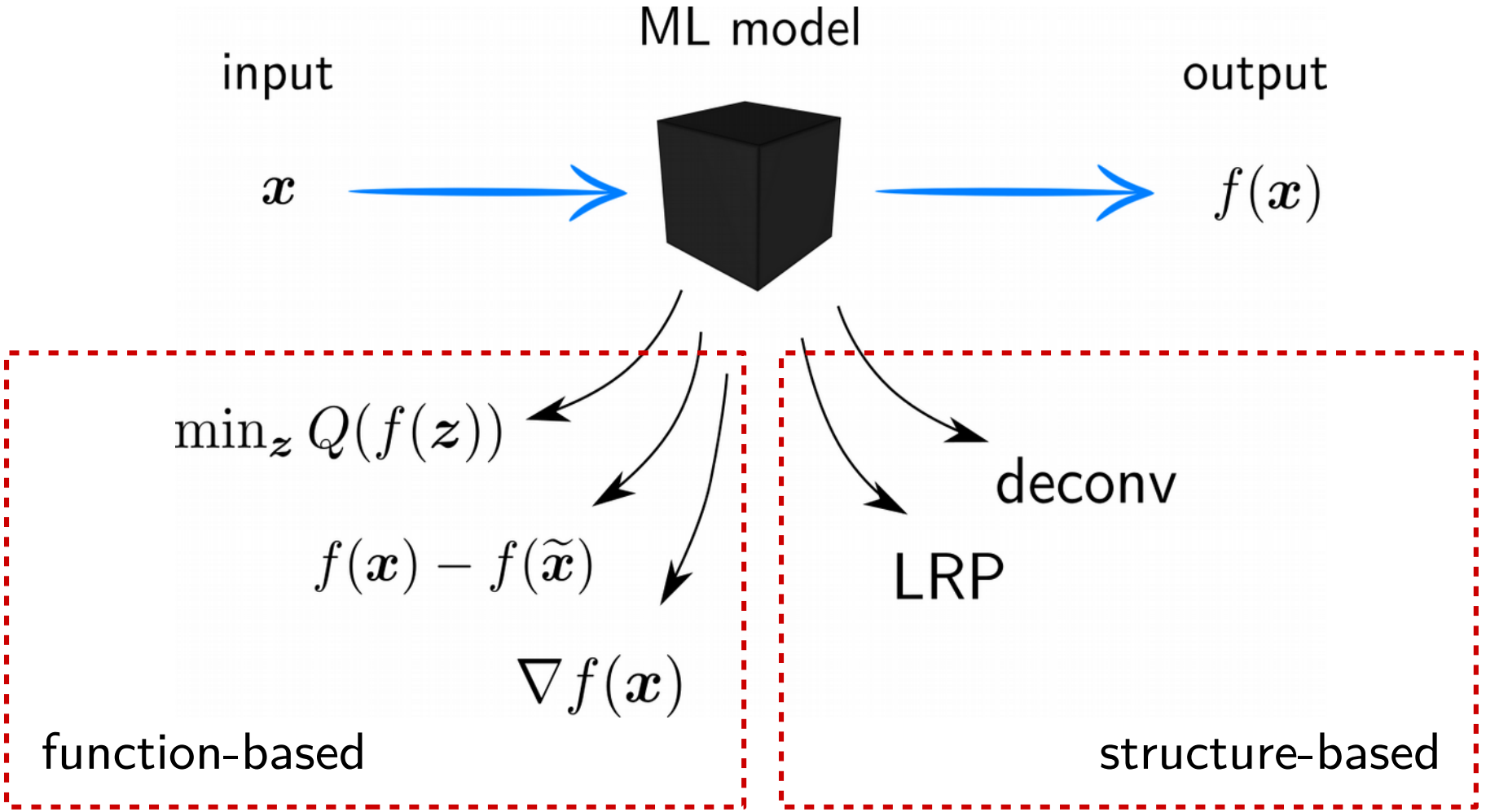
Fisher classifier (pretrained) deep net



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

Pascal VOC-2007 Horse Images

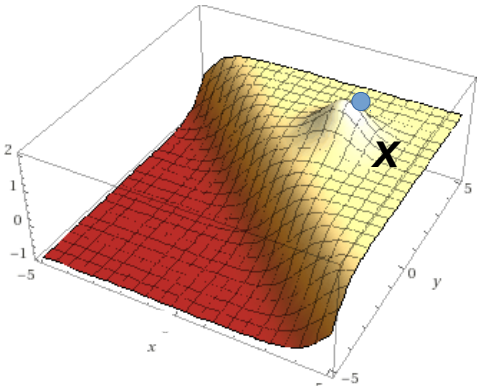
Techniques to Explain Deep Nets



Function-Based Methods: Limitations

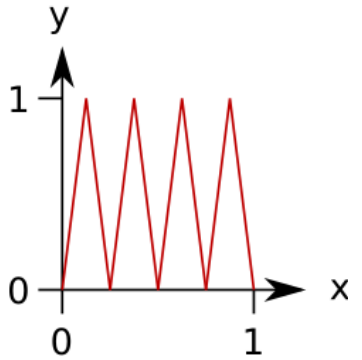
1. **Local vs. global variations**

   Global effects are not visible when looking at the function f(x) locally.



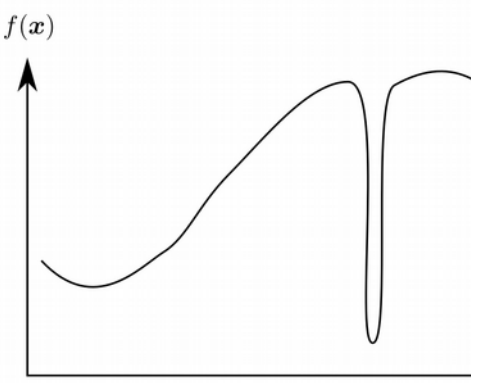
2. **Shattered gradients**

   Function local variations grows exponentially with depth.

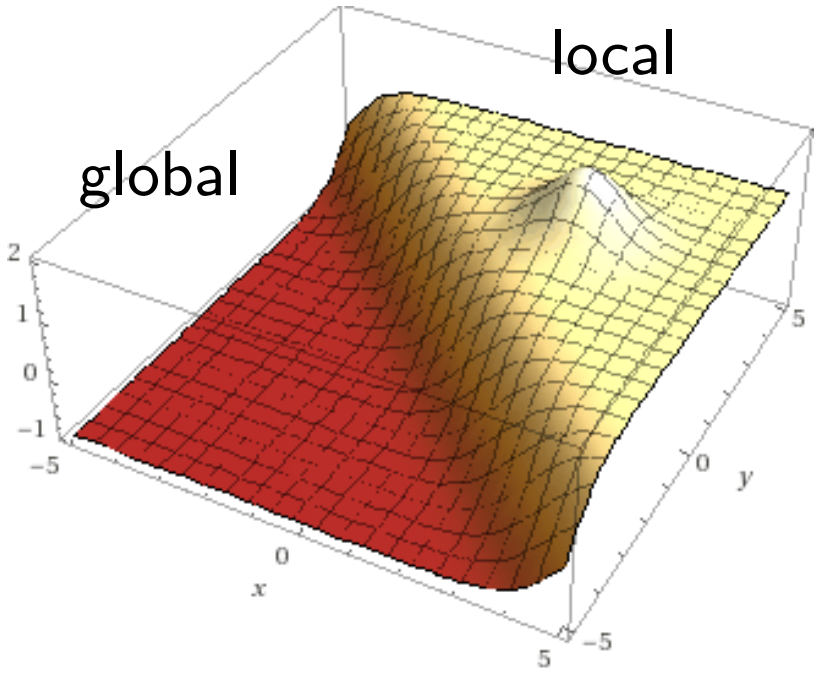


3. **Adversarial examples**

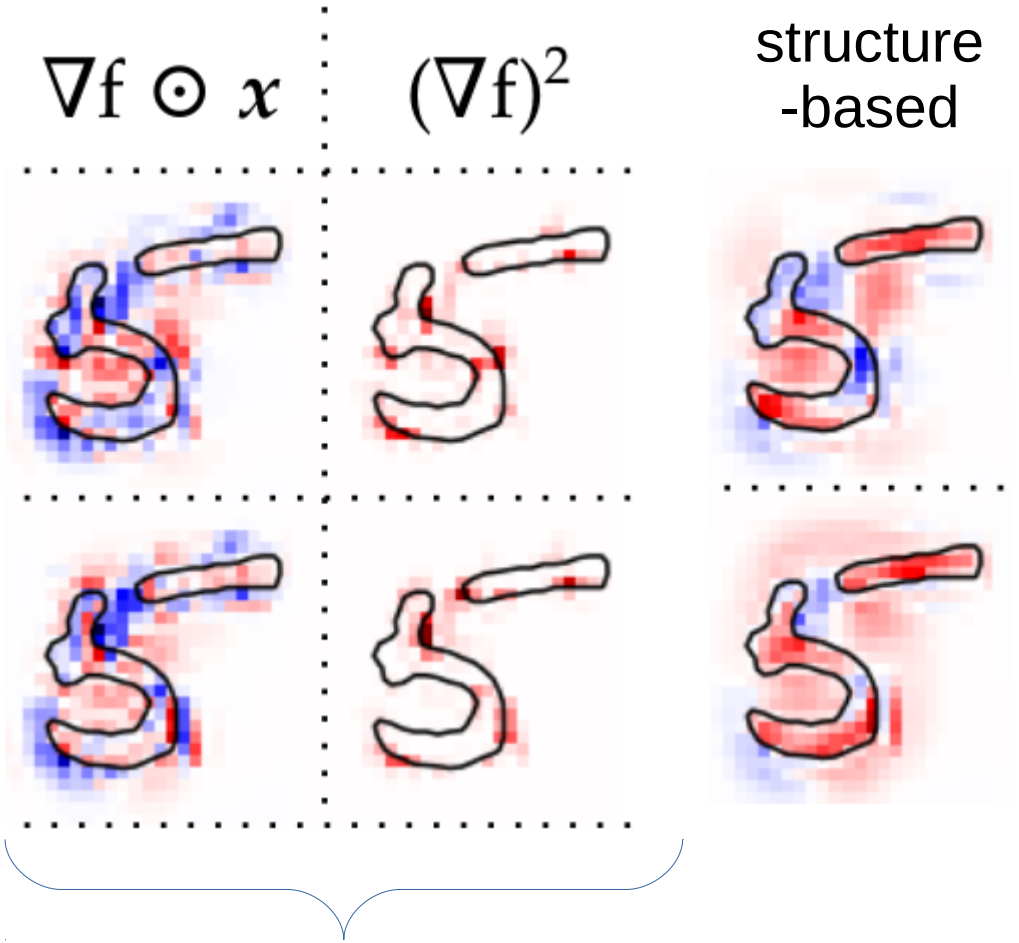
   Input that produce incorrect output can be found easily.



Local vs. Global Variations

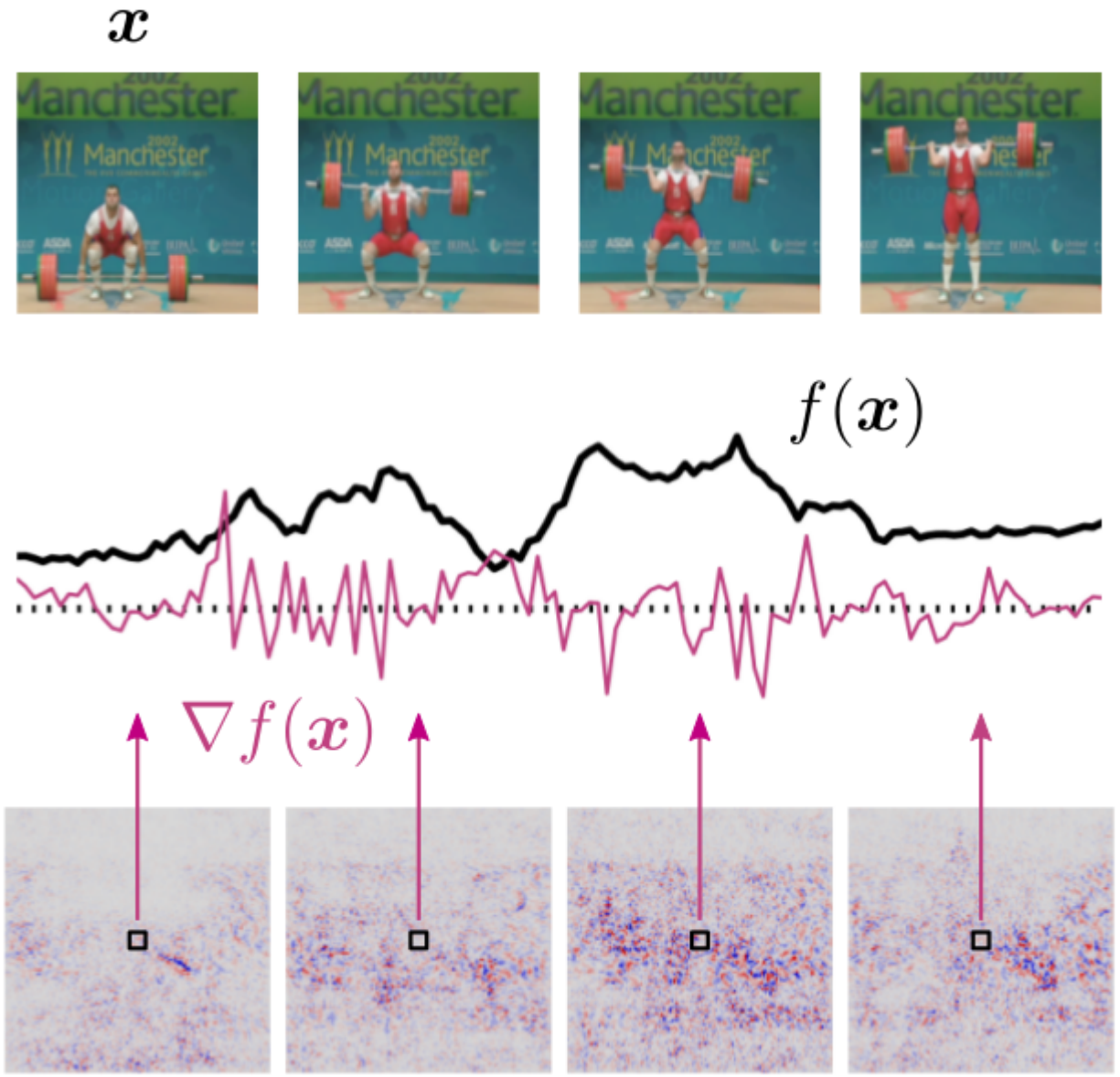


evidence for the class “5”:

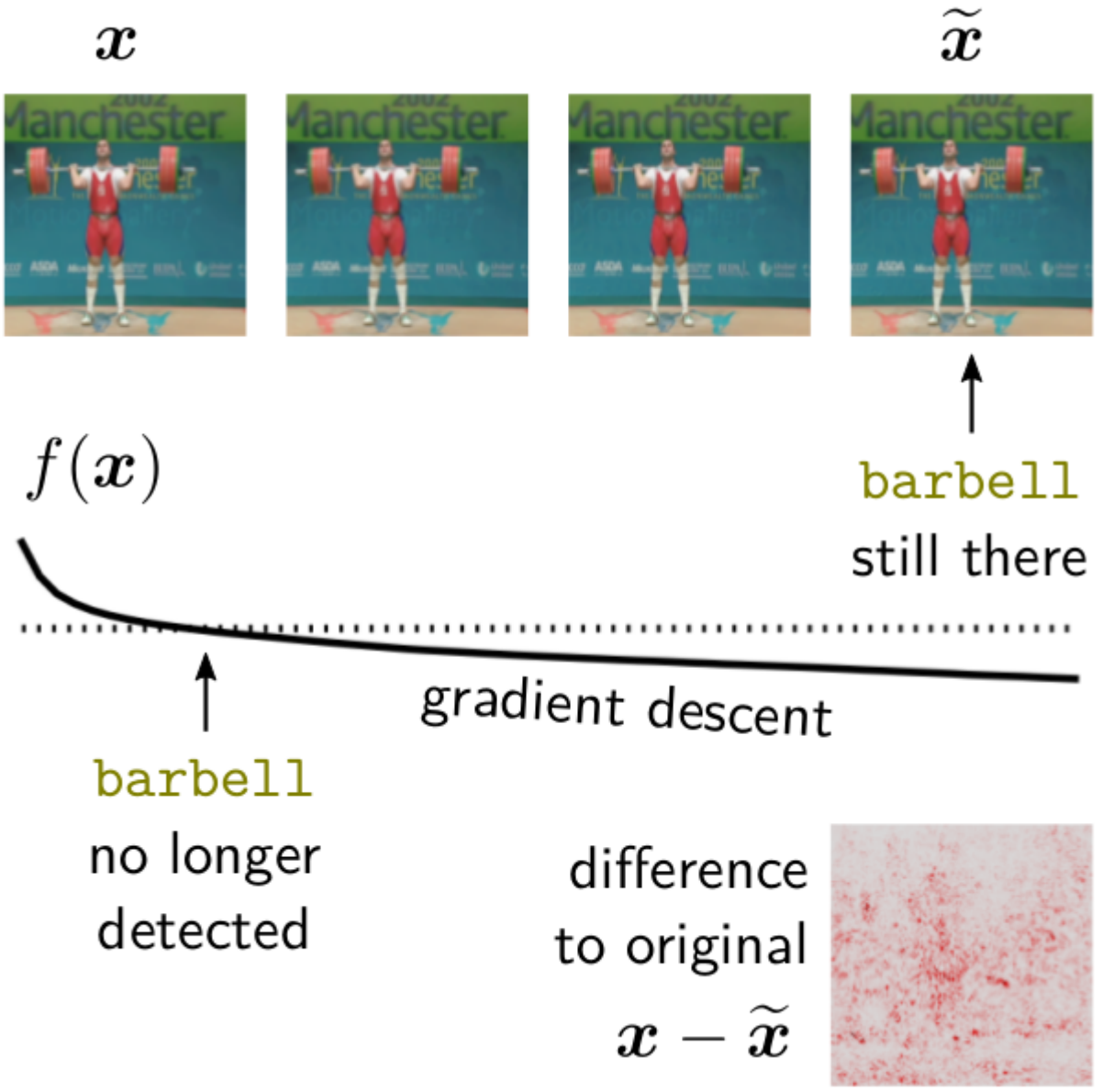


misses global factors

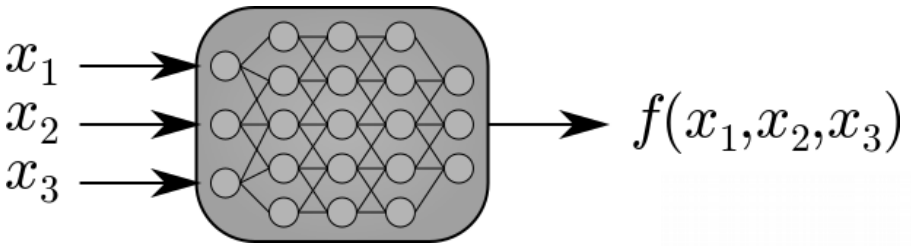
Shattered Gradients



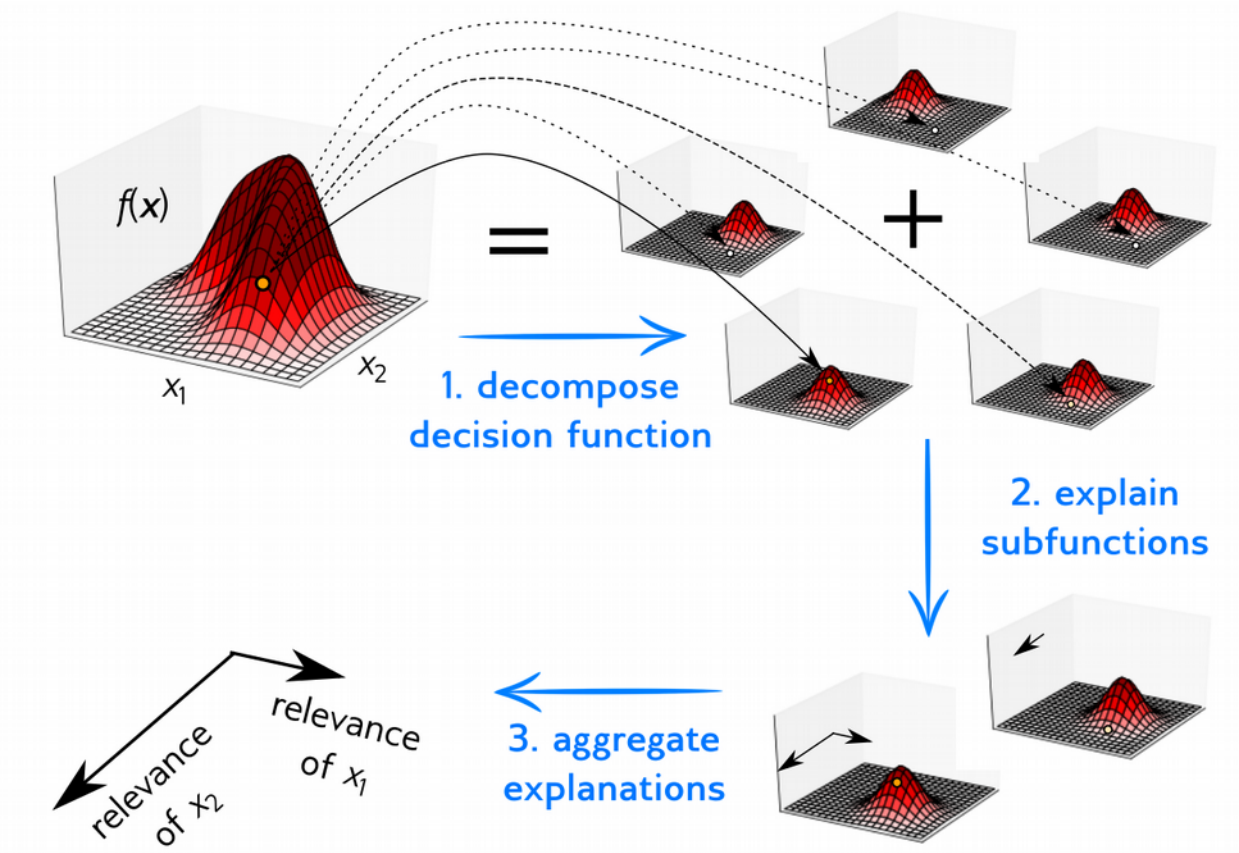
Adversarial Examples



Beyond Function: Reusing Model Structure



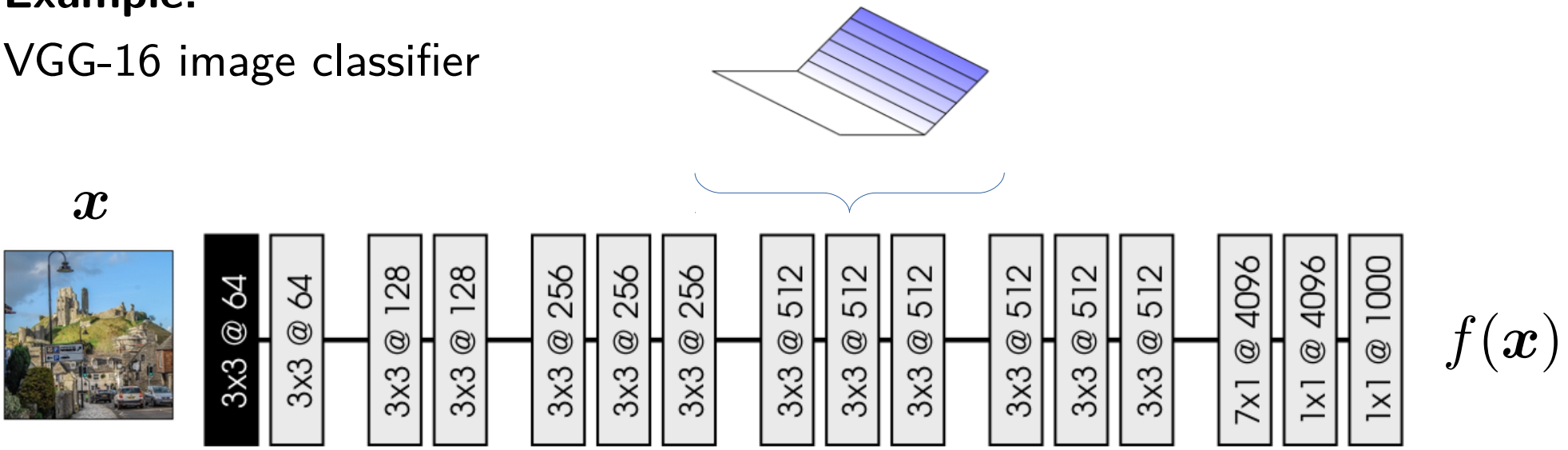
model is a composition of neurons. This can be exploited to make explanation easier.



Deep Rectifier Network Classifiers

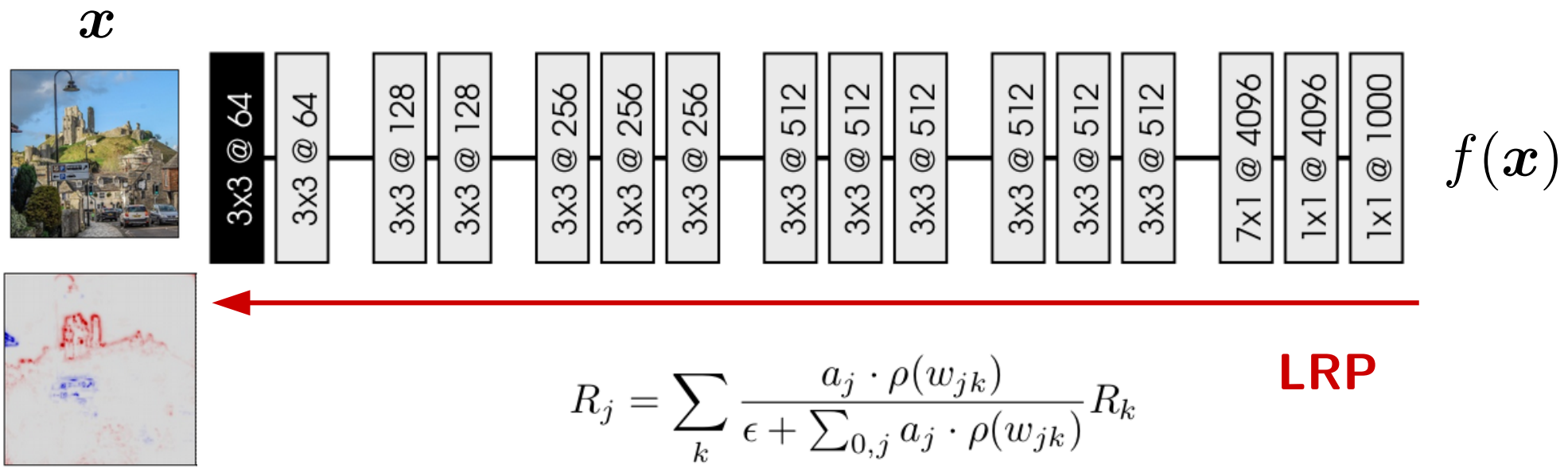
Example:

VGG-16 image classifier



structure: composition of layers consisting of many interconnected neurons.

Layer-wise Relevance Propagation



$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

LRP

Engineered backpropagation rule

G Montavon, A Binder, S Lapuschkin, W Samek, KR Müller. LRP: An Overview in Explainable AI, Springer LNCS 11700, 2019

Layer-Wise Relevance Propagation

Neuron function:

$$a_k = \max\left(0, \sum_{0,j} a_j w_{jk}\right)$$

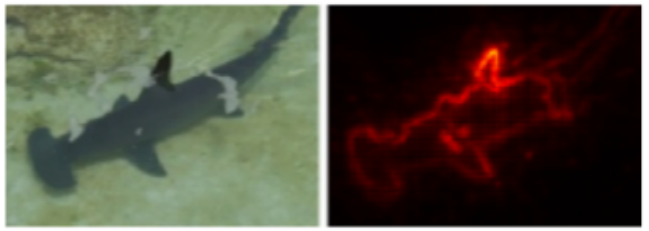
assign relevance if the neuron is
activated and if the network **reacts** to it.

Engineered
propagation rule:

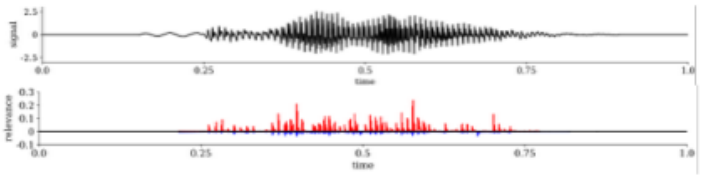
$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

LRP for Different Applications

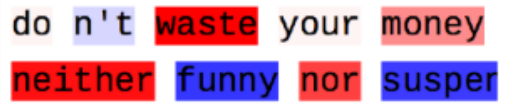
General Images (Bach' 15, Lapuschkin'16)



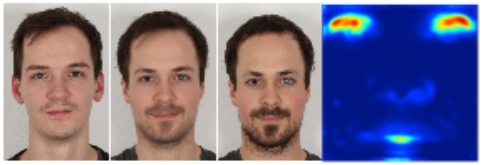
Speech (Becker'18)



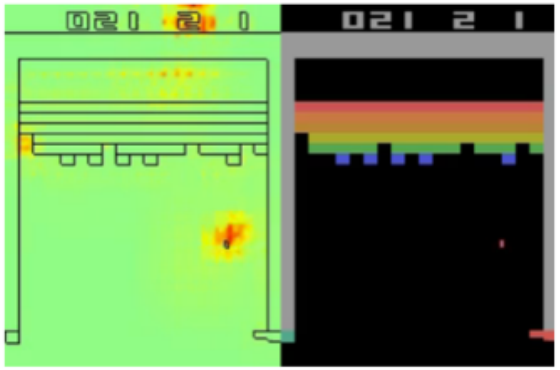
Text Analysis (Arras'16 &17)



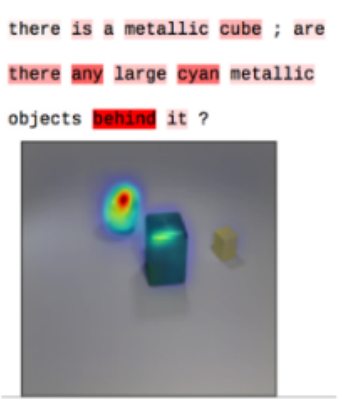
Morphing (Seibold'18)



Games (Lapuschkin'19)

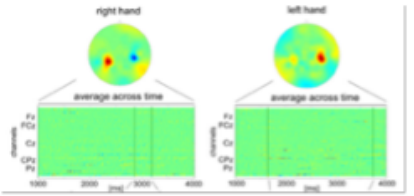


VQA (Samek'19)

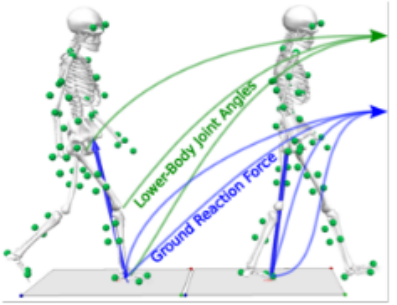


Video (Anders'18)

EEG (Sturm'16)

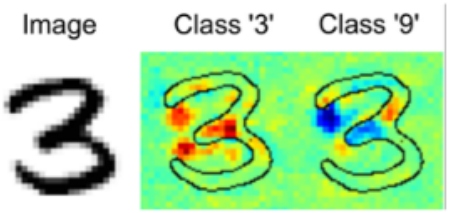


Gait Patterns (Horst'19)

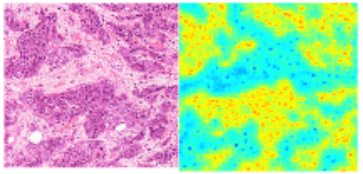


Faces (Lapuschkin'17)

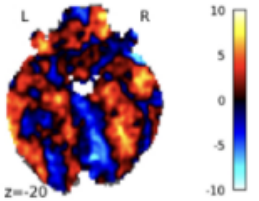
Digits (Bach' 15)



Histopathology (Binder'18)



fMRI (Thomas'18)



assign relevance if the neuron is
activated and if the network **reacts** to it.

Engineered
propagation rule:

$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

Theoretically Justified ?

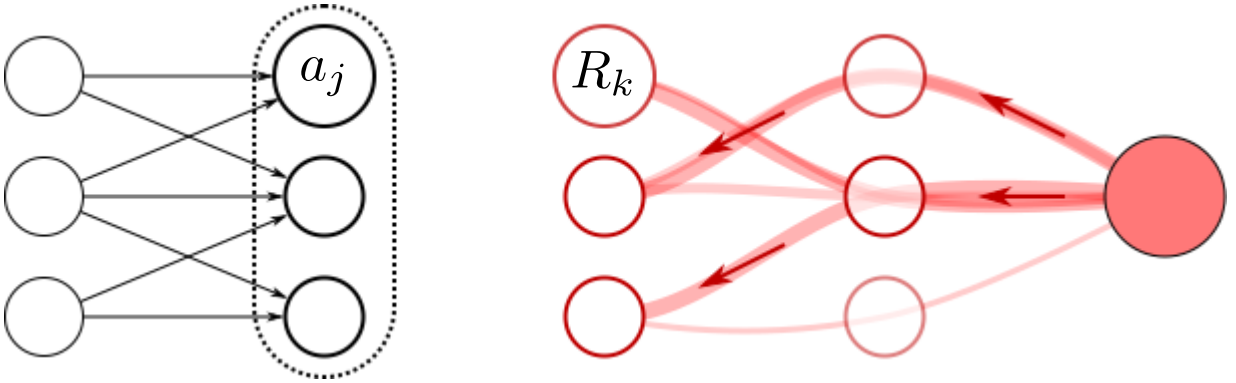
Simple Taylor Decomposition

$$\boldsymbol{x} \mapsto f(\boldsymbol{x})$$

$$f(\boldsymbol{x}) = f(\widetilde{\boldsymbol{x}}) + \sum_{i=1}^{d} [\nabla f(\widetilde{\boldsymbol{x}})]_i \cdot (x_i - \widetilde{x}_i) + \dots$$

Deep Taylor Decomposition

$$\boldsymbol{a} \mapsto R_k(\boldsymbol{a})$$

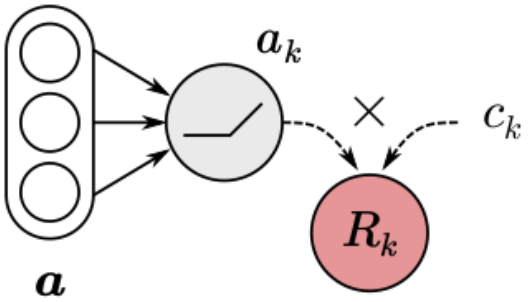


$$R_k(\boldsymbol{a}) = R_k(\widetilde{\boldsymbol{a}}) + \sum_j [\nabla R_k(\widetilde{\boldsymbol{a}})]_j \cdot (a_j - \widetilde{a}_j) + \ldots$$

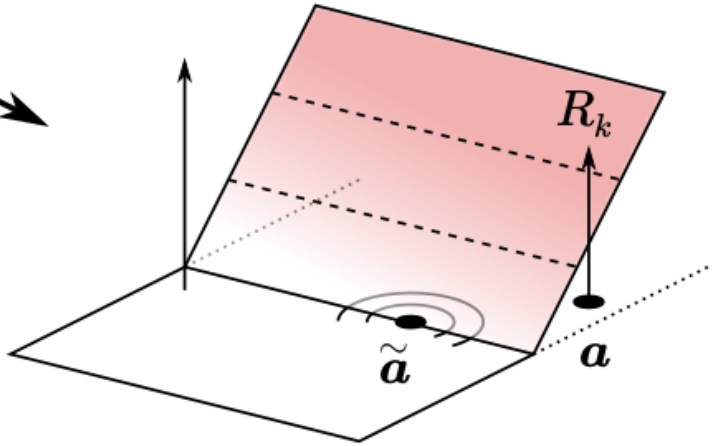
G Montavon, S Lapuschkin, A Binder, W Samek, KR Müller. Explaining NonLinear Classification Decisions with Deep Taylor Decomposition, Pattern Recognition, 65:211–222, 2017

Deep Taylor Decomposition

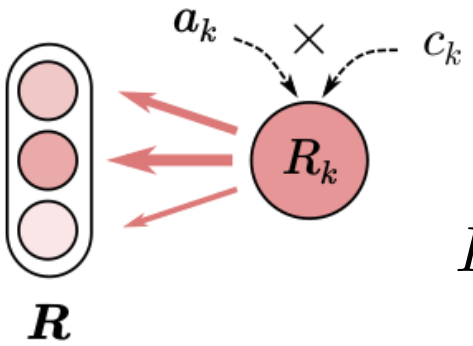
(a) DTD relevance model



(b) Taylor expansion



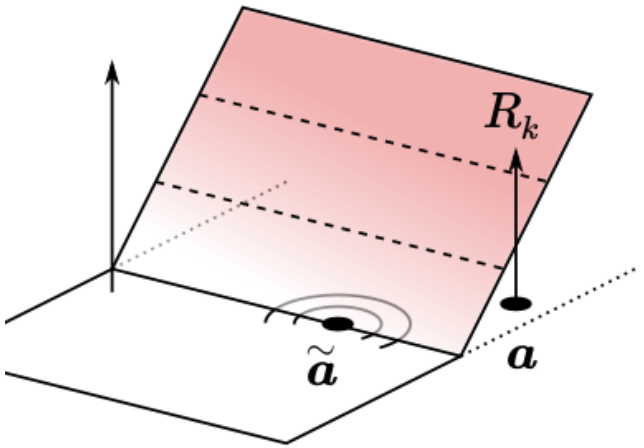
(c) relevance propagation



$$R_k(\boldsymbol{a}) = R_k(\widetilde{\boldsymbol{a}}) + \sum_j [\nabla R_k(\widetilde{\boldsymbol{a}})]_j \cdot (a_j - \widetilde{a}_j) + \ldots$$

Deep Taylor Decomposition

DTD view

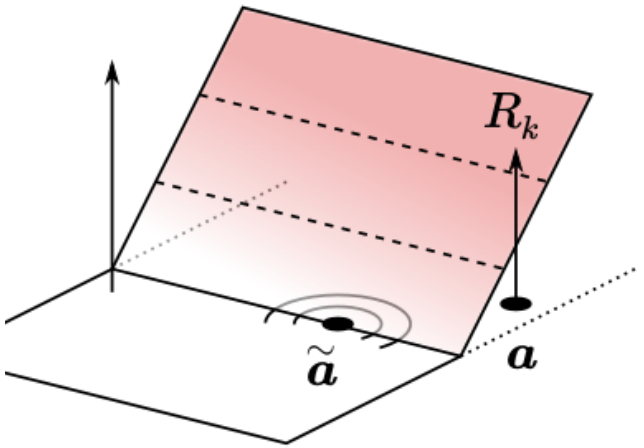


LRP view

$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

Deep Taylor Decomposition

DTD view



LRP view

$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

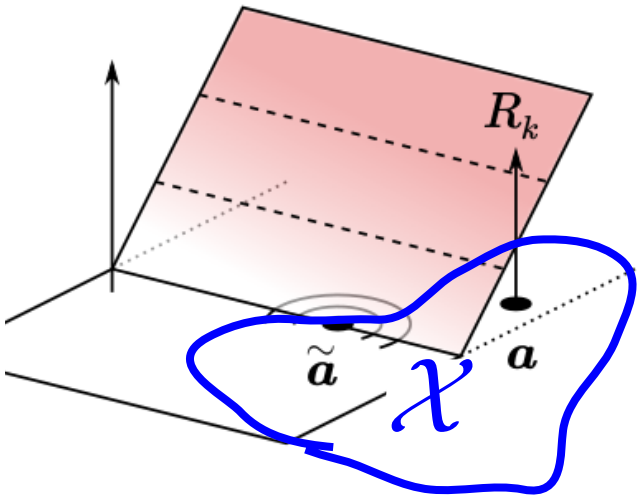
$$\widetilde{\boldsymbol{a}}^{(k)} = \mathbf{0} \quad \Longleftrightarrow \quad \rho = (\cdot), \epsilon = 0$$

$$\widetilde{\boldsymbol{a}}^{(k)} = \boldsymbol{a} - t \cdot \boldsymbol{a} \quad \Longleftrightarrow \quad \rho = (\cdot), \epsilon = (t^{-1} - 1) \cdot a_k$$

$$\widetilde{\boldsymbol{a}}^{(k)} = \boldsymbol{a} - t \cdot \boldsymbol{a} \odot \mathbf{1}_{\boldsymbol{w}_k \succ \mathbf{0}} \quad \Longleftrightarrow \quad \rho = \max(0, \cdot)$$

Deep Taylor Decomposition

DTD view



LRP view

$$R_j = \sum_k \frac{a_j \cdot \rho(w_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(w_{jk})} R_k$$

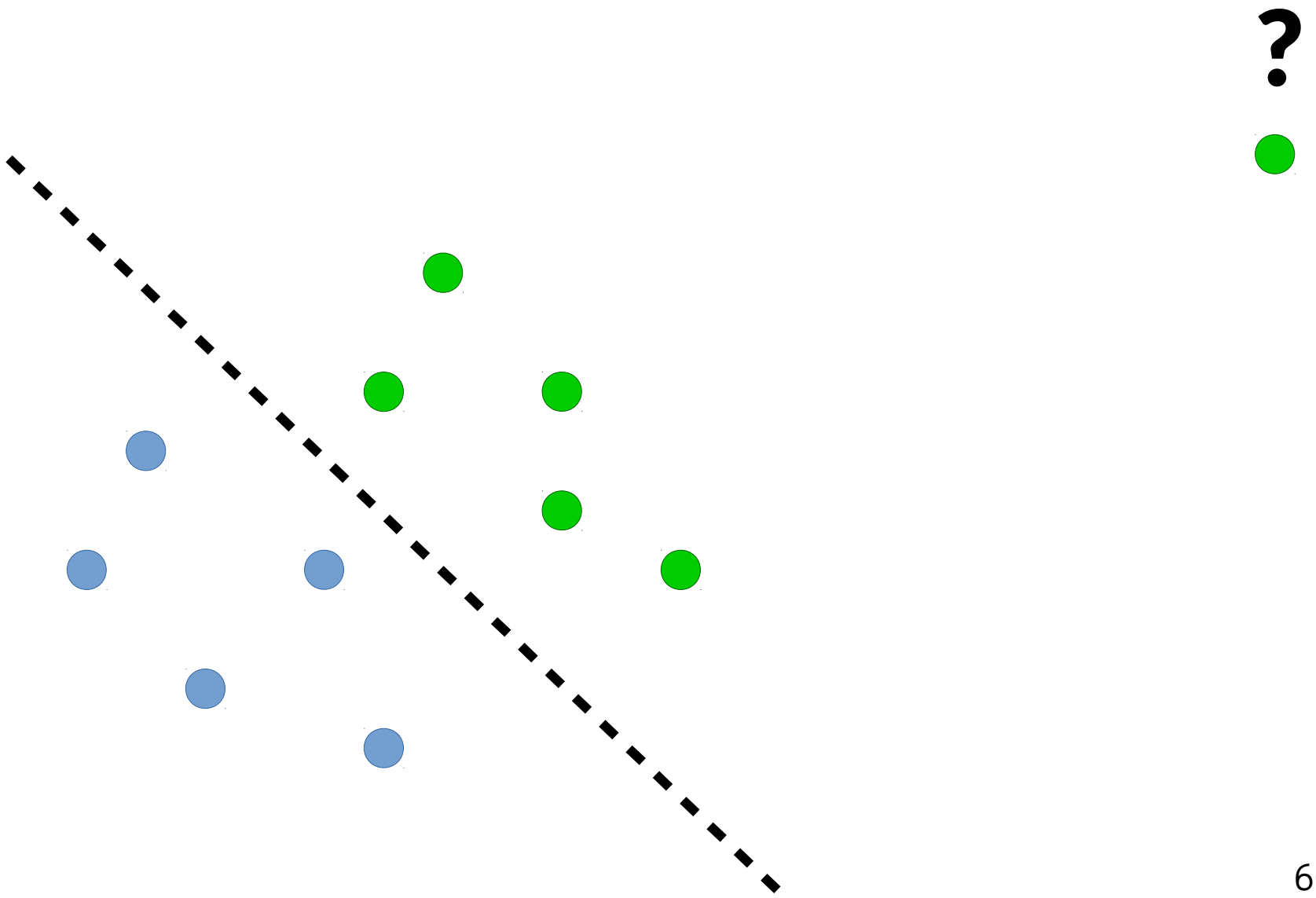
$$\widetilde{\boldsymbol{a}}^{(k)} = \mathbf{0} \quad \Longleftrightarrow \quad \rho = (\cdot), \epsilon = 0$$

$$\widetilde{\boldsymbol{a}}^{(k)} = \boldsymbol{a} - t \cdot \boldsymbol{a} \quad \Longleftrightarrow \quad \rho = (\cdot), \epsilon = (t^{-1} - 1) \cdot a_k$$

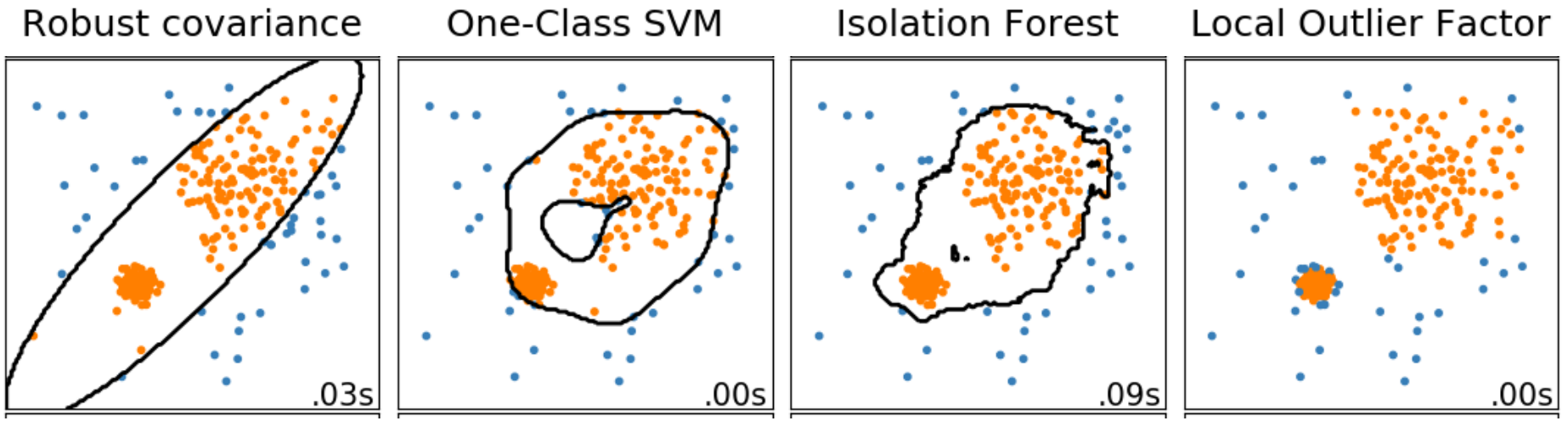
$$\widetilde{\boldsymbol{a}}^{(k)} = \boldsymbol{a} - t \cdot \boldsymbol{a} \odot \mathbf{1}_{\boldsymbol{w}_k \succ \mathbf{0}} \quad \Longleftrightarrow \quad \rho = \max(0, \cdot)$$

Beyond DNN Classifiers

Anomaly Detection



Anomaly Detection



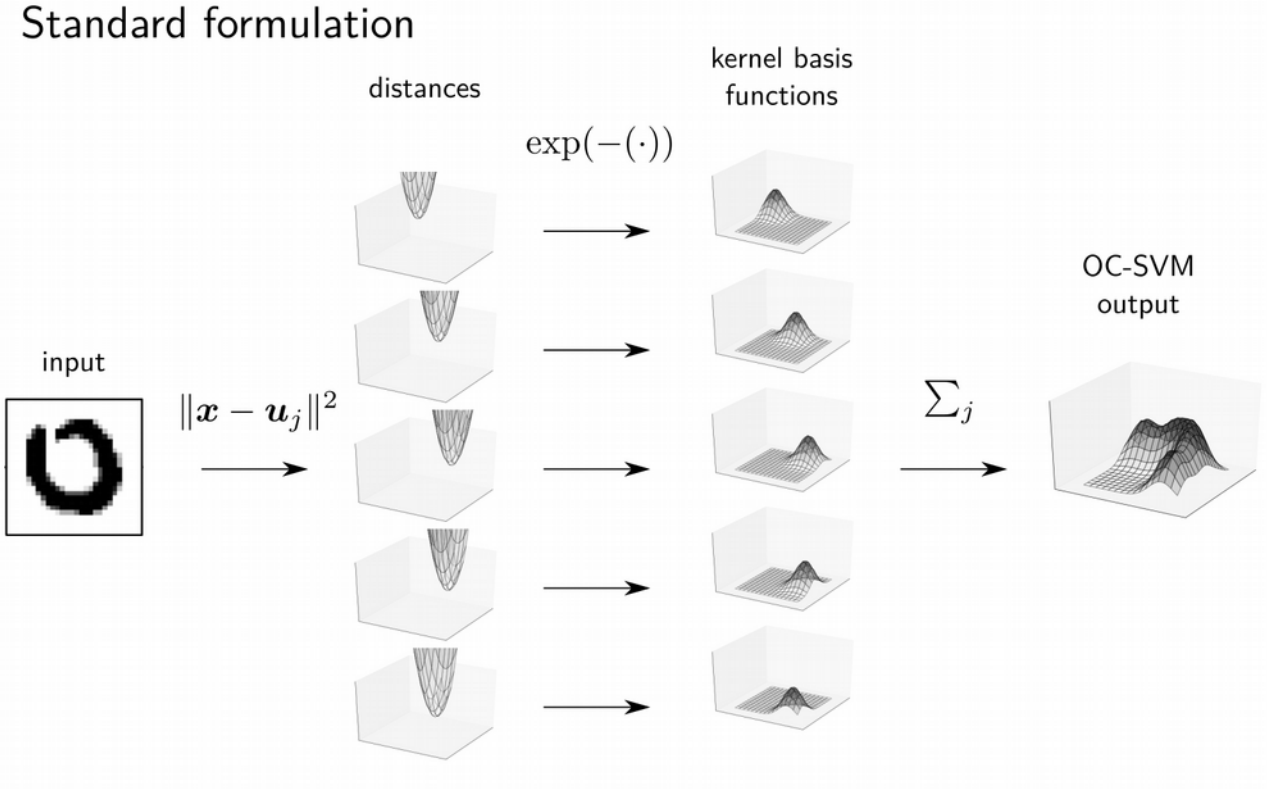
Source: scikit-learn.org

Traditionally solved with, kernels, decision trees, **not** neural networks.

Question:

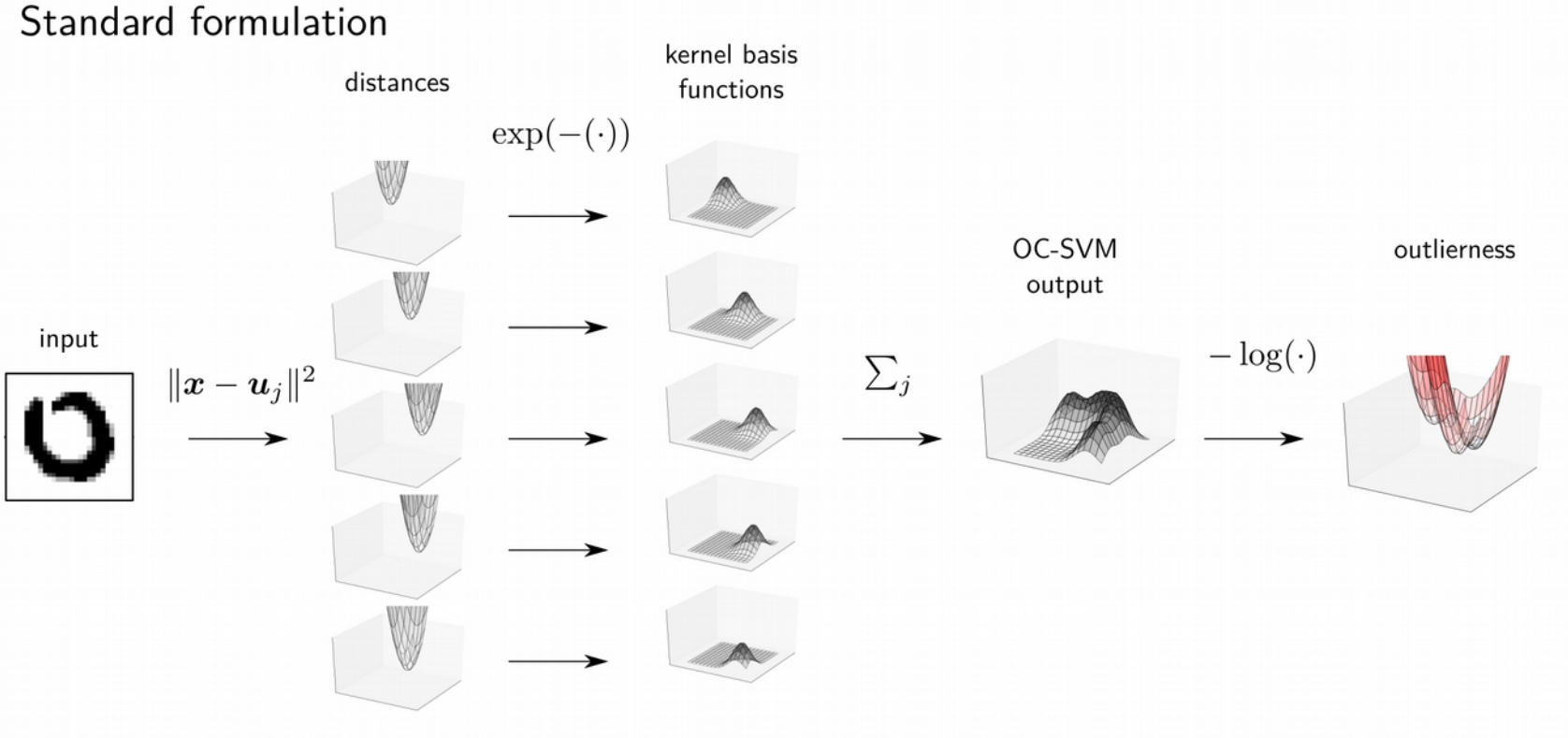
Can we still extract a structure to guide the process of explanation?

One-Class SVMs



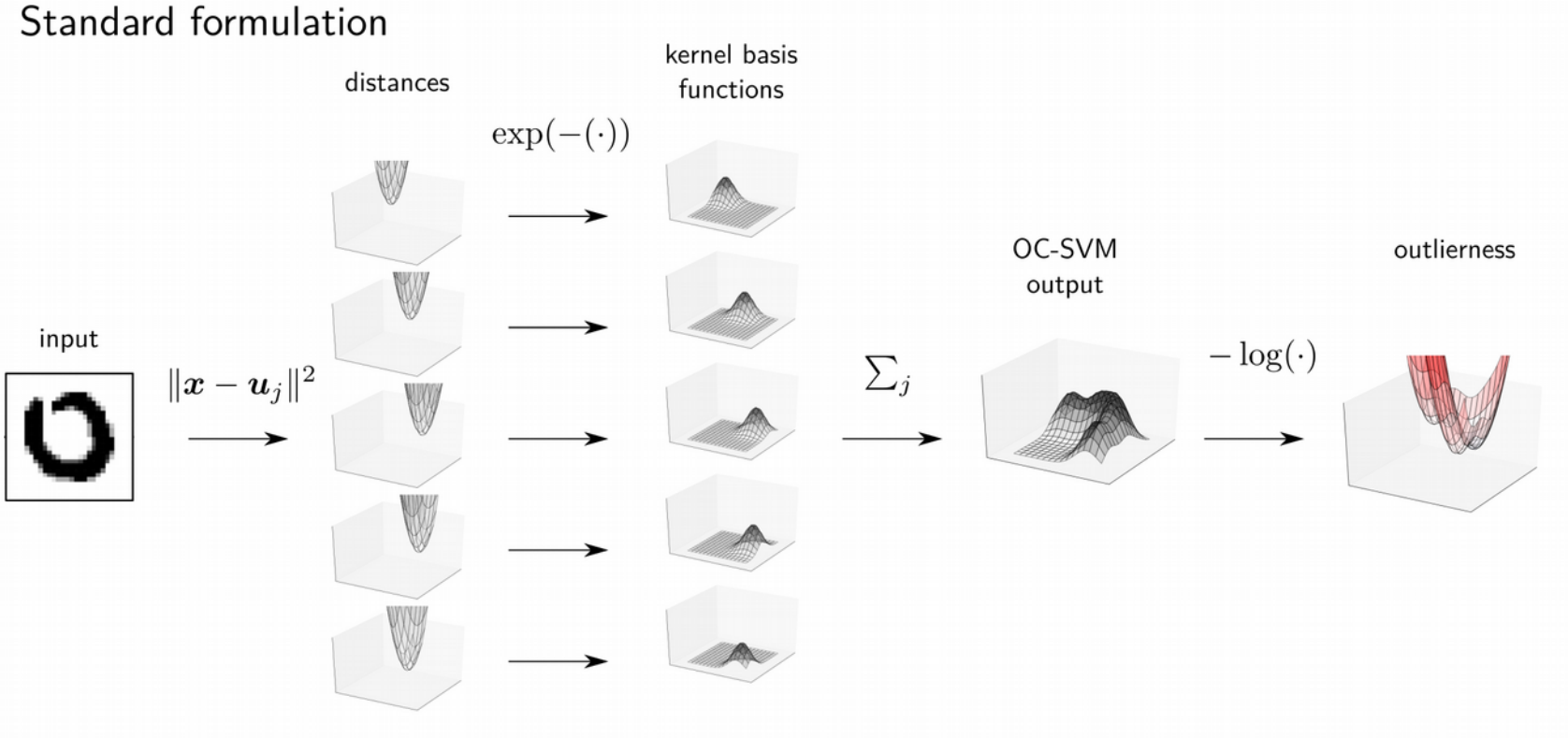
J Kauffmann, KR Müller, G Montavon. Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models Pattern Recognition, 107198, 2020

One-Class SVMs



J Kauffmann, KR Müller, G Montavon. Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models Pattern Recognition, 107198, 2020

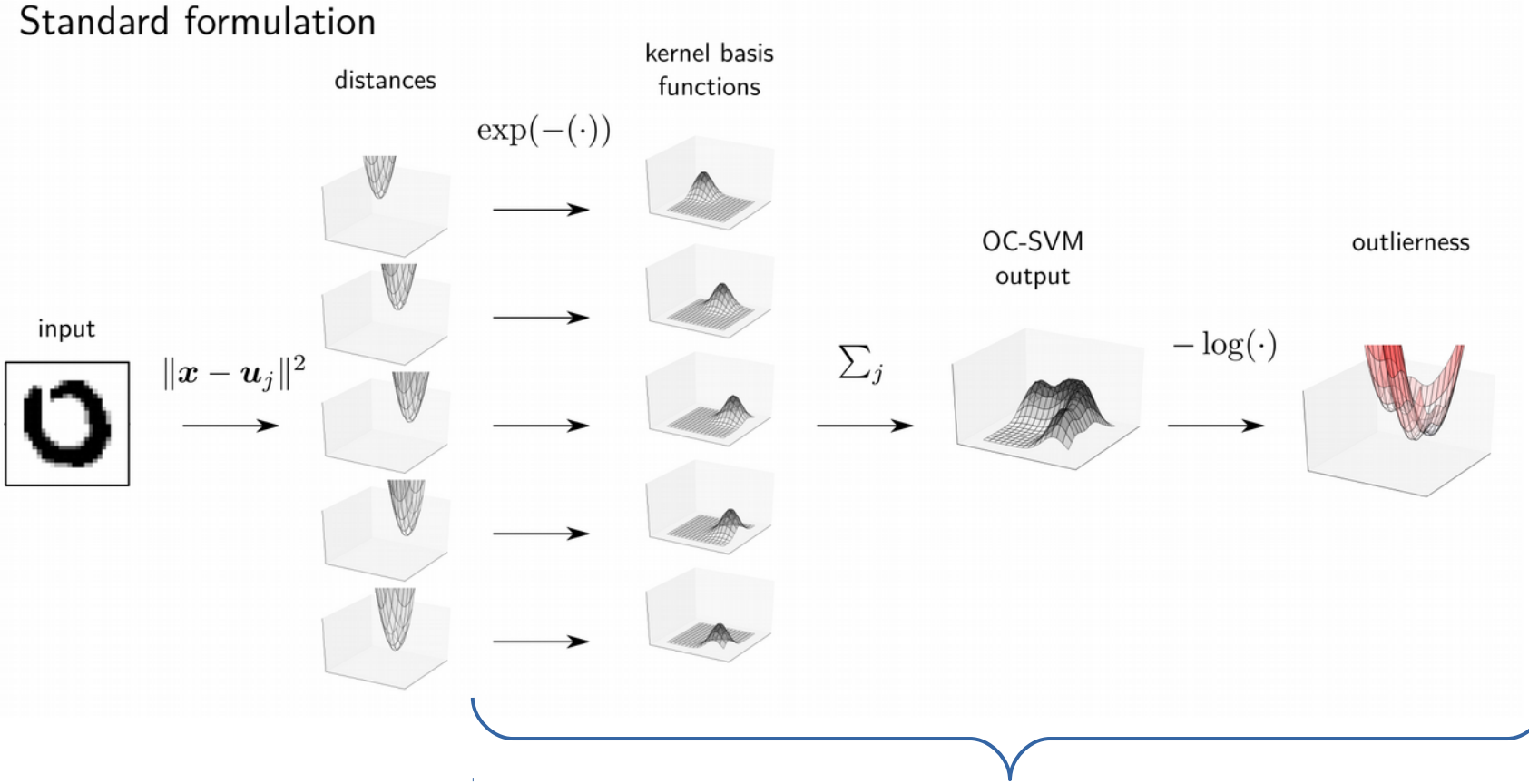
One-Class SVMs



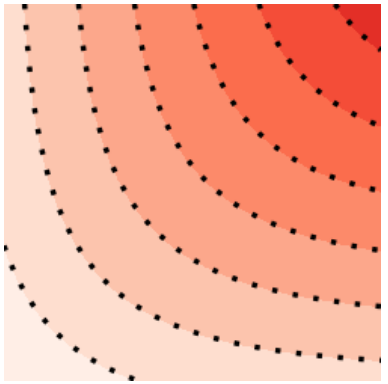
Problem: Most mappings are nonlinear $\rightarrow$ hard to design a LRP propagation scheme that can be justified in terms of Taylor expansions.

J Kauffmann, KR Müller, G Montavon. Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models Pattern Recognition, 107198, 2020

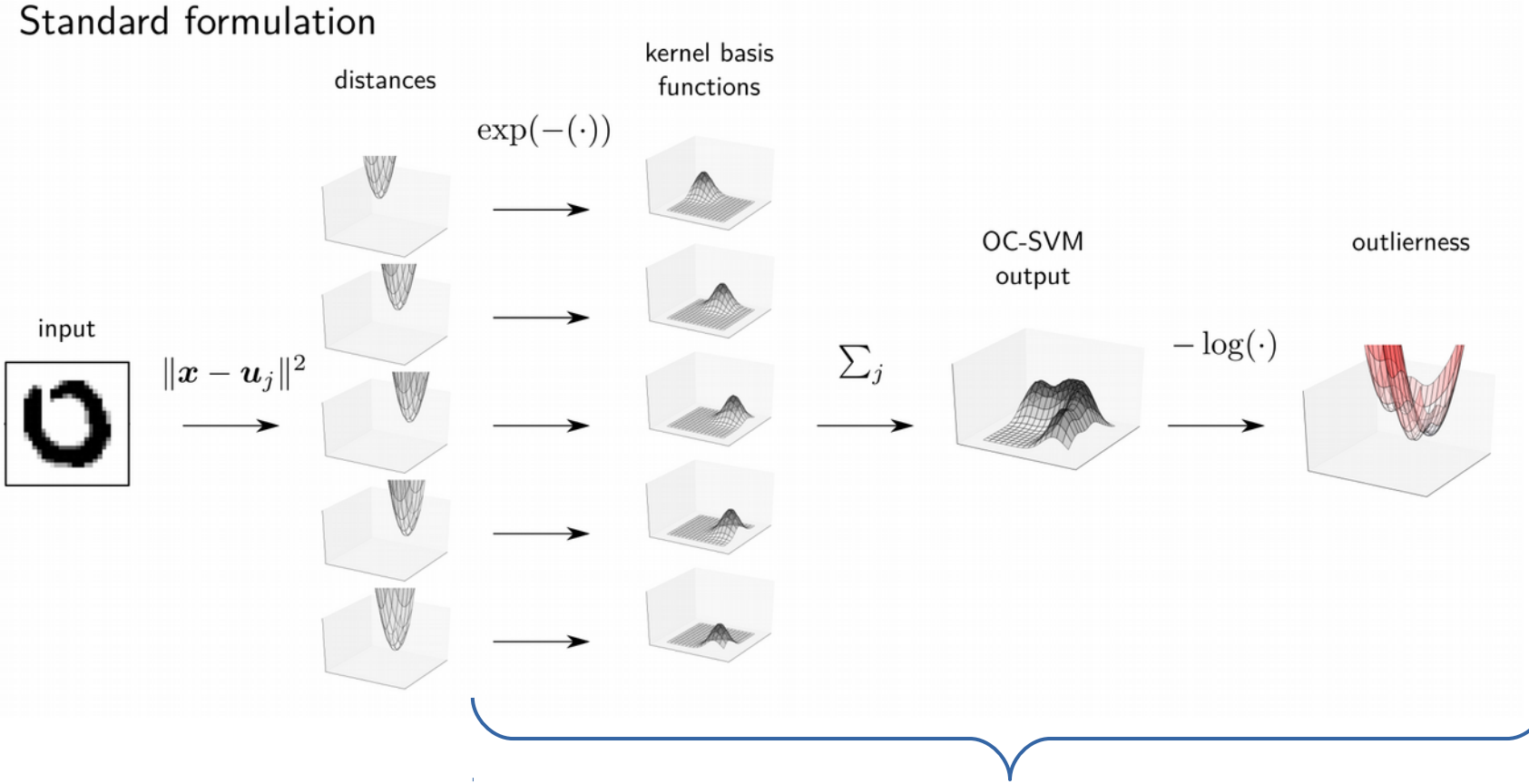
One-Class SVMs



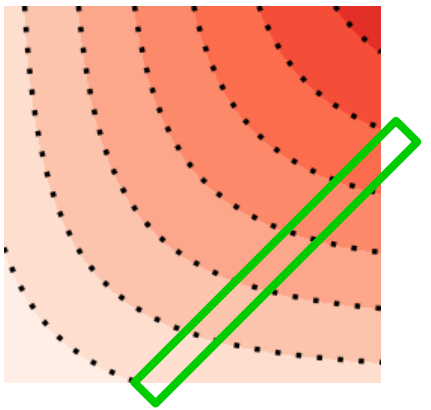
"LogSumExp" pooling



One-Class SVMs



"LogSumExp" pooling



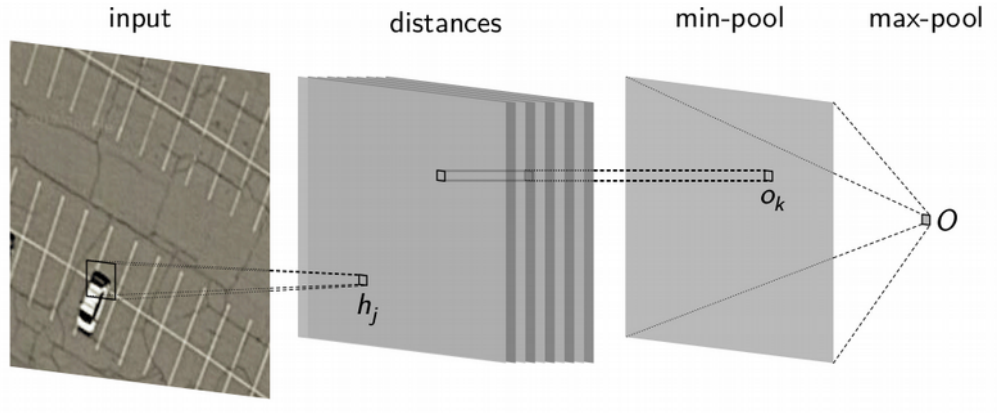
locally linear!

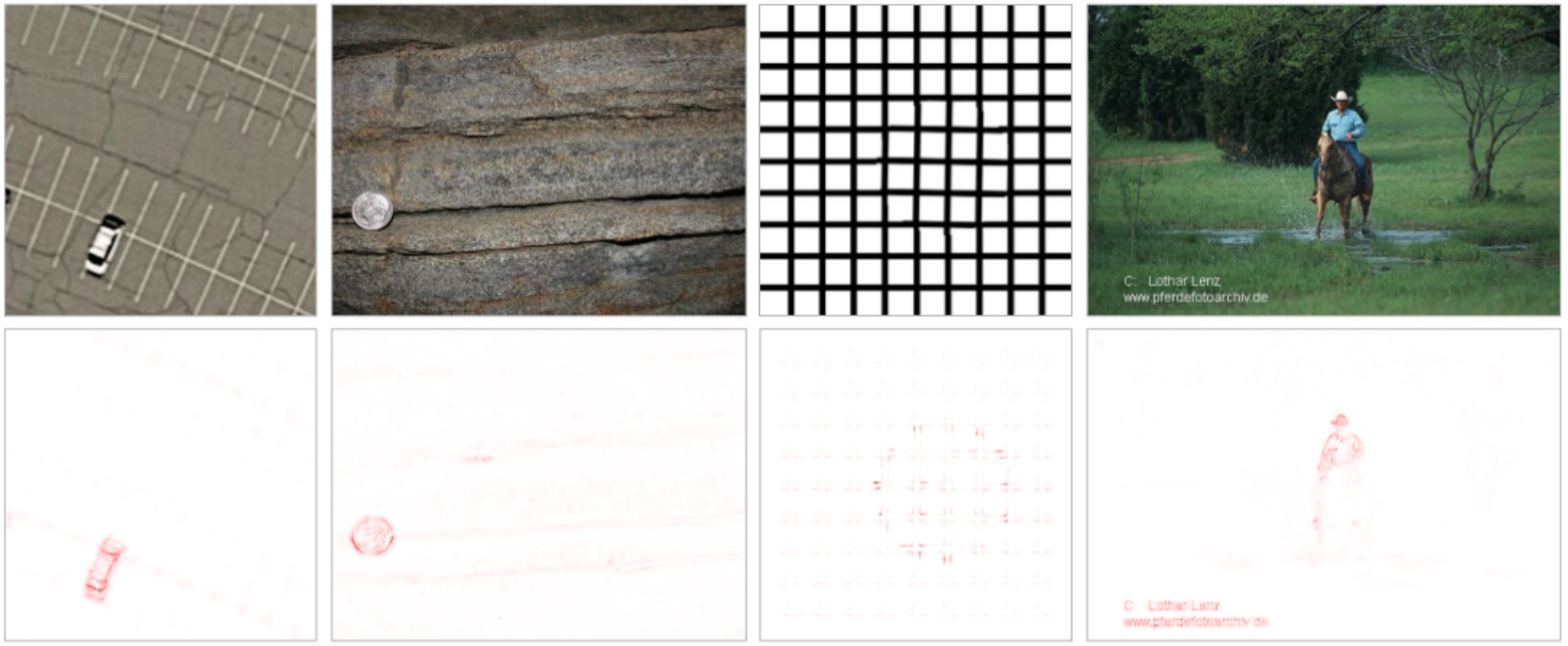
Examples of Explained Anomalies

"Neuralized" LRP can explain a variety of anomalies (stroke artefacts, digits artefacts, ...).

J Kauffmann, KR Müller, G Montavon. Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models Pattern Recognition, 107198, 2020

Examples of Explained Anomalies

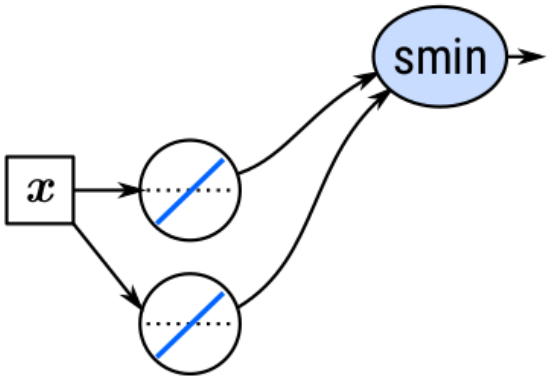




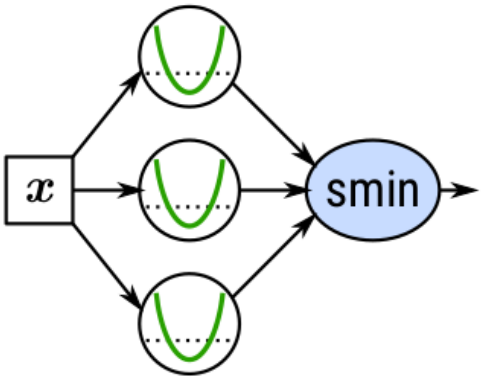
J Kauffmann, KR Müller, G Montavon. Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models Pattern Recognition, 107198, 2020

Beyond DNN Classifiers (Overview)

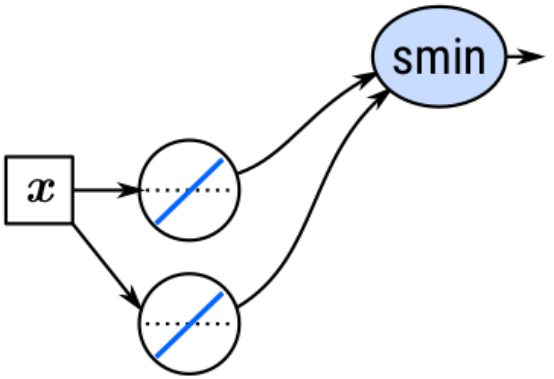
(i) Logistic Classifier



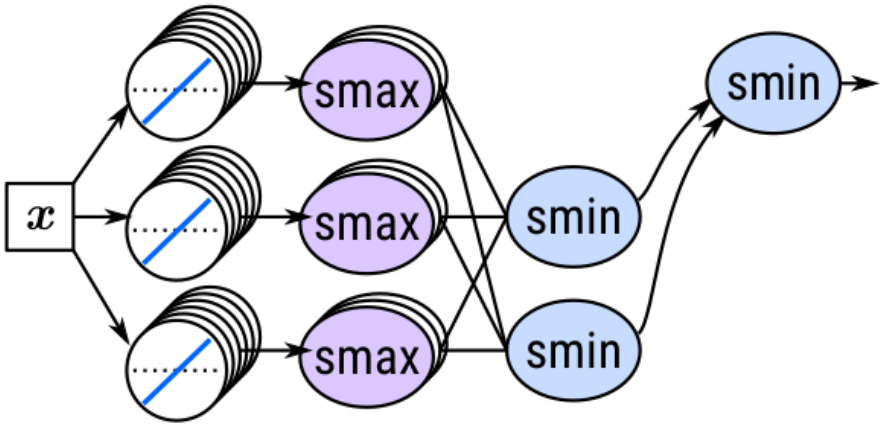
(ii) One-Class SVMs

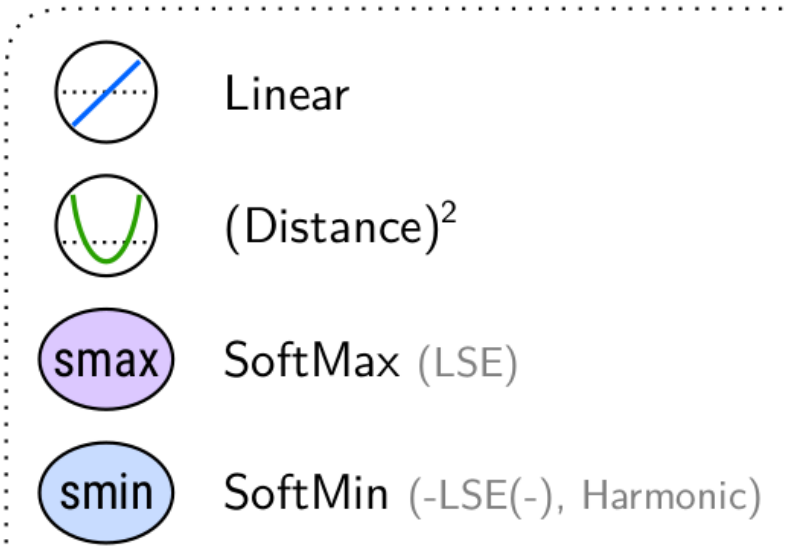


(iii) K-Means



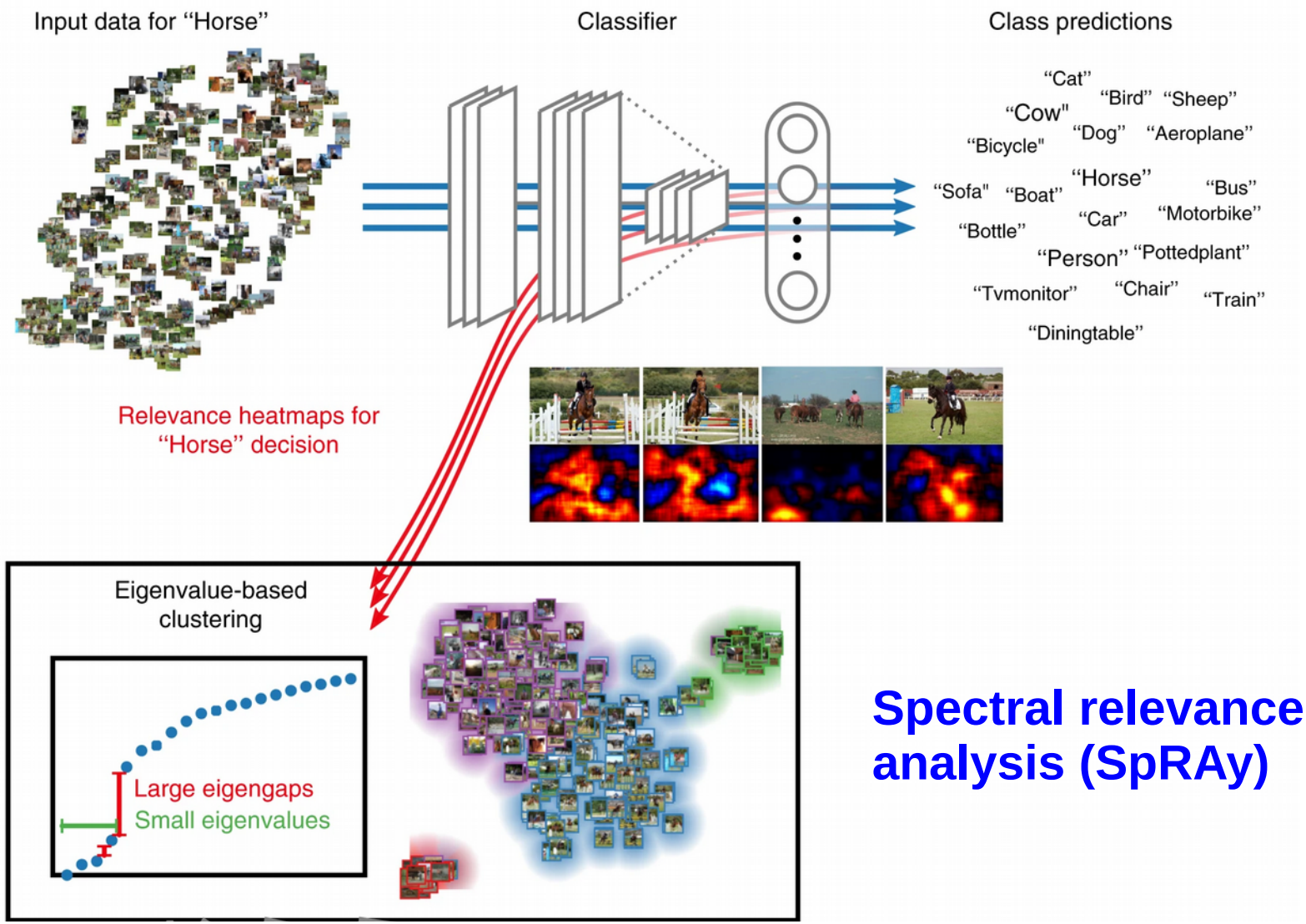
(iv) Kernel K-Means





Dataset-Wide Analyses

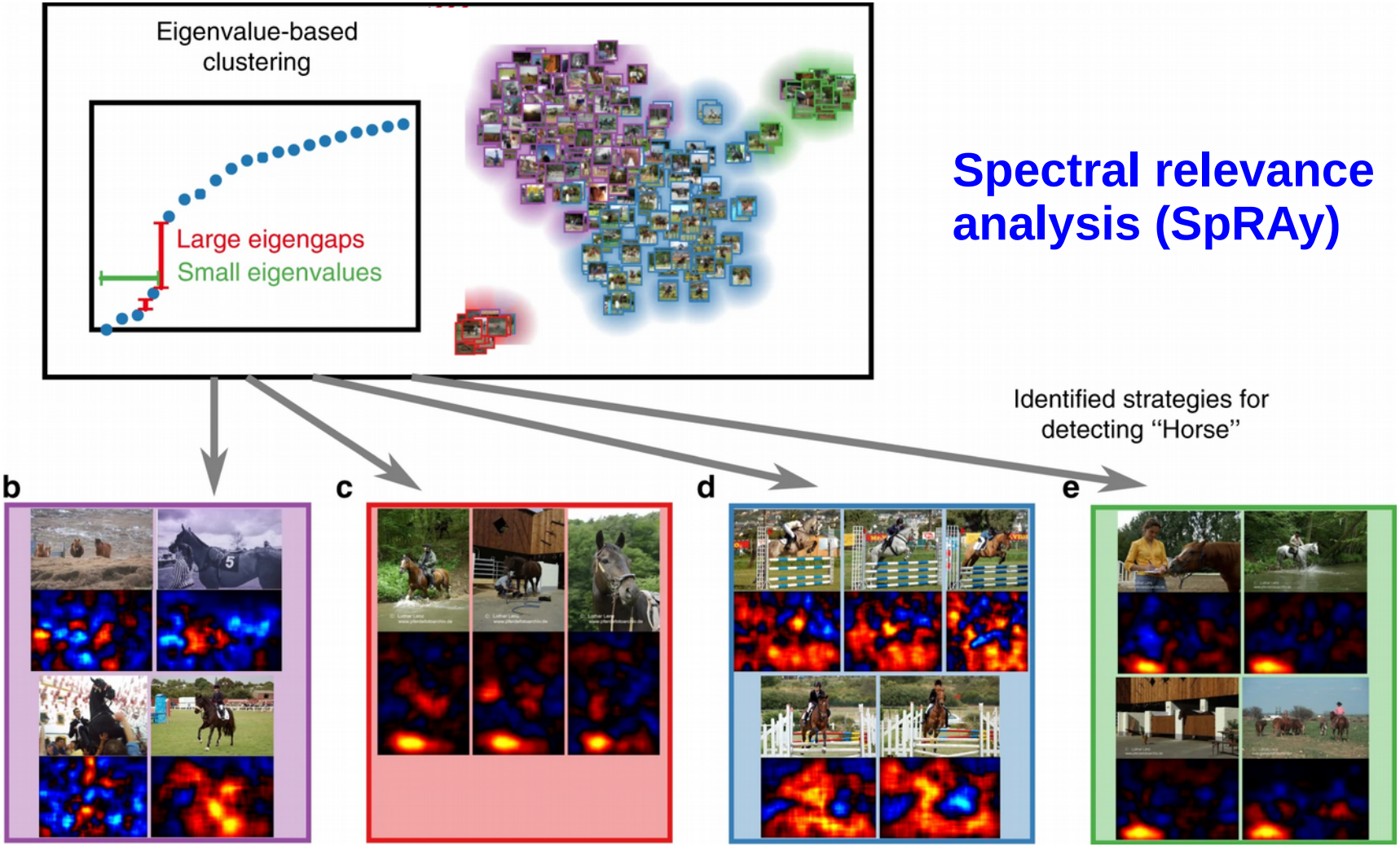
SpRAy (Spectral Relevance Analysis)



Spectral relevance analysis (SpRAy)

S Lapuschkin, S Wäldchen, A Binder, G Montavon, W Samek, KR Müller. Unmasking Clever Hans Predictors and Assessing What Machines Really Learn
Nature Communications, 10:1096, 2019

SpRAy (Spectral Relevance Analysis)



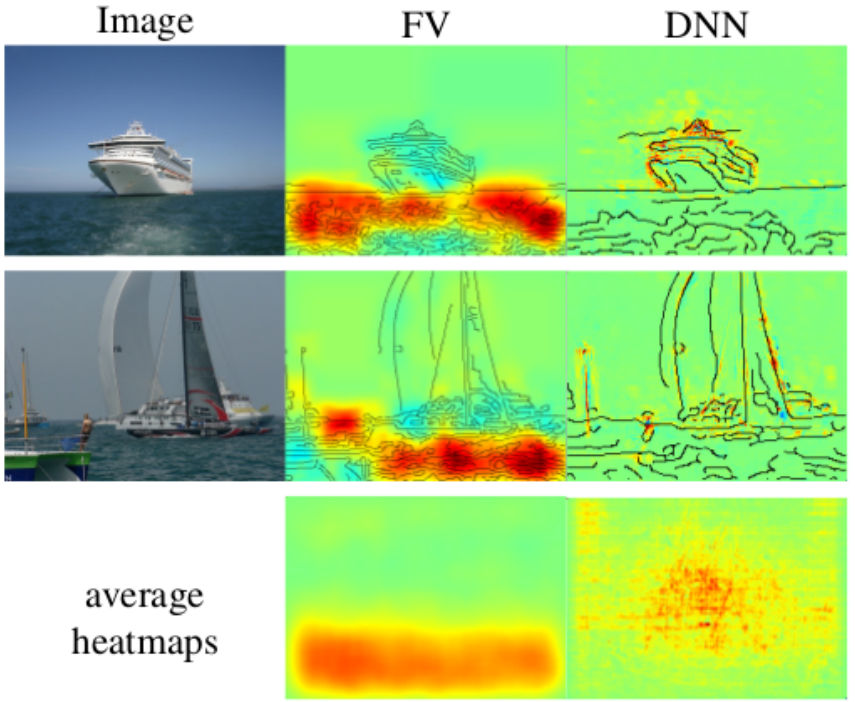
Spectral relevance analysis (SpRAy)

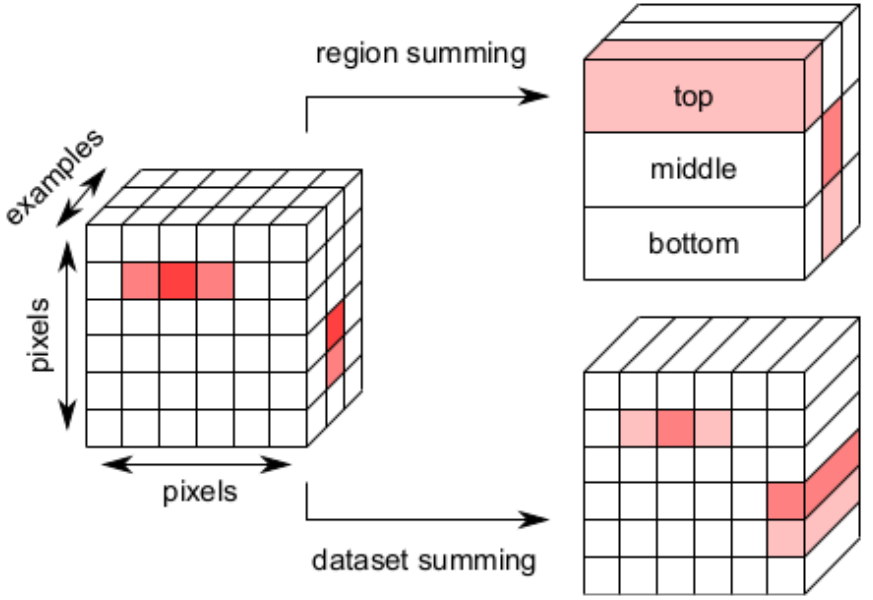
S Lapuschkin, S Wäldchen, A Binder, G Montavon, W Samek, KR Müller. Unmasking Clever Hans Predictors and Assessing What Machines Really Learn
Nature Communications, 10:1096, 2019

Relevance Pooling

Idea: analyze the overall classifier behavior by pooling relevance in pixel and dataset space.

This analysis is possible due to the conservation property of LRP.





S Lapuschkin, A Binder, G Montavon, KR Müller, W Samek. Analyzing Classifiers: Fisher Vectors and Deep Neural Networks, IEEE CVPR, 2912-2920, 2016

Application: Quantifying Context Use

how important is context ?

how important is context ?

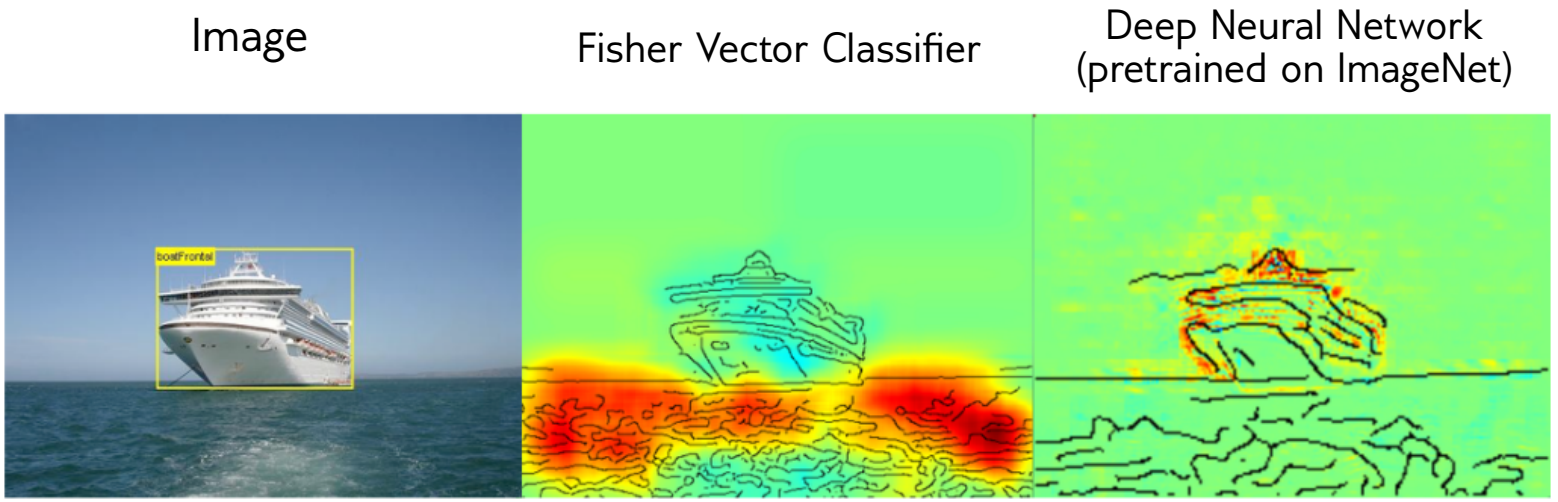
classifier

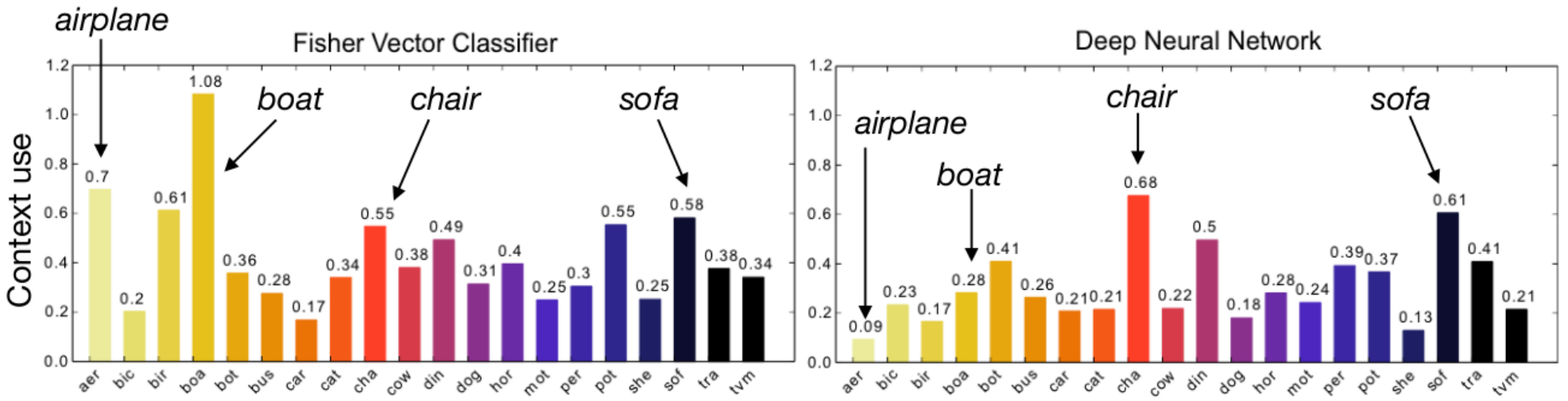
LRP decomposition allows meaningful pooling over bbox !

$$\sum_i R_i = f(x)$$

$$\text{importance of context} = \frac{\text{relevance outside bbox}}{\text{relevance inside bbox}}$$

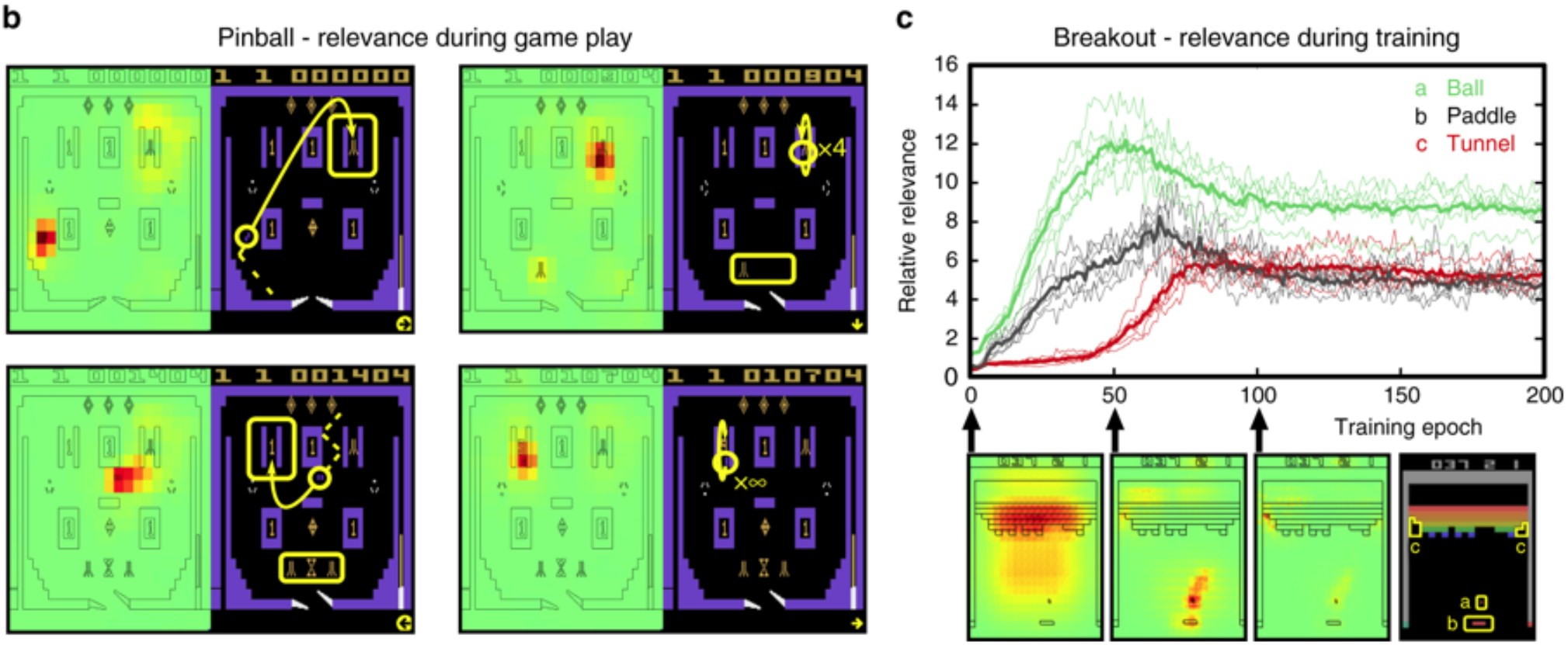
Application: Quantifying Context Use





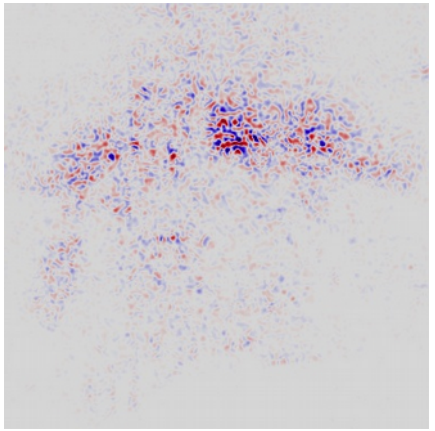
S Lapuschkin, A Binder, G Montavon, KR Müller, W Samek. Analyzing Classifiers: Fisher Vectors and Deep Neural Networks, IEEE CVPR, 2912-2920, 2016

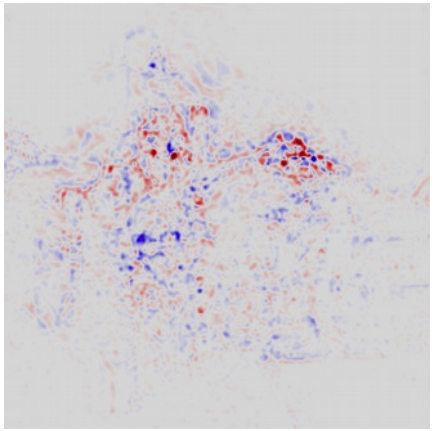
Application: Monitoring RL Strategies

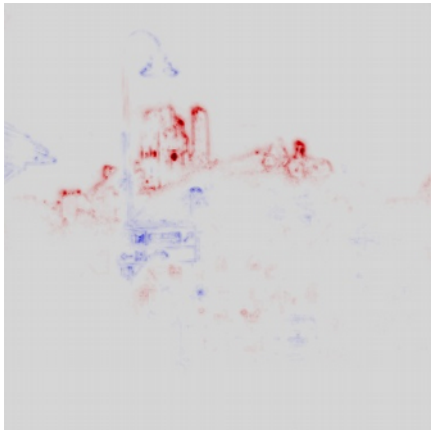


Lapuschkin et al. Unmasking Clever Hans Predictors and Assessing What Machines Really Learn. Nature Communications, 2019

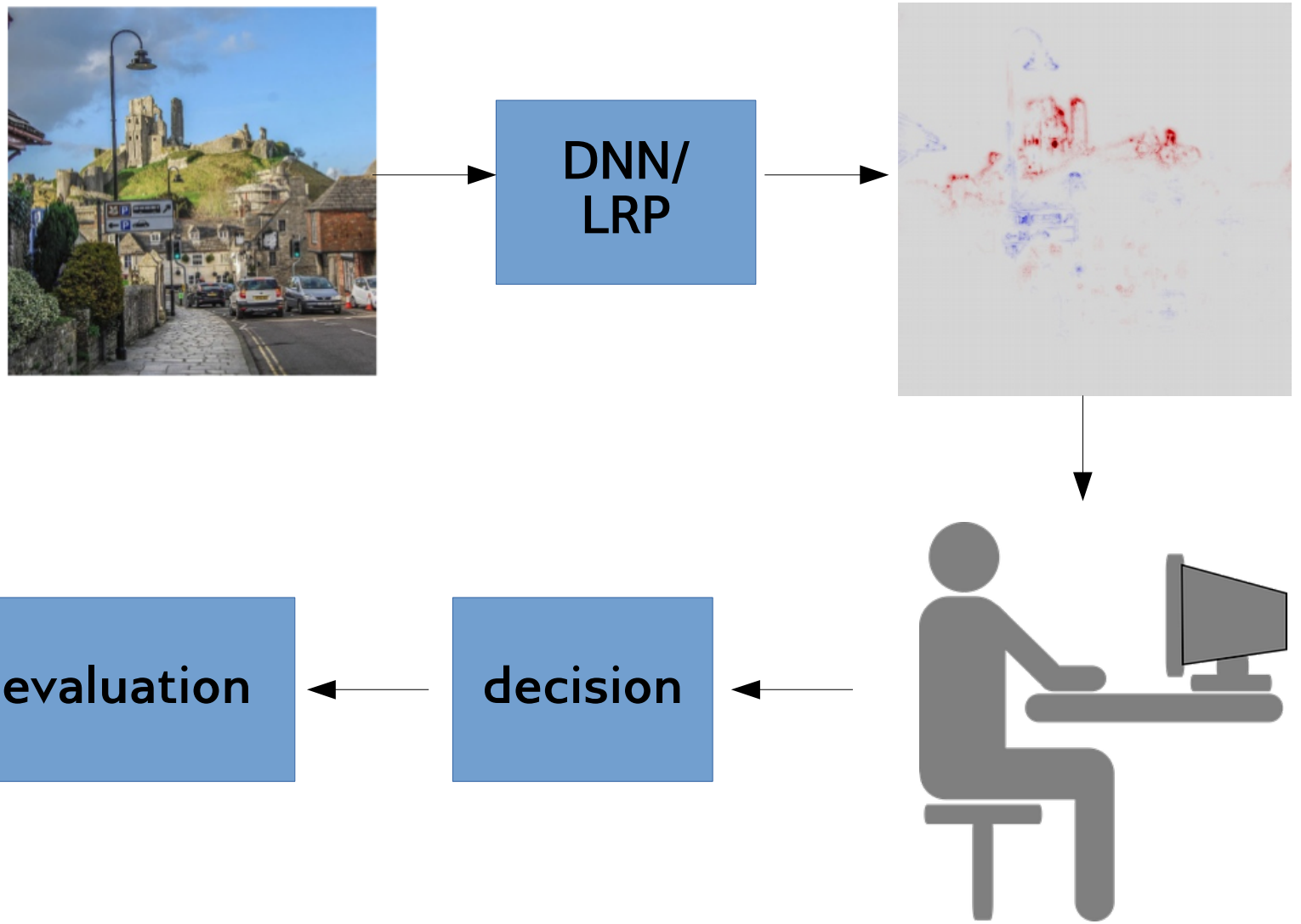
How to Select the Best Explanation?







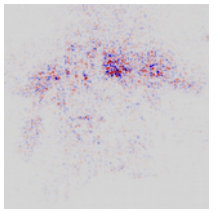
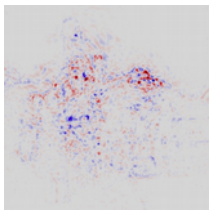
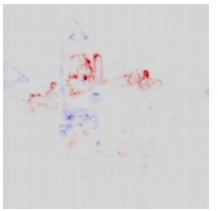
Approach 1: End-to-End Evaluation



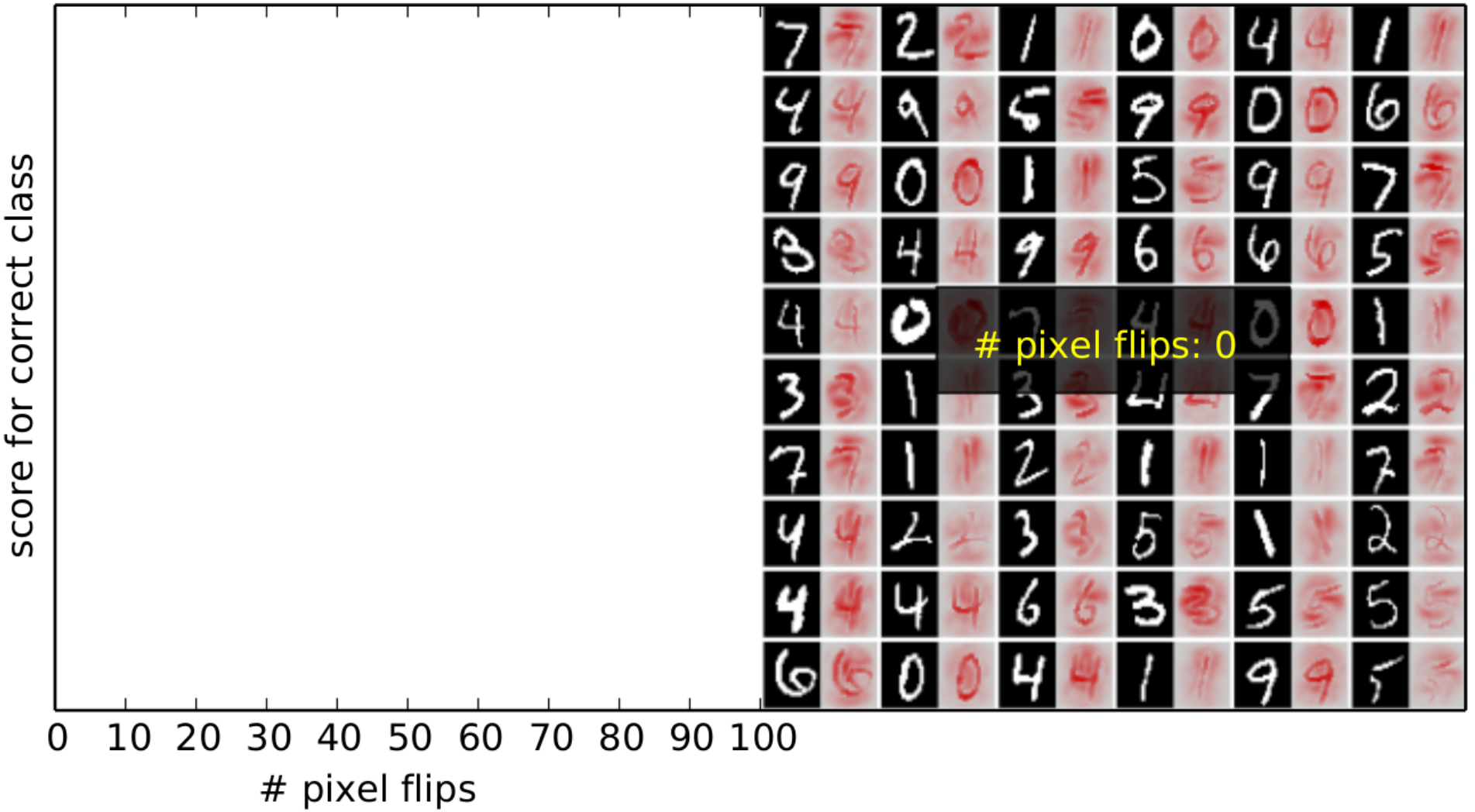
Approach 2: Axioms

candidate methods

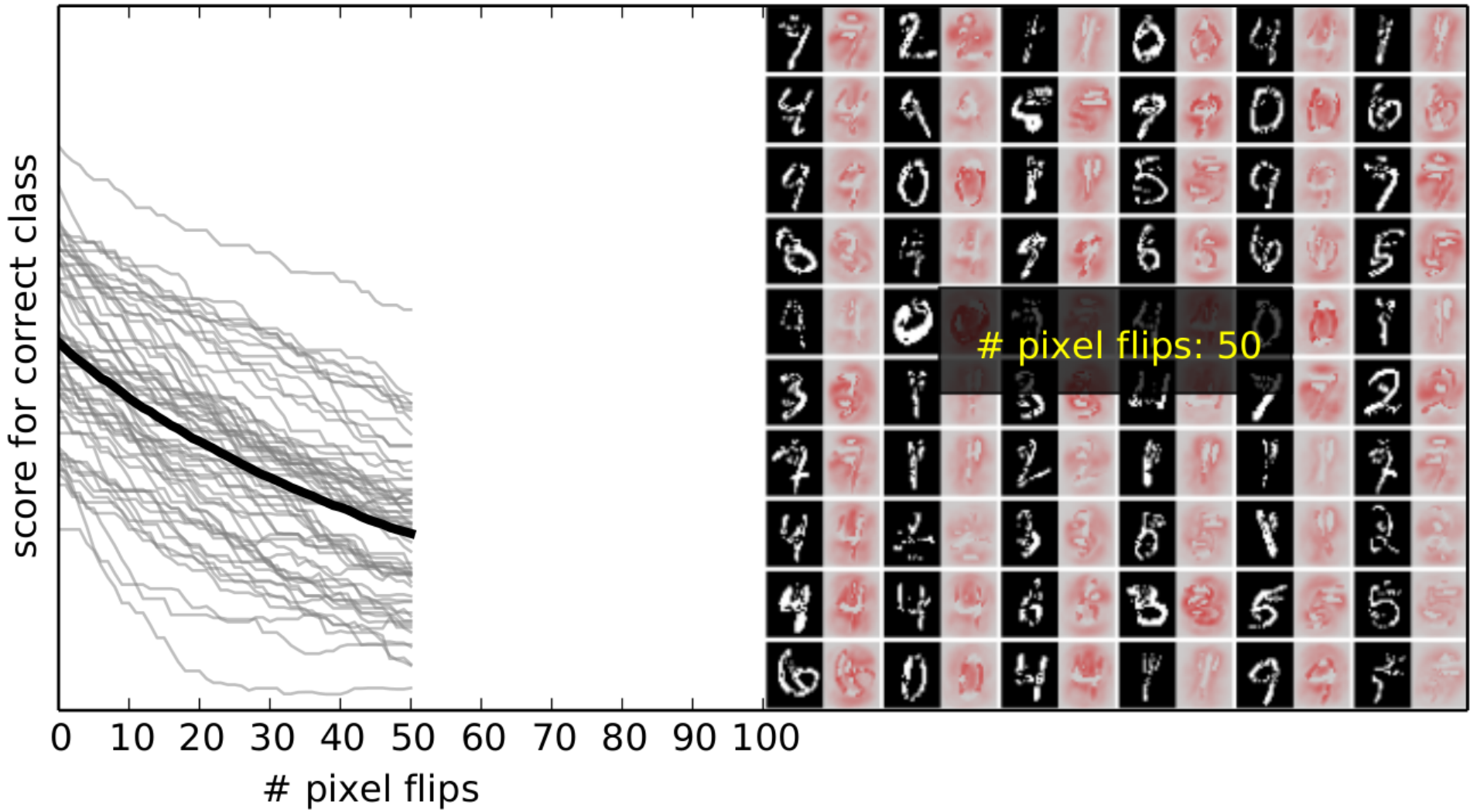
axioms

				
$\sum_{i=1}^{d} R_i + \varepsilon = f(\boldsymbol{x})$	✓	✓	✓	✓
$\sum_{i=1}^{d} \lvert R_i \rvert \leq A$	✗	✓	✓	✓
$R_i(\boldsymbol{x}) \approx R_i(\boldsymbol{x} + \boldsymbol{\varepsilon})$	✗	✓	✓	✓
$R_i(\boldsymbol{x}; f_\theta) = R_i(\boldsymbol{x}; f_{\theta'})$	✓	✗	✗	✓

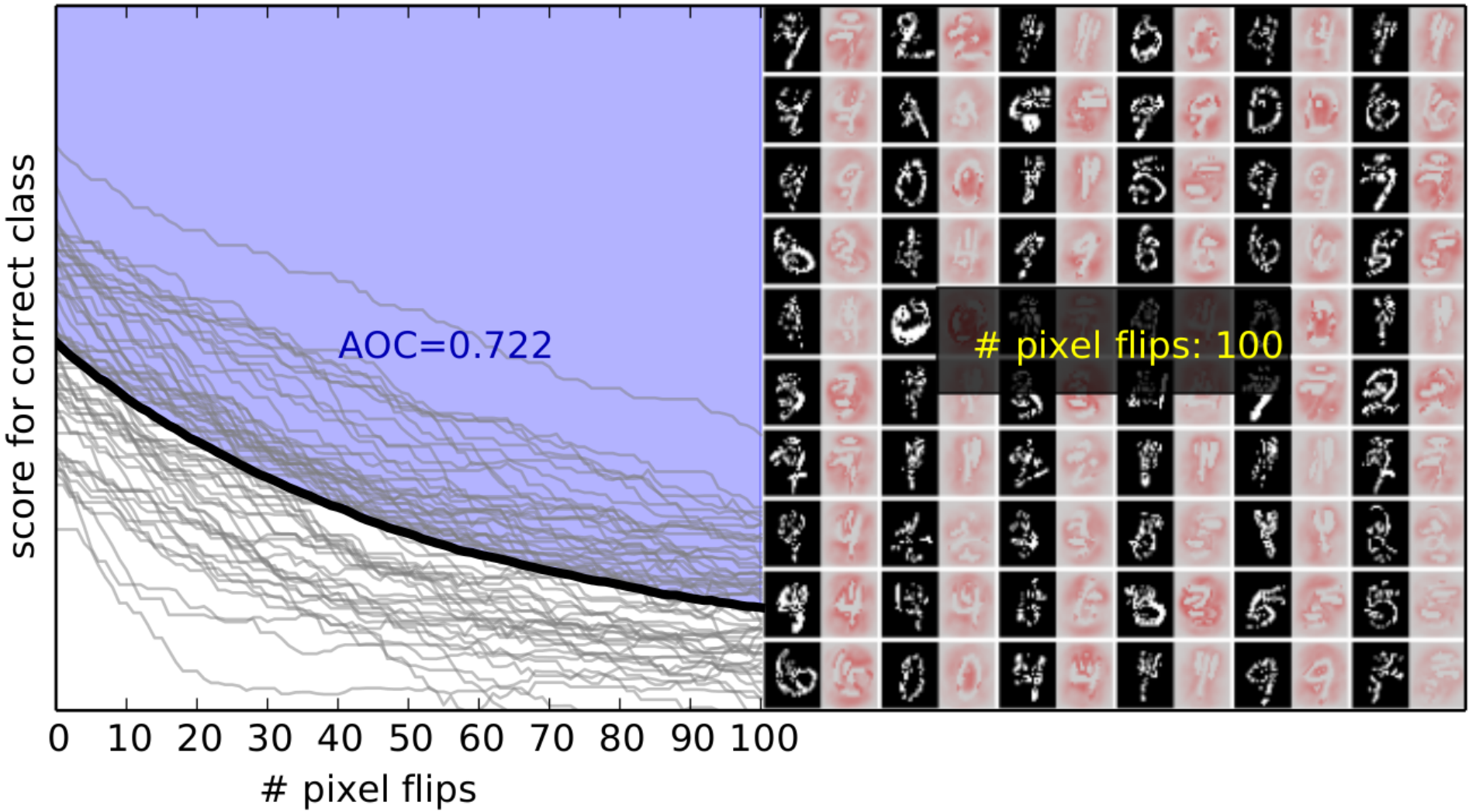
Approach 3: Pixel-Flipping



Approach 3: Pixel-Flipping



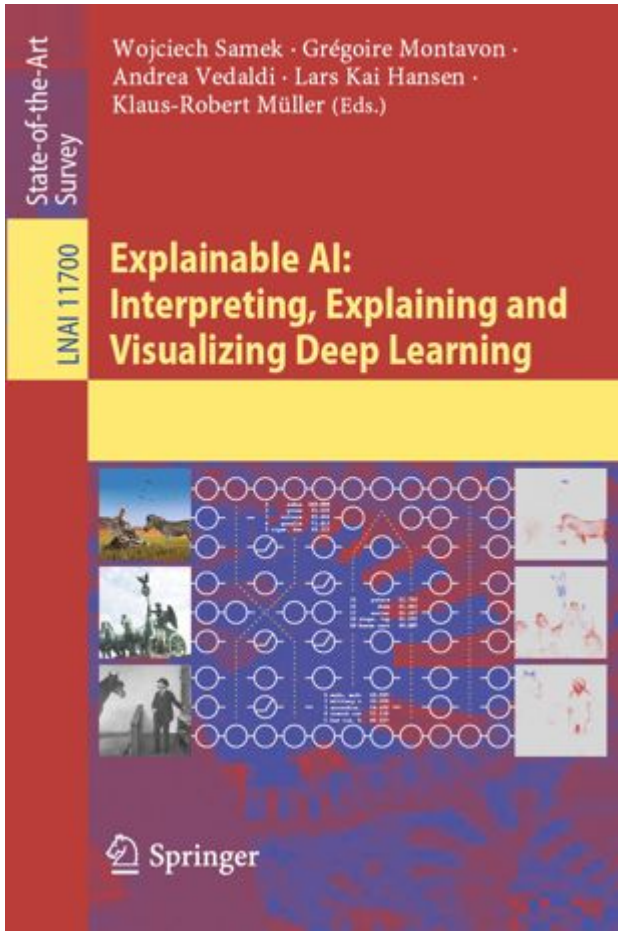
Approach 3: Pixel-Flipping



Summary

- Optimizing a neural network to minimize the training objective is not sufficient, we need to make sure it delivers good generalization performance.
- Typical neural network training has self-regularizing properties, e.g. cross-entropy loss, noise in the training procedure.
- We also need to ensure that the neural network uses the correct features and do not exploit spurious correlations in the given dataset.
- This can be sometimes achieved by introducing prior knowledge, however, we also need to verify what the network has learned by making it transparent and inspecting the structure of its predictions.

New Book



Explainable AI: Interpreting, Explaining and Visualizing Deep Learning

Editors: **Samek**, W., **Montavon**, G., **Vedaldi**, A., **Hansen**, L.K., **Muller**, K.-R. (Eds.)

https://link.springer.com/book/10.1007/978-3-030-28954-6

Check our website

Online demos, tutorials, code examples, etc.

and tutorial papers

G Montavon, W Samek, KR Müller. Methods for Interpreting and Understanding Deep Neural Networks Digital Signal Processing, 73:1-15, 2018

G Montavon, A Binder, S Lapuschkin, W Samek, KR Müller. Layer-Wise Relevance Propagation: An Overview in Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Springer LNCS 11700, 2019