Seminar Explainable AI

Module 8

Selected Methods Part 4

Sensitivity – Gradients

Andreas Holzinger

Human-Centered AI Lab (Holzinger Group)

Institute for Medical Informatics/Statistics, Medical University Graz, Austria

and

Explainable AI-Lab, Alberta Machine Intelligence Institute, Edmonton, Canada

Remark

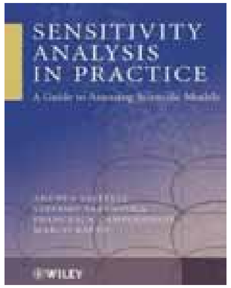
This is the version for printing and reading. The lecture version is didactically different.

AGENDA

- 00 Reflection
- 01 Sensitivity Analysis
- 02 Gradients: General overview
- 03 Gradients: DeepLIFT
- 04 Gradients: Grad-CAM
- 05 Integrated Gradients

01 Sensitivity Analysis

What is Sensitivity Analysis?

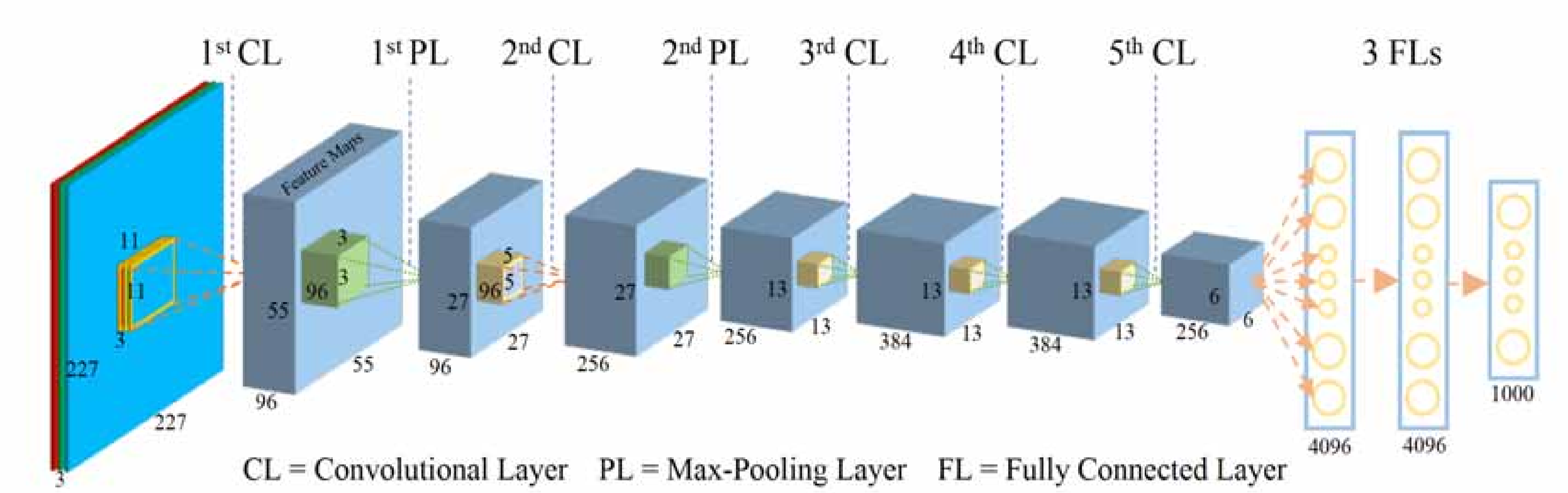

SENSITIVITY ANALYSIS IN PRACTICE
A Guide to Assessing Scientific Models
WILEY

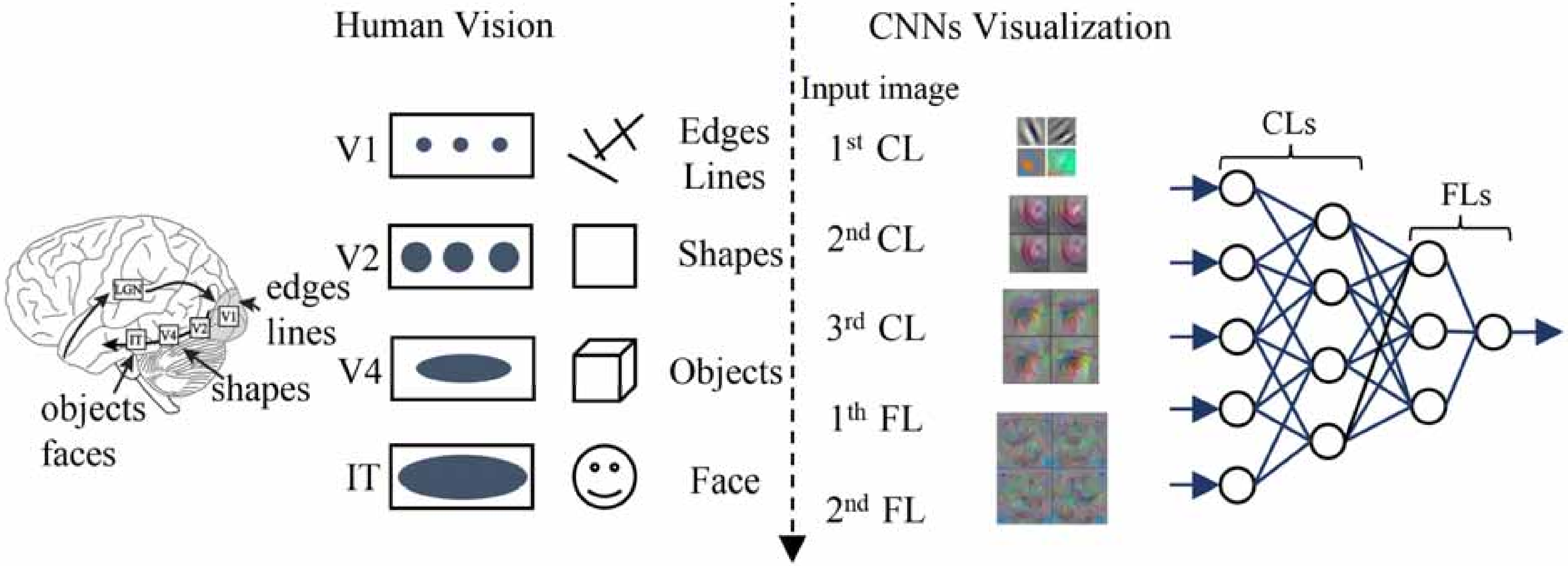
- Sensitivity analysis (SA) is a classic, versatile and broad field with long tradition and can be used for a variety of different purposes, including:
  - Robustness testing (very important for ML)
  - Understanding the relationship between input and output
  - Reducing uncertainty

Andrea Saltelli, Stefano Tarantola, Francesca Campolongo & Marco Ratto 2004. Sensitivity analysis in practice: a guide to assessing scientific models. Chichester, England.

- Remember: NN=nonlinear function approximators using gradient descent to minimize the error in such a function approximation
- To students this seems to be “new” – but it has a long history:
  - Chain rule = back-propagation was invented by Leibniz (1676) and L’Hopital (1696)
  - Calculus and Algebra have long been used to solve optimization problems and gradient descent was introduced by Cauchy (1847)
  - This was used to fuel machine learning in the 1940ies > perceptron – but was limited to linear functions, therefore
  - Learning nonlinear functions required the development of a multilayer perceptron and methods to compute the gradient through such a model
  - This was elaborated by LeCun (1985), Parker (1985), Rummelhart (1986) and Hinton (1986)

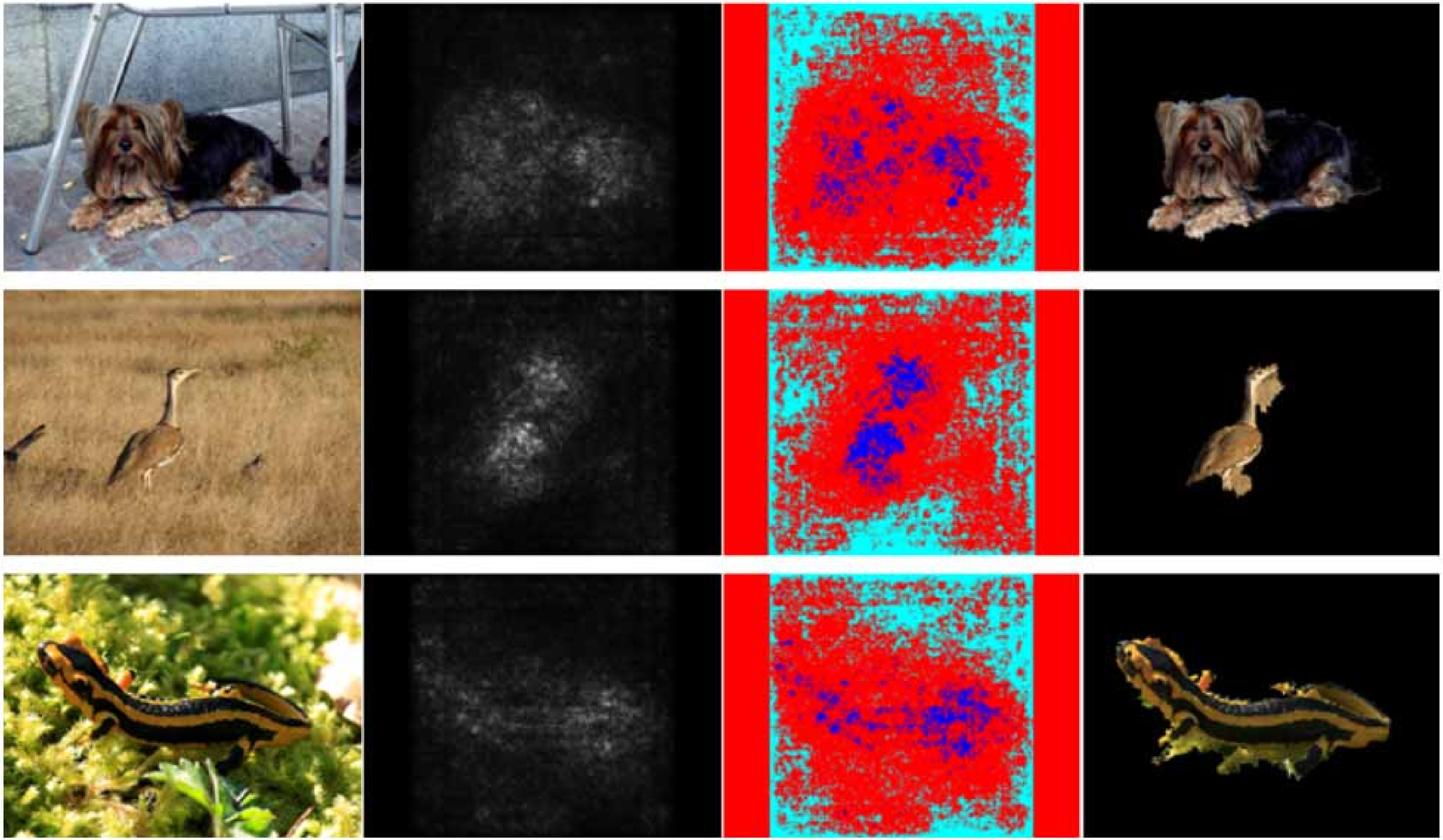
What are Saliency Maps?





Zhuwei Qin, Fuxun Yu, Chenchen Liu & Xiang Chen 2018. How convolutional neural network see the world-A survey of convolutional neural network visualization methods. *arXiv preprint arXiv:1804.11191*.

What are Saliency Maps?

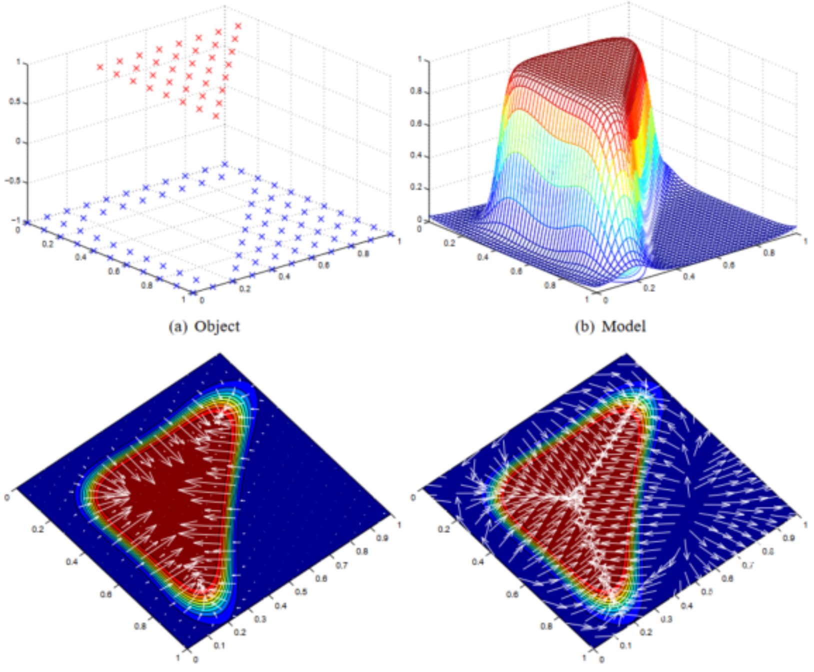


Karen Simonyan, Andrea Vedaldi & Andrew Zisserman 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv:1312.6034*.

Problem: How to cope with local non-linearities?

For $\mathrm{var}_f(x_0) = k(x_0,x_0) - k_*^T(K+\Sigma)^{-1}k_*$ the derivative is given by[3]

$$\nabla \mathrm{var}_f(x)|_{x=x_0} = \frac{\partial \mathrm{var}_f}{\partial x_{0,j}} = \left(\frac{\partial}{\partial x_{0,j}} k(x_0,x_0)\right) - 2*k_*^T(K+\Sigma)^{-1}\frac{\partial}{\partial x_{0,j}}k_* \quad \text{for } j \in \{1,\dots,d\}.$$

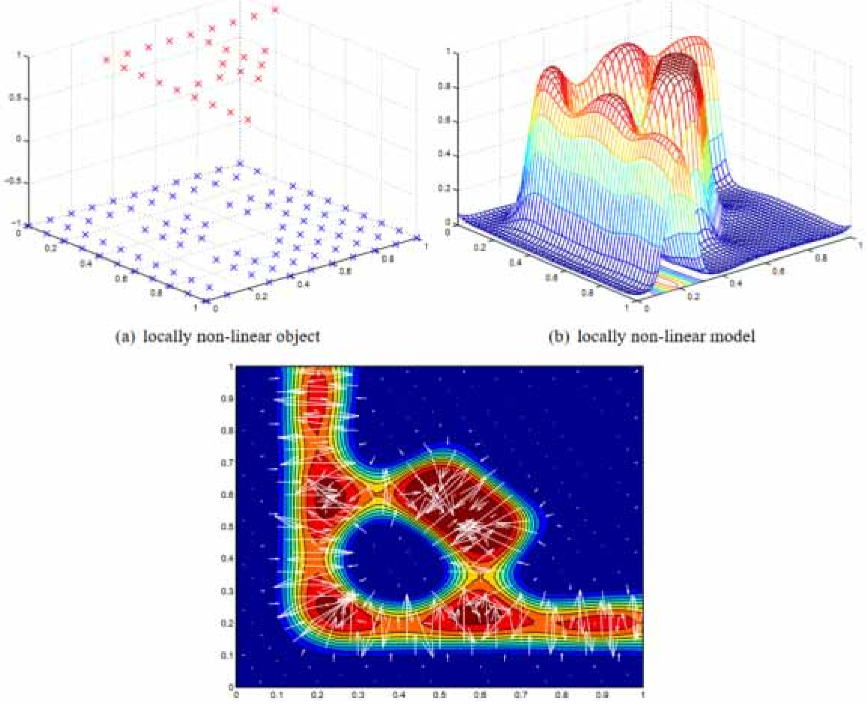


(a) Object

(b) Model

(c) Local explanation vectors

(d) Direction of explanation vectors



(a) locally non-linear object

(b) locally non-linear model

(c) locally non-linear explanation

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen & Klaus-Robert Mueller 2010. How to explain individual classification decisions. *Journal of machine learning research (JMLR)*, 11, (6), 1803-1831.

Principle of Sensitivity Analysis (SA)

- Let us consider a function f,
- a data point $x = (x1, \dots, xd)$ and the prediction
- $f(x1, \dots, xd)$
- Now, SA measures the local variation of the function along each input dimension:
- $Ri = \left(\frac{\partial f}{\partial xi} | x = x\right)^2$
- With other words, SA produces local explanations for the prediction of a differentiable function f using the squared norm of its gradient w.r.t. the inputs $x : S(x) / krxfk2$.
- The saliency map S produced with this method describes the extent to which variations in the input would produce a change in the output $S(x) \propto \|\nabla_{\mathbf{x}} f\|^2$

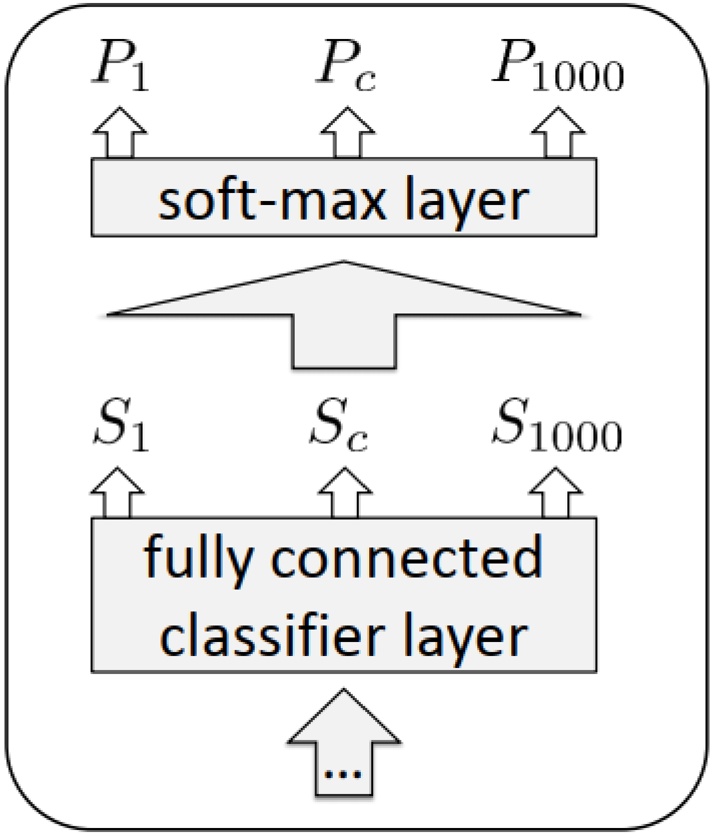
Muriel Gevrey, Ioannis Dimopoulos & Sovan Lek 2003. Review and comparison of methods to study the contribution of variables in artificial neural network models. Ecological modelling, 160, (3), 249-264.

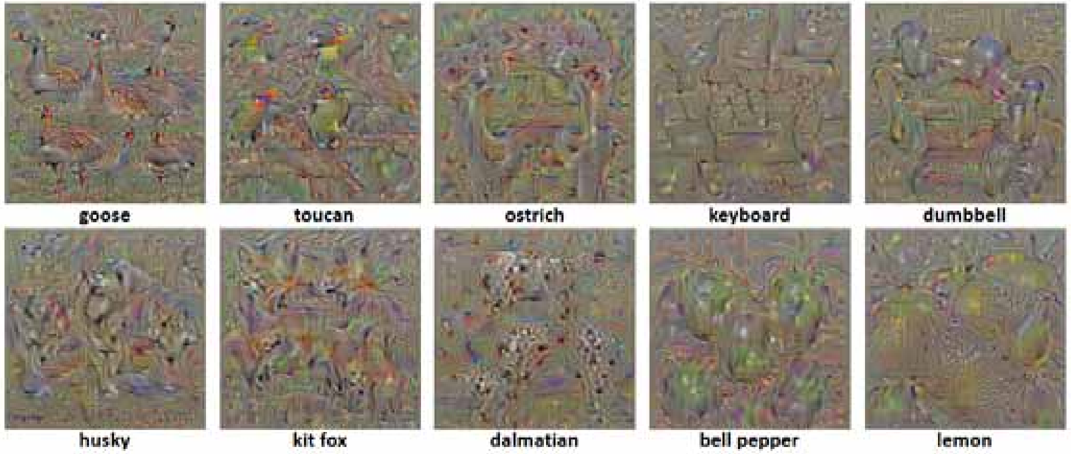
What do we want to know?

- Given an image classification (ConvNet), we aim to answer two questions:
  - What does a class model look like?
  - What makes an image belong to a class?
- To this end, we visualise:
  - Canonical image of a class
  - Class saliency map for a given image and class
- Both visualisations are based on the class score derivative w.r.t. the input image (computed using back-prop)

Simonyan, Vedaldi, Zisserman: Visualizing Saliency Maps

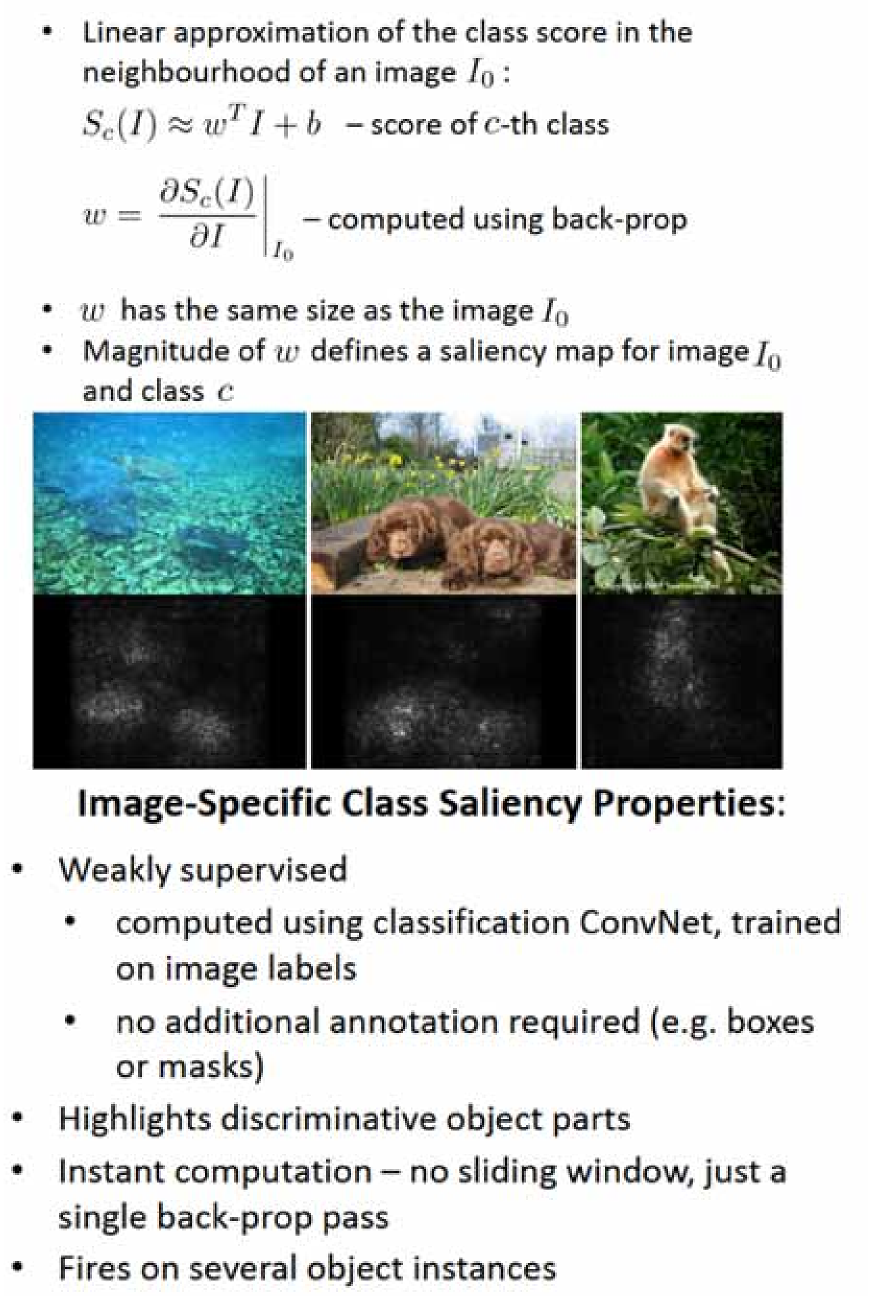
- We compute a (regularised) image I with a high class score $S_c(I)$: $\arg\max_I S_c(I) - \lambda\|I\|_2^2$ [Erhan et al., 2009]
- Optimised using gradient descent, initialised with the zero image
- Gradient $\partial S_c(I)/\partial I$ is computed using back-prop
- Maximising soft-max score $\arg\max_I P_c(I)$ leads to worse visualisation
- We visualise a ConvNet trained on ImageNet ILSVRC 2013 (1000 classes)





Karen Simonyan, Andrea Vedaldi & Andrew Zisserman 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv:1312.6034*.

Simonyan & Zissserman (2014):



- Linear approximation of the class score in the neighbourhood of an image I_0 :

$S_c(I) \approx w^T I + b$ – score of c-th class

$w = \left.\frac{\partial S_c(I)}{\partial I}\right|_{I_0}$ – computed using back-prop

- w has the same size as the image I_0
- Magnitude of w defines a saliency map for image I_0 and class c

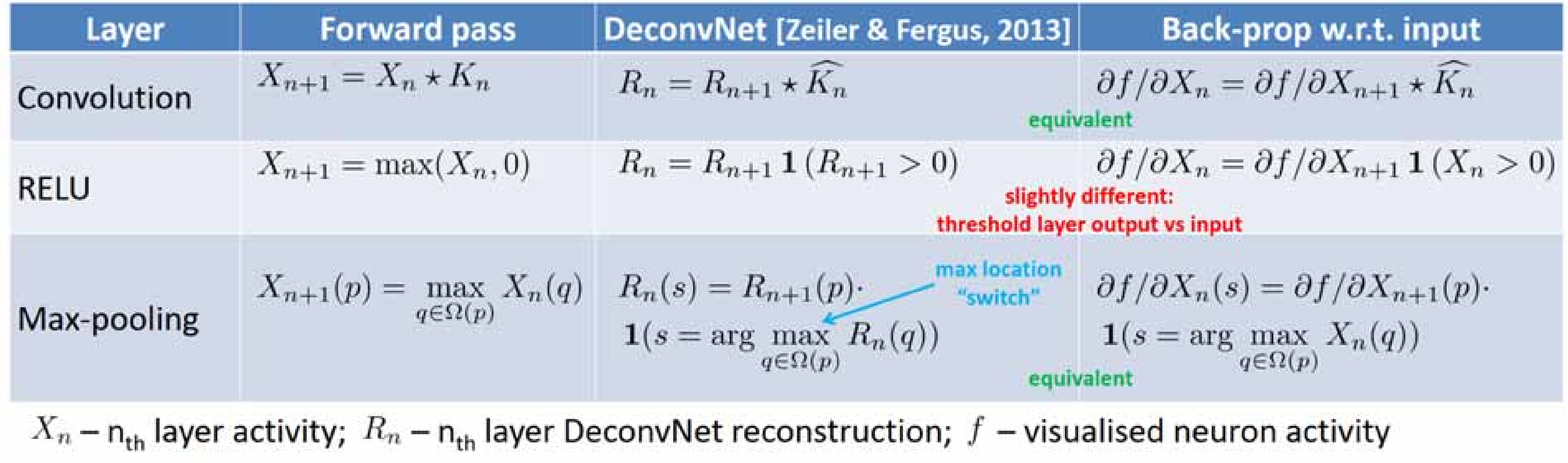
Image-Specific Class Saliency Properties:

- Weakly supervised
  - computed using classification ConvNet, trained on image labels
  - no additional annotation required (e.g. boxes or masks)
- Highlights discriminative object parts
- Instant computation – no sliding window, just a single back-prop pass
- Fires on several object instances

- Given an image and a saliency map:
  1. Saliency map is thresholded to obtain foreground / background masks
  2. GraphCut colour segmentation [Boykov and Jolly, 2001] is initialised with the masks
  3. Object localisation: bounding box of the largest foreground connected component
- GraphCut propagates segmentation from the most salient areas of the object
- ILSVRC 2013 localisation accuracy: 46.4%
  - weak supervision: ground-truth bounding boxes were not used for training
  - saliency maps for top-5 predicted classes were used to compute five bounding box predictions

Karen Simonyan & Andrew Zisserman 2014. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

Relation to de-convolutional networks



Layer	Forward pass	DeconvNet [Zeiler & Fergus, 2013]	Back-prop w.r.t. input
Convolution	$X_{n+1} = X_n \star K_n$	$R_n = R_{n+1} \star \widehat{K_n}$	$\partial f / \partial X_n = \partial f / \partial X_{n+1} \star \widehat{K_n}$
RELU	$X_{n+1} = \max(X_n, 0)$	$R_n = R_{n+1} \mathbf{1}(R_{n+1} > 0)$	$\partial f / \partial X_n = \partial f / \partial X_{n+1} \mathbf{1}(X_n > 0)$
Max-pooling	$X_{n+1}(p) = \max\limits_{q \in \Omega(p)} X_n(q)$	$R_n(s) = R_{n+1}(p) \cdot$ $\mathbf{1}(s = \arg \max\limits_{q \in \Omega(p)} R_n(q))$	$\partial f / \partial X_n(s) = \partial f / \partial X_{n+1}(p) \cdot$ $\mathbf{1}(s = \arg \max\limits_{q \in \Omega(p)} X_n(q))$

equivalent

slightly different:
threshold layer output vs input

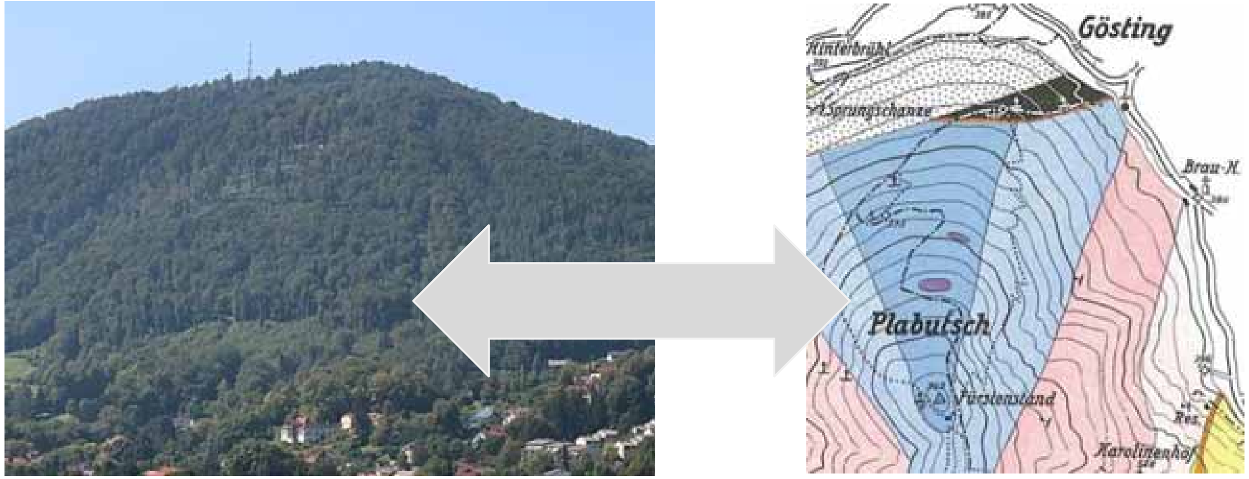
max location
"switch"

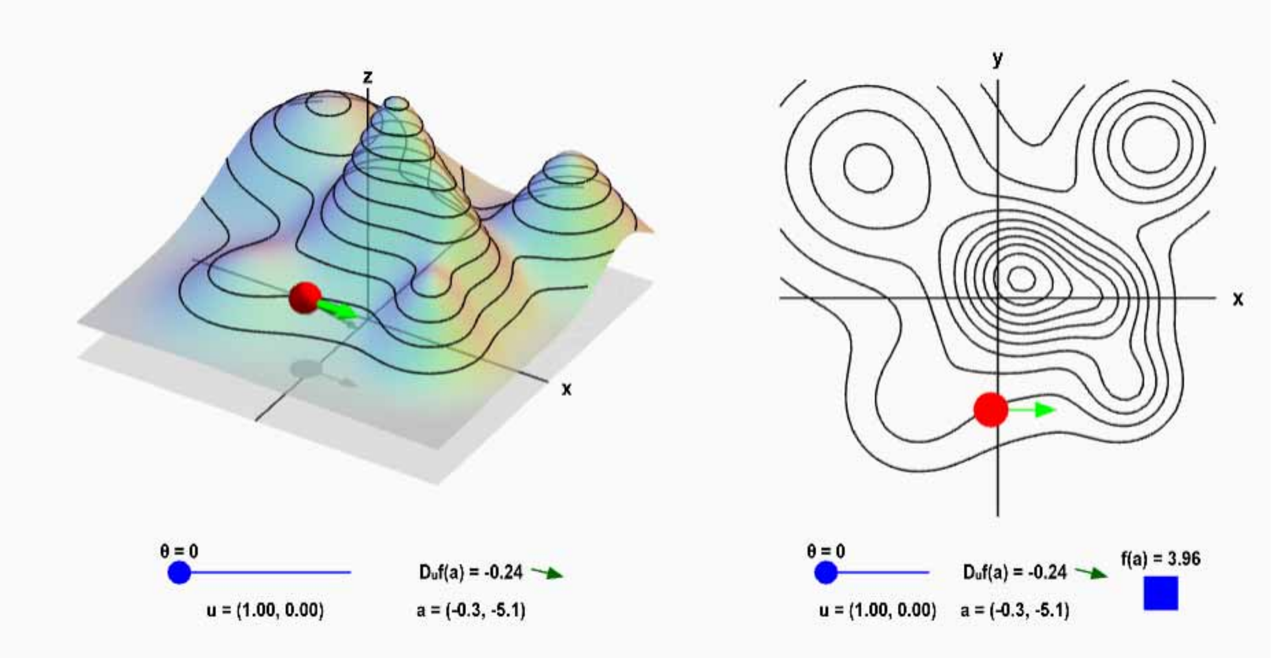
equivalent

X_n – n$_{th}$ layer activity; R_n – n$_{th}$ layer DeconvNet reconstruction; f – visualised neuron activity

02 Gradients General Overview

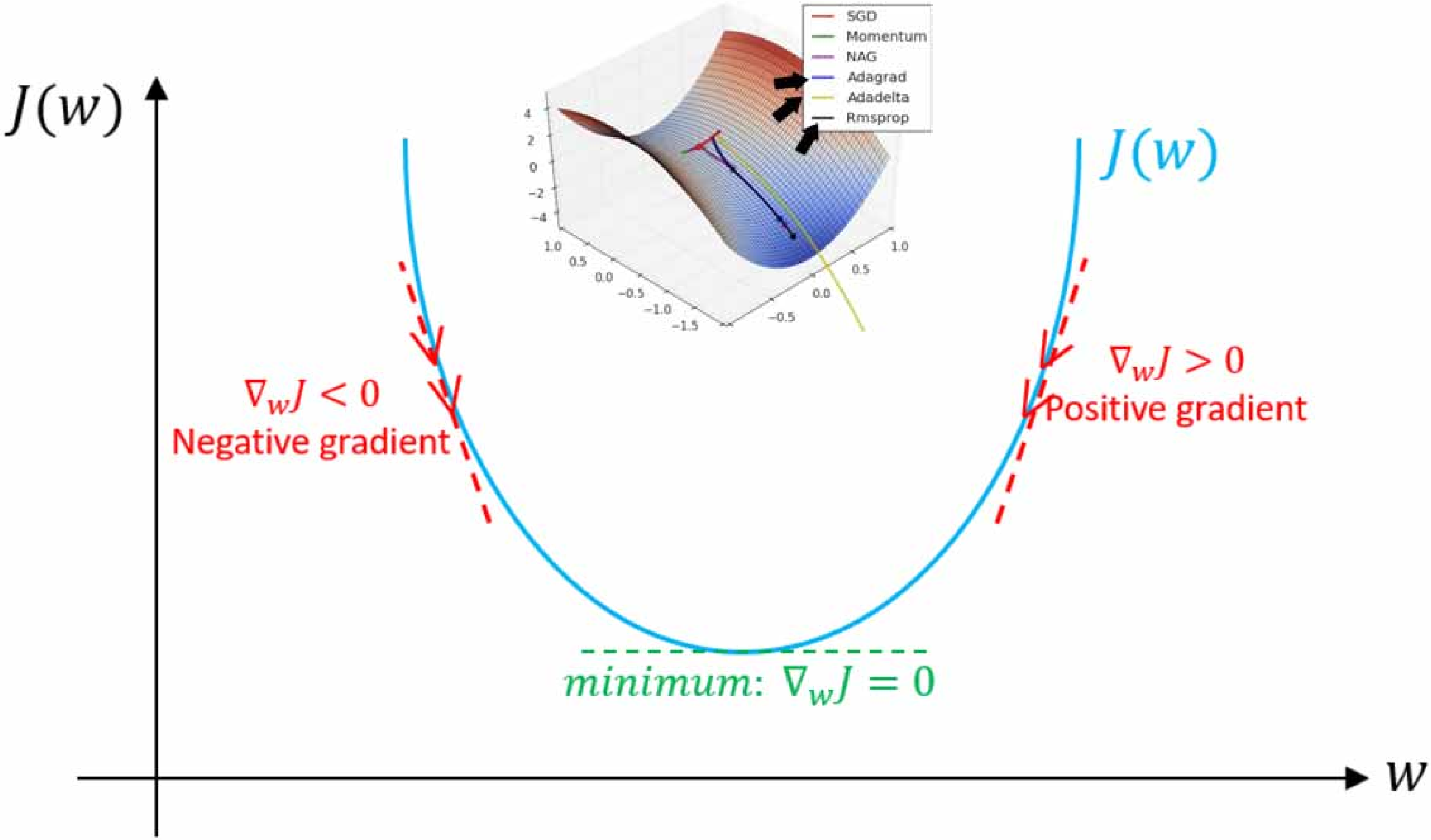
Topographical maps, level curves, gradients





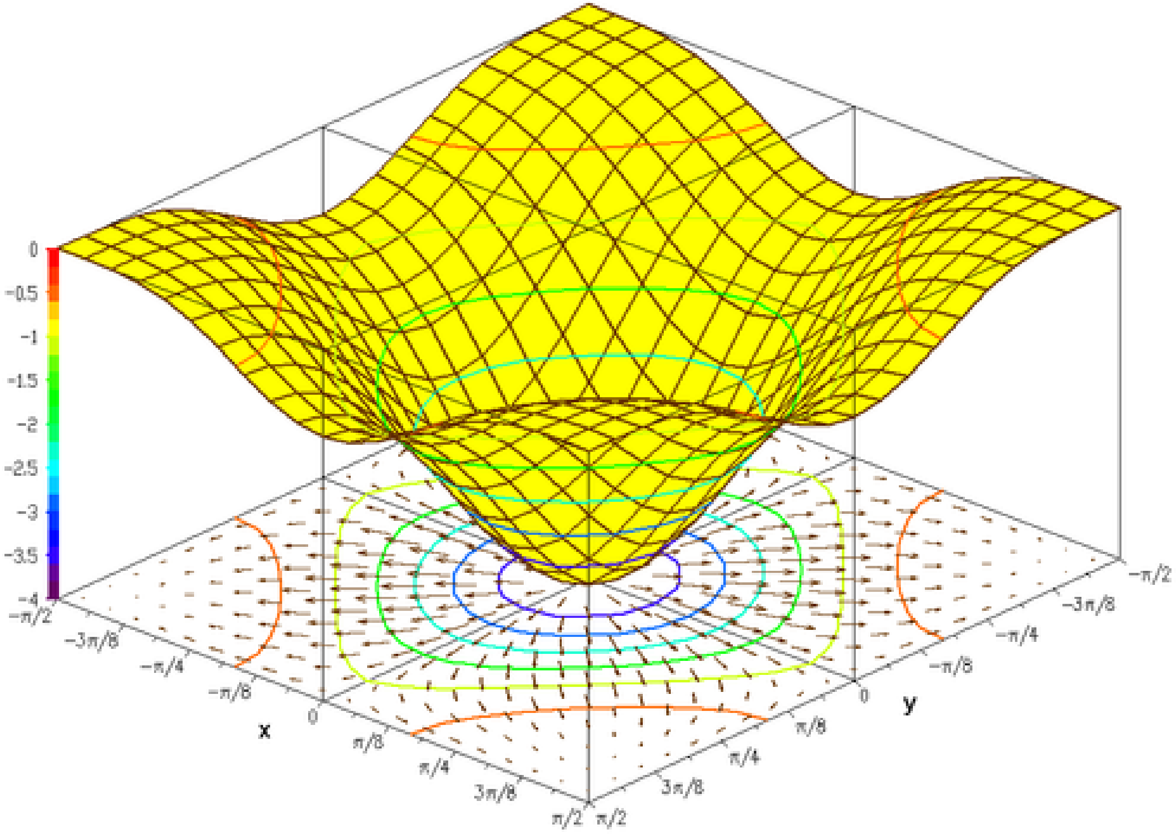
https://mathinsight.org/applet/gradient_directional_derivative_mountain

How to find a local minima?

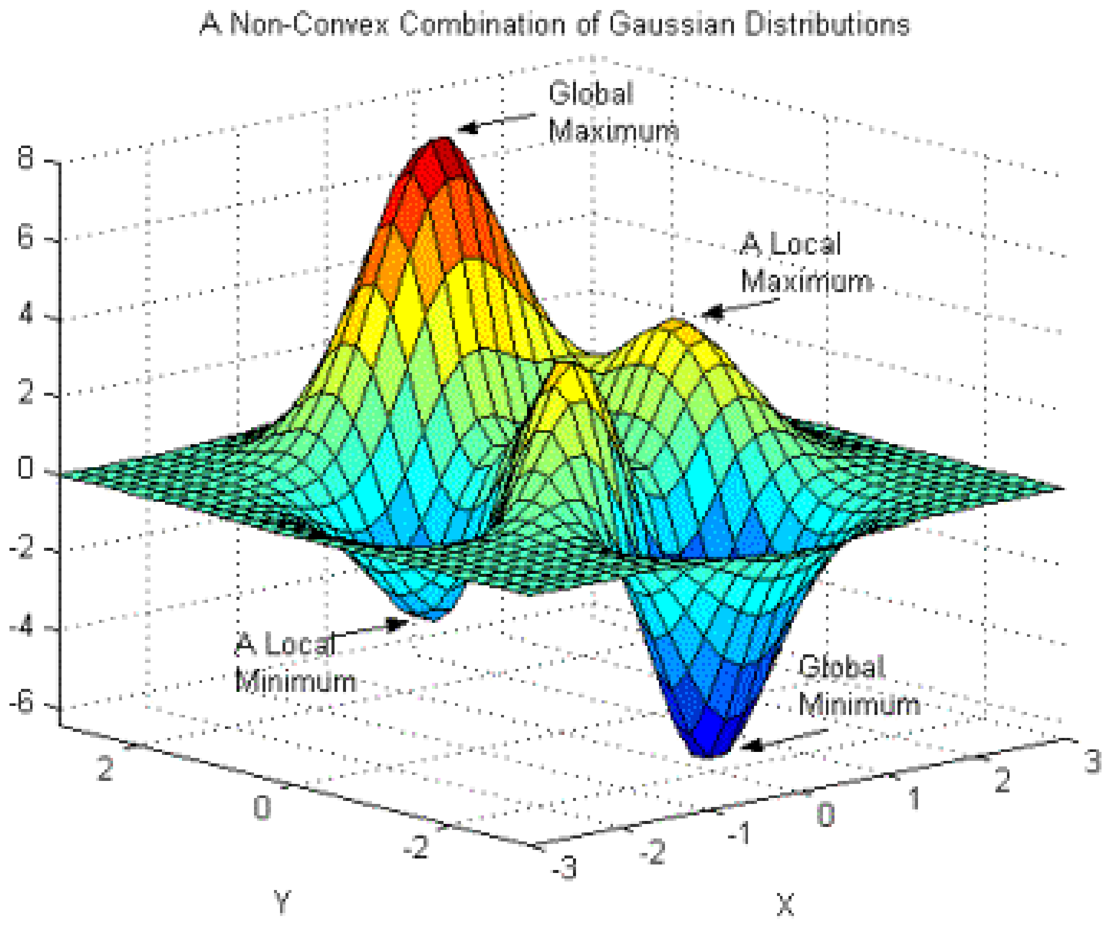


Example

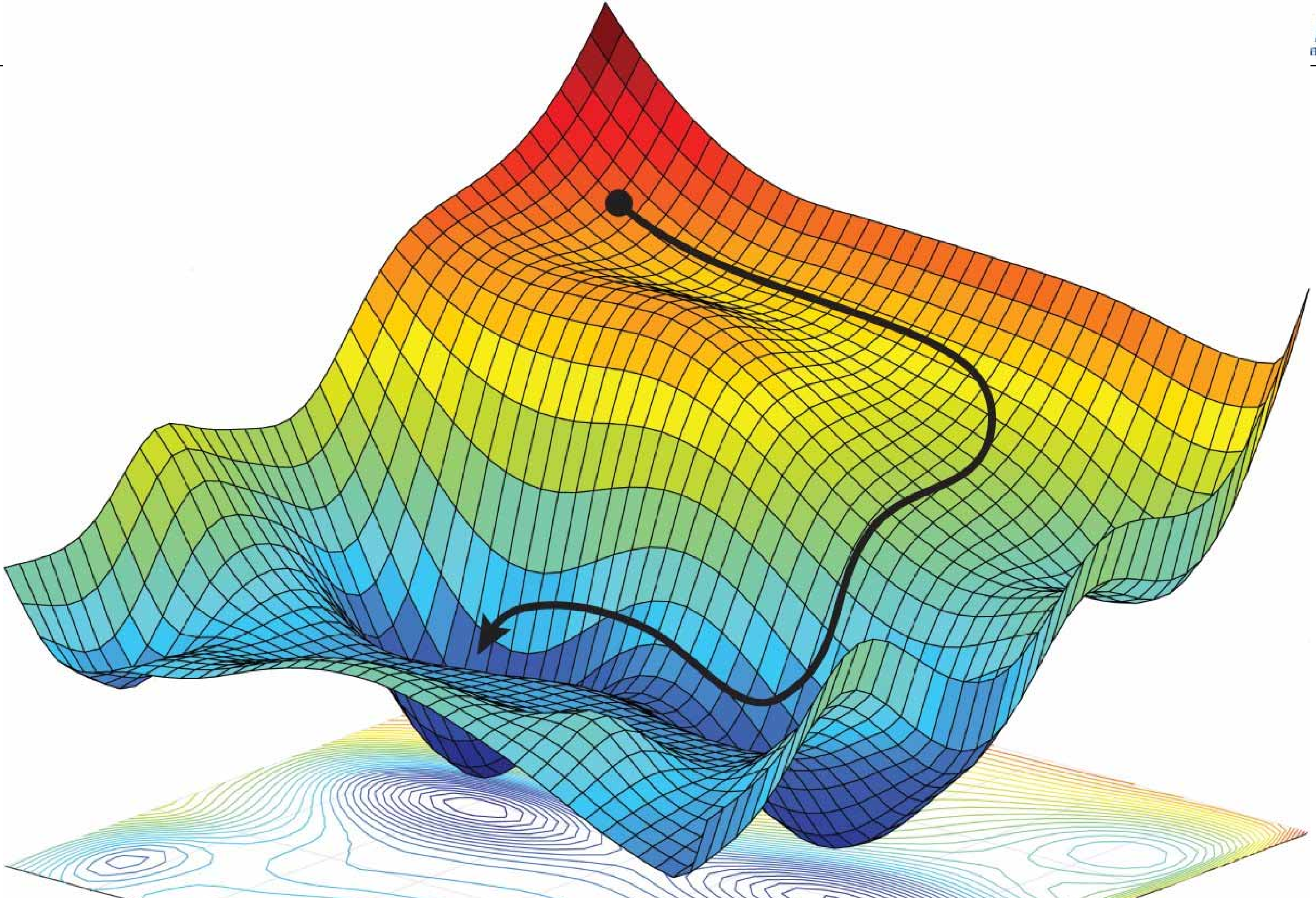
$$\big(\nabla f(x)\big) \cdot \mathbf{v} = D_{\mathbf{v}} f(x)$$



Finding the GLOBAL minimum is difficult !

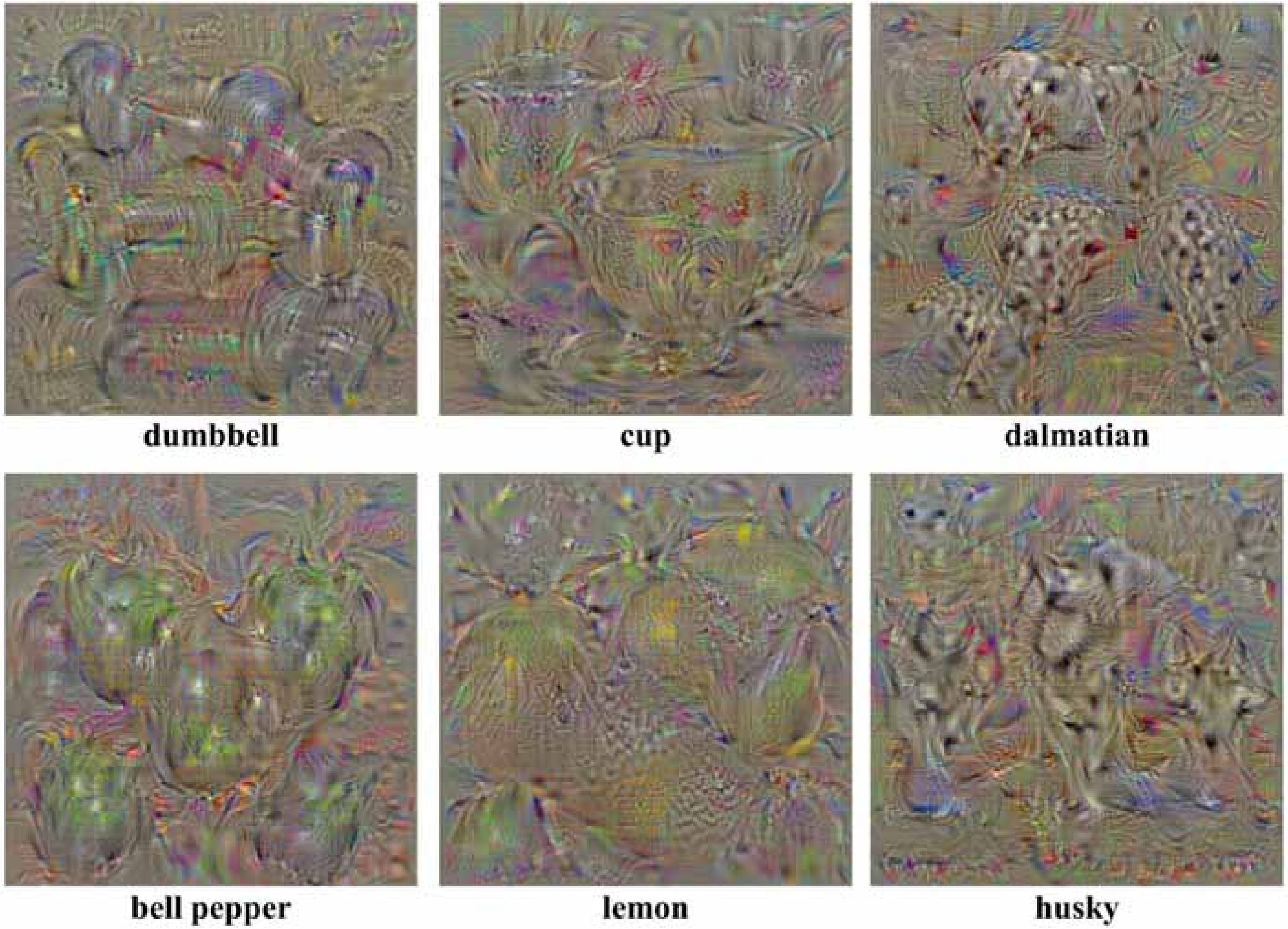


https://www.khanacademy.org/math/multivariable-calculus/multivariable-derivatives/partial-derivative-and-gradient-articles/a/the-gradient



$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \mathbb{E}_i[f_i(x)] = \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right\}$$

Gradients > Sensitivity Analysis > Heatmapping

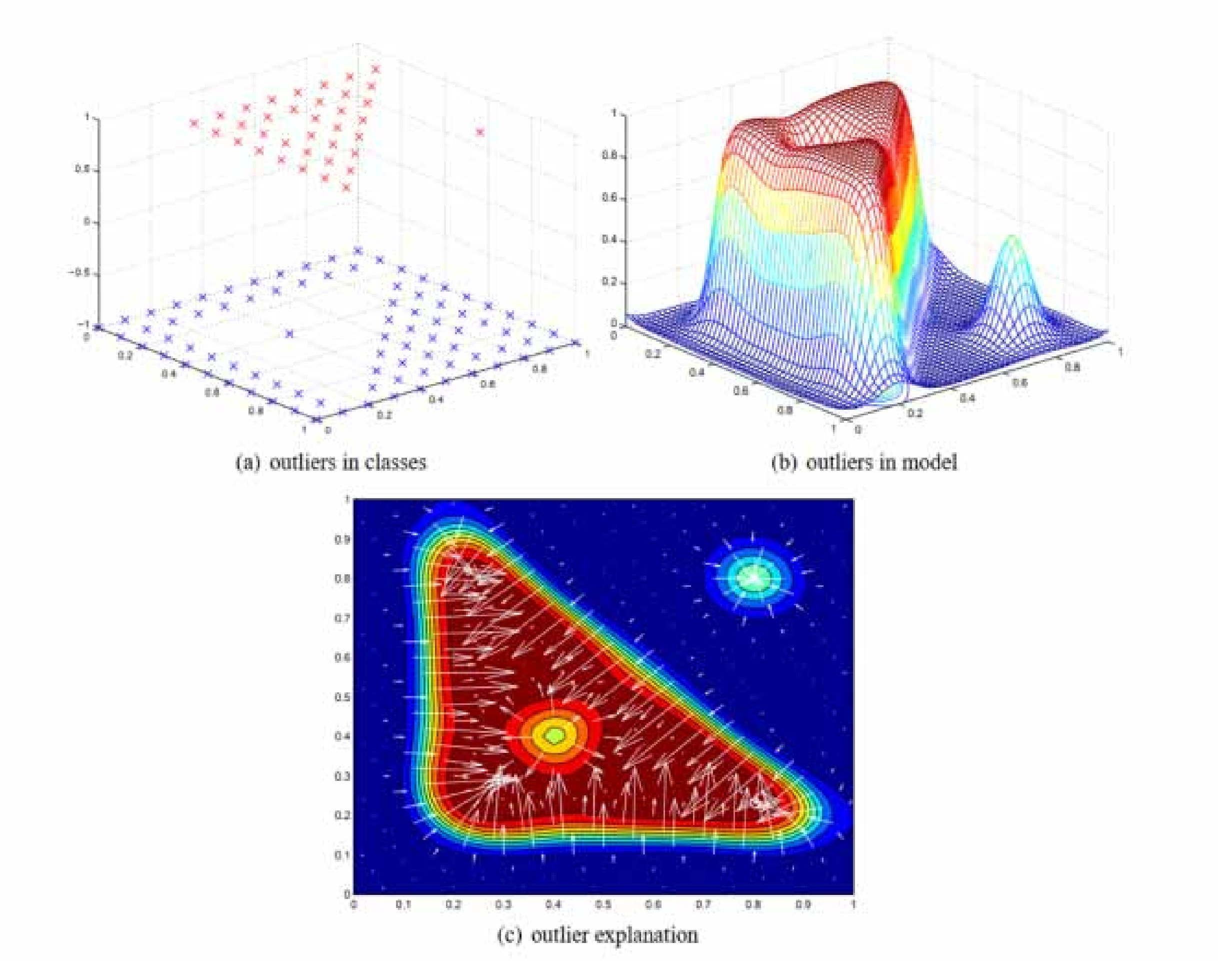


dumbbell cup dalmatian

bell pepper lemon husky

Karen Simonyan, Andrea Vedaldi & Andrew Zisserman 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv:1312.6034.

Gradients



(a) outliers in classes

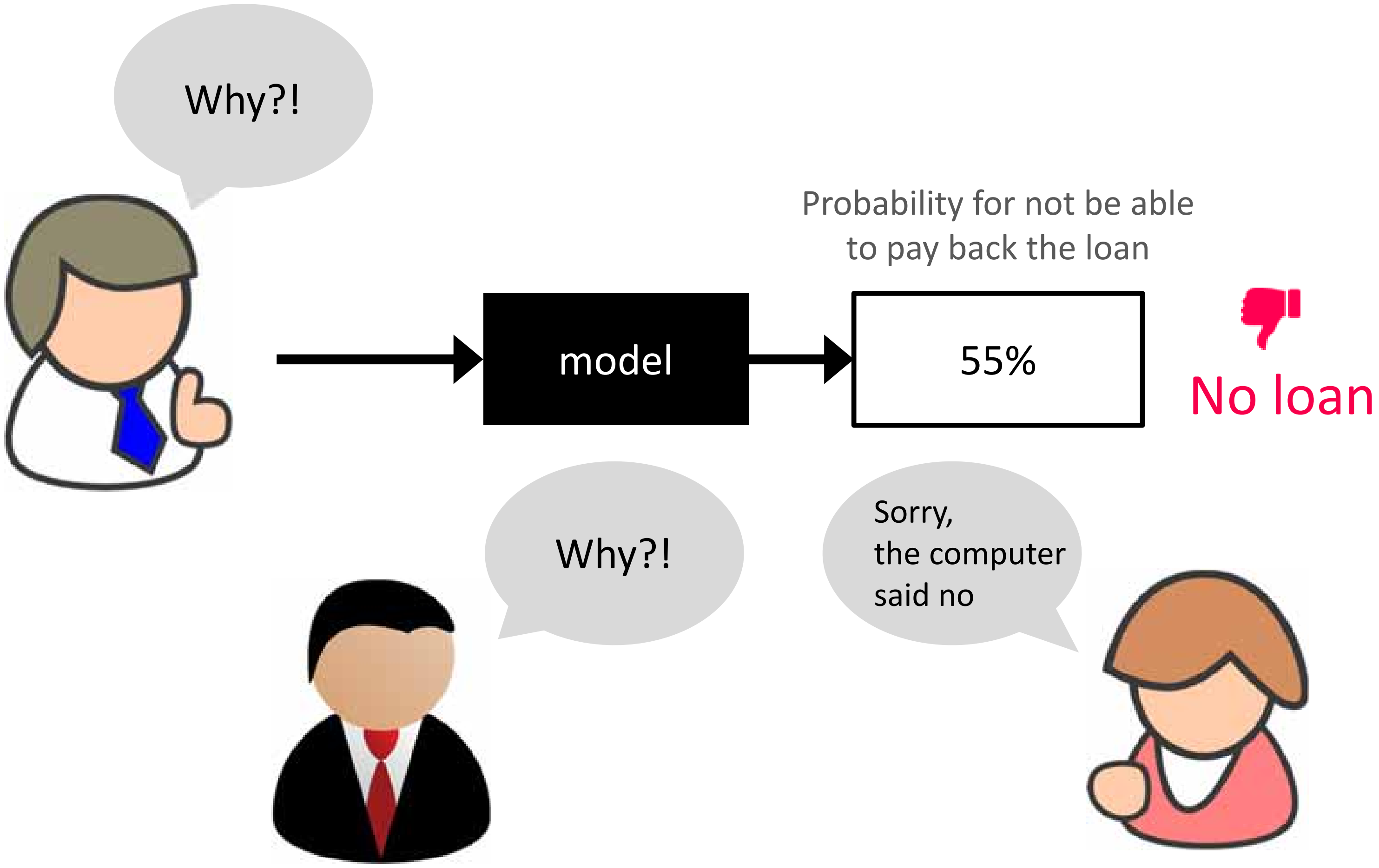
(b) outliers in model

(c) outlier explanation

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen & Klaus-Robert Mueller 2010. How to explain individual classification decisions. Journal of machine learning research (JMLR), 11, (6), 1803-1831.

03 Gradients: DeepLIFT Deep Learning Important FeaTures

Typical Recommender Systems Scenario Black-Box



Why?!

model

Probability for not be able to pay back the loan

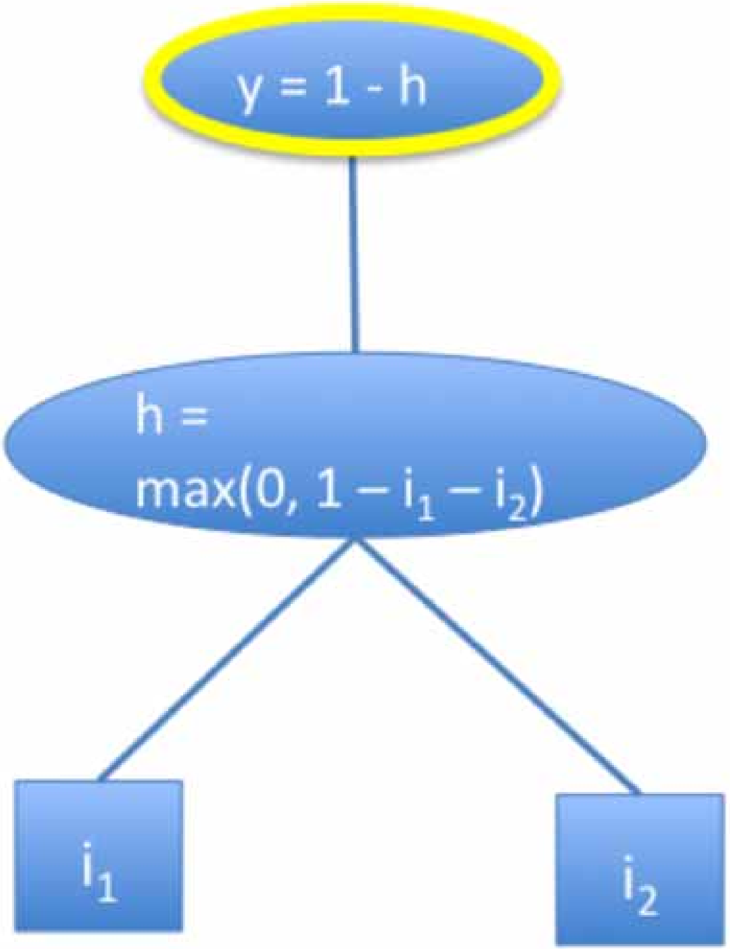
55%

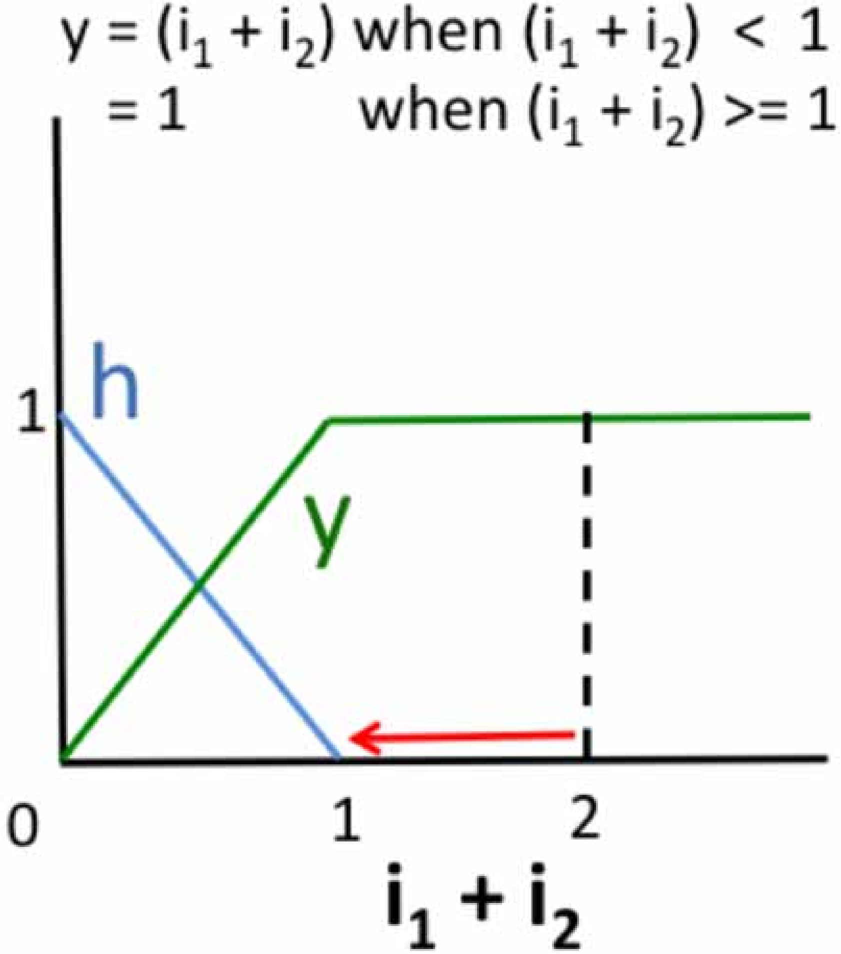
👎 No loan

Why?!

Sorry, the computer said no

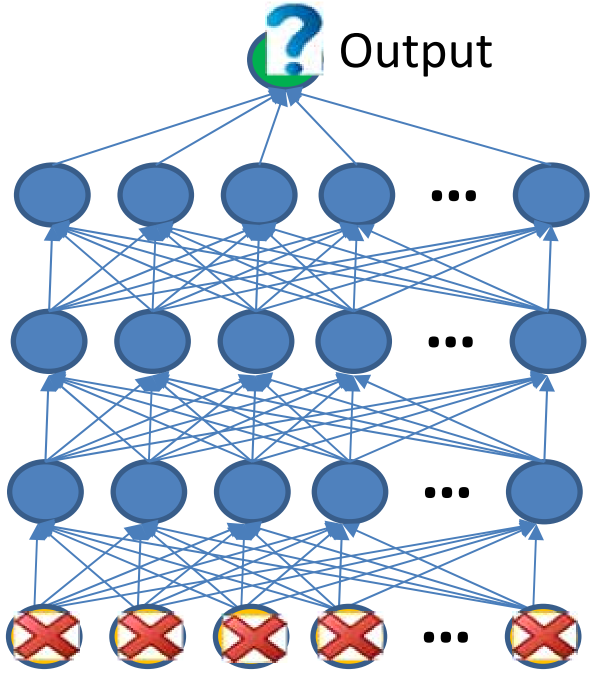
Motivation: The Saturation Problem





Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. *arXiv:1704.02685*.

- First idea: perturbation



Yellow = inputs

Examples

1) Zeiler & Fergus, 2013
2) LIME (Ribeiro et al., 2016)
3) Zintgraf et al., 2017

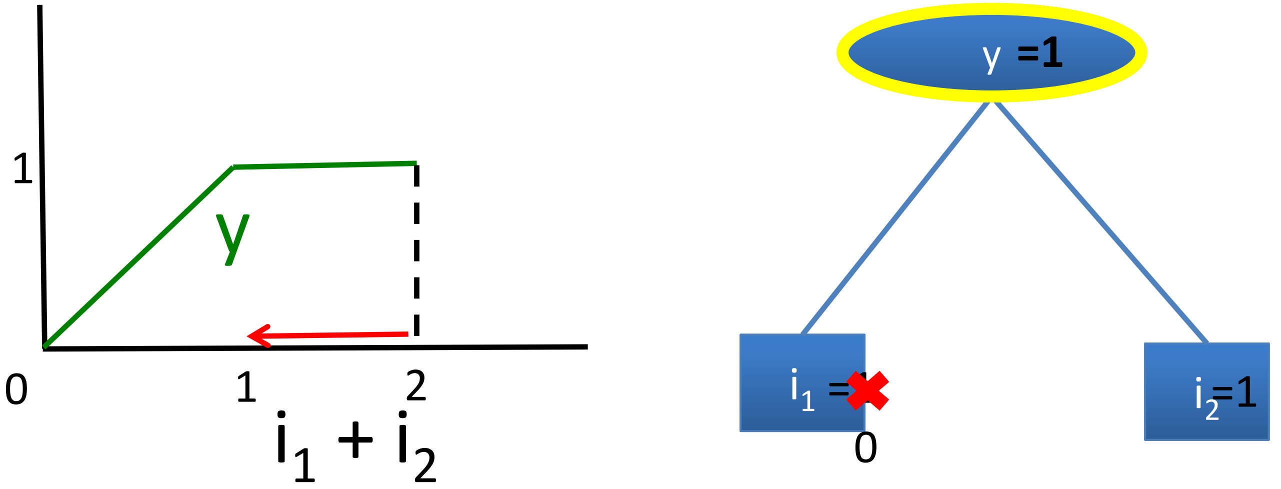
Drawbacks

1) Computational efficiency - requires one forward prop for each perturbation

2) Saturation

Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

Saturation problem illustrated

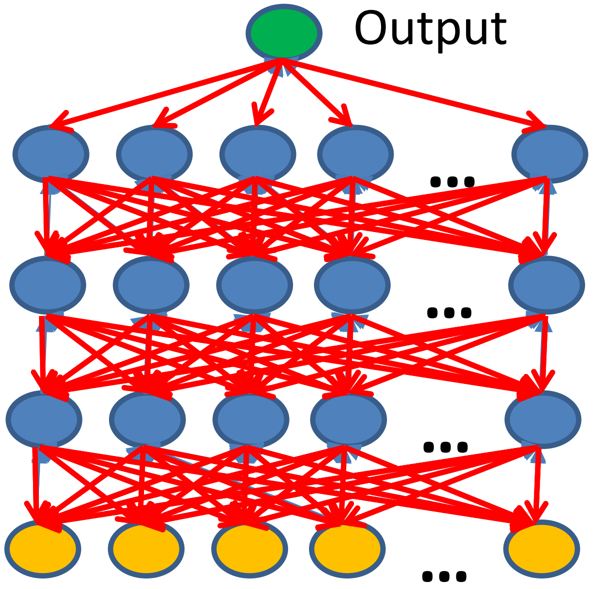


Avoiding saturation means perturbing combinations of inputs → increased computational cost

Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

How can we find the important parts of the input for a given prediction?

Second idea backpropagate importance



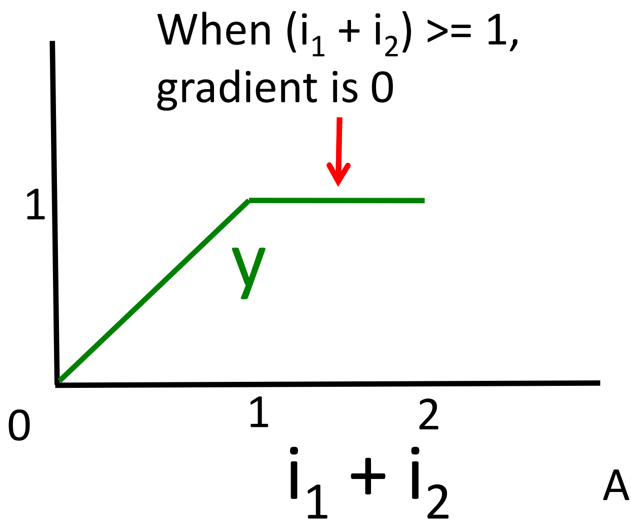
Yellow = inputs

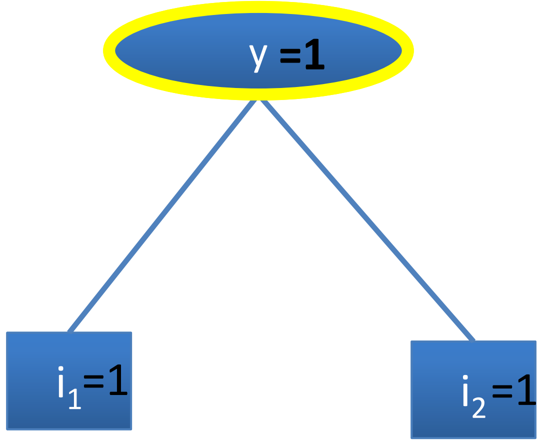
Examples:

- Gradients (Simonyan et al.)
- Deconvolutional Networks (Zeiler & Fergus)
- Guided Backpropagation (Springenberg et al.)
- Layerwise Relevance Propagation (Bach et al.)
- Integrated Gradients (Sundararajan et al.)
- **DeepLIFT (Learning Important FeaTures)**
    - https://github.com/kundajelab/deeplift

Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

Saturation revisited


When $(i_1 + i_2) >= 1$, gradient is 0



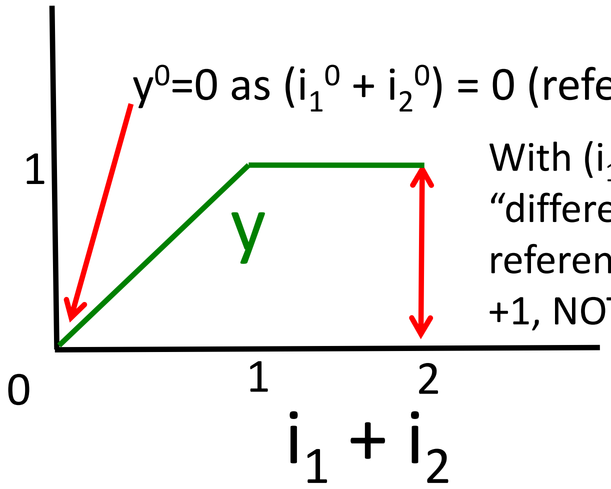
Affects:

- Gradients
- Deconvolutional Networks
- Guided Backpropagation
- Layerwise Relevance Propagation

Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

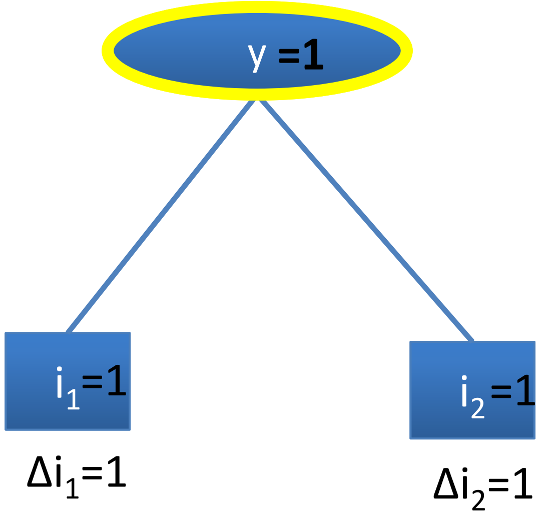
DeepLIFT

Reference: $i_1^0=0$ & $i_2^0=0$



$y^0=0$ as $(i_1^0 + i_2^0) = 0$ (reference)

With $(i_1 + i_2) = 2$, the "difference from reference" (Δy) is +1, NOT 0



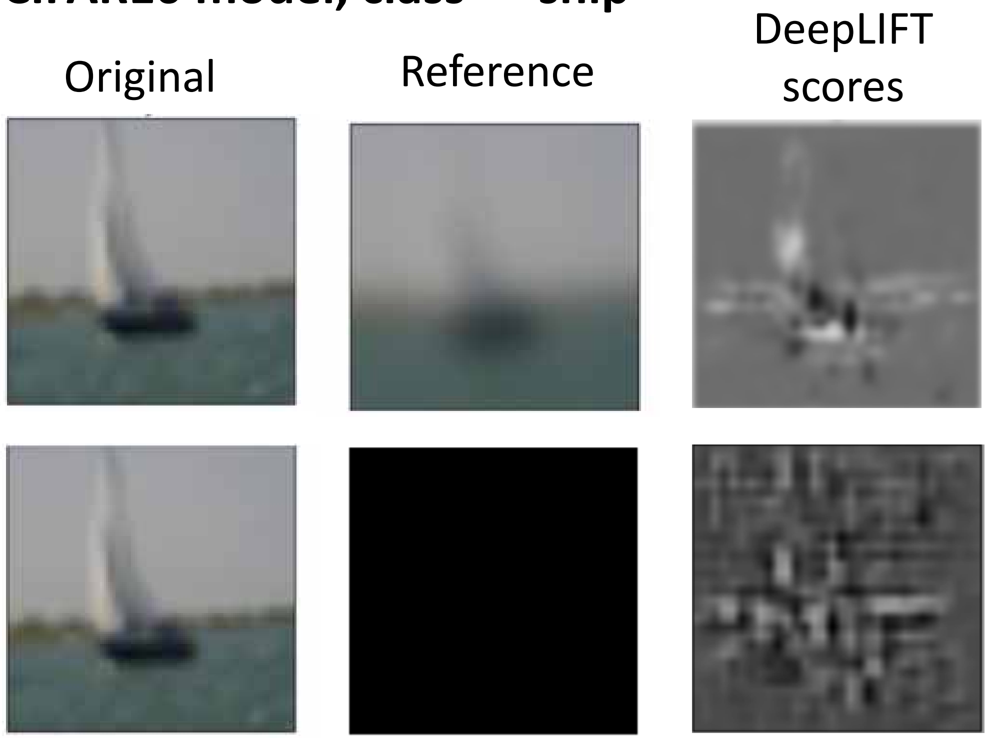
$C_{\Delta i1 \Delta y}=0.5=C_{\Delta i2 \Delta y}$

See paper for detailed backpropagation rules

Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

Choice of reference matters!

CIFAR10 model, class = “ship”



Suggestions on how to pick a reference:

- MNIST: all zeros (background)
- **Consider using a distribution of references**
  - E.g. multiple references generated by shuffling a genomic sequence

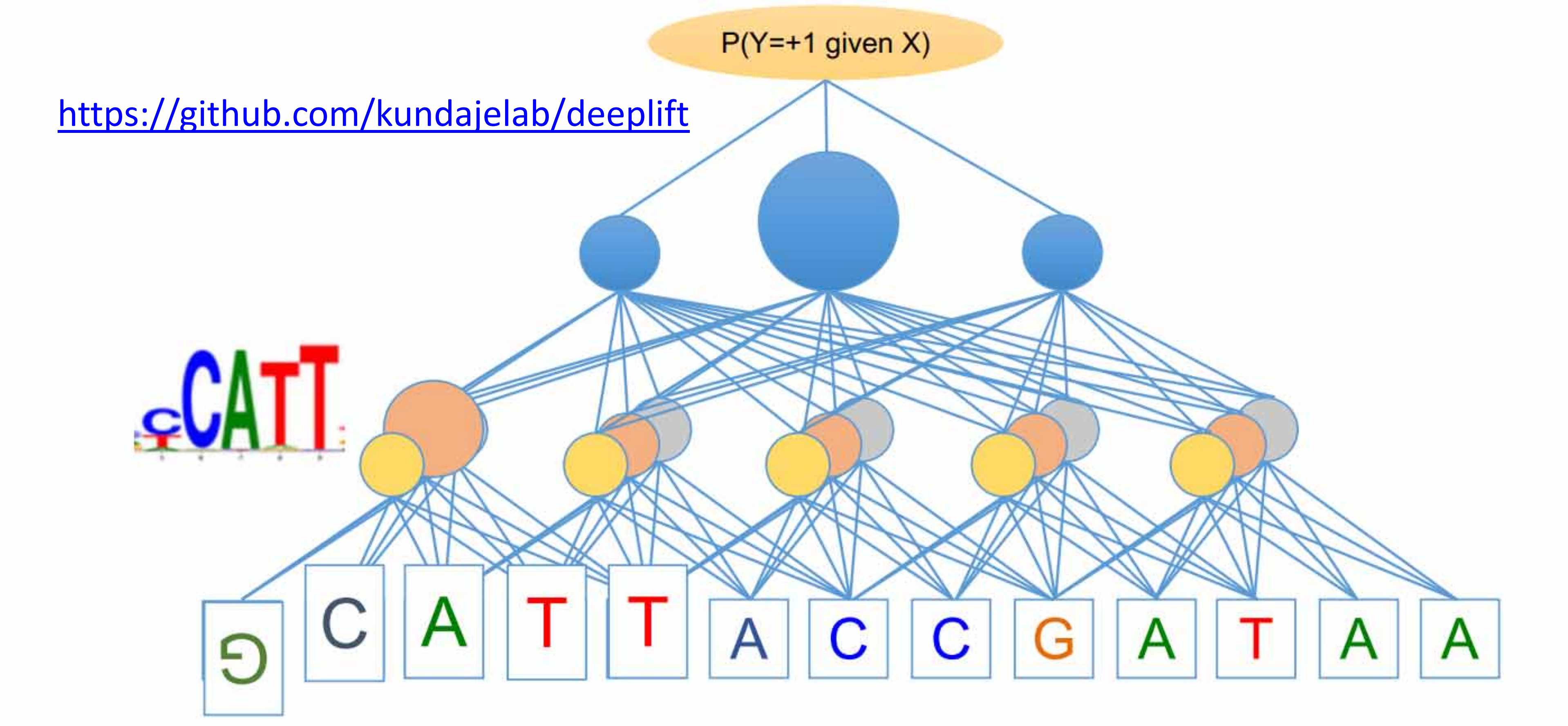
Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

Example for Interpretation: which features are important?

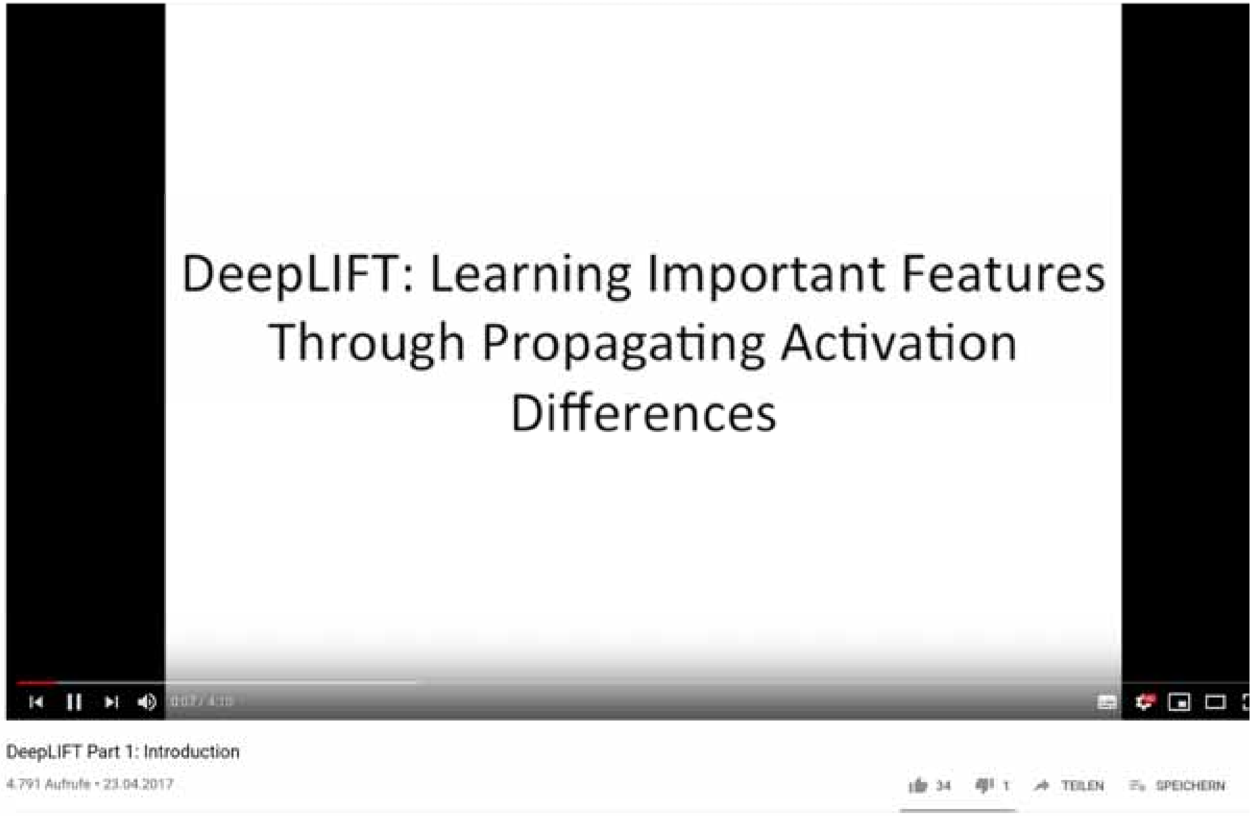
https://vimeo.com/238275076

https://pypi.org/project/deeplift

https://github.com/kundajelab/deeplift



Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.



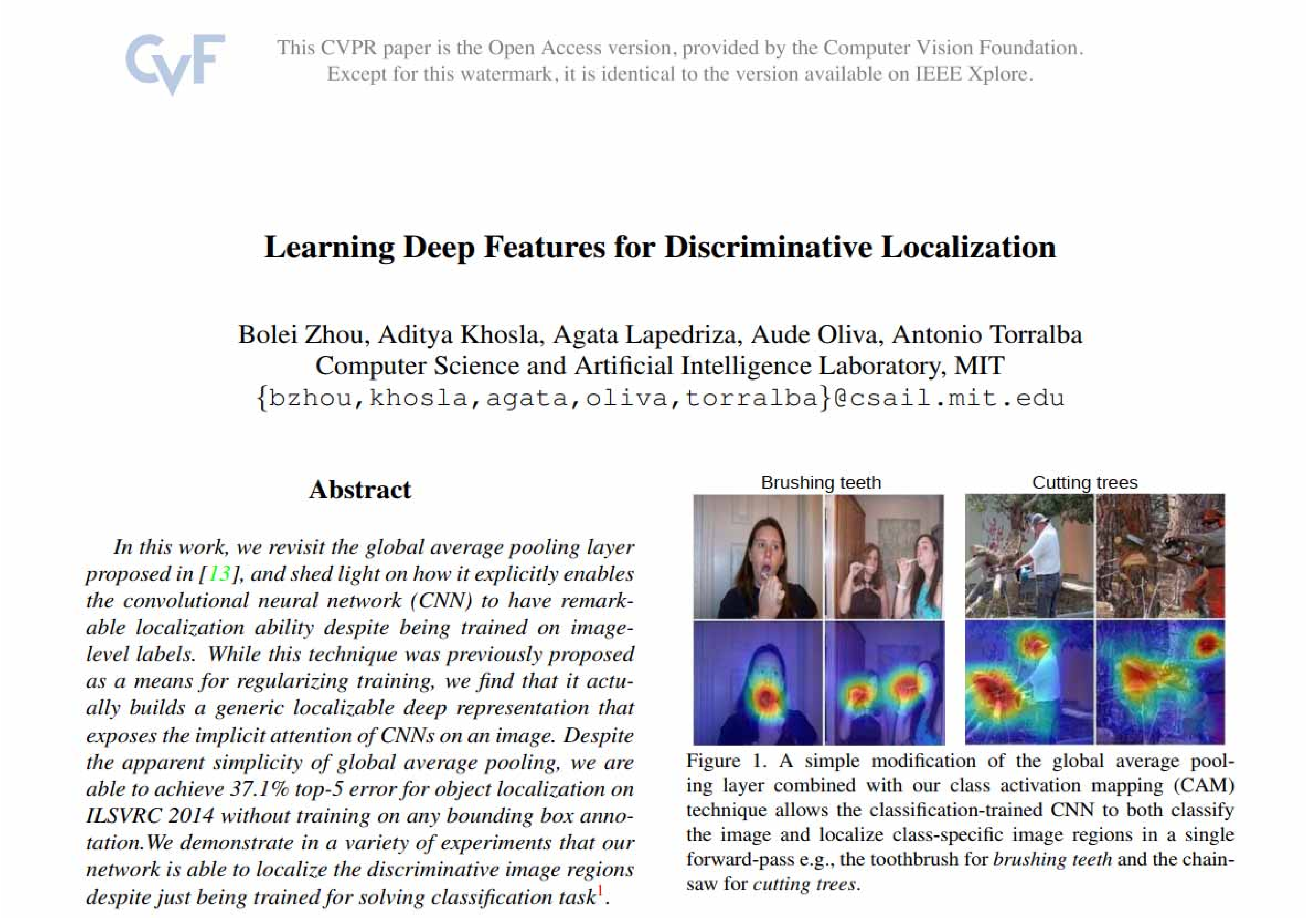
DeepLIFT: Learning Important Features Through Propagating Activation Differences

DeepLIFT Part 1: Introduction

4.791 Aufrufe • 23.04.2017

https://www.youtube.com/watch?v=v8cxYjNZAXc&list=PLJLjQOkqSRTP3cLB2cOOi_bQFw6KPGKML

04 Gradients: Grad-CAM



Learning Deep Features for Discriminative Localization

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba
Computer Science and Artificial Intelligence Laboratory, MIT
{bzhou,khosla,agata,oliva,torralba}@csail.mit.edu

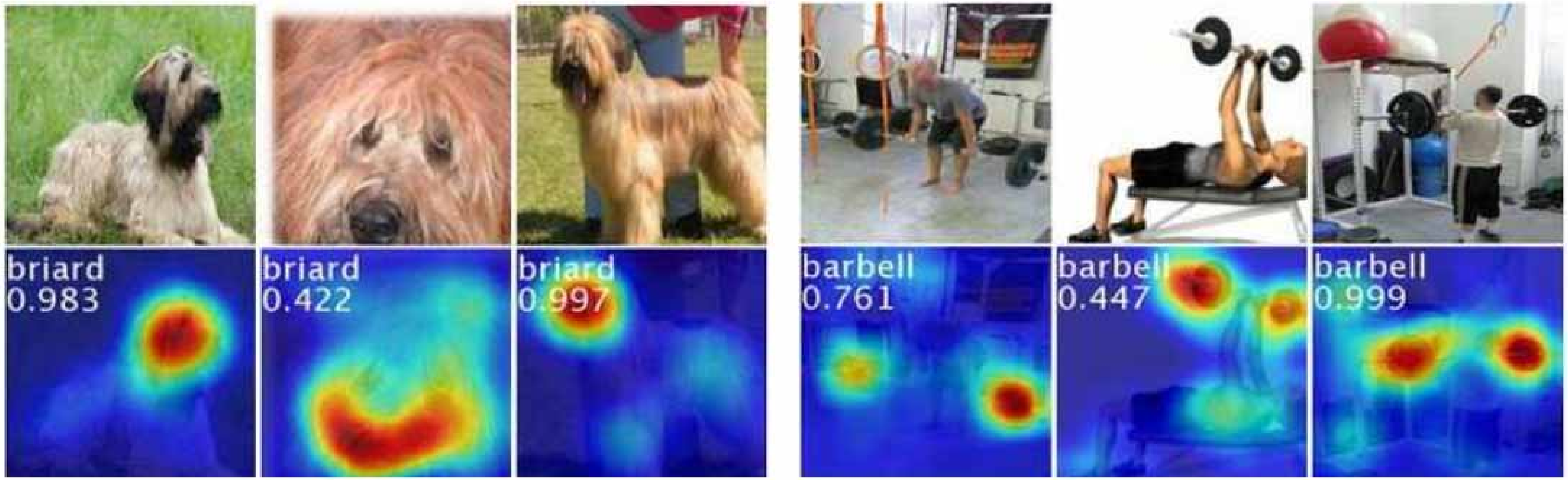
Abstract

In this work, we revisit the global average pooling layer proposed in [13], and shed light on how it explicitly enables the convolutional neural network (CNN) to have remarkable localization ability despite being trained on image-level labels. While this technique was previously proposed as a means for regularizing training, we find that it actually builds a generic localizable deep representation that exposes the implicit attention of CNNs on an image. Despite the apparent simplicity of global average pooling, we are able to achieve 37.1% top-5 error for object localization on ILSVRC 2014 without training on any bounding box annotation.We demonstrate in a variety of experiments that our network is able to localize the discriminative image regions despite just being trained for solving classification task[1].

Figure 1. A simple modification of the global average pooling layer combined with our class activation mapping (CAM) technique allows the classification-trained CNN to both classify the image and localize class-specific image regions in a single forward-pass e.g., the toothbrush for *brushing teeth* and the chainsaw for *cutting trees*.

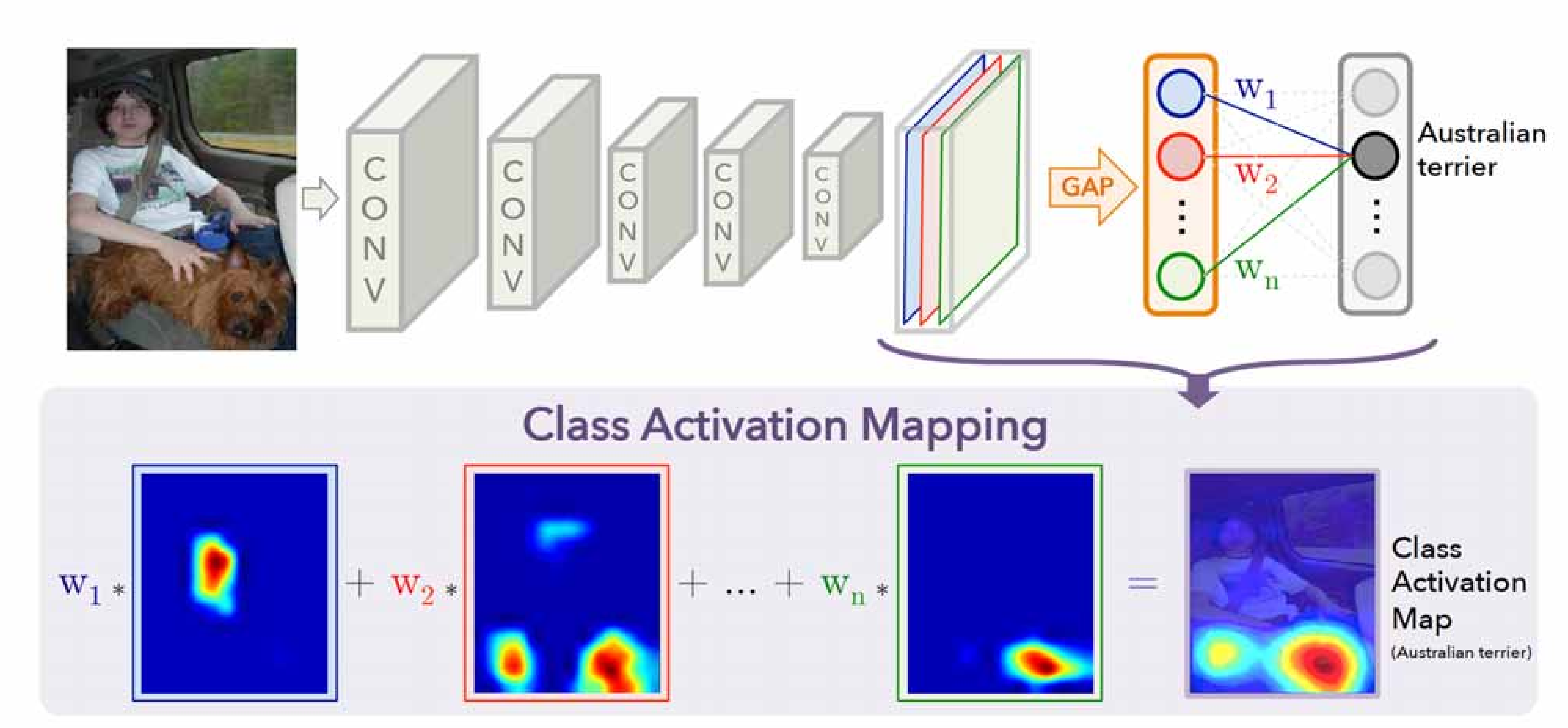
Class Activation Mapping (CAM)

- CAM relies on heatmaps highlighting image pixels for a particular class, and uses global average pooling (GAP) in CNNs.
- A class activation map for a particular category indicates the discriminative image regions used by the CNN to identify that exact category (see figure below and see next slide for the procedure).
- GAP outputs the spatial average of the feature map of each unit at the **last layer** of the CNN. A weighted sum of these values is used to generate the final output. Similarly, a weighted sum of the feature maps of the last convolutional layer to obtain the class activation maps is computed.



Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva & Antonio Torralba 2016. Learning deep features for discriminative localization. Proceedings of the IEEE conference on computer vision and pattern recognition, 2921-2929.

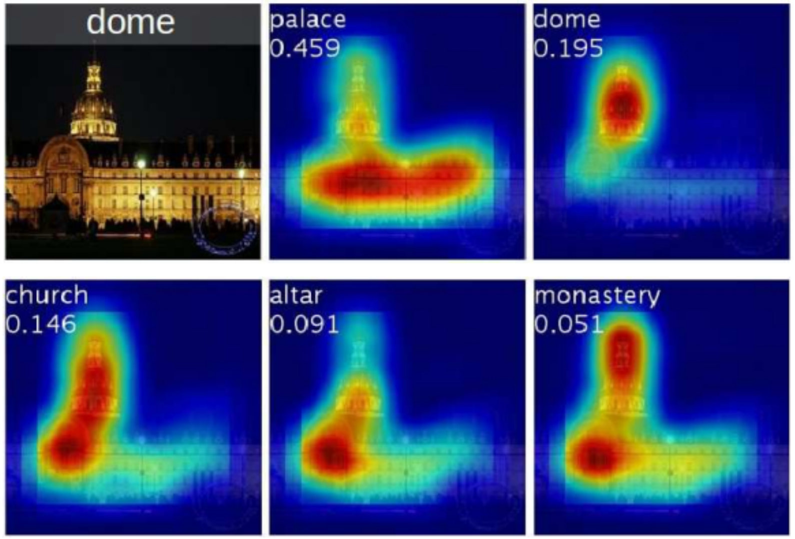
Class Activation Mapping (CAM)



Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva & Antonio Torralba 2016. Learning deep features for discriminative localization. Proceedings of the IEEE conference on computer vision and pattern recognition, 2921-2929.

Class Activation Mapping (CAM)

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x,y) = \sum_{x,y} \sum_k w_k^c f_k(x,y)$$



$$M_c(x,y) = \sum_k w_k^c f_k(x,y)$$

Thus, $S_c = \sum_{x,y} M_c(x,y)$, and hence $M_c(x,y)$ directly indicates the importance of the activation at spatial grid (x,y) leading to the classification of an image to class c.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva & Antonio Torralba 2016. Learning deep features for discriminative localization. Proceedings of the IEEE conference on computer vision and pattern recognition, 2921-2929.

Disadvantages

- The drawback of CAM is that it requires changing the network structure and then retraining it. This also implies that current architectures which don’t have the final convolutional layer — global average pooling layer — linear dense layer — structure, can’t be directly employed for this heat map technique. The technique is constrained to visualization of the latter stages of image classification or the final convolutional layers.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva & Antonio Torralba. Learning deep features for discriminative localization. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016. 2921-2929.

http://cnnlocalization.csail.mit.edu

https://jacobgil.github.io/deeplearning/class-activation-maps

https://www.youtube.com/watch?v=COjUB9Izk6E

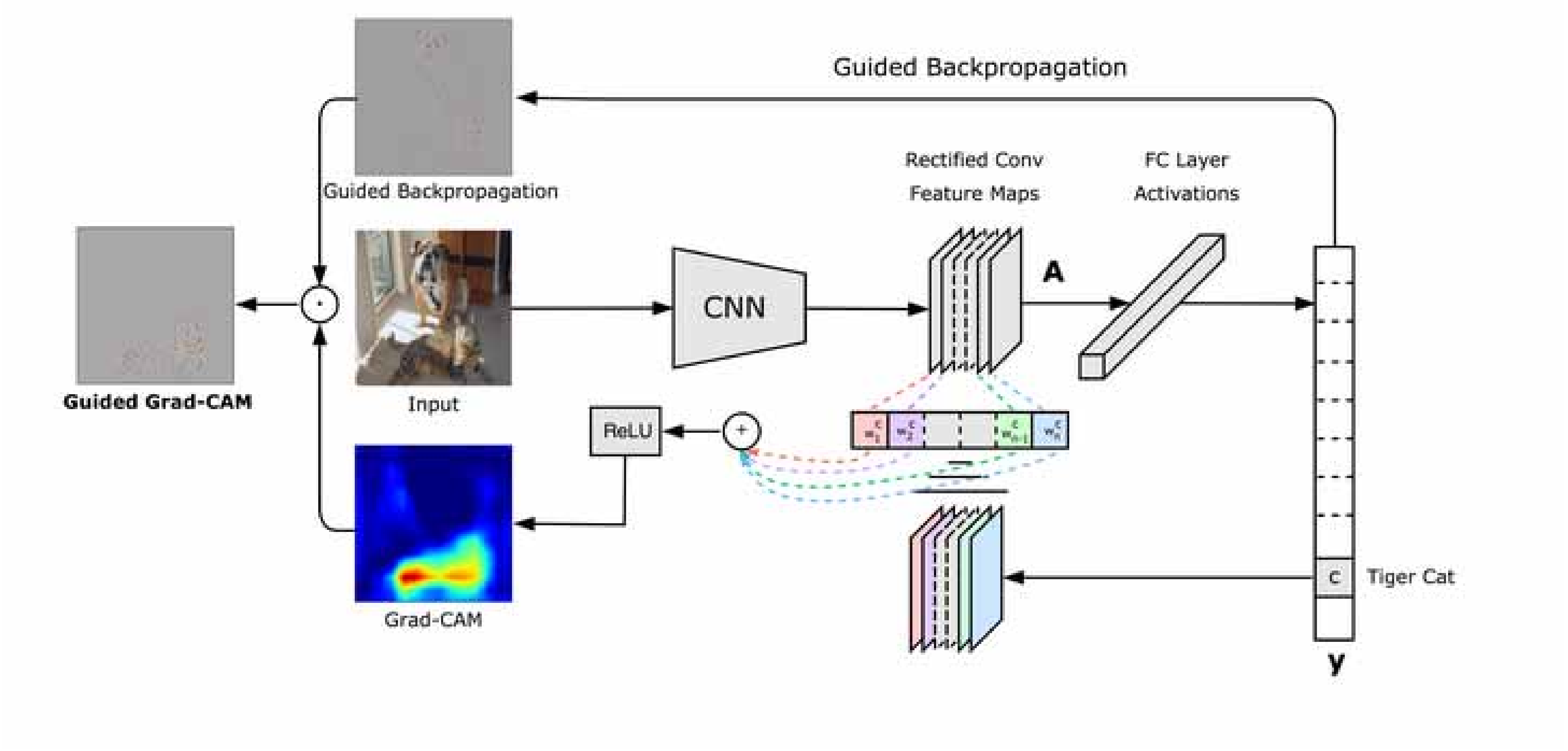
More information (additional material for the interested student)

Lecture 12 | Visualizing and Understanding

121.540 Aufrufe • 11.08.2017

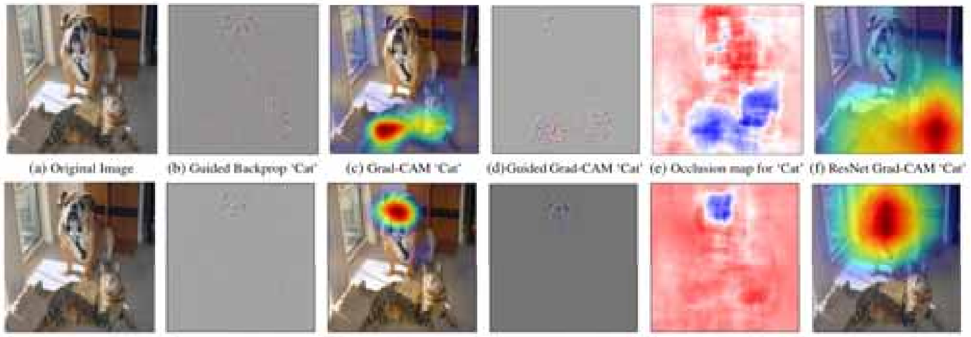
https://www.youtube.com/watch?v=6wcs6szJWMY

Grad-CAM (first paper)

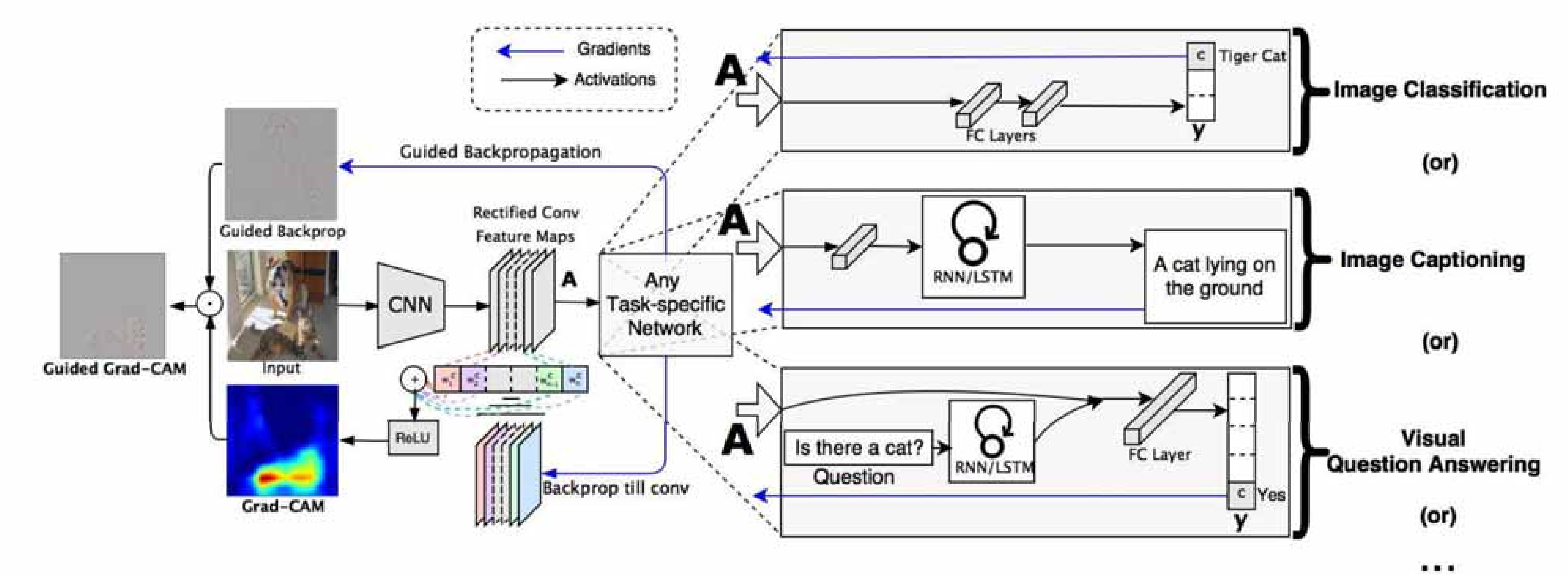


Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh & Dhruv Batra 2016. Grad-CAM: Why did you say that? arXiv:1611.07450.

Grad-CAM big picture



(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d)Guided Grad-CAM 'Cat' (e) Occlusion map for 'Cat' (f) ResNet Grad-CAM 'Cat'



Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh & Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. ICCV, 2017. 618-626.

Grad-CAM is a generalization of CAM

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

$$S^c = \sum_k \underbrace{w_k^c}_{\text{class feature weights}} \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{A_{ij}^k}_{\text{feature map}}$$

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}}$$

$$S^c = \frac{1}{Z}\sum_i\sum_j \underbrace{\sum_k w_k^c A_{ij}^k}_{L^c_{\text{CAM}}}$$

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh & Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. ICCV, 2017. 618-626.

05 Integrated Gradients

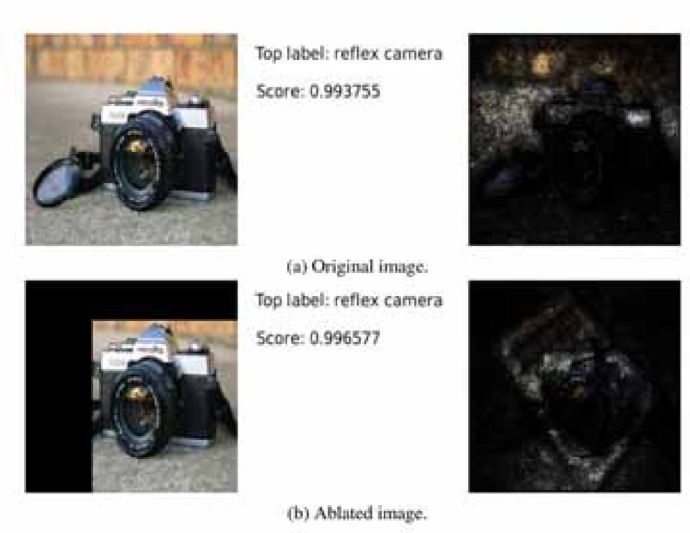
Integrated Gradients

- combines the Implementation Invariance of Gradients along with the Sensitivity of techniques e.g. LRP, or DeepLift
- Formally, suppose we have a function $F : R^n \rightarrow [0; 1]$ that represents a deep network. Specifically, let $x \in R^n$ be the input at hand, and $x' \in R^n$ be the baseline input. For image networks, the baseline could be the black image, while for text models it could be the zero embedding vector.
- We consider the straight line path (in Rn) from the baseline x' to the input x, and compute the gradients at all points along the path. Integrated gradients are obtained by *cumulating* these gradients. Specifically, integrated gradients are defined as the path integral of the gradients along the straight line path from the baseline to the input x.

$$\text{IntegratedGrads}_i(x) ::= (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} \, d\alpha \qquad (1)$$

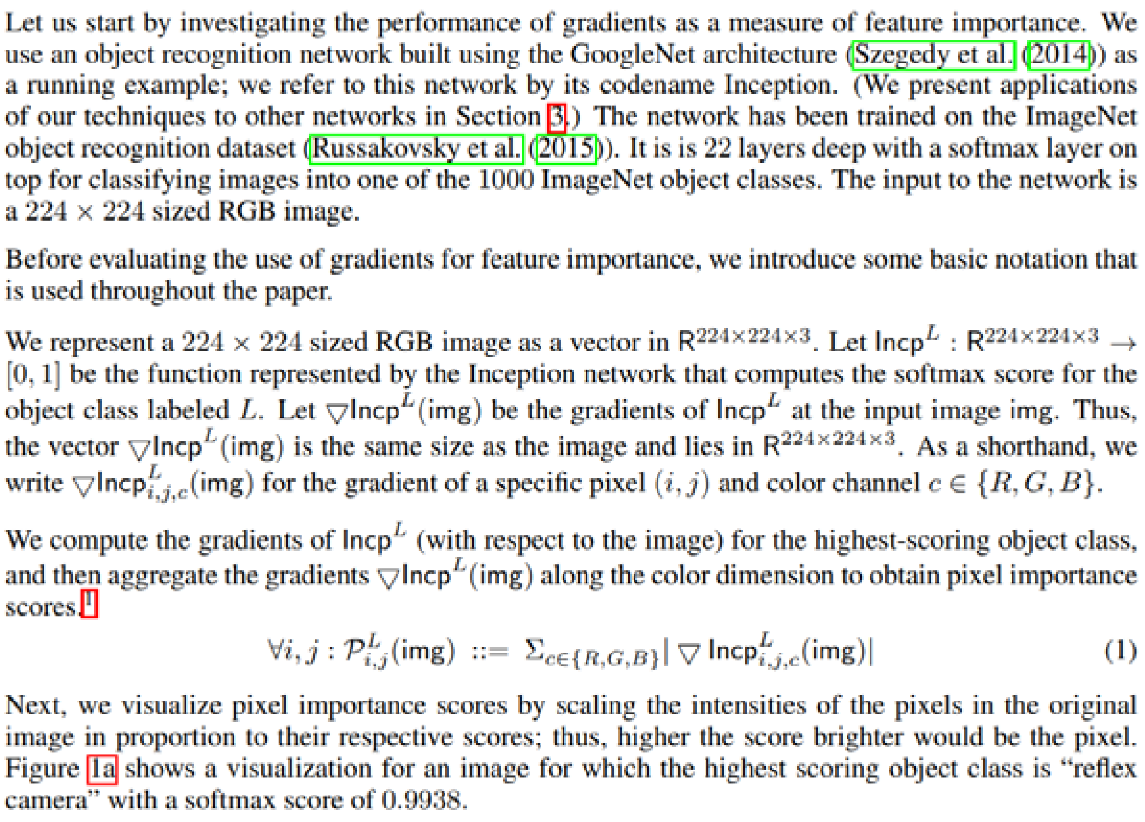
Mukund Sundararajan, Ankur Taly & Qiqi Yan. Axiomatic attribution for deep networks. Proceedings of the 34th International Conference on Machine Learning, 2017. JMLR, 3319-3328.

Gradients of counterfactuals



(a) Original image.

(b) Ablated image.



Let us start by investigating the performance of gradients as a measure of feature importance. We use an object recognition network built using the GoogleNet architecture (Szegedy et al. (2014)) as a running example; we refer to this network by its codename Inception. (We present applications of our techniques to other networks in Section 3.) The network has been trained on the ImageNet object recognition dataset (Russakovsky et al. (2015)). It is is 22 layers deep with a softmax layer on top for classifying images into one of the 1000 ImageNet object classes. The input to the network is a 224 × 224 sized RGB image.

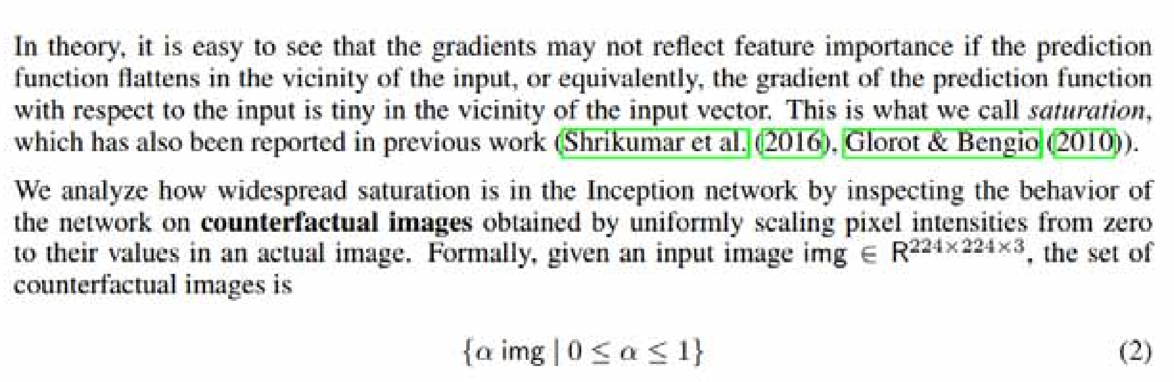
Before evaluating the use of gradients for feature importance, we introduce some basic notation that is used throughout the paper.

We represent a 224 × 224 sized RGB image as a vector in $\mathsf{R}^{224\times224\times3}$. Let $\mathsf{Incp}^L : \mathsf{R}^{224\times224\times3} \rightarrow [0,1]$ be the function represented by the Inception network that computes the softmax score for the object class labeled L. Let $\triangledown\mathsf{Incp}^L(\mathsf{img})$ be the gradients of Incp^L at the input image img. Thus, the vector $\triangledown\mathsf{Incp}^L(\mathsf{img})$ is the same size as the image and lies in $\mathsf{R}^{224\times224\times3}$. As a shorthand, we write $\triangledown\mathsf{Incp}^L_{i,j,c}(\mathsf{img})$ for the gradient of a specific pixel (i,j) and color channel $c \in \{R,G,B\}$.

We compute the gradients of Incp^L (with respect to the image) for the highest-scoring object class, and then aggregate the gradients $\triangledown\mathsf{Incp}^L(\mathsf{img})$ along the color dimension to obtain pixel importance scores.[1]

$$\forall i,j : \mathcal{P}^L_{i,j}(\mathsf{img}) \ ::= \ \Sigma_{c\in\{R,G,B\}} |\triangledown \mathsf{Incp}^L_{i,j,c}(\mathsf{img})| \qquad (1)$$

Next, we visualize pixel importance scores by scaling the intensities of the pixels in the original image in proportion to their respective scores; thus, higher the score brighter would be the pixel. Figure 1a shows a visualization for an image for which the highest scoring object class is "reflex camera" with a softmax score of 0.9938.



In theory, it is easy to see that the gradients may not reflect feature importance if the prediction function flattens in the vicinity of the input, or equivalently, the gradient of the prediction function with respect to the input is tiny in the vicinity of the input vector. This is what we call *saturation*, which has also been reported in previous work (Shrikumar et al. (2016), Glorot & Bengio (2010)).

We analyze how widespread saturation is in the Inception network by inspecting the behavior of the network on **counterfactual images** obtained by uniformly scaling pixel intensities from zero to their values in an actual image. Formally, given an input image $\mathsf{img} \in \mathsf{R}^{224\times224\times3}$, the set of counterfactual images is

$$\{\alpha\ \mathsf{img} \mid 0 \le \alpha \le 1\} \qquad (2)$$

Mukund Sundararajan, Ankur Taly & Qiqi Yan 2016. Gradients of counterfactuals. arXiv:1611.02639.

Thank you!