Interpretability

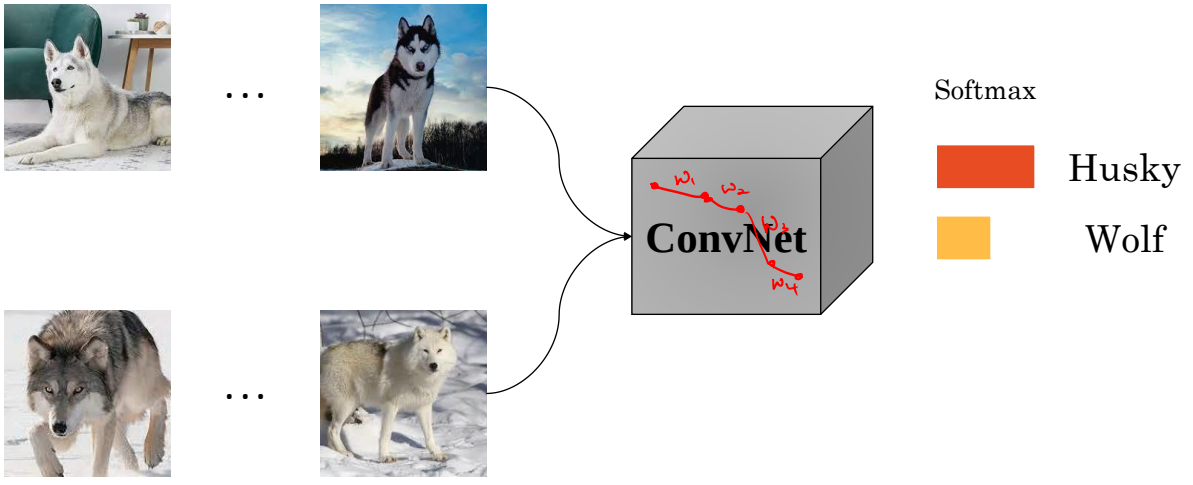
ADVANCED ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

1

Black box model

- Husky vs. Wolf classifier

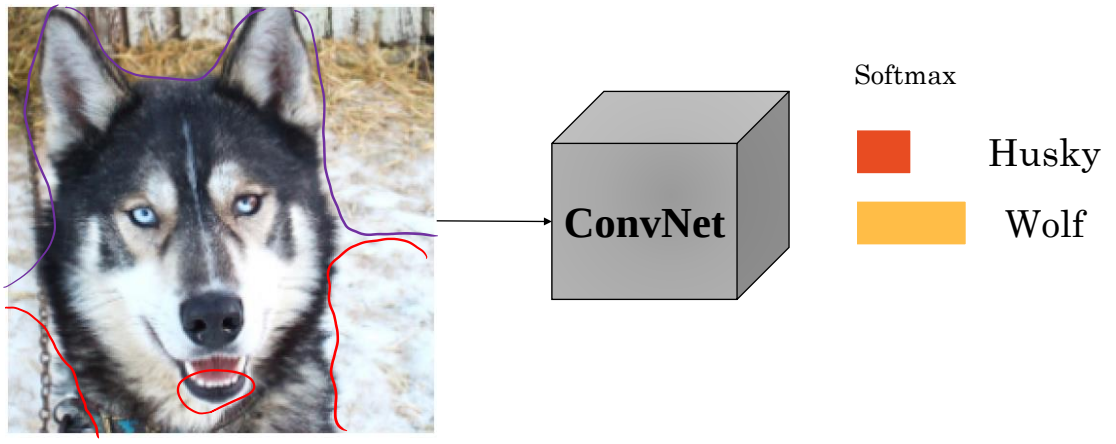


2

2

Black box model

- Why should I trust it?

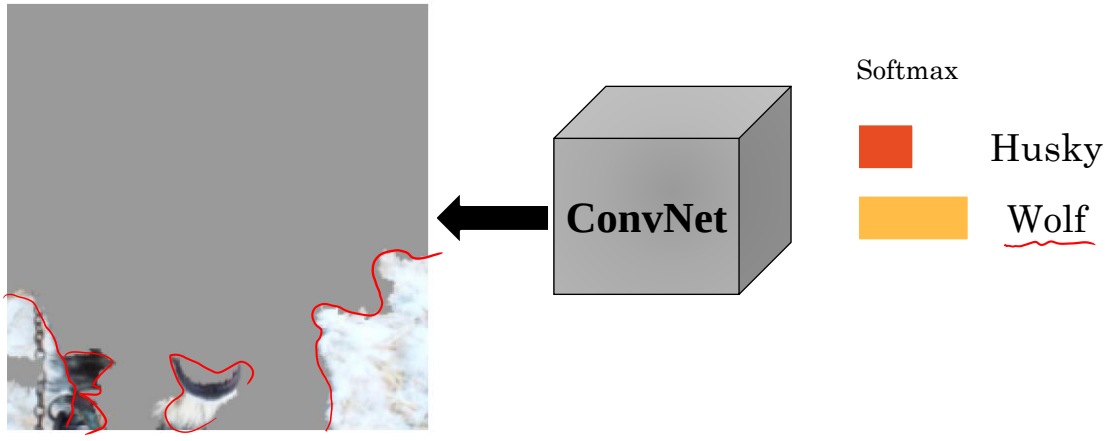


3

3

Black box model

- Bad model's prediction and explanation



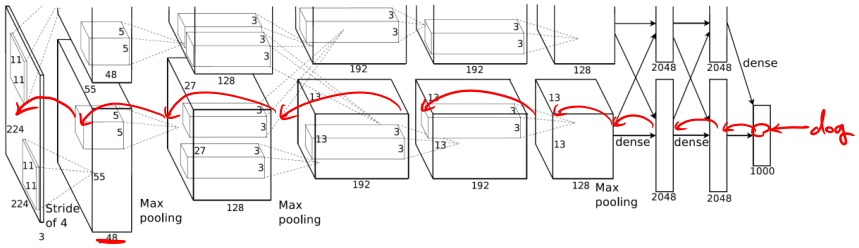
Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should I trust you?" Explaining the predictions of any classifier.

4

4

Gradient-based visualization

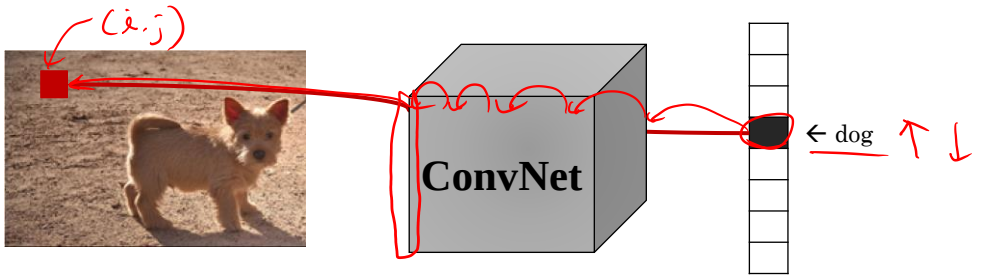
- ILSVRC-2013 dataset
  - 1.2M training images
  - 1000 classes
- ConvNet
  - conv64-conv256-conv256-conv256-conv256-full4096-full4096-full1000 (modified from below)



5

Gradient-based visualization

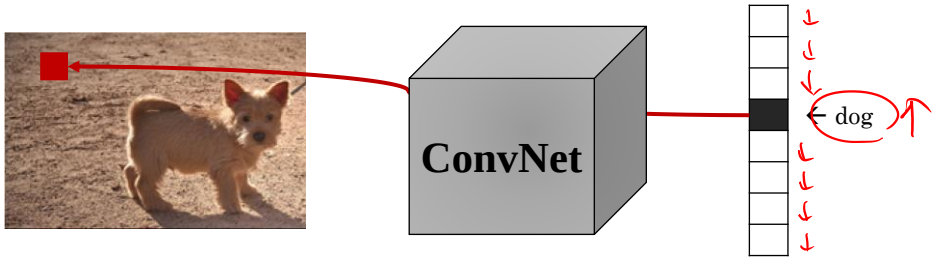
- Assume we have a trained network
  - Given an image I_0
  - A class c and a class score function $S(I)$
  - Back-propagation of the score
    - $w(i,j) = \left.\frac{\partial S_c}{\partial I}\right|_{I_0(i,j)}$



6

Gradient-based visualization

- $S_c(I)$ is the unnormalized class score, rather than soft-max probability
  - If soft-max is used, the maximization of the class posterior can be achieved by minimizing the scores of other classes

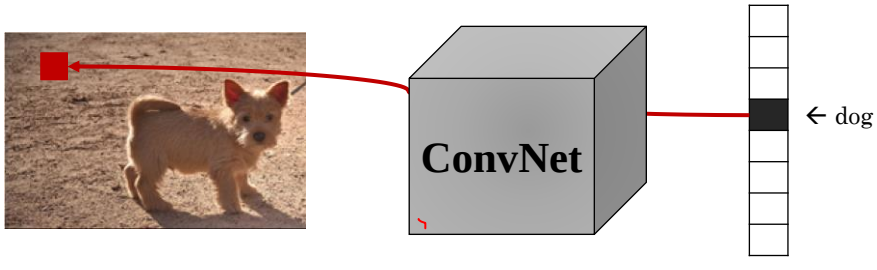


7

Saliency Map

- Single channel image
  - $M(i,j) = |w(i,j)|$
- Multi-channel image
  - $M(i,j) = \max w(i,j,ch)$

ch is a channel index $ch \in \{R, G, B\}$



8

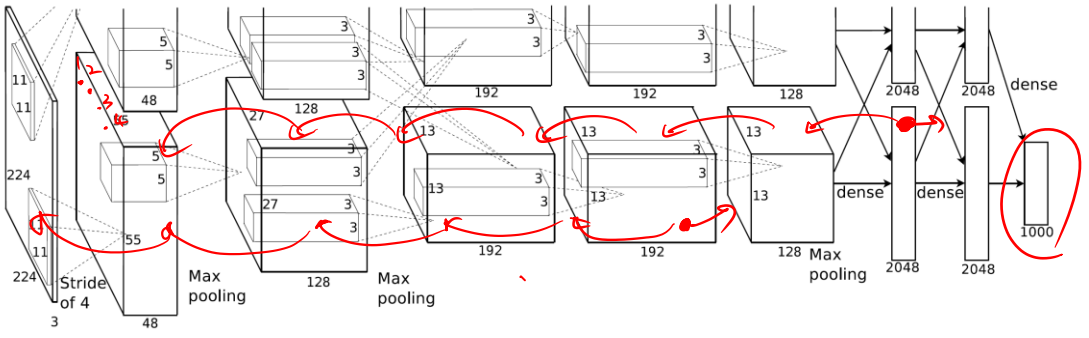
Saliency Map

Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps.

9

Visualizing via optimization

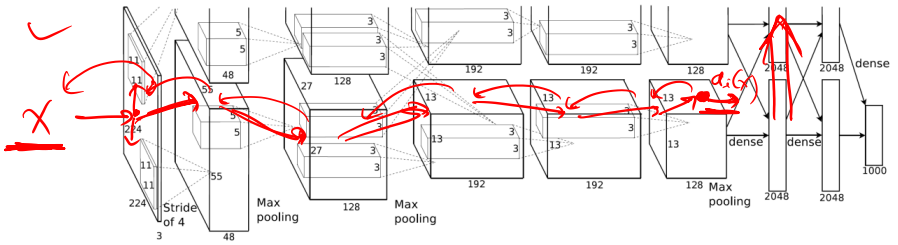
- What if we back-propagate an activation for some unit?
  - an image $x \in \mathbb{R}^{H \times W \times C}$ $C = 3$ (R, G, B)
  - an activation $a_i(x)$ for some unit i, where for simplicity i is an index that runs over all units on all layers.



10

Visualizing via optimization

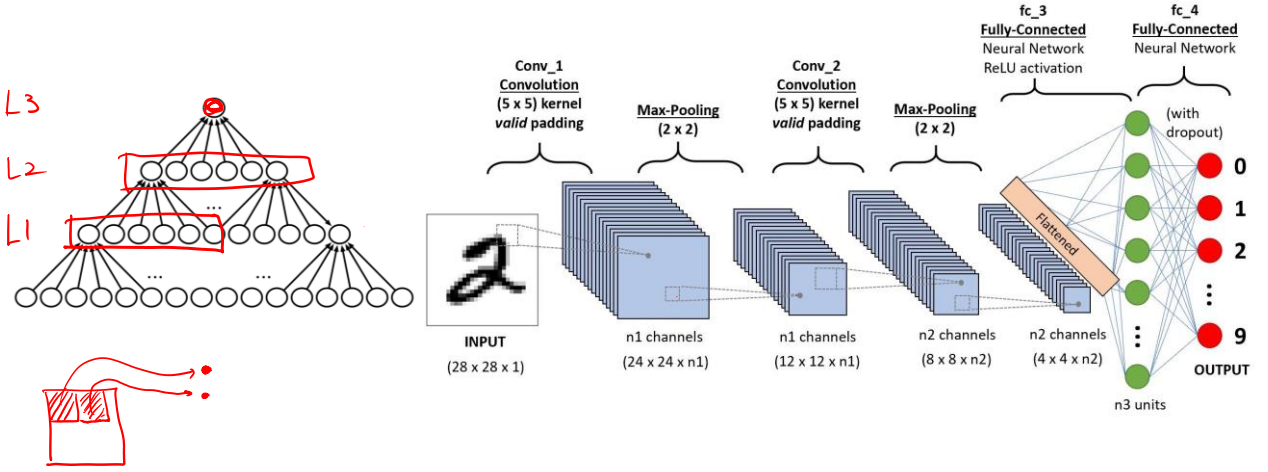
- Optimization problem
  - $x^* = \arg\max_x (a_i(x))$
  - where an activation $a_i(x)$ for some unit i
- Gradient ascent
  - $x \leftarrow x + \eta \frac{\partial a_i}{\partial x}$



11

Sparse interaction

- In ConvNet, the size of kernel is smaller than the input



12

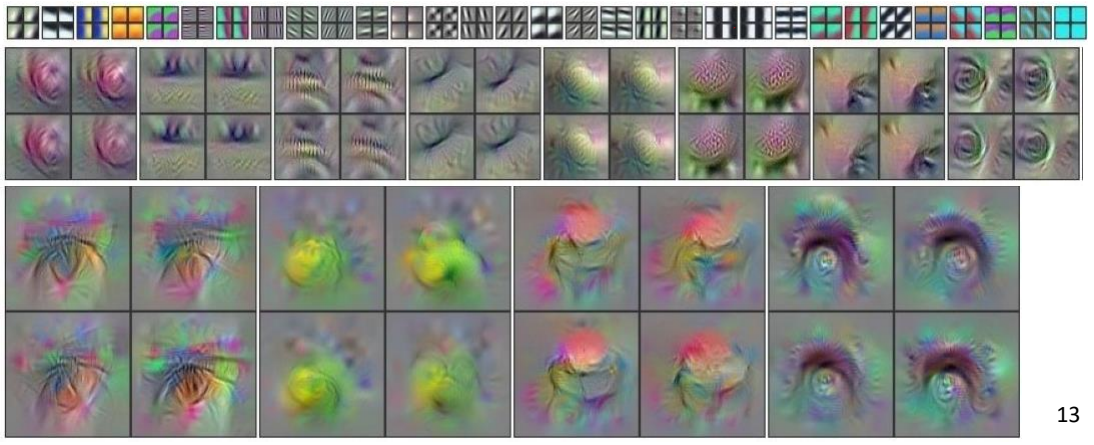
Visualization of example features

- Selected visualizations from 4 random runs
  - The images reflect the true sizes of the features at different layers.

Layer 1

Layer 2

Layer 3

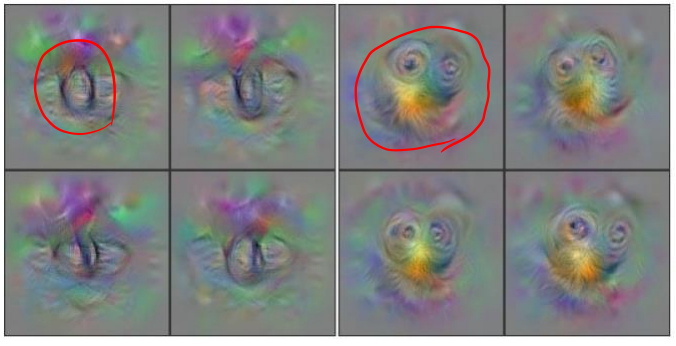


13

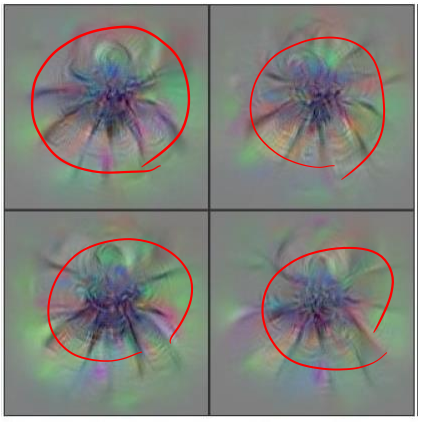
13

Visualization of example features

Layer 4



Layer 5

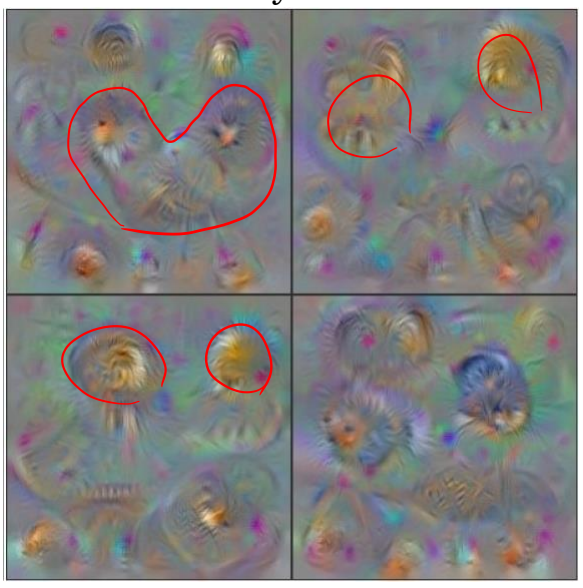


14

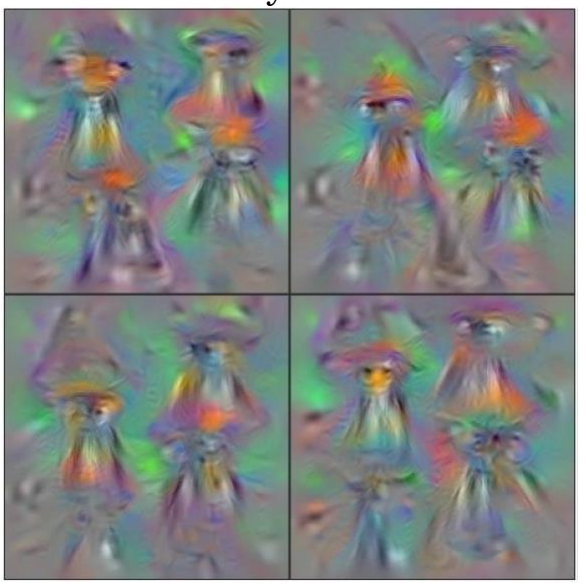
14

Visualization of example features

Layer 6



Layer 7

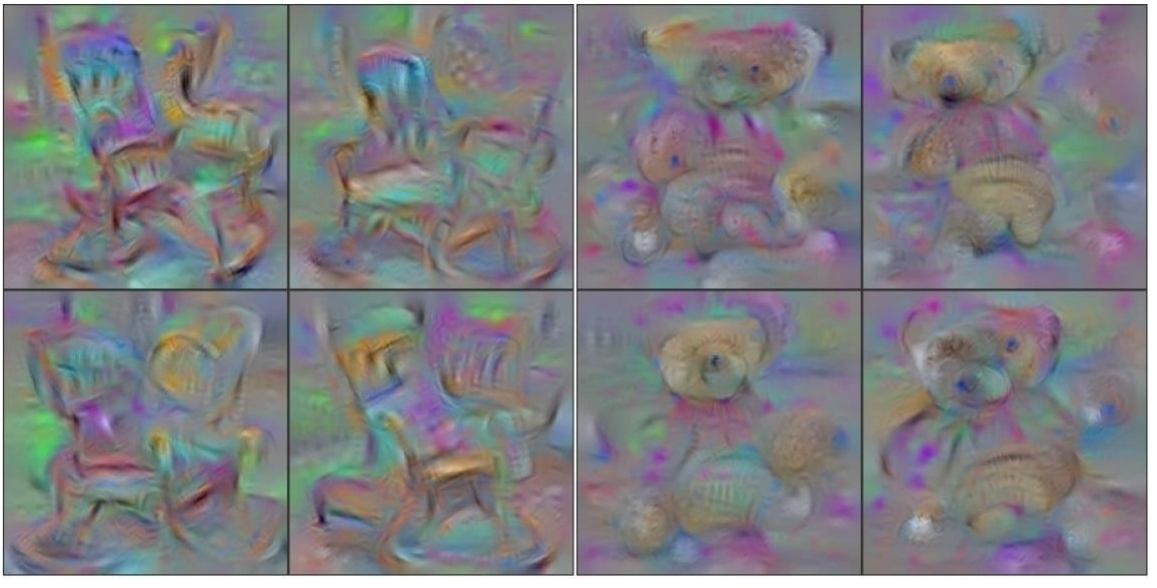


15

15

Visualization of example features

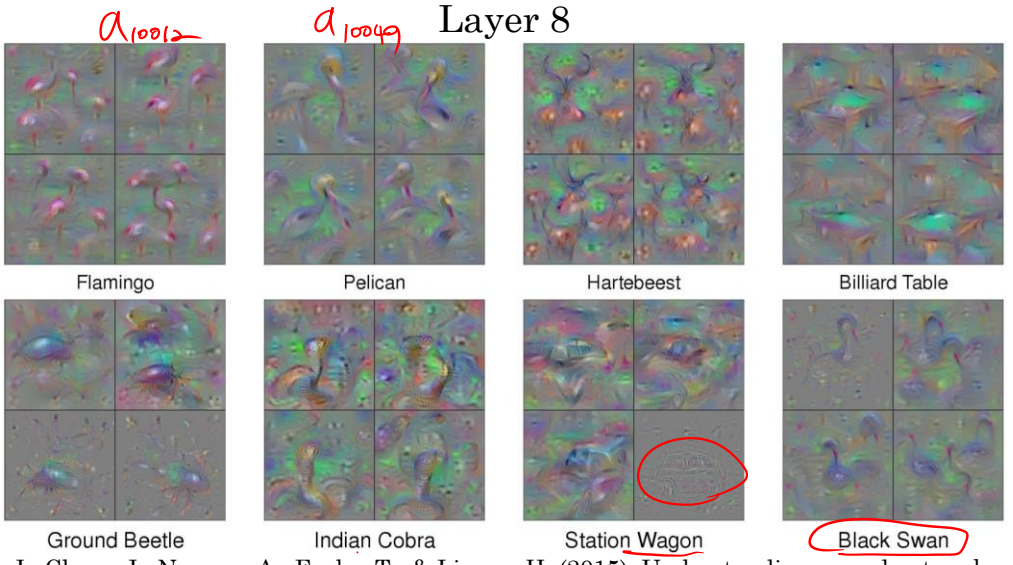
Layer 8



16

16

Visualization of example features



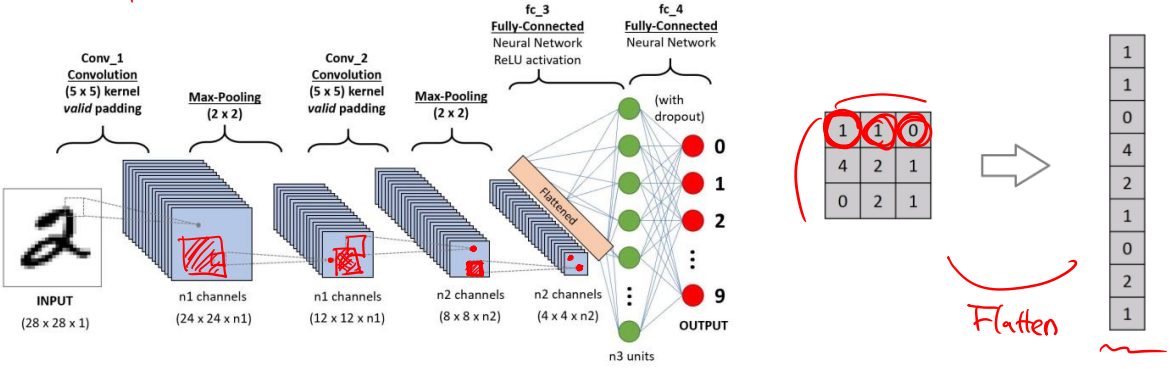
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding neural networks through deep visualization.

17

Flatten layer

- Bridge between ConvNet and DenseNet
  - Do we lose something?

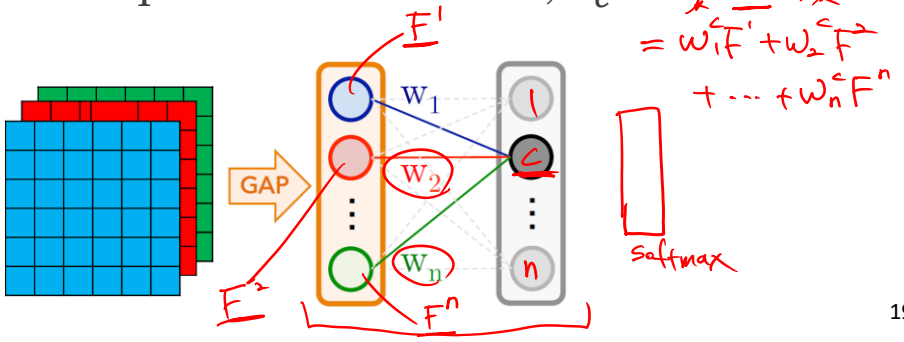
Spatial information



18

Global average pooling (GAP)

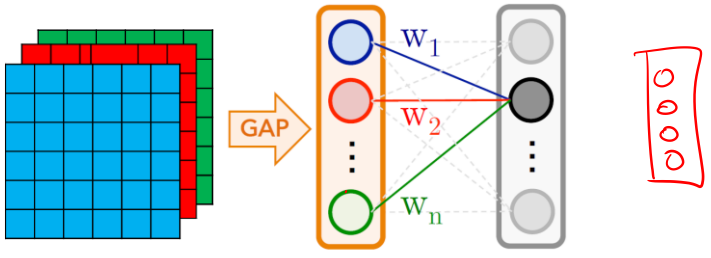
- It outputs the spatial average of the feature map of each kernel at the last convolutional layer
  - $f_k(i,j)$: activation of kernel k in the last conv at (i,j)
  - GAP: $F^k = \sum_{i,j} f_k(i,j)$
  - For a class c, the input to the softmax, $S_c = \sum_k w_k^c F_k$



19

Class activation mapping (CAM)

- $F^k = \sum_{i,j} f_k(i,j)$
- $S_c = \sum_k w_k^c F_k = \sum_k w_k^c \sum_{i,j} f_k(i,j) = \sum_{i,j} \sum_k w_k^c f_k(i,j)$
- CAM for class c: $M_c(i,j) = \sum_k w_k^c f_k(i,j)$
  - It indicates the importance of the activation at spatial grid (i,j) leading to the classification of an image to class c.

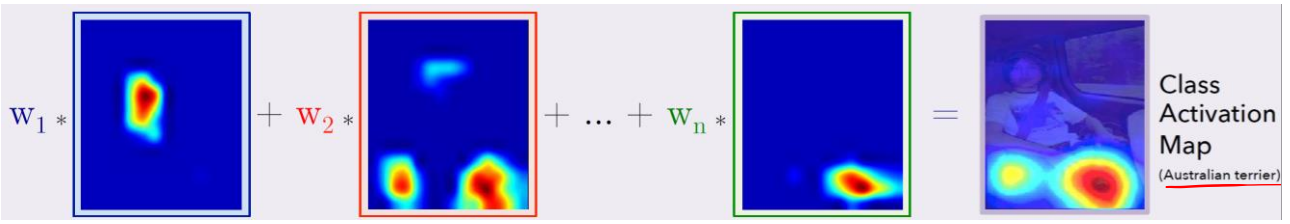


20

Class activation mapping (CAM)

- Each kernel to be activated by some visual pattern within its receptive field
  - $M_c(i,j) = \sum_k w_k^c f_k(i,j)$
  - CAM is a weighted linear sum of the presence of these visual patterns at different spatial locations

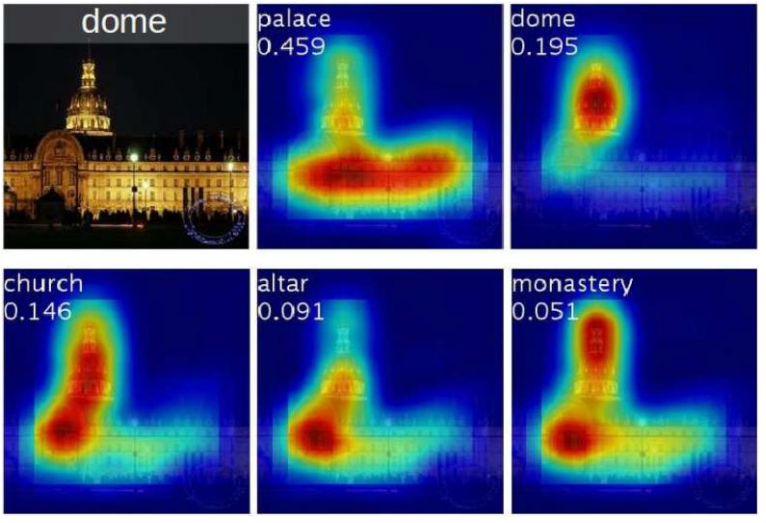
Target: Australian terrier



21

Class activation mapping (CAM)

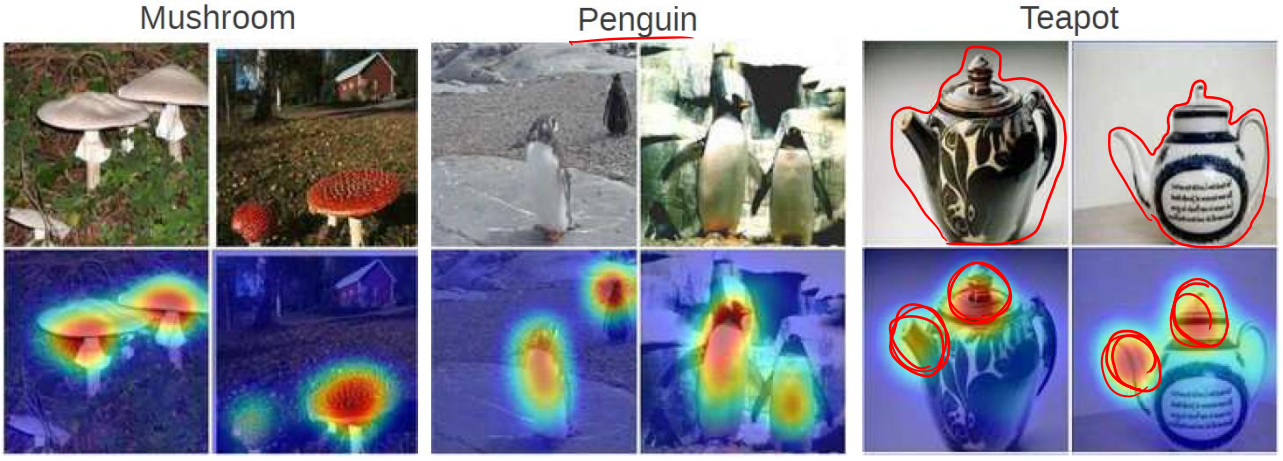
- Examples from top 5 predicted categories with ground-truth



22

Class activation mapping (CAM)

- Generic localization ability



23

Class activation mapping (CAM)

- Visual question answering

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization.

24

Class activation mapping (CAM)

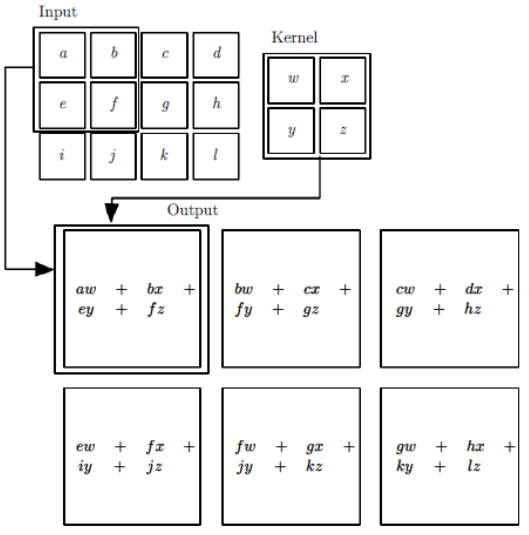
- What the CNN is looking?

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization.

25

Convolution

- The convolution operation
  - $S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n)$
  - I: 2D-input
  - K: 2D-kernel



26

Deconvolution

- $\vec{x} = [x_1\ x_2\ x_3\ x_4\ x_5\ x_6]^T$
- $\vec{w} = [w_1\ w_2\ w_3]^T$
- Convolution with stride 1 without padding
  - $\vec{y} = [y_1\ y_2\ y_3\ y_4]^T \in \mathbb{R}^{4\times 1}$

$y_1 = w_1x_1 + w_2x_2 + w_3x_3$

$y_2 = w_1x_2 + w_2x_3 + w_3x_4$

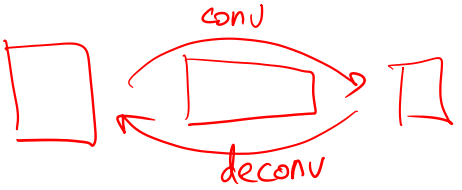
$y_3 = w_1x_3 + w_2x_4 + w_3x_5$

$y_4 = w_1x_4 + w_2x_5 + w_3x_6$

27

Deconvolution

- $y_1 = w_1x_1 + w_2x_2 + w_3x_3$
- $y_2 = w_1x_2 + w_2x_3 + w_3x_4$
- $y_3 = w_1x_3 + w_2x_4 + w_3x_5$
- $y_4 = w_1x_4 + w_2x_5 + w_3x_6$



$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & w_3 & 0 & 0 & 0 \\ 0 & w_1 & w_2 & w_3 & 0 & 0 \\ 0 & 0 & w_1 & w_2 & w_3 & 0 \\ 0 & 0 & 0 & w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

28

Deconvolution

- $\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & w_3 & 0 & 0 & 0 \\ 0 & w_1 & w_2 & w_3 & 0 & 0 \\ 0 & 0 & w_1 & w_2 & w_3 & 0 \\ 0 & 0 & 0 & w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$
- $\vec{y} = \vec{W}\vec{x}$, $\vec{x} = \vec{W}^{-1}\vec{y}$
- Inverse of non-square matrix?
  - In general, no.

29

Deconvolution

- $\vec{W} \in \mathbb{R}^{m \times n}$, if
  - Each of rows and columns are linearly independent
    - Full rank
  - $m < n$
    - $W^{-1} = W^T(WW^T)^{-1}$
  - $m > n$
    - $W^{-1} = (W^TW)^{-1}W^T$
- $\vec{x} = \vec{W}^{-1}\vec{y} = W^T(WW^T)^{-1}\vec{y}$

30

Deconvolution

- $\vec{x} = \vec{W}^{-1}\vec{y} = W^T(WW^T)^{-1}\vec{y}$
- Complexity of $n \times n$ matrix inversion
  - Optimized CW-like algorithms: $O(n^{2.373})$
- Assumption: strong and irresponsible(?)

$WW^T = I$ $\quad$ $WW^{-1} = I$

$\vec{x} = W^T\vec{y}$

31

31

Deconvolution (Transposed convolution)

- $$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & w_3 & 0 & 0 & 0 \\ 0 & w_1 & w_2 & w_3 & 0 & 0 \\ 0 & 0 & w_1 & w_2 & w_3 & 0 \\ 0 & 0 & 0 & w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ w_3 & w_2 & w_1 & 0 \\ 0 & w_3 & w_2 & w_1 \\ 0 & 0 & w_3 & w_2 \\ 0 & 0 & 0 & w_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

(6×4) $\quad$ (4×1) $\quad$ (6×1)

32

32

Transposed convolution

- $$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ w_3 & w_2 & w_1 & 0 \\ 0 & w_3 & w_2 & w_1 \\ 0 & 0 & w_3 & w_2 \\ 0 & 0 & 0 & w_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$
- Complexity of matrix multiplication
  - $m \times n$ matrix and $n \times p$ matrix
  - $O(mnp)$

33

33

Transposed convolution

- $$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} w_1 & 0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ w_3 & w_2 & w_1 & 0 \\ 0 & w_3 & w_2 & w_1 \\ 0 & 0 & w_3 & w_2 \\ 0 & 0 & 0 & w_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$x_1 = w_1 y_1 \quad x_2 = w_2 y_1 + w_1 y_2 \quad x_3 = w_3 y_1 + w_2 y_2 + w_1 y_3$

$x_4 = w_3 y_2 + w_2 y_3 + w_1 y_4, \quad x_5 = w_3 y_3 + w_2 y_4, \quad x_6 = w_3 y_4$

34

34

Transposed convolution

- $x_1 = w_1 y_1$
- $x_2 = w_2 y_1 + w_1 y_2$
- $x_3 = w_3 y_1 + w_2 y_2 + w_1 y_3$
- $x_4 = w_3 y_2 + w_2 y_3 + w_1 y_4$
- $x_5 = w_3 y_3 + w_2 y_4$
- $x_6 = w_3 y_4$

$\vec{y}' = [0\ 0\ y_1\ y_2\ y_3\ y_4\ 0\ 0]^T$

$\vec{w}' = [w_3\ w_2\ w_1]^T$

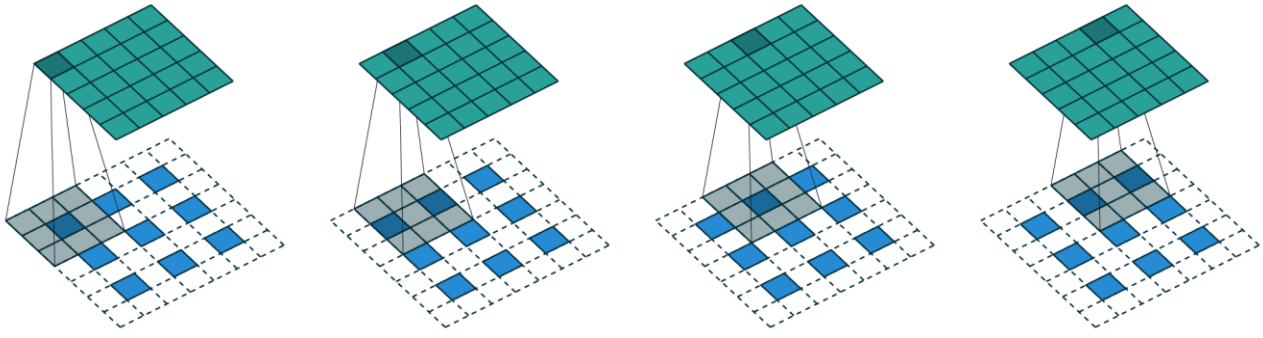
$\vec{x} = \vec{w}' \circledast \vec{y}'$

35

35

Transposed convolution

- The transposed of convolving a 3×3 kernel over a 5×5 input padded with a 1×1 border of zeros using 2 strides



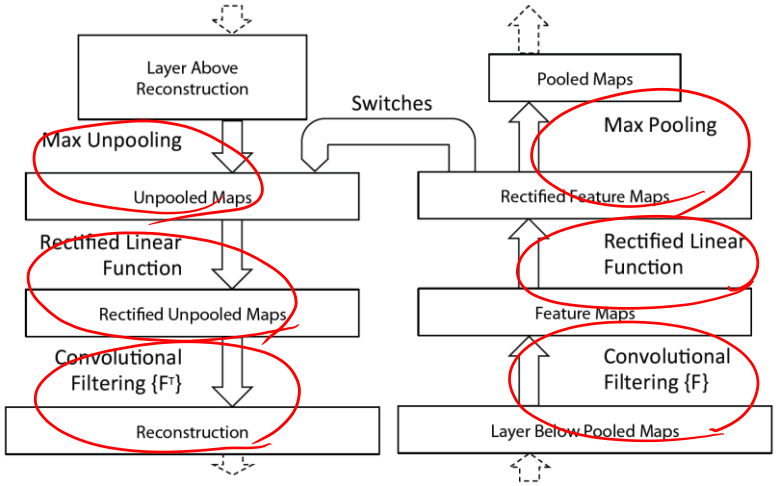
Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning.

36

36

Deconvnet

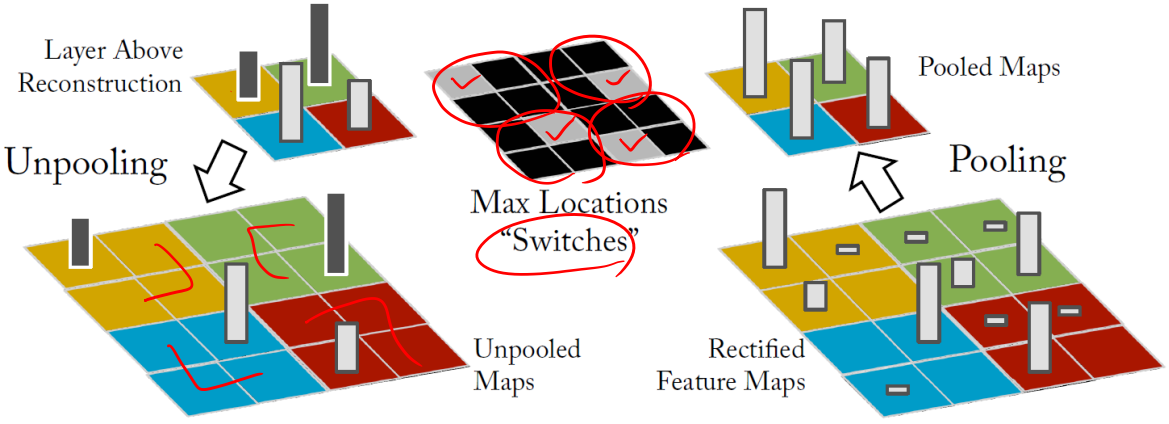
- The deconvnet will reconstruct an approximate version of the convnet features from the layer beneath.



37

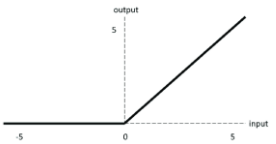
Unpooling

- Approximate inverse by recording the locations of the maxima within each pooling region in a set of **switch** variables.



38

Backward ReLU



- Backward ReLU is NOT inverse of ReLU
- Examine which units contribute to the loss

4	-3
-10	2

4	0
0	2

Switch

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

-10	0
0	7

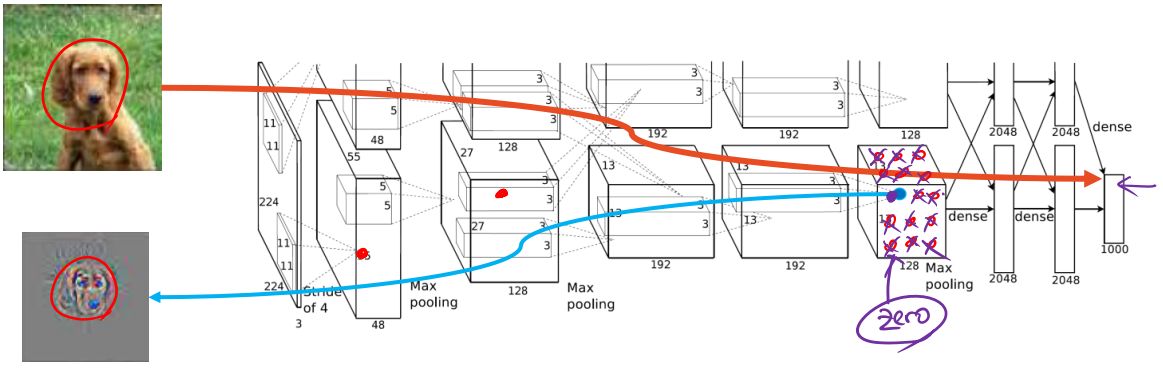
-10	-5
8	7

39

39

Deconvnet

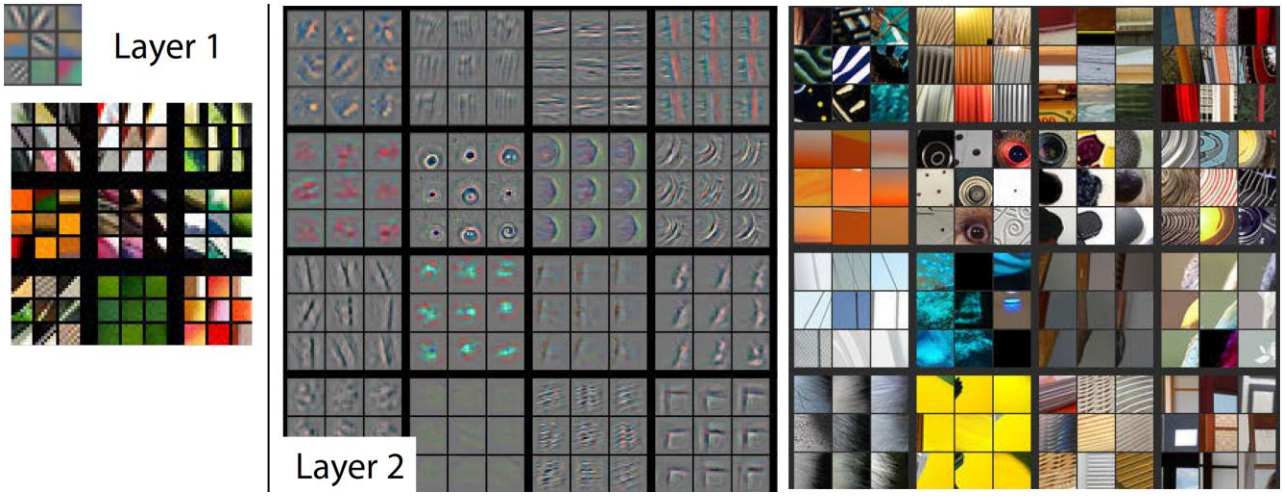
- Set all other activations in the layer to zero and pass the feature maps as input to the attached deconvnet layer



Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. 40

40

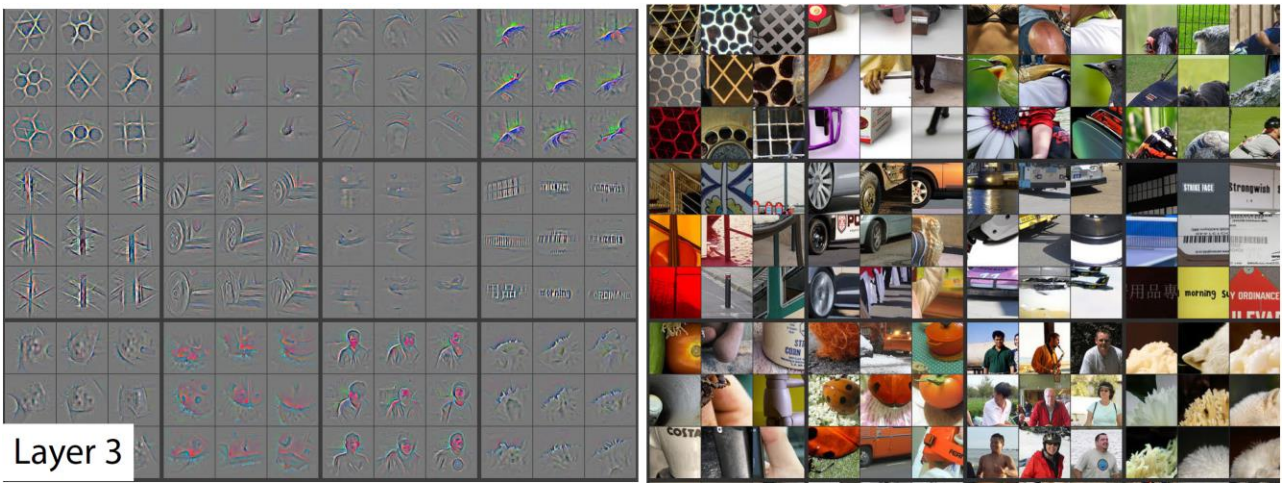
Deconvnet



Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. 41

41

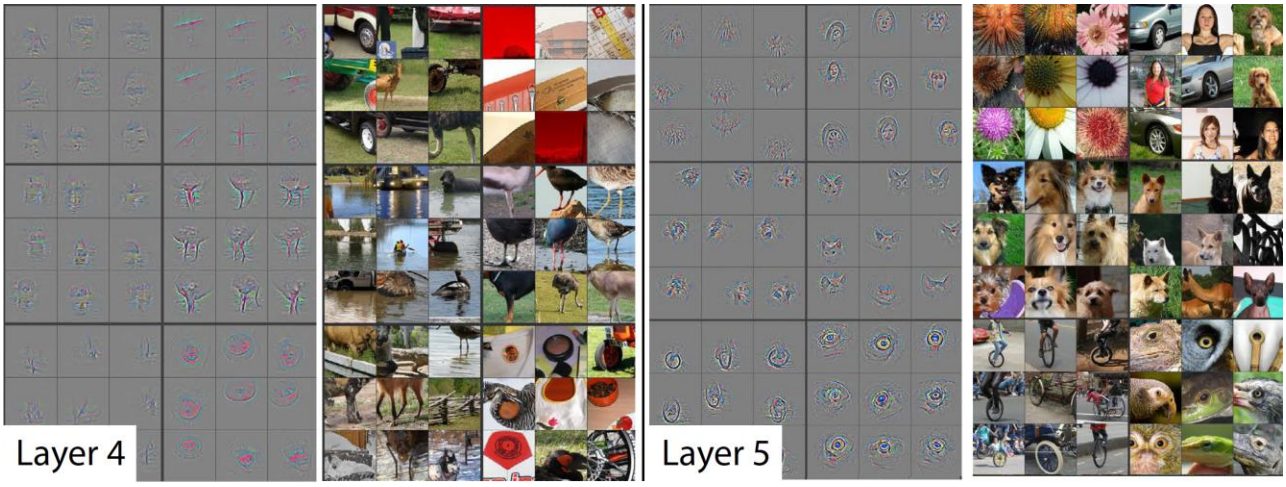
Deconvnet



Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. 42

42

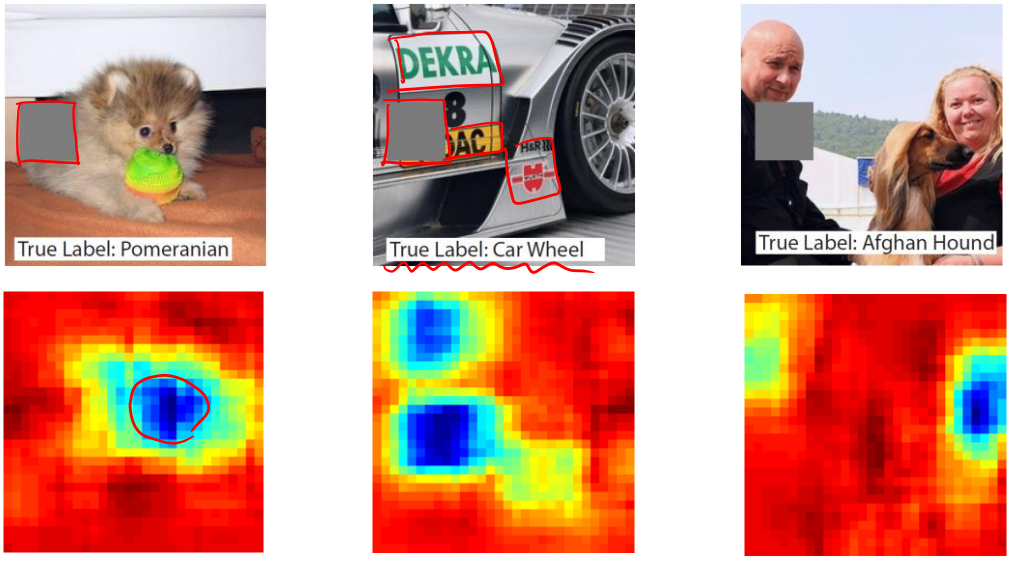
Deconvnet



Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. 43

43

Occlusion sensitivity



Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. 44

44

Deep Visualization Toolbox

yosinski.com/deepvis

#deepvis


Jason Yosinski


Jeff Clune


Anh Nguyen


Thomas Fuchs


Hod Lipson

Cornell University

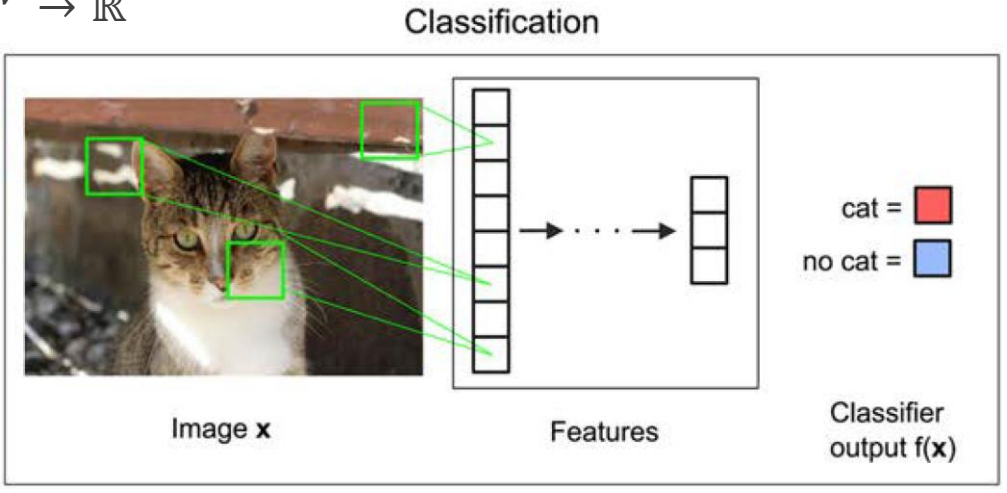
UNIVERSITY OF WYOMING

NASA Jet Propulsion Laboratory
California Institute of Technology

45

Pixel-wise decomposition

- Classifier with confidence
  - $f: \mathbb{R}^V \rightarrow \mathbb{R}$

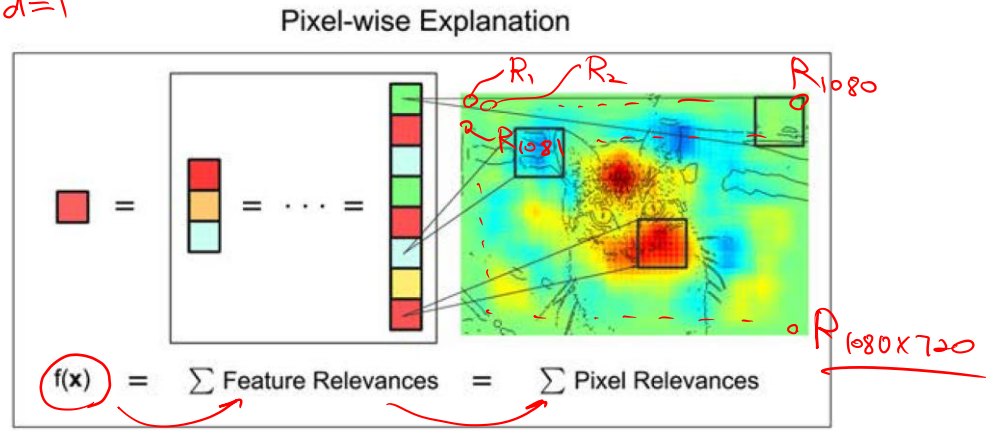


46

46

Pixel-wise decomposition

- Decompose the prediction $f(x)$ as a sum of terms of the separate input dimensions x_d respectively pixels
  - $f(x) \approx \sum_{d=1}^{V} R_d$



47

47

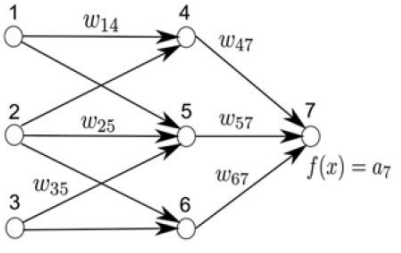
Layer-wise relevance propagation (LRP)

- Assumption
  - We have a relevance score $R_d^{(l+1)}$ at layer $l+1$
- $f(x) = \cdots = \sum_{d \in l+1} R_d^{(l+1)} = \sum_{d \in l} R_d^{(l)} = \sum_{d=1} R_d^{(1)}$

$f(x) = R_7^{(3)}$

$R_7^{(3)} = R_4^{(2)} + R_5^{(2)} + R_6^{(2)}$

$= R_1^{(1)} + R_2^{(1)} + R_3^{(1)}$



48

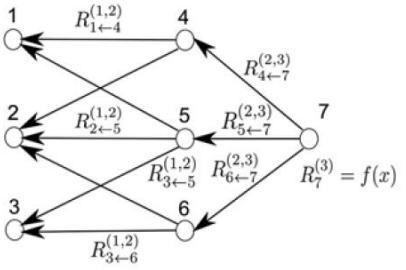
48

Layer-wise relevance propagation

- Assumptions
  - The layer-wise relevance propagation between neurons i and j can be sent along each connection

- $R_i^{(l)} = \sum_{k \in (i,k)} R_{i \leftarrow k}^{(l,l+1)}$
  - $R_3^{(1)} = R_{3 \leftarrow 5}^{(1,2)} + R_{3 \leftarrow 6}^{(1,2)}$

- $R_k^{(l+1)} = \sum_{i \in (i,k)} R_{i \leftarrow k}^{(l,l+1)}$
  - $R_4^{(2)} = R_{1 \leftarrow 4}^{(1,2)} + R_{2 \leftarrow 4}^{(1,2)}$

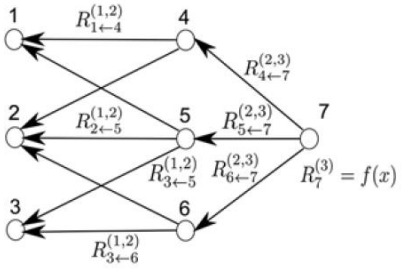


49

Layer-wise relevance propagation

- $R_i^{(l)} = \sum_{k \in (i,k)} R_{i \leftarrow k}^{(l,l+1)}$
- $R_k^{(l+1)} = \sum_{i \in (i,k)} R_{i \leftarrow k}^{(l,l+1)}$
- $\sum_k R_k^{(l+1)} = \sum_k \sum_{i \in (i,k)} R_{i \leftarrow k}^{(l,l+1)}$

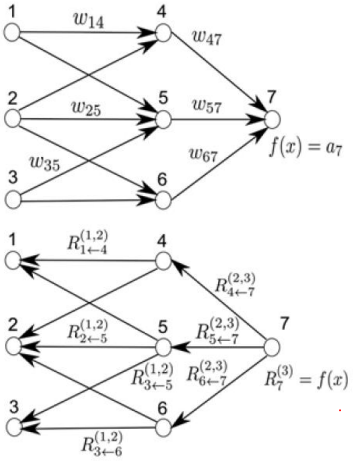
$$= \sum_{i \in (i,k)} \sum_k R_{i \leftarrow k}^{(l,l+1)} = \sum_i R_i^{(l)}$$



50

Layer-wise relevance propagation

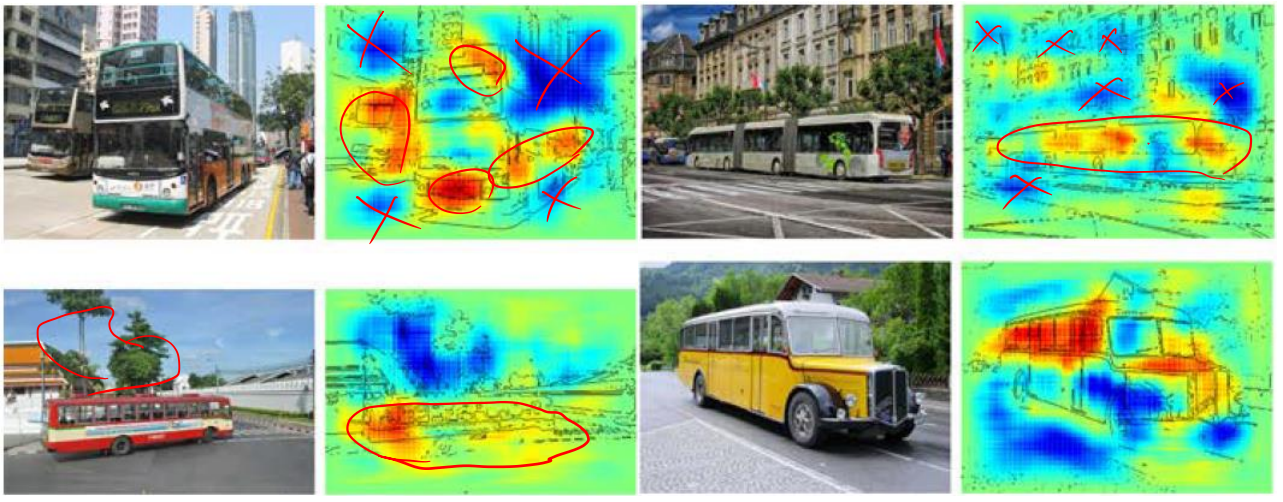
- A neuron i inputs $a_i w_{ik}$ to neuron k, provided that i has a forward connection to k
  - $R_{i\leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \frac{a_i w_{ik}}{\sum_h a_h w_{hk}}$
  - $R_{4\leftarrow 7}^{(2,3)} = R_7^{(3)} \frac{a_4 w_{47}}{\sum_{h\in\{4,5,6\}} a_h w_{h7}}$
- $R_i^{(l)} = \sum_{k\in(i,k)} R_{i\leftarrow k}^{(l,l+1)} = \sum_{k\in(i,k)} \frac{a_i w_{ik}}{\sum_h a_h w_{hk}} R_k^{(l+1)}$



Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation.

51

LRP for buses



Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation.

52

LRP for airplanes

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation.

53