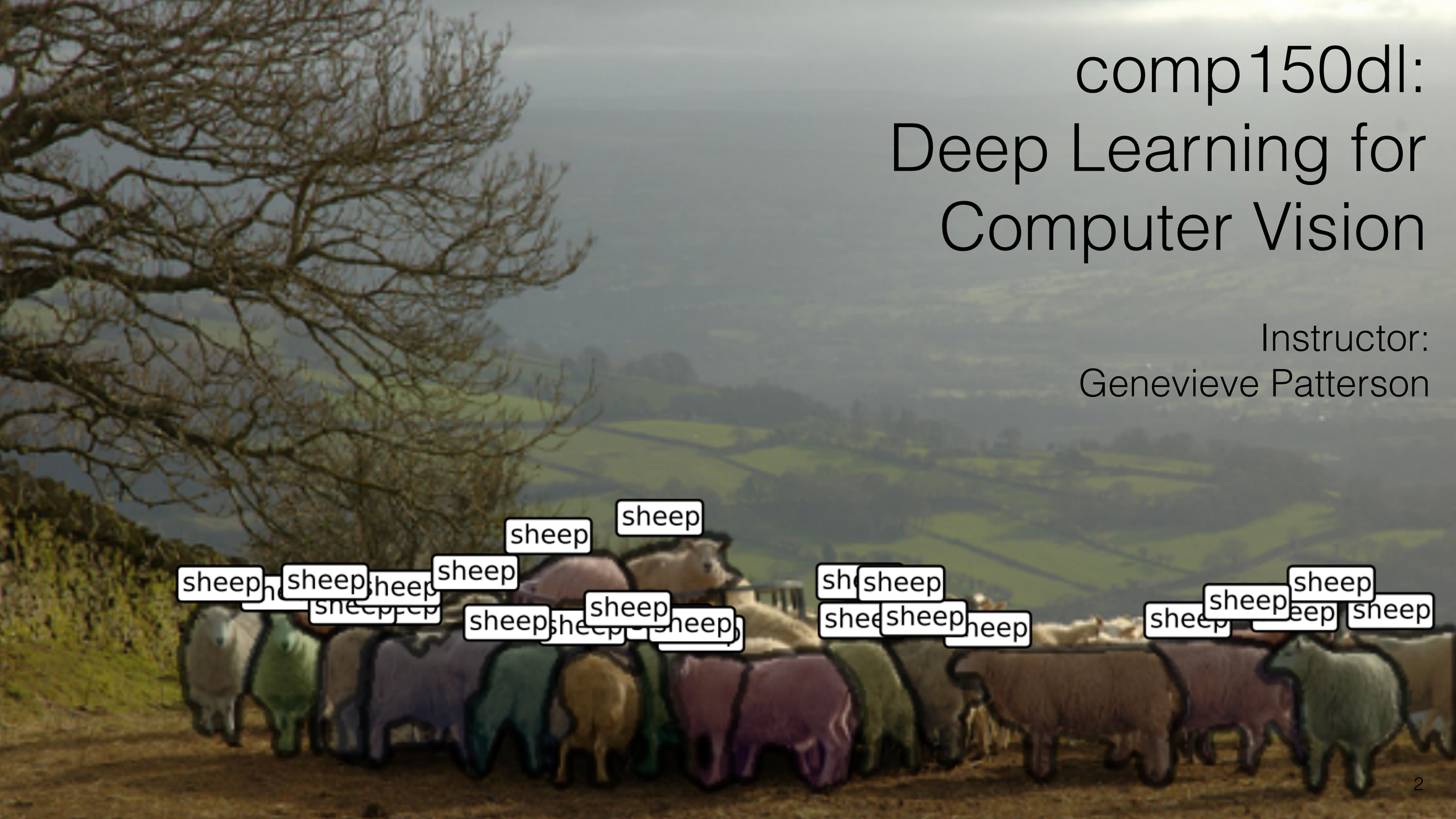
# comp150dl: Deep Learning for Computer Vision

Instructor:
Genevieve Patterson



# Ridiculously Brief History of Computer Vision

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

Artificial Intelligence Group
Vision Memo. No. 100.

July 7, 1966

# THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

# Parts-and-shape models

- Model:
  - Object as a set of parts
  - Relative locations between parts
  - Appearance of part

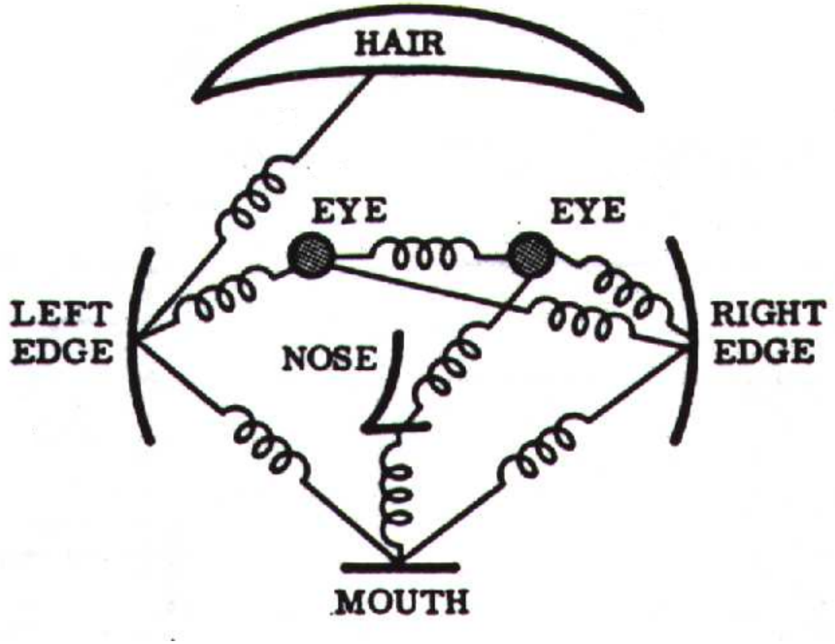
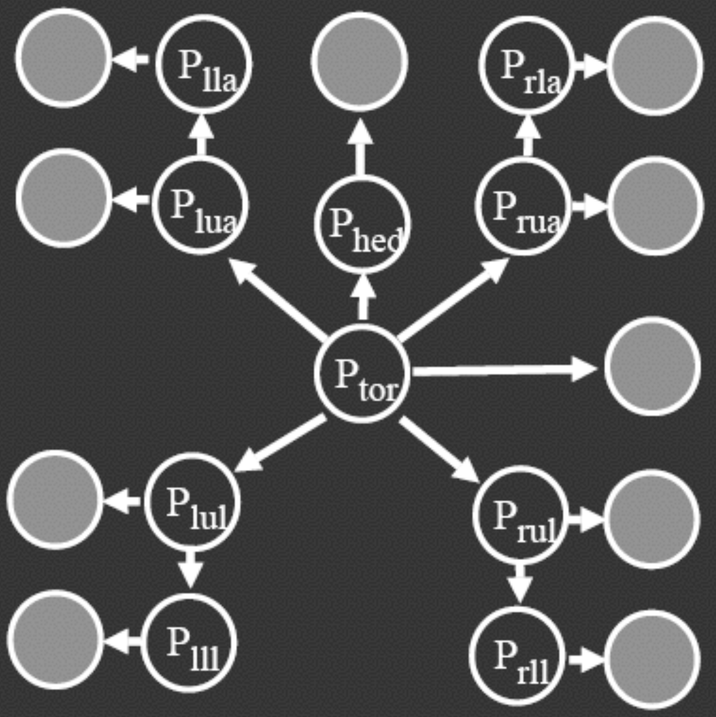


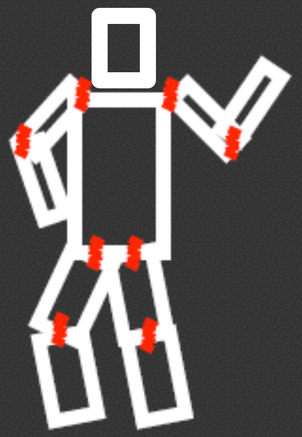


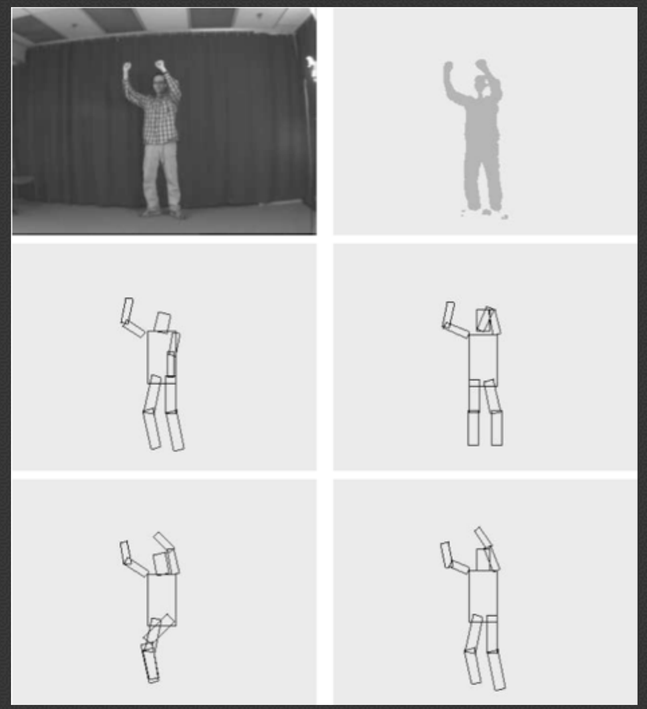

Figure from [Fischler & Elschlager 73]

# Pictorial structure model

Fischler and Elschlager(73), Felzenszwalb and Huttenlocher(00)



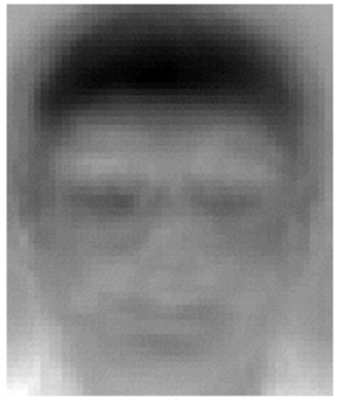


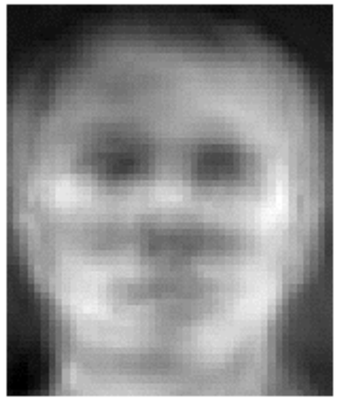
$$\Pr(P_{tor}, P_{arm}, \ldots | \mathrm{Im}) \propto \prod_{i,j} \Pr(P_i | P_j) \prod_i \Pr(\mathrm{Im}(P_i))$$

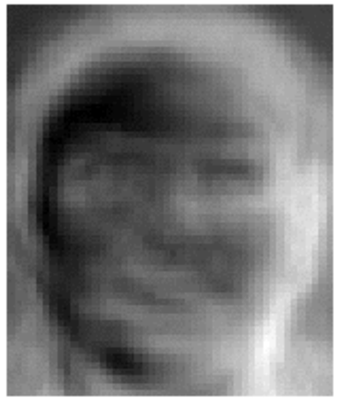
part geometry

part appearance

# Eigenfaces (Turk & Pentland, 1991)







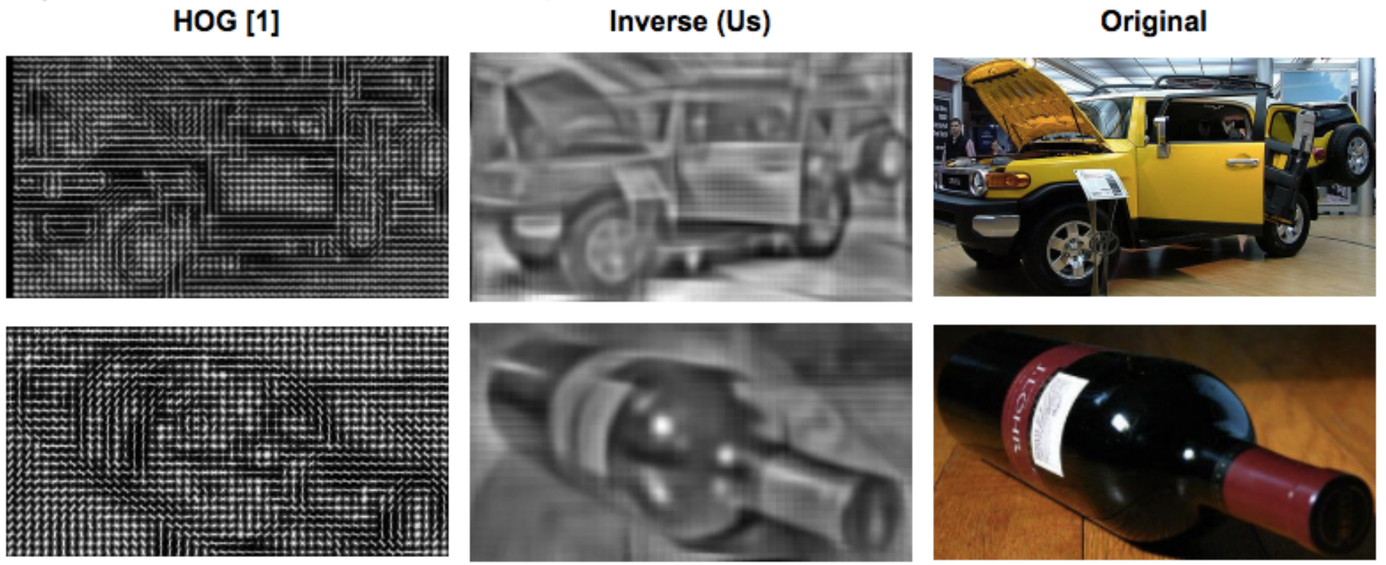
# Local features for object instance recognition - SIFT




D. Lowe (1999, 2004)

# Carefully Considered Features

- Histogram of Oriented Gradients



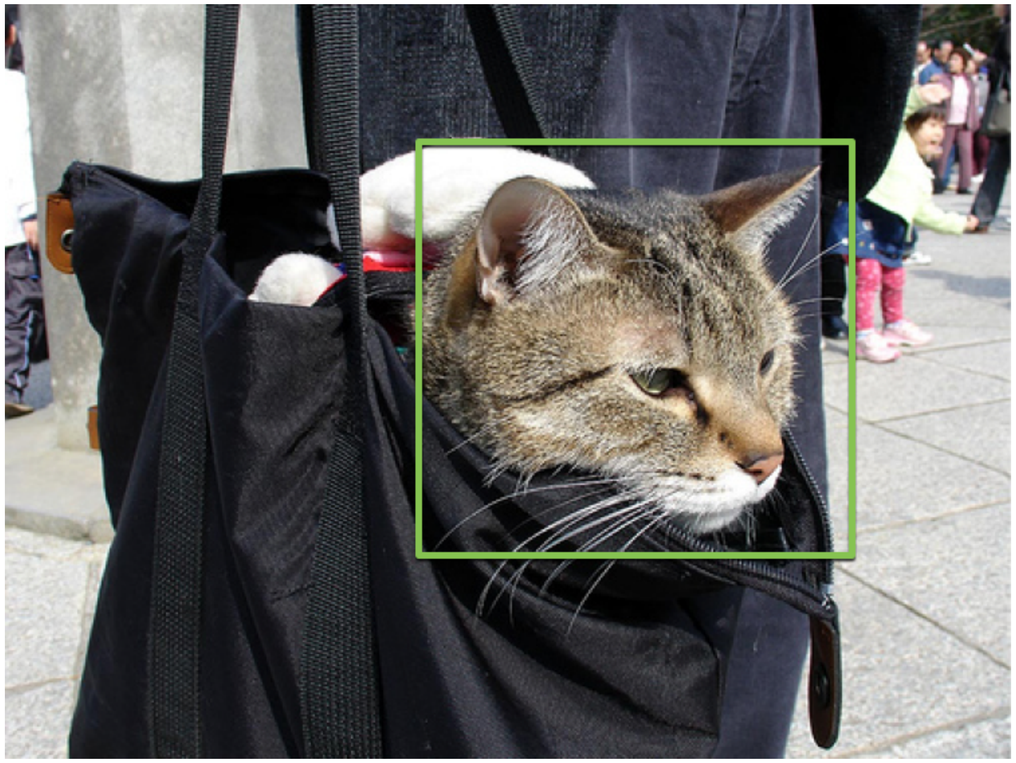

- Self-Similarity

# Canonical Challenges

# Classification Challenge

is there a cat?

# Detection Challenge



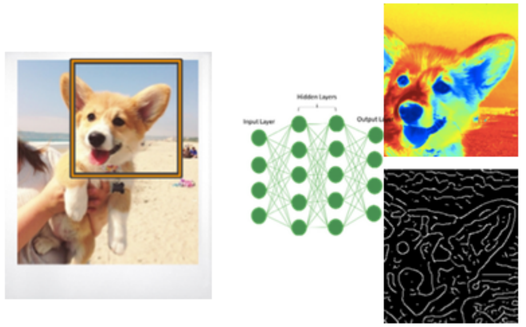
# Segmentation Challenge

# Training Pipeline

Dataset

*[puppy, glacier , ..., sailboat]*

Feature Extraction



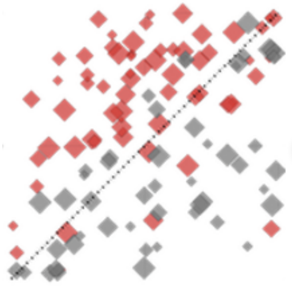
Feature

$$\begin{bmatrix} 0.543 \\ 0.789 \\ \vdots \\ 0.125 \end{bmatrix}$$

$$\begin{bmatrix} 0.543 & 0.789 & \dots & 0.125 \\ 0.462 & 0.821 & \dots & 0.976 \\ & \ddots & & \\ 0.538 & 0.975 & \dots & 0.125 \end{bmatrix} \begin{bmatrix} true \\ false \\ \vdots \\ true \end{bmatrix}$$

Dataset Features

Dataset Labels



Classifier

Repeat for many items …

Library of Classifiers

# Sliding window approaches

- Turk and Pentland, 1991
- Belhumeur, Hespanha, & Kriegman, 1997
- Schneiderman & Kanade 2004
- Viola and Jones, 2000

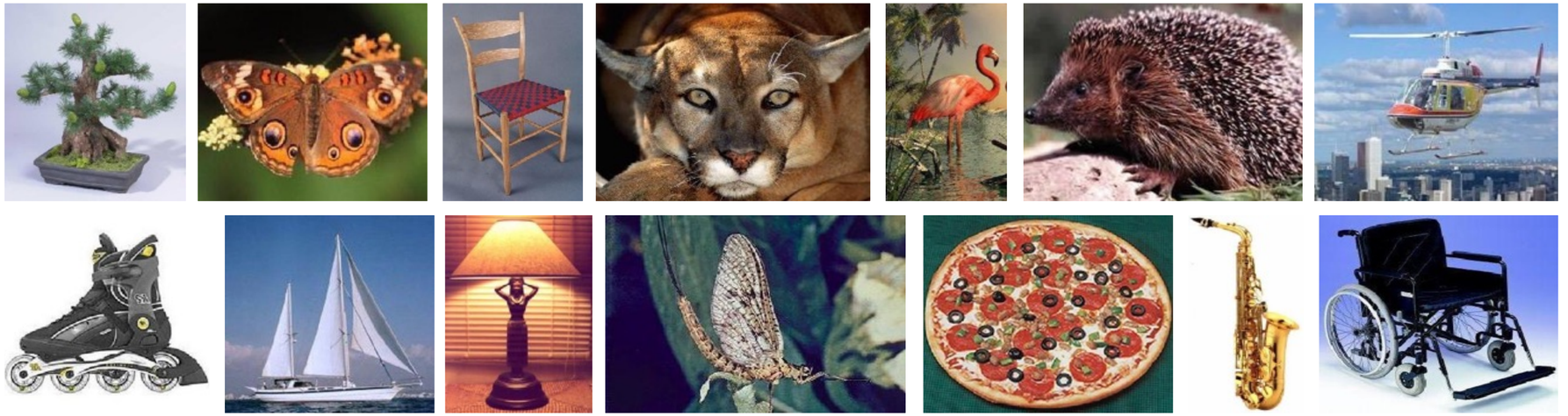
# Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution




Lazebnik, Schmid & Ponce (CVPR 2006)

# Caltech101 dataset



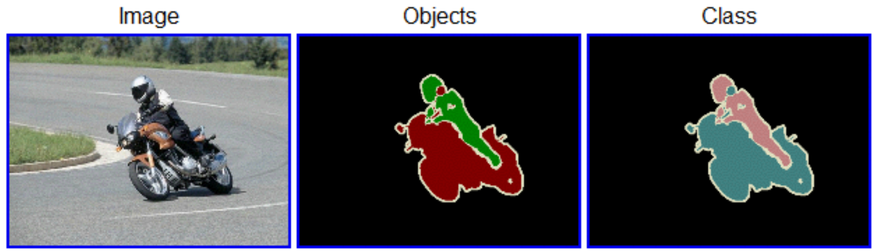
Multi-class classification results (30 training images per class)

| | Weak features (16) | | Strong features (200) | |
|---|---|---|---|---|
| Level | Single-level | Pyramid | Single-level | Pyramid |
| 0 | 15.5 ±0.9 | | 41.2 ±1.2 | |
| 1 | 31.4 ±1.2 | 32.8 ±1.3 | 55.9 ±0.9 | 57.0 ±0.8 |
| 2 | 47.2 ±1.1 | 49.3 ±1.4 | 63.6 ±0.9 | **64.6** ±0.8 |
| 3 | 52.2 ±0.8 | **54.0** ±1.1 | 60.3 ±0.9 | 64.6 ±0.7 |

# The PASCAL Visual Object Classes Challenge 2009 (VOC2009)

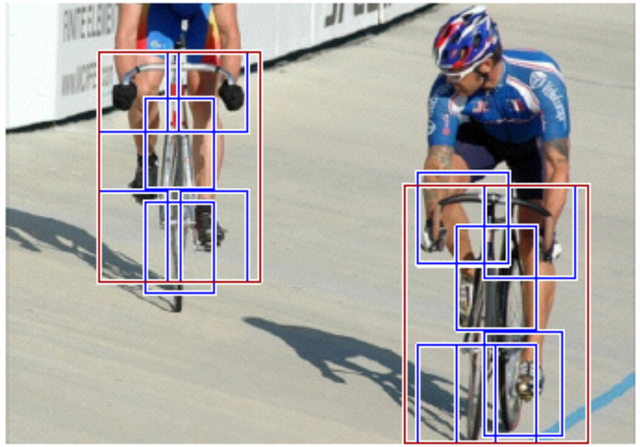
- Twenty object categories (aeroplane to TV/monitor)

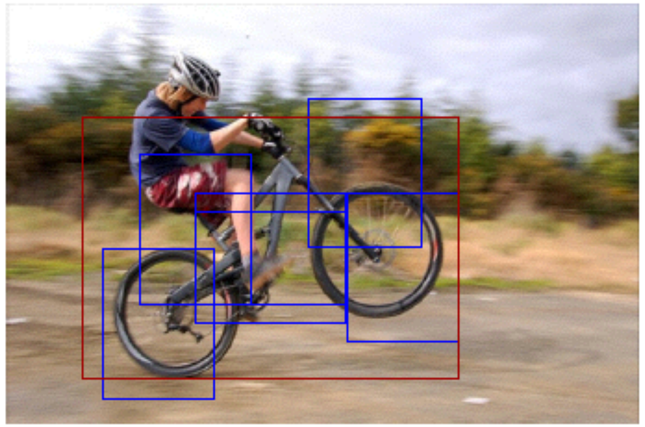
- Three challenges:
  - Classification challenge (is there an X in this image?)
  - Detection challenge (draw a box around every X)
  - Segmentation challenge

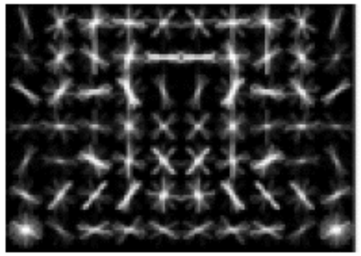


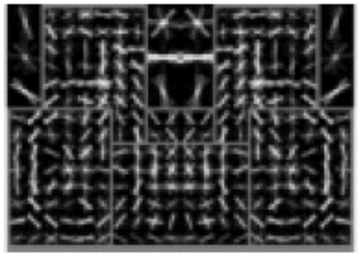

Slides from Noah Snavely

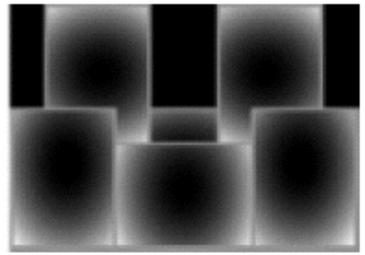
# Discriminatively trained part-based

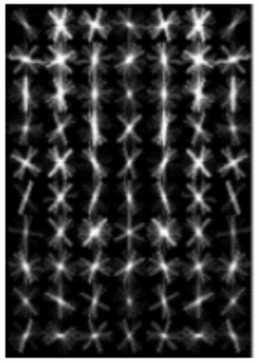


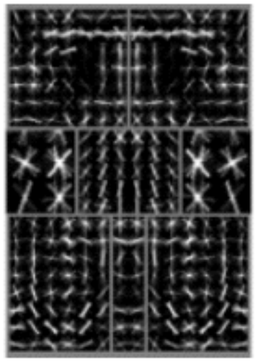


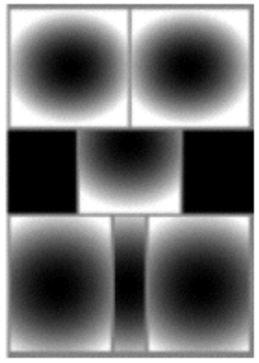










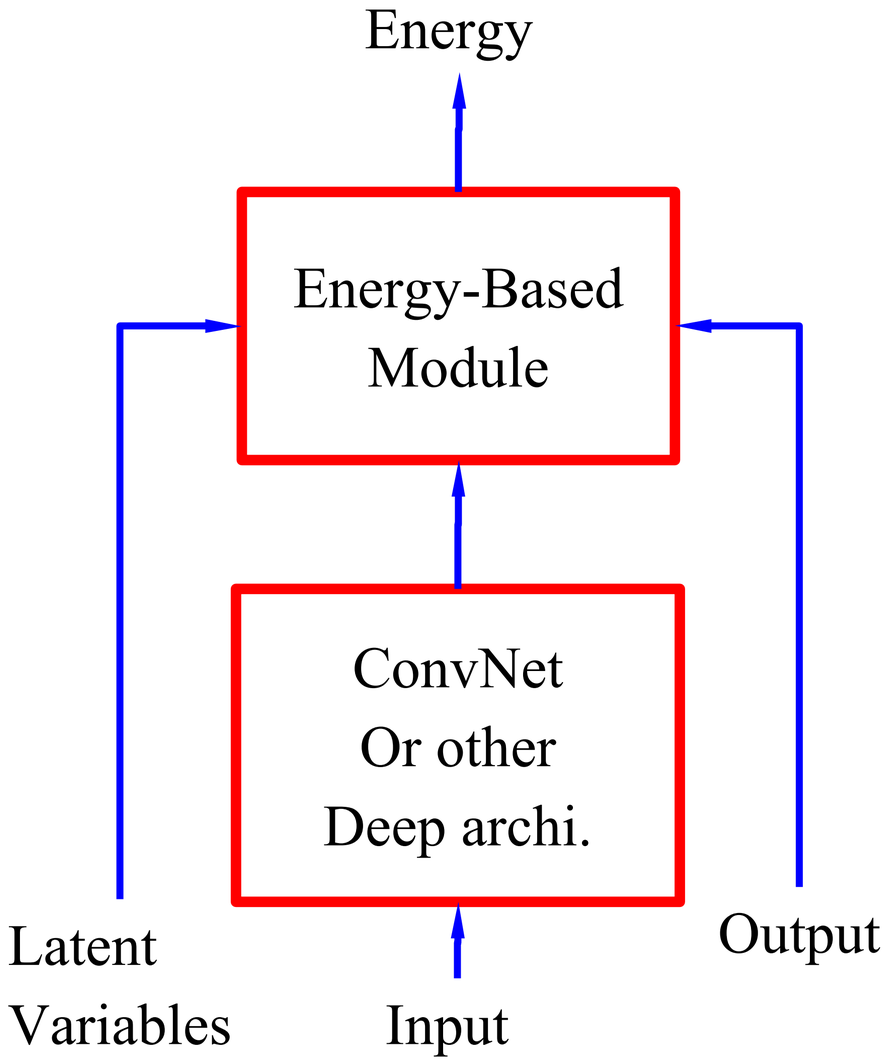


P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, **"Object Detection with Discriminatively Trained Part-Based Models,"** PAMI 2009

# Why Deep Networks?

# Global (End-to-End) Learning: Energy-Based Models.






Making every single module in the system trainable.

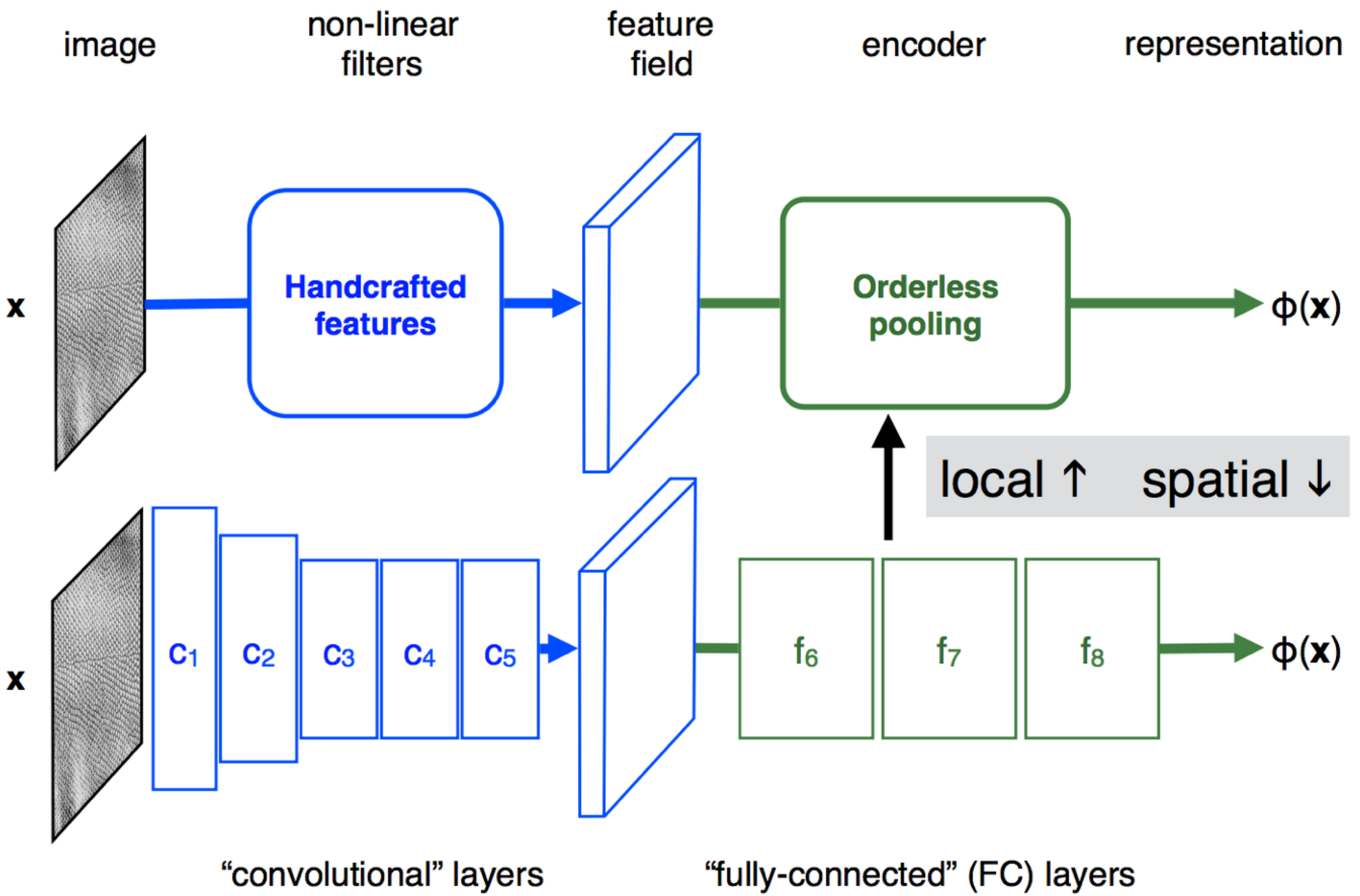
Every module is trained simultaneously so as to optimize a global loss function.

Includes the feature extractor, the recognizer, and the contextual post-processor (graphical model)

Problem: back-propagating gradients through the graphical model.

# Components of End-to-End Learning

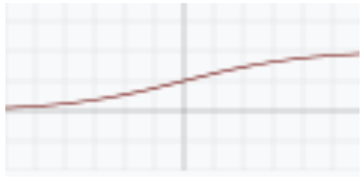
# Texture representations vs CNNs



# Feedforward Neural Network

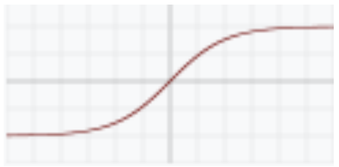
- Logistic Function

$$y = \frac{1}{1 + e^{-x}}$$



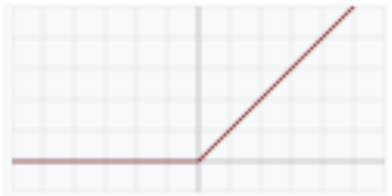
- tanh

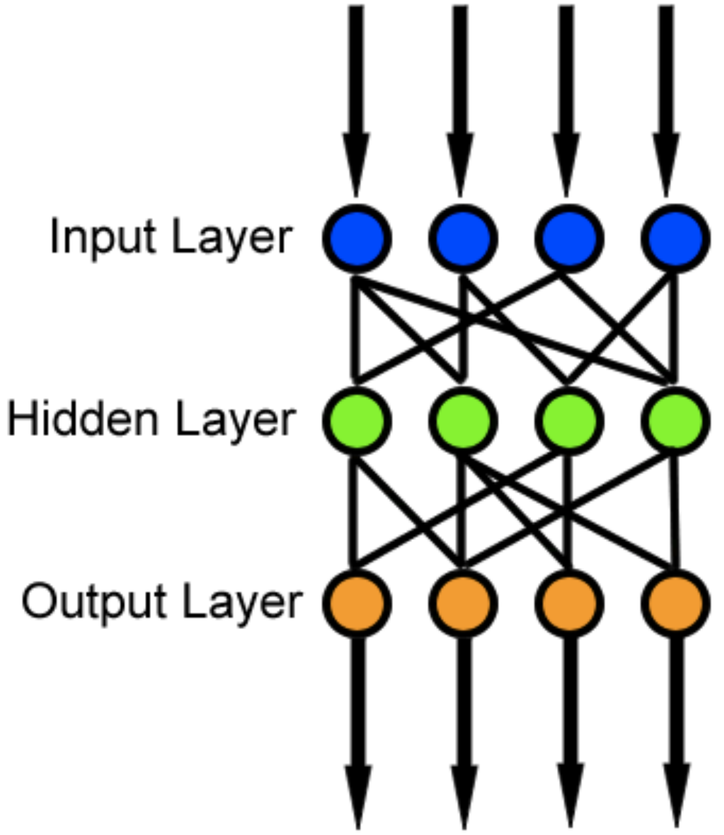
$$\phi(v_i) = \tanh\left(\beta_1 + \beta_0 \sum_j v_{i,j} x_j\right)$$



- ReLU

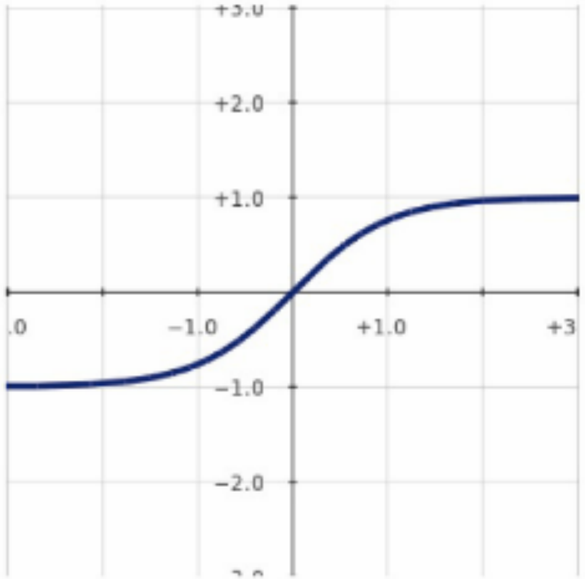
$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$$

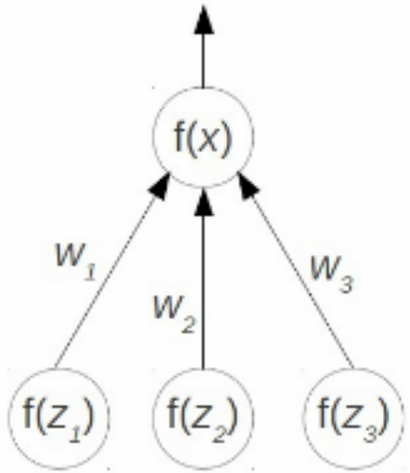




# Neurons

$f(x) = \tanh(x)$

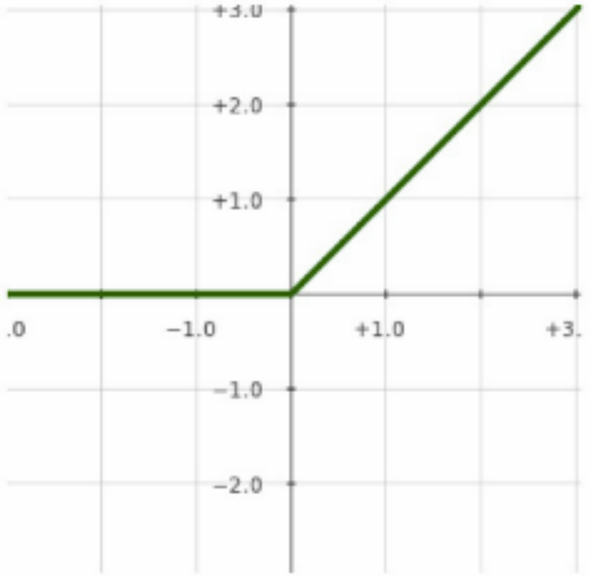



$x = w_1 f(z_1) + w_2 f(z_2) + w_3 f(z_3)$

$x$ is called the total input to the neuron, and $f(x)$ is its output

$f(x) = \max(0, x)$




Very bad (slow to train)

Very good (quick to train)

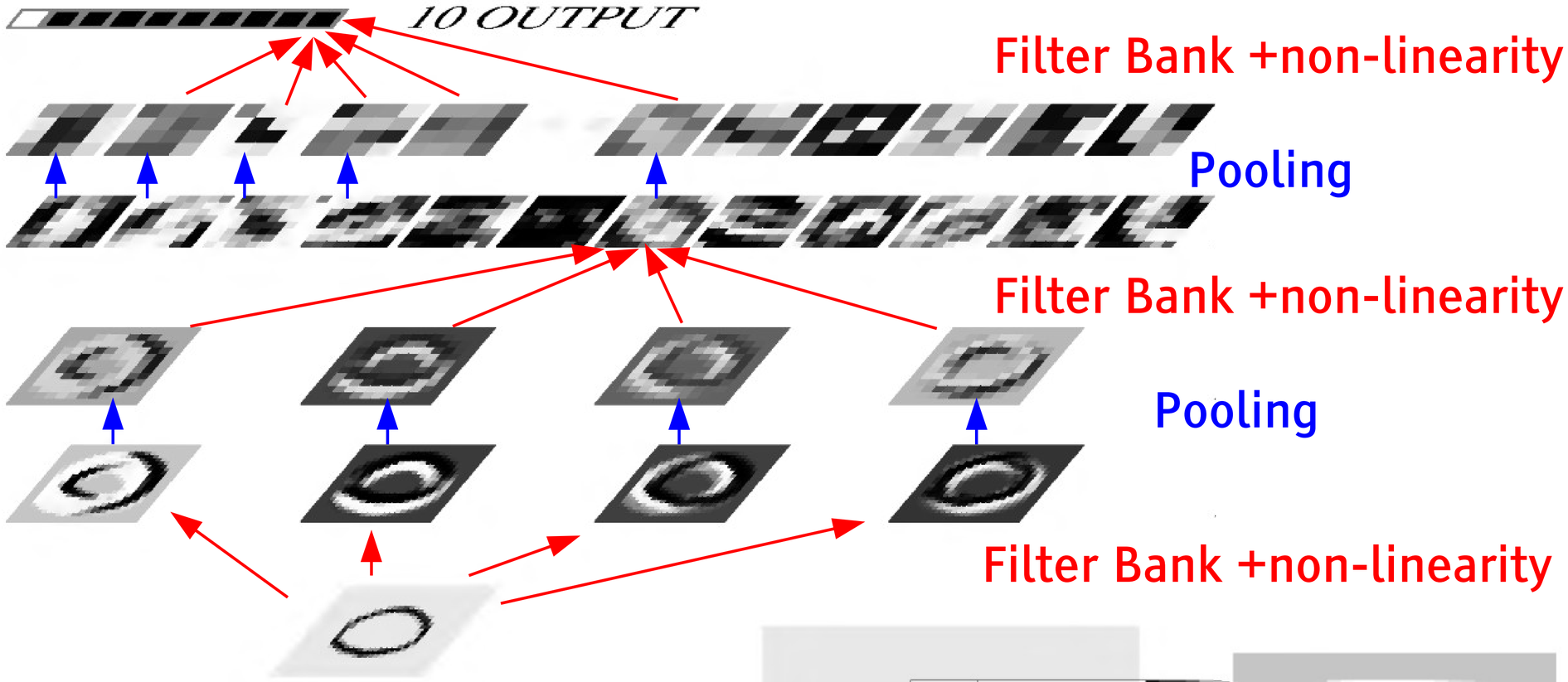
Perceptron

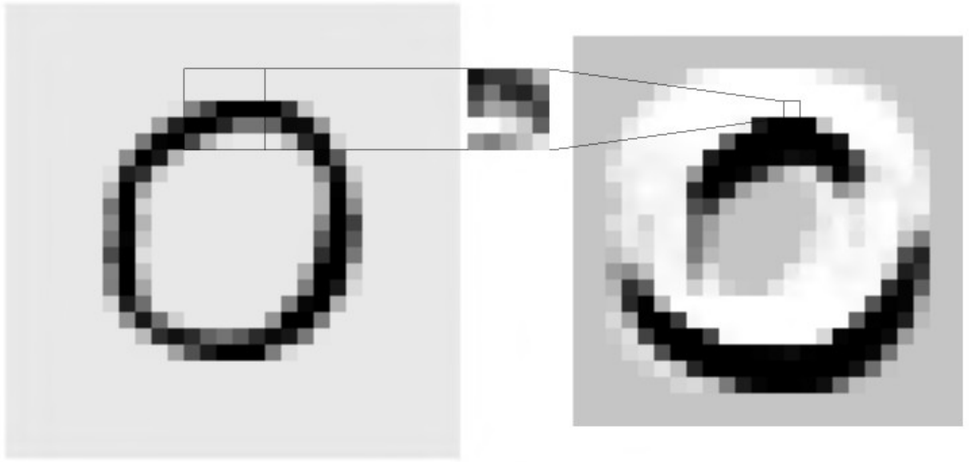
Rectified Linear Unit (ReLU)

Figure from Karpathy 2015

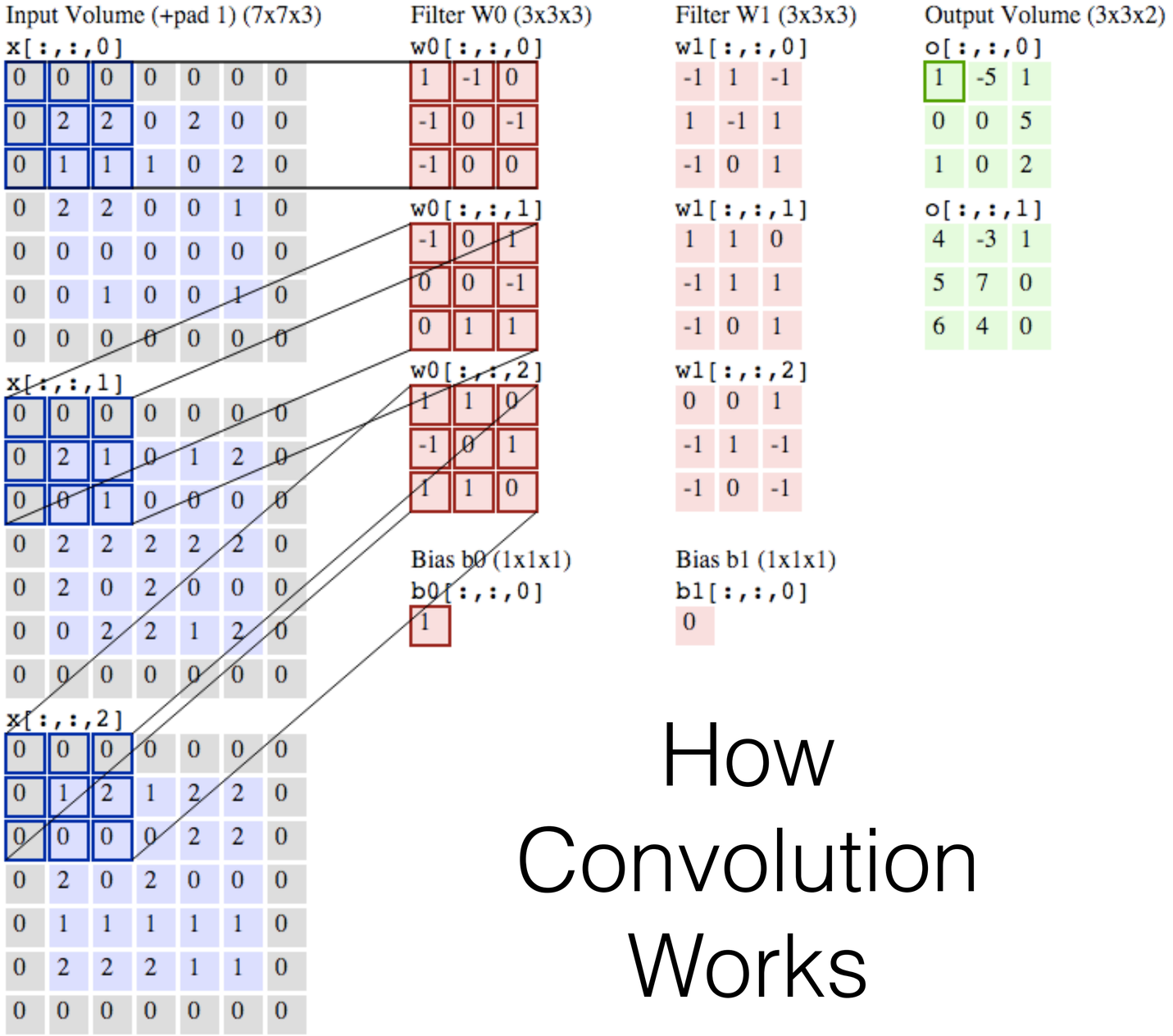
# Convolutional Network





[LeCun et al. NIPS 1989]



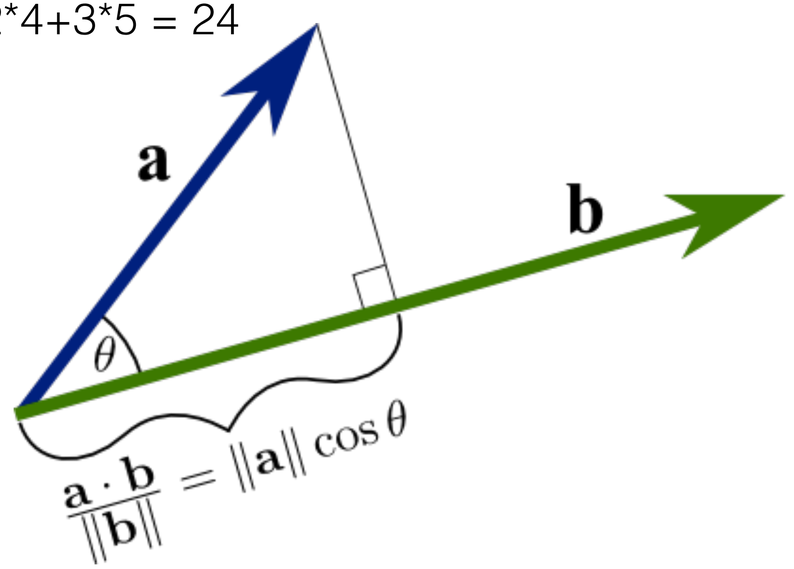

# How Convolution Works

Figure from Karpathy 2015

# Image Filters

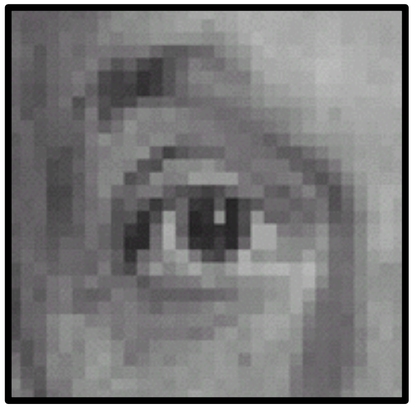
# The Dot Product

- Also called ‘scalar product’
- Sum of the product of each element of two sequences
- (1,2,3) • (3,4,5) = 1*3+2*4+3*5 = 24
- a = (1,2)
- b = (3,1)
- a • b = 5




- Dot product is the length of **a** projected on **b**

# Practice with linear filters



Original

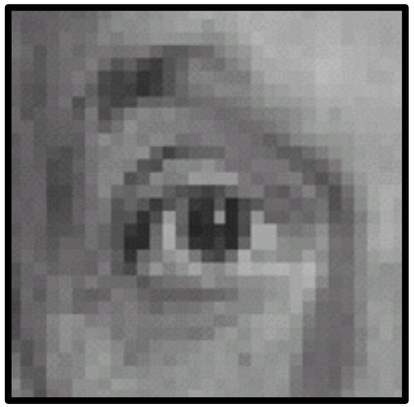
| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

?



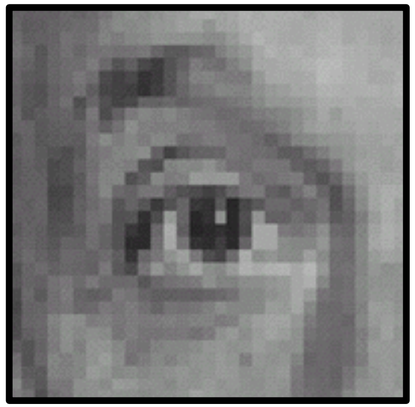
Source: D. Lowe

# Practice with linear filters



Original

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

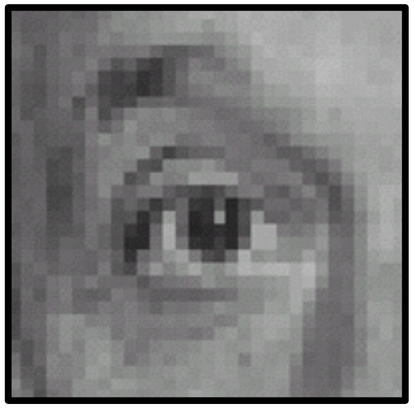


Filtered
(no change)



Source: D. Lowe

# Practice with linear filters



Original

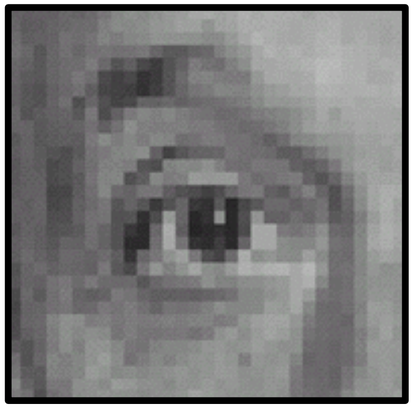
| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 0 | 0 |

?



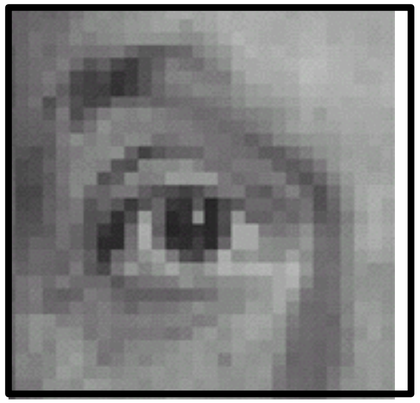
Source: D. Lowe

# Practice with linear filters



Original

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 0 | 0 |



Shifted left
By 1 pixel

Source: D. Lowe

# Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

$$g[\cdot,\cdot]$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



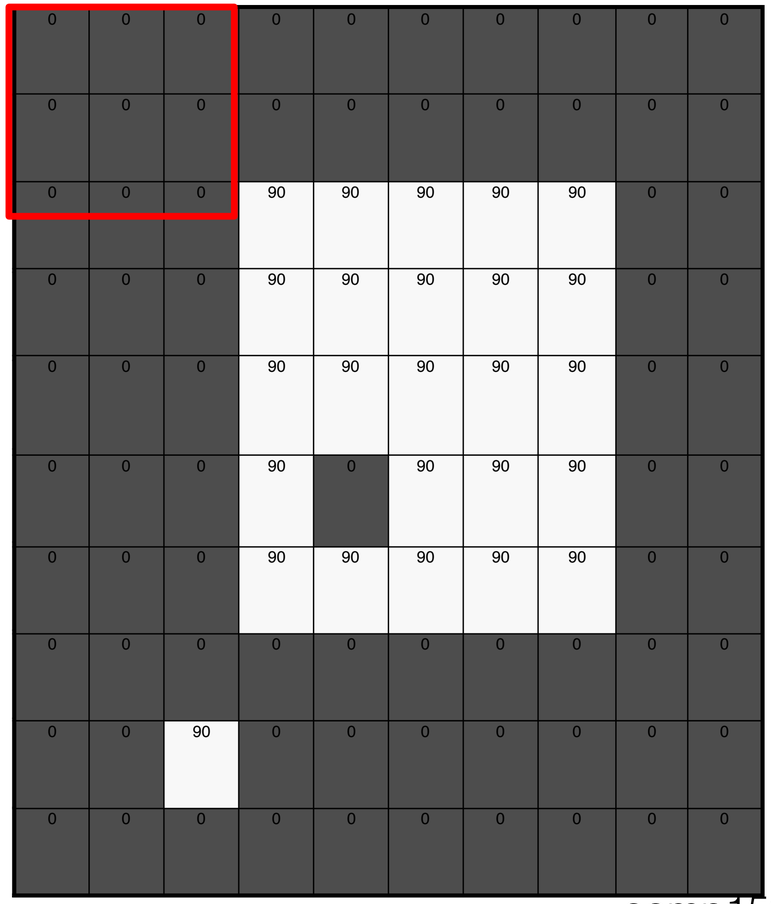
Slide credit: David Lowe (UBC)

# Smoothing with box filter

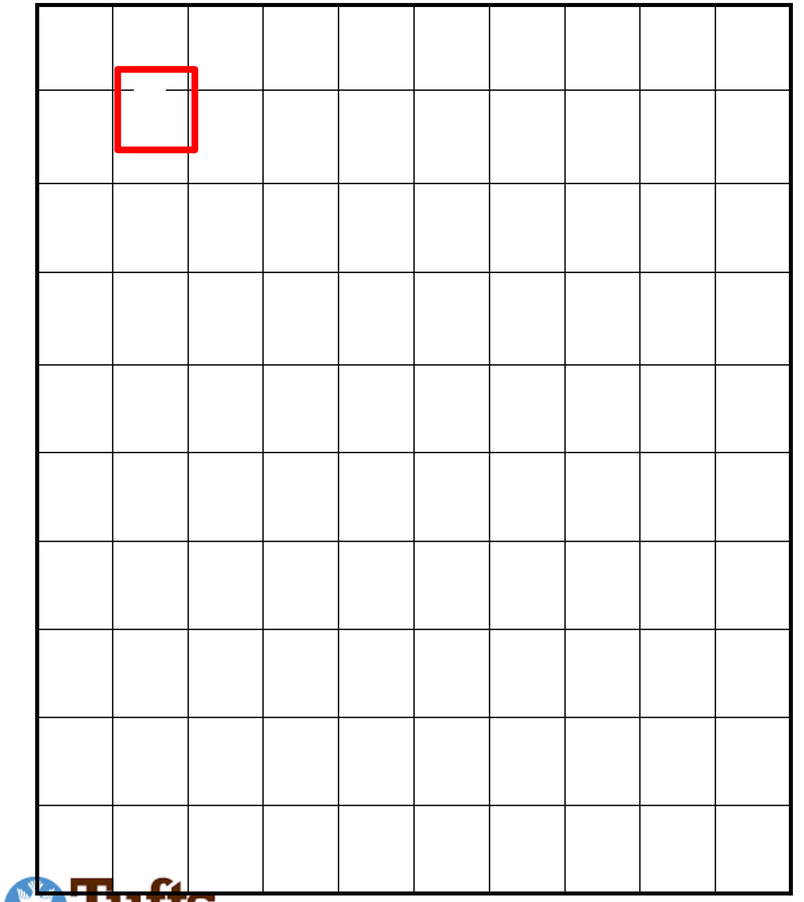
# Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[.,.]$




$h[.,.]$





Credit: S. Seitz

# Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$h[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | 0 | 10 | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

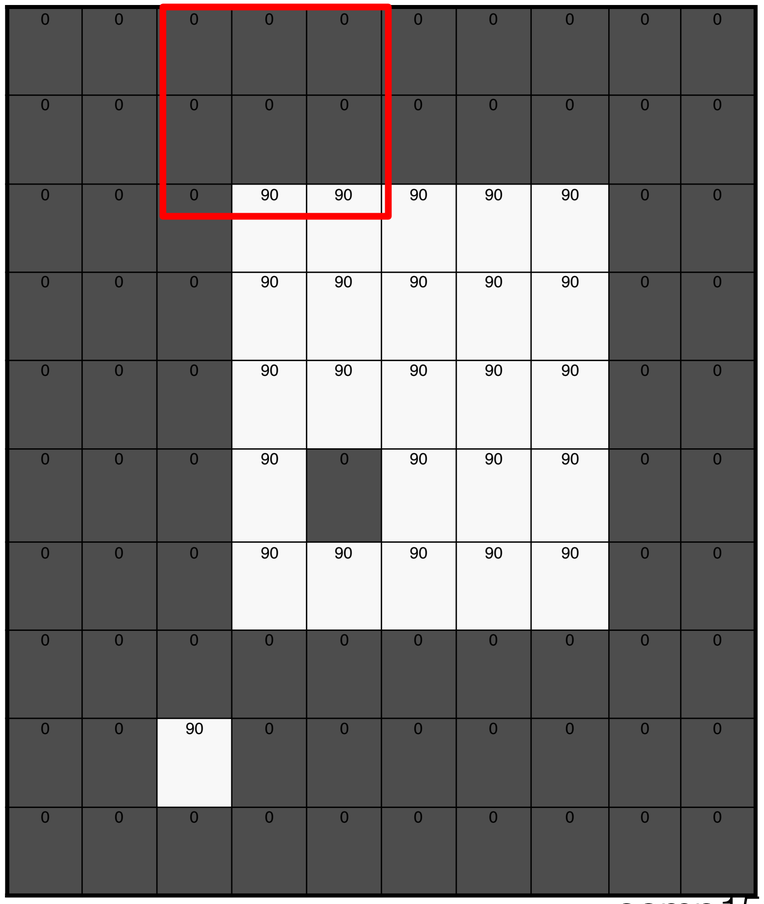


Credit: S. Seitz

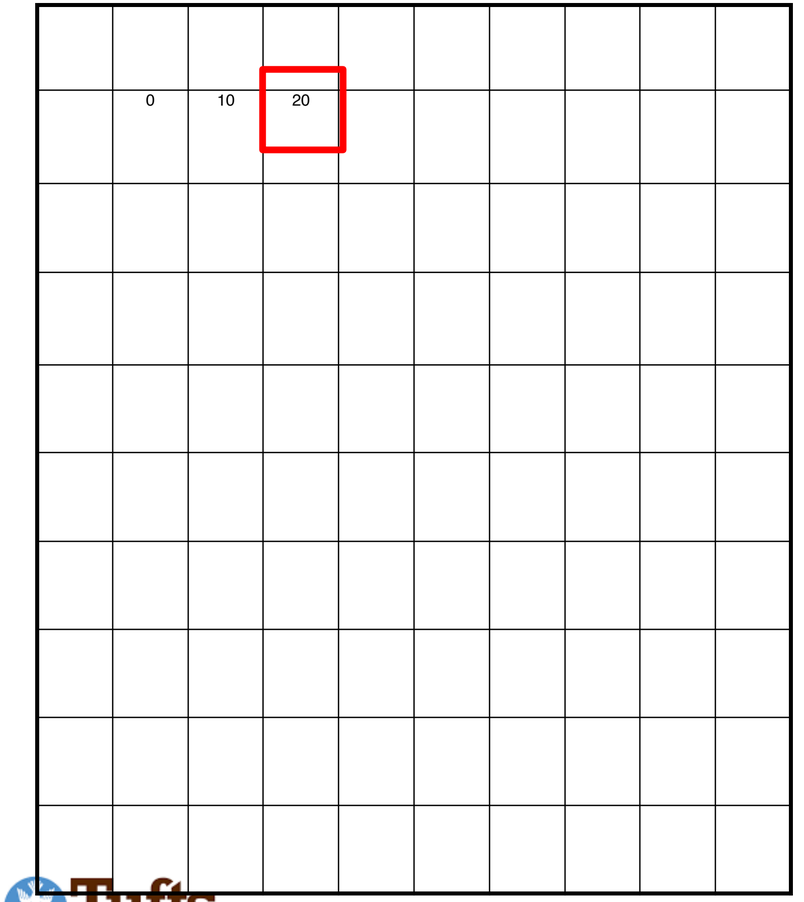
# Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[.,.]$




$h[.,.]$



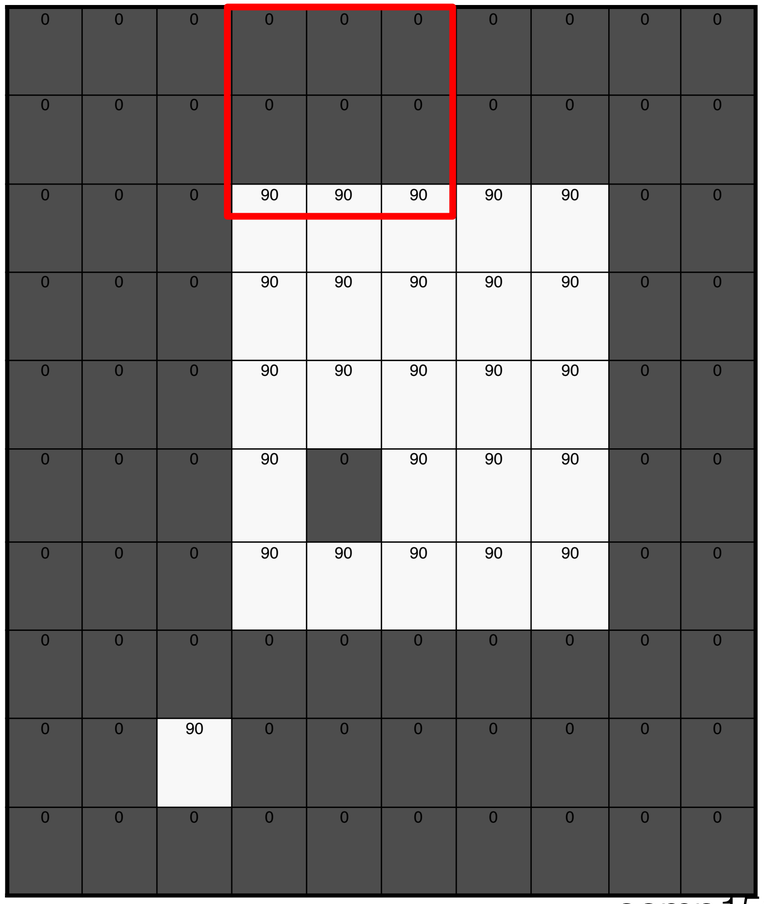



Credit: S. Seitz

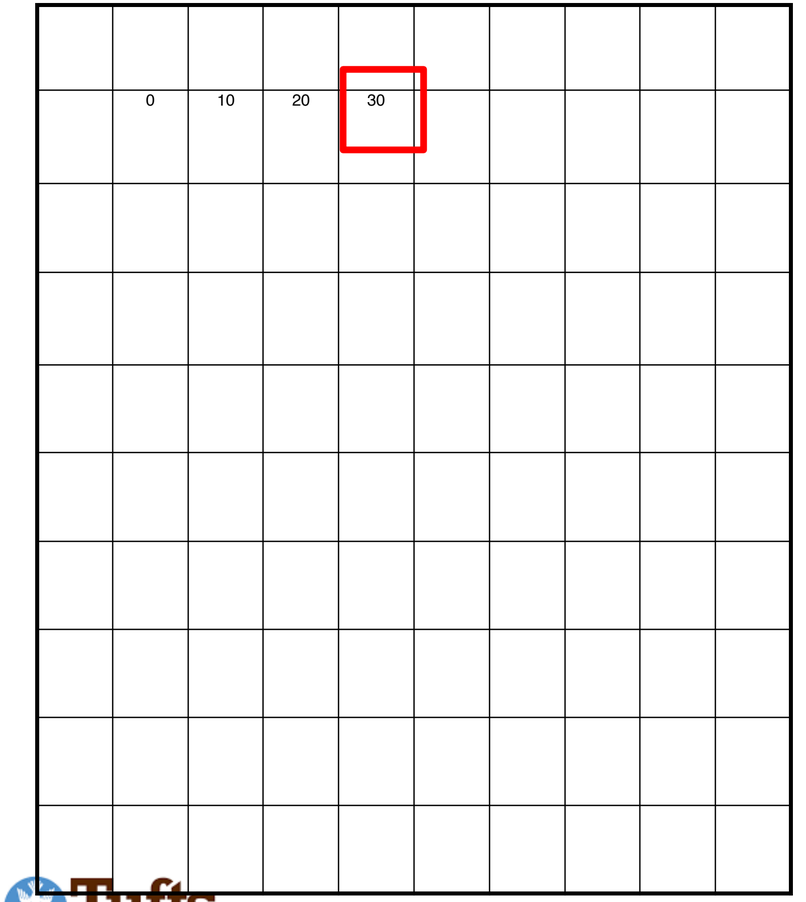
# Image filtering

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[\cdot,\cdot]$




$h[\cdot,\cdot]$



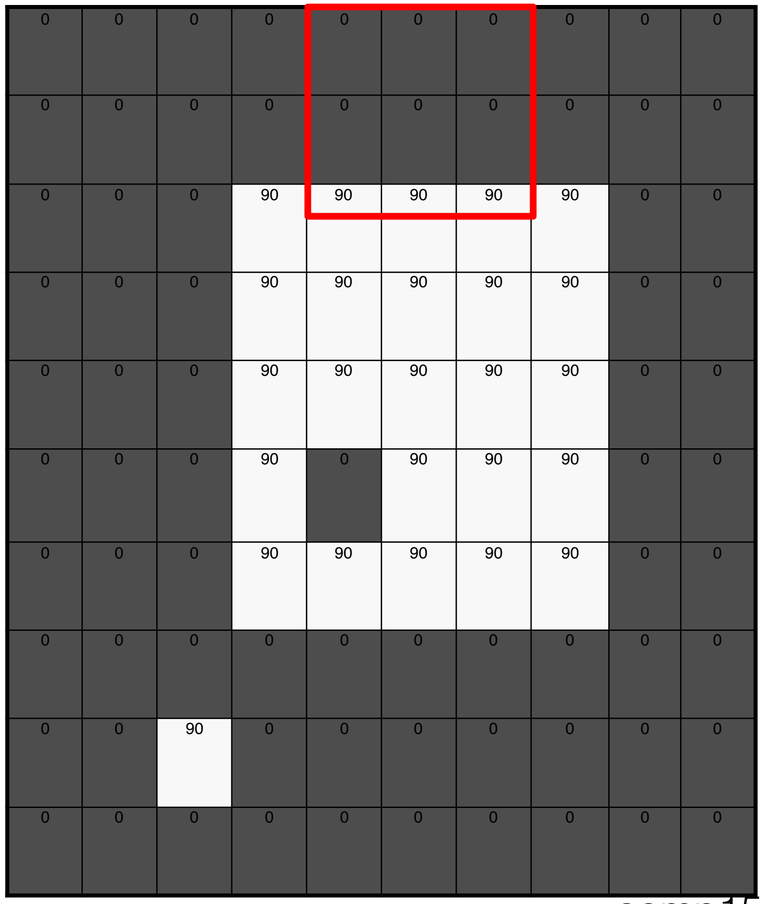



Credit: S. Seitz

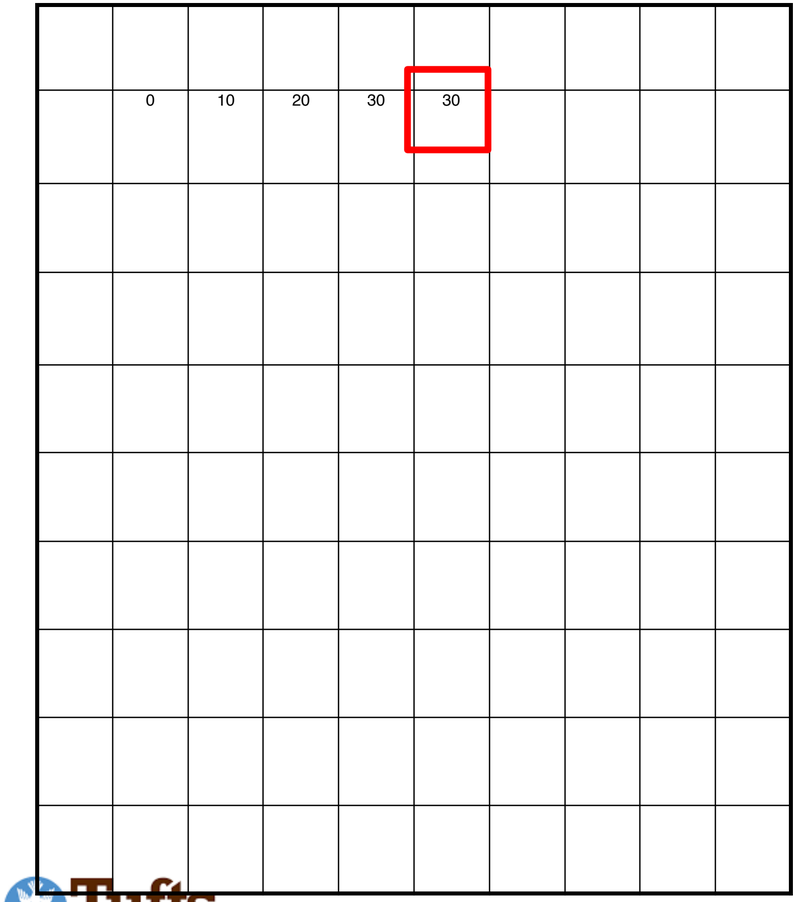
# Image filtering

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[\cdot,\cdot]$




$h[\cdot,\cdot]$






Credit: S. Seitz

# Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

$f[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$h[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | 0 | 10 | 20 | 30 | 30 | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | ? | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

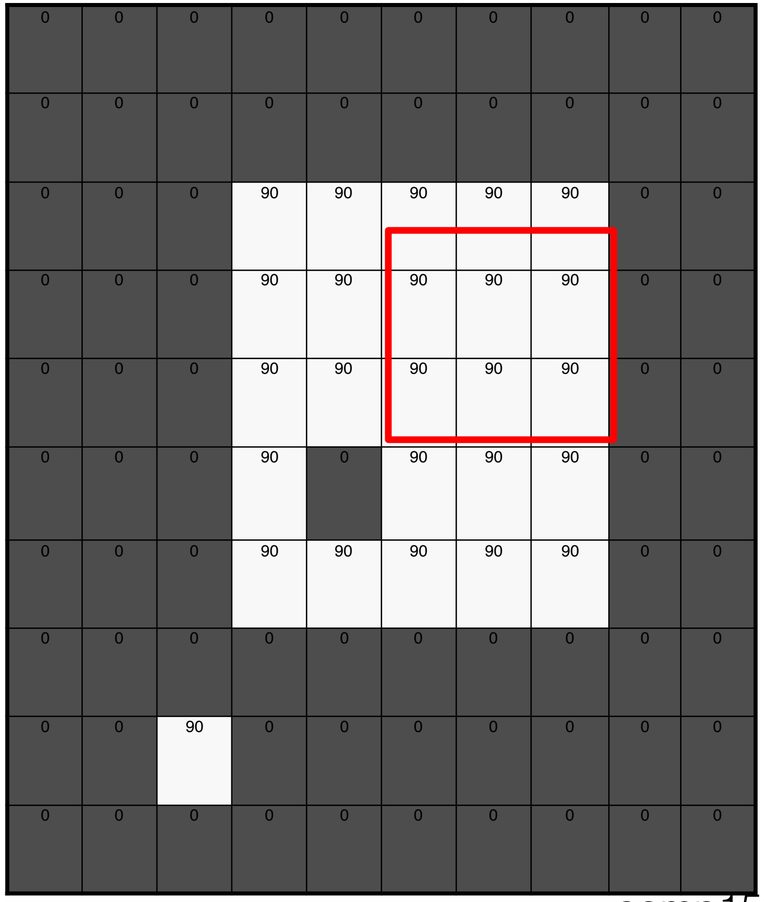


Credit: S. Seitz

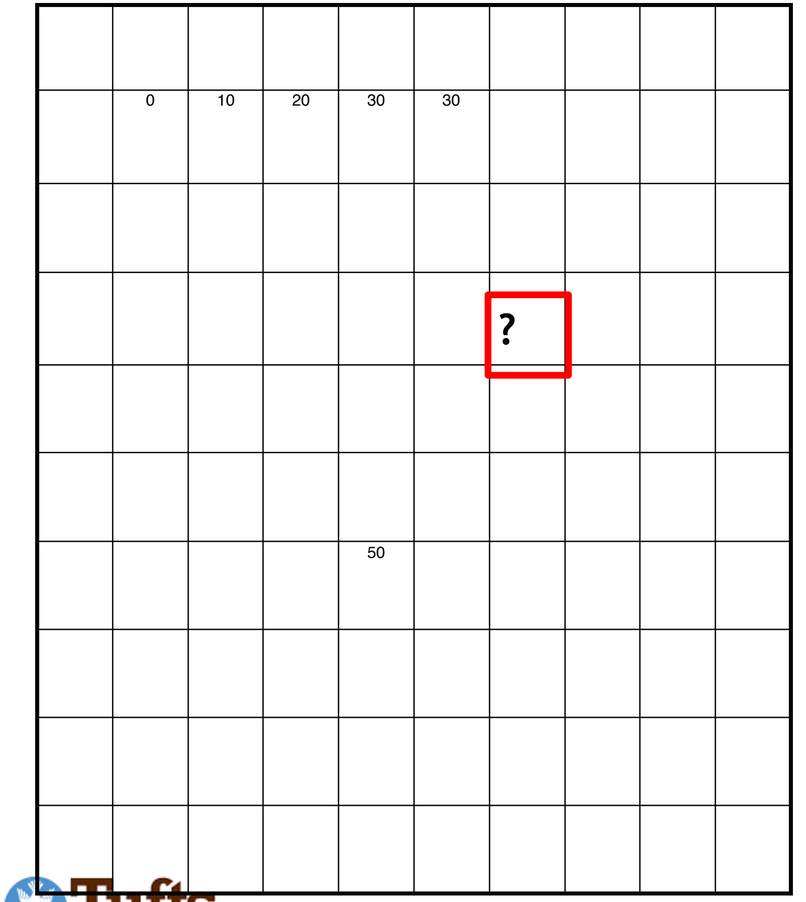
# Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$f[\cdot,\cdot]$




$h[\cdot,\cdot]$






Credit: S. Seitz

# Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

$f[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$h[\cdot,\cdot]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | 0 | 10 | 20 | 30 | 30 | 30 | 20 | 10 | |
| | 0 | 20 | 40 | 60 | 60 | 60 | 40 | 20 | |
| | 0 | 30 | 60 | 90 | 90 | 90 | 60 | 30 | |
| | 0 | 30 | 50 | 80 | 80 | 90 | 60 | 30 | |
| | 0 | 30 | 50 | 80 | 80 | 90 | 60 | 30 | |
| | 0 | 20 | 30 | 50 | 50 | 60 | 40 | 20 | |
| | 10 | 20 | 30 | 30 | 30 | 30 | 20 | 10 | |
| | 10 | 10 | 10 | 0 | 0 | 0 | 0 | 0 | |
| | | | | | | | | | |

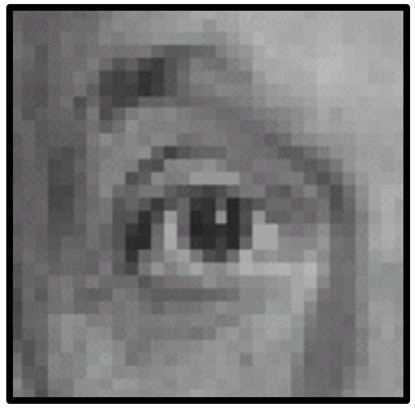


Credit: S. Seitz

# What else is possible with Filters?

- Really important for photo editing!
  - Enhance images
    - Denoise, resize, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns —-> Convolutional Networks!!
    - Template matching

# Practice with linear filters



Original

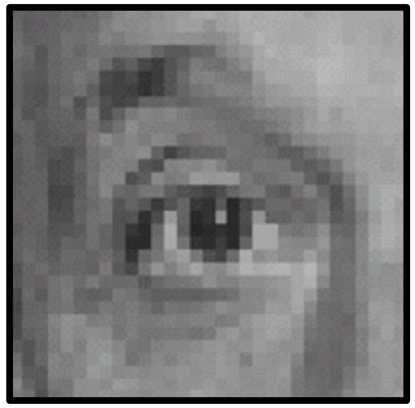
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad ?$$

(Note that filter sums to 1)



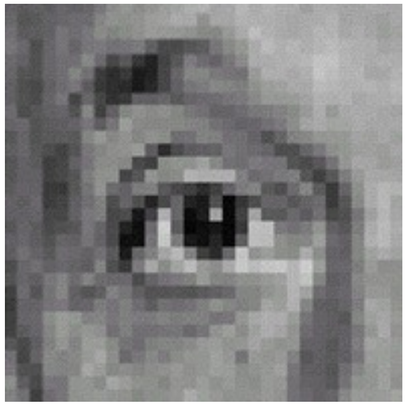
Source: D. Lowe

# Practice with linear filters



Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



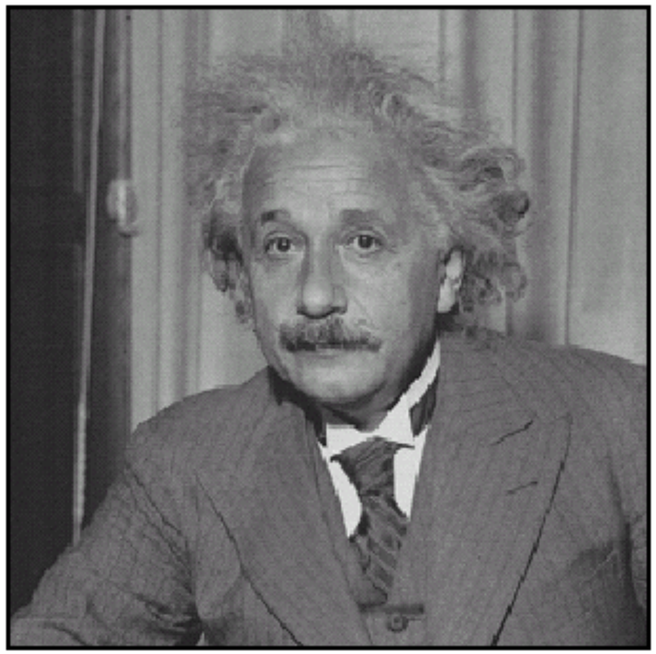
**Sharpening filter**

- Accentuates differences with local average

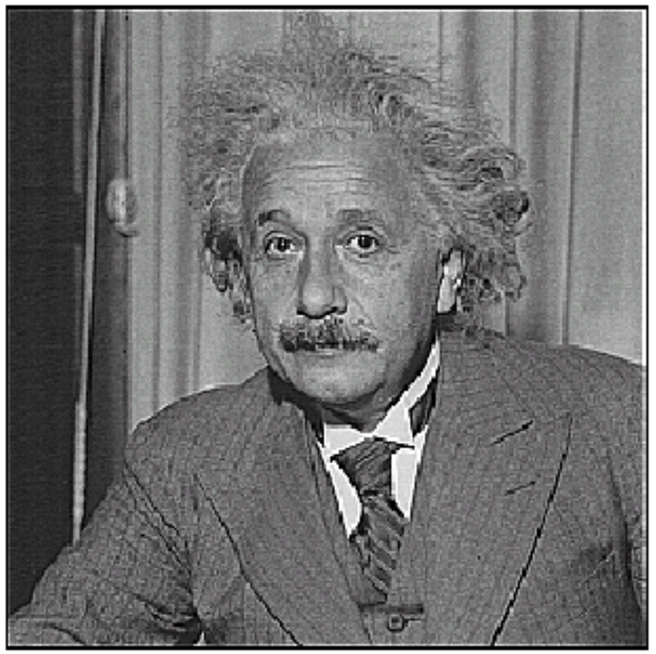


Source: D. Lowe

# Sharpening



before



after



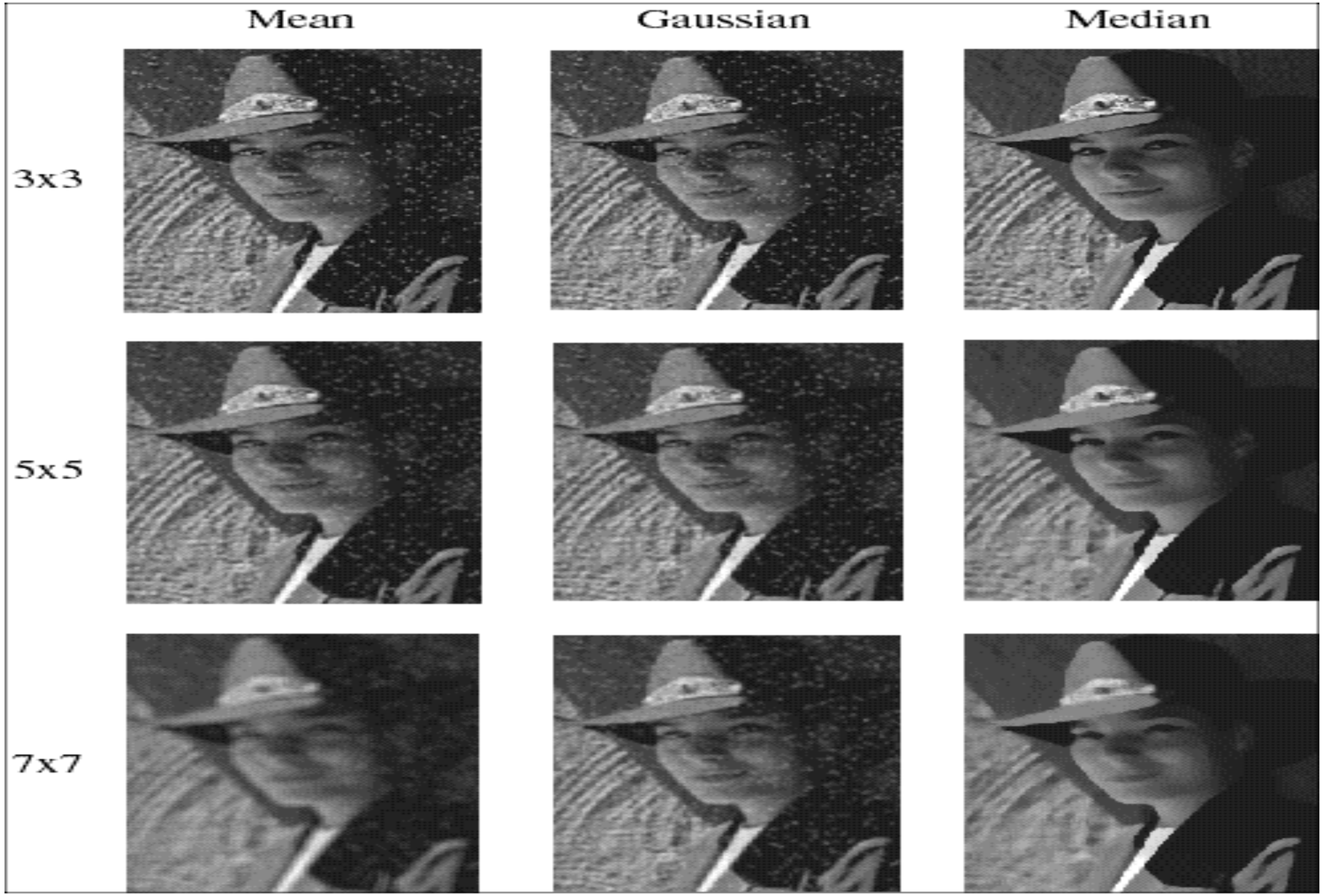
Source: D. Lowe

# Noise

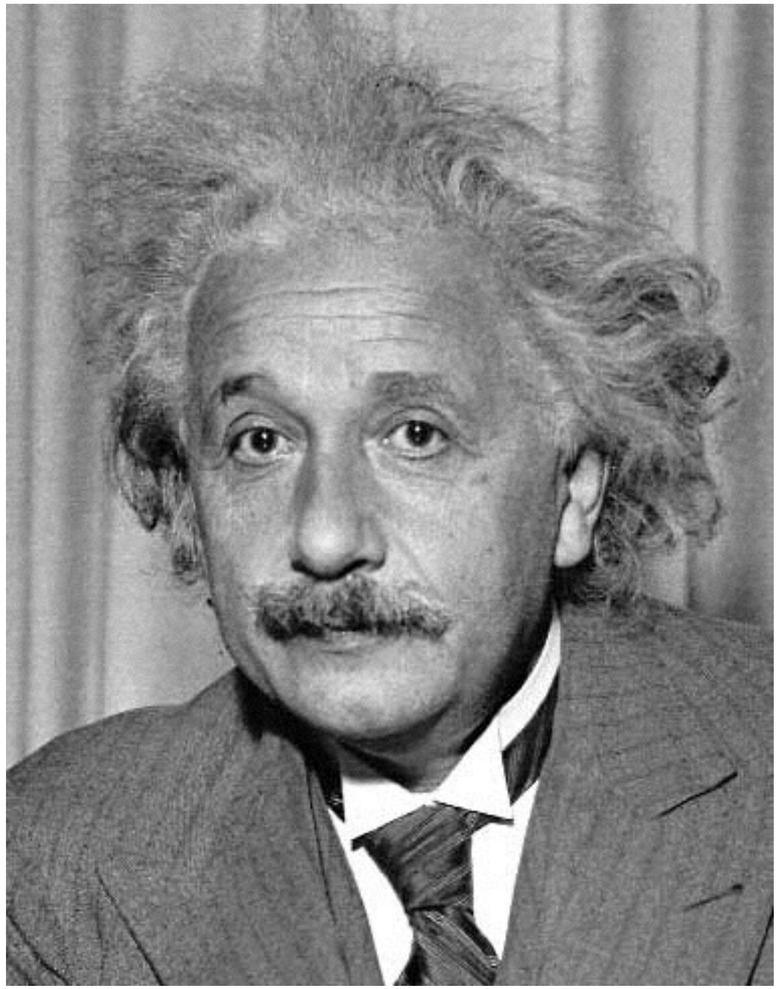
# Median filters

- A **Median Filter** operates over a window by selecting the median intensity in the window.
- What advantage does a median filter have over a mean filter?

# Comparison: salt and pepper noise

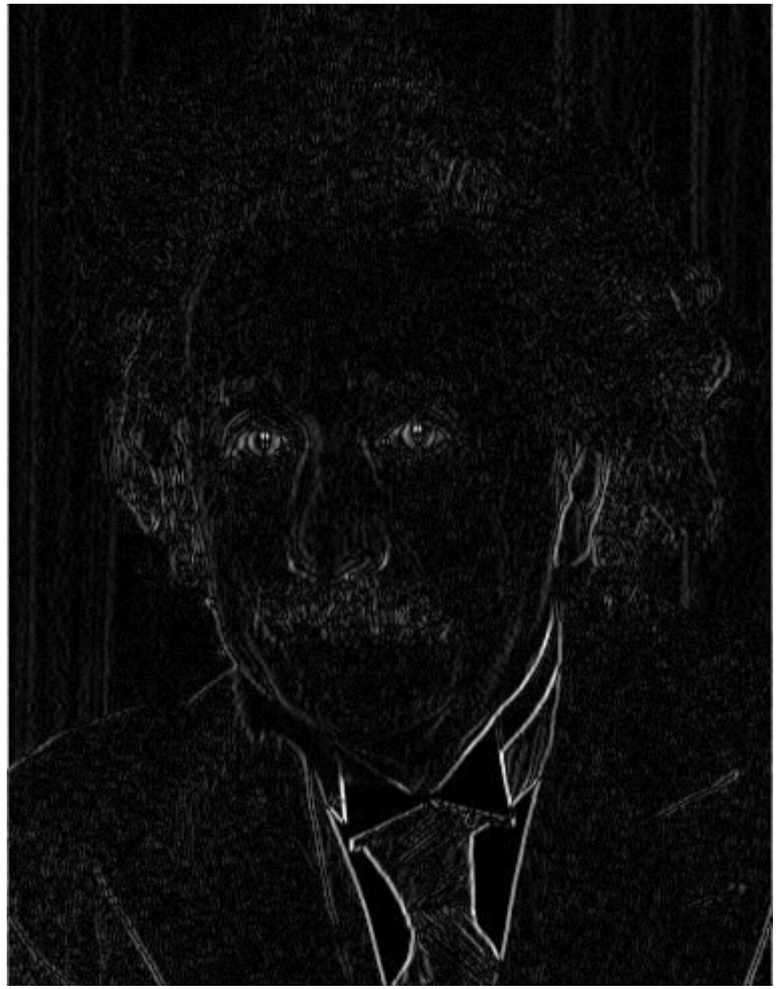


# Other filters



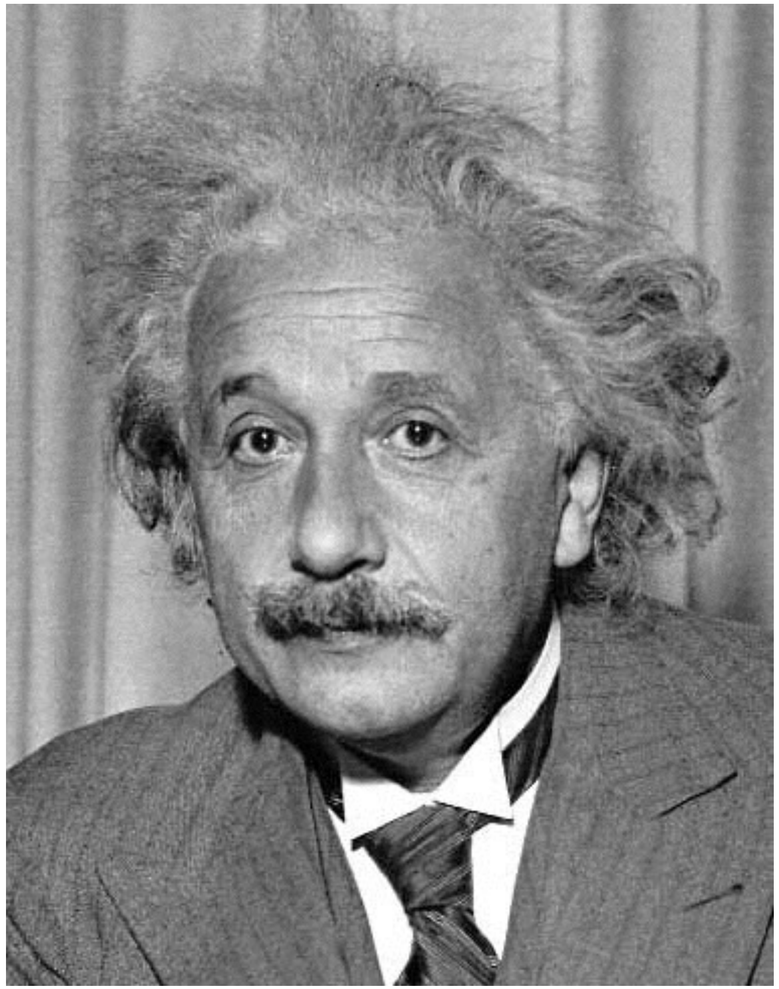
| | | |
|---|---|---|
| 1 | 0 | -1 |
| 2 | 0 | -2 |
| 1 | 0 | -1 |

Sobel



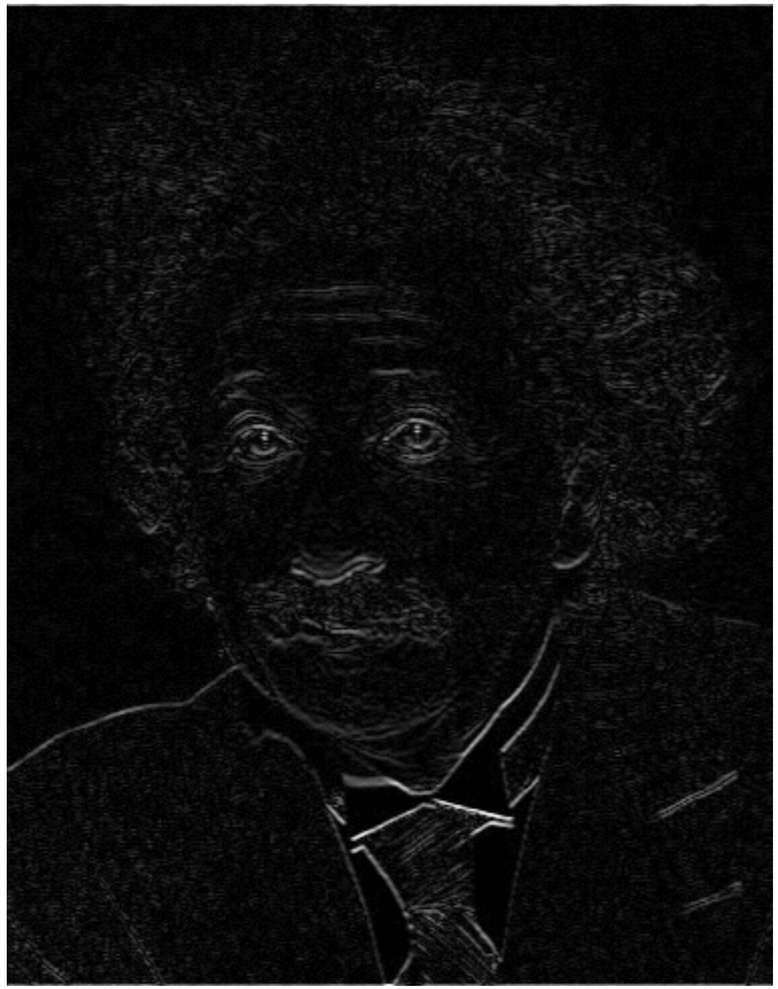
Vertical Edge
(absolute value)

# Other filters



| | | |
|---|---|---|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Sobel



Horizontal Edge
(absolute value)

# Key properties of linear filters

**Linearity:**

$\texttt{filter}(f_1 + f_2) = \texttt{filter}(f_1) + \texttt{filter}(f_2)$

**Shift invariance:** same behavior regardless of pixel location

$\texttt{filter}(\texttt{shift}(f)) = \texttt{shift}(\texttt{filter}(f))$



Source: S. Lazebnik

# More properties

- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
  - But particular filtering implementations might break this equality
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: a * $(b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k (a * b)$
- Identity: unit impulse $e = [0, 0, 1, 0, 0]$,
  $a * e = a$



Source: S. Lazebnik

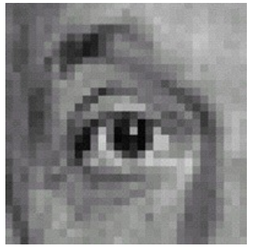
# Practical matters

- What about near the edge?
  - the filter window falls off the edge of the image
  - need to extrapolate
  - methods:
    - clip filter (black)
    - wrap around
    - copy edge
    - reflect across edge



Source: S. Marschner

# Take-home messages about filters



$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

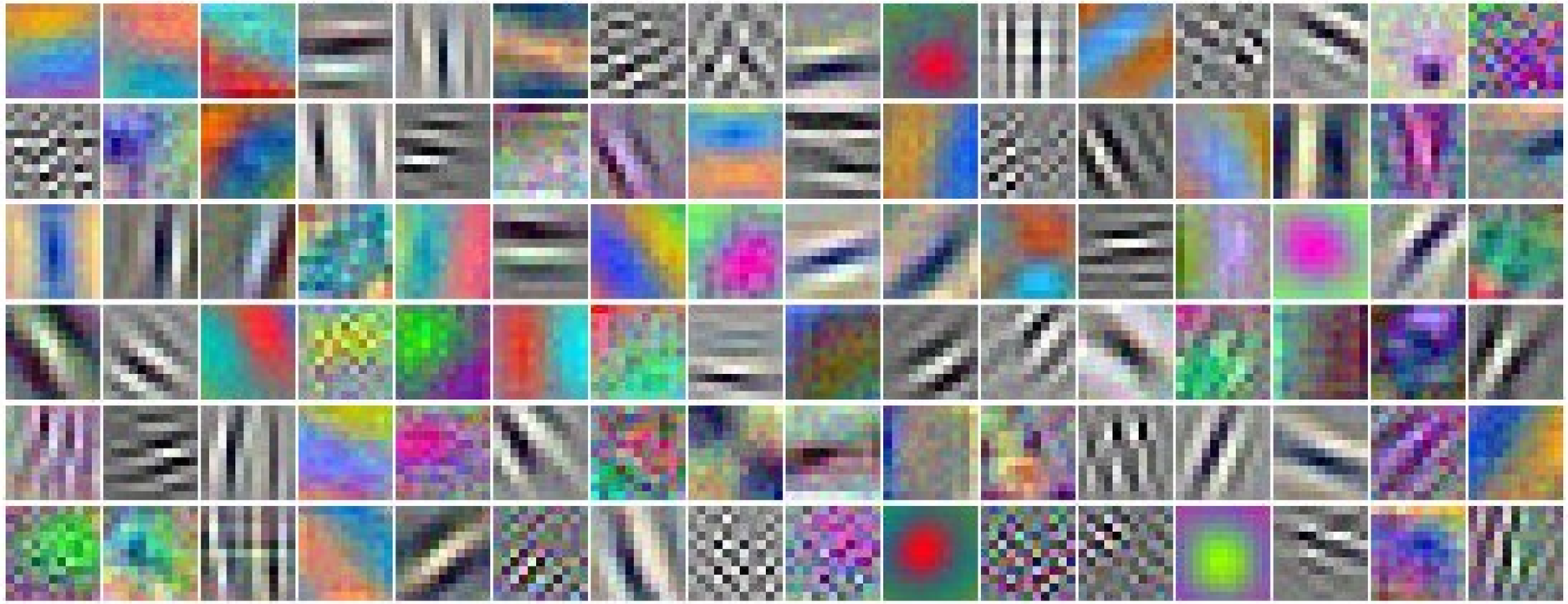
- Linear filtering is sum of dot product at each position
  - Can smooth, sharpen, translate (among many other uses)

- Be aware of details for filter size, extrapolation, cropping

# Kernels: Layer 1 (11x11)

Y LeCun

- Layer 1: 3x96 kernels, RGB->96 feature maps, 11x11 Kernels, stride 4



# Current
# Computer Vision

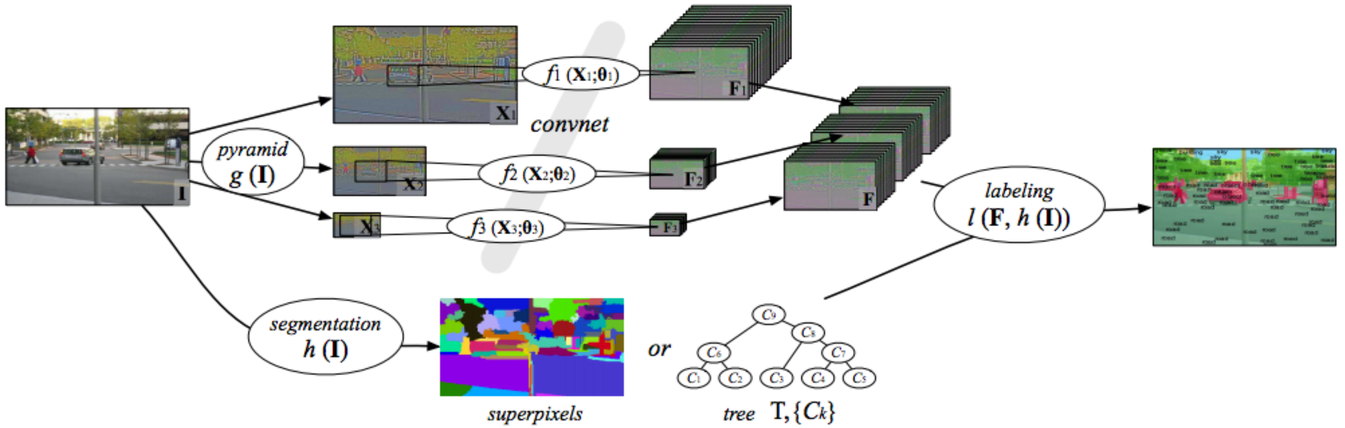



Fig. 8. More results using our multiscale convolutional network and a flat CRF on the Stanford Background Dataset.

This time though, the reviewers were particularly clueless, or negatively biased, or both. I was very sure that this paper was going to get good reviews because: 1) it has two simple and generally applicable ideas for segmentation ("purity tree" and "optimal cover"); 2) it uses no hand-crafted features (it's all learned all the way through. Incredibly, this was seen as a negative point by the reviewers!); 3) it beats all published results on 3 standard datasets for scene parsing; 4) it's an order of magnitude faster than the competing methods.

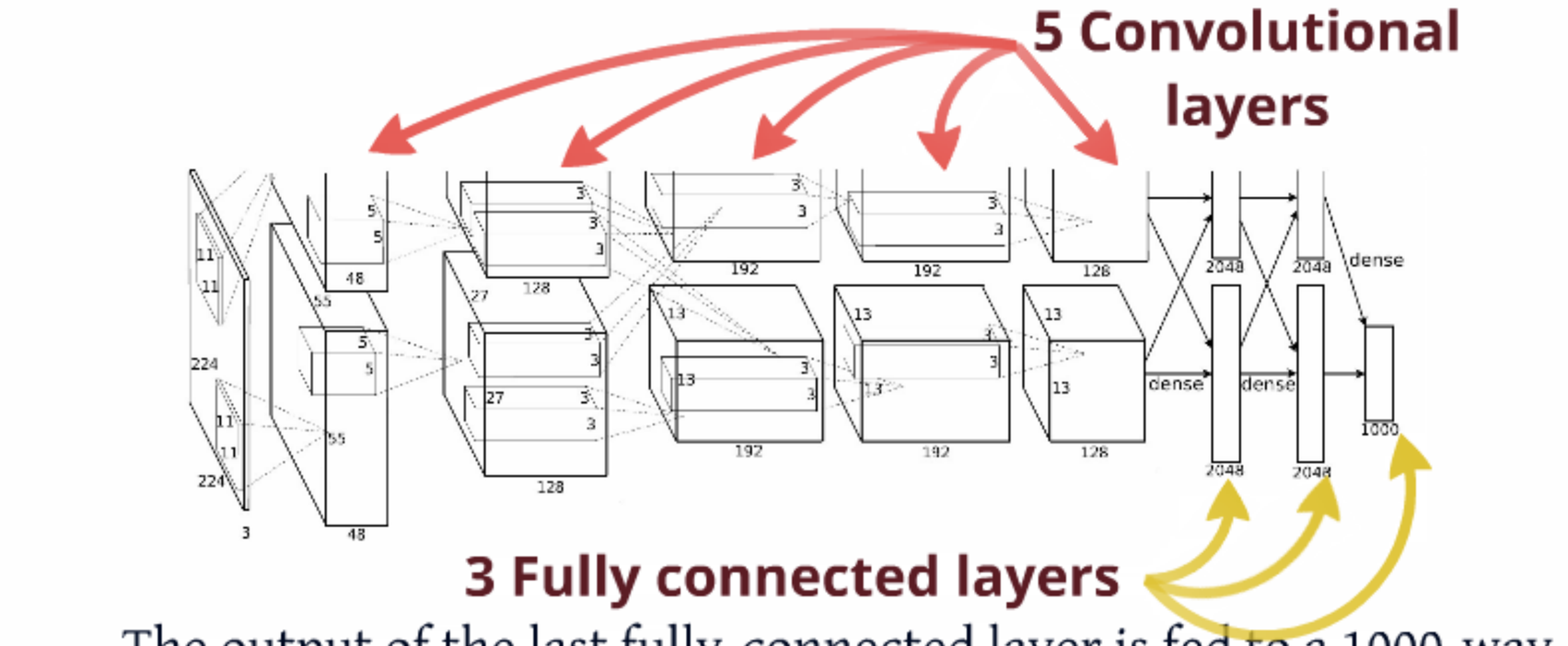
If that is not enough to get good reviews, I just don't know what is.

So, I'm giving up on submitting to computer vision conferences altogether.

**Learning Hierarchical Features for Scene Labeling**

Clement Farabet, Camille Couprie, Laurent Najman, Yann LeCun [PAMI '13]

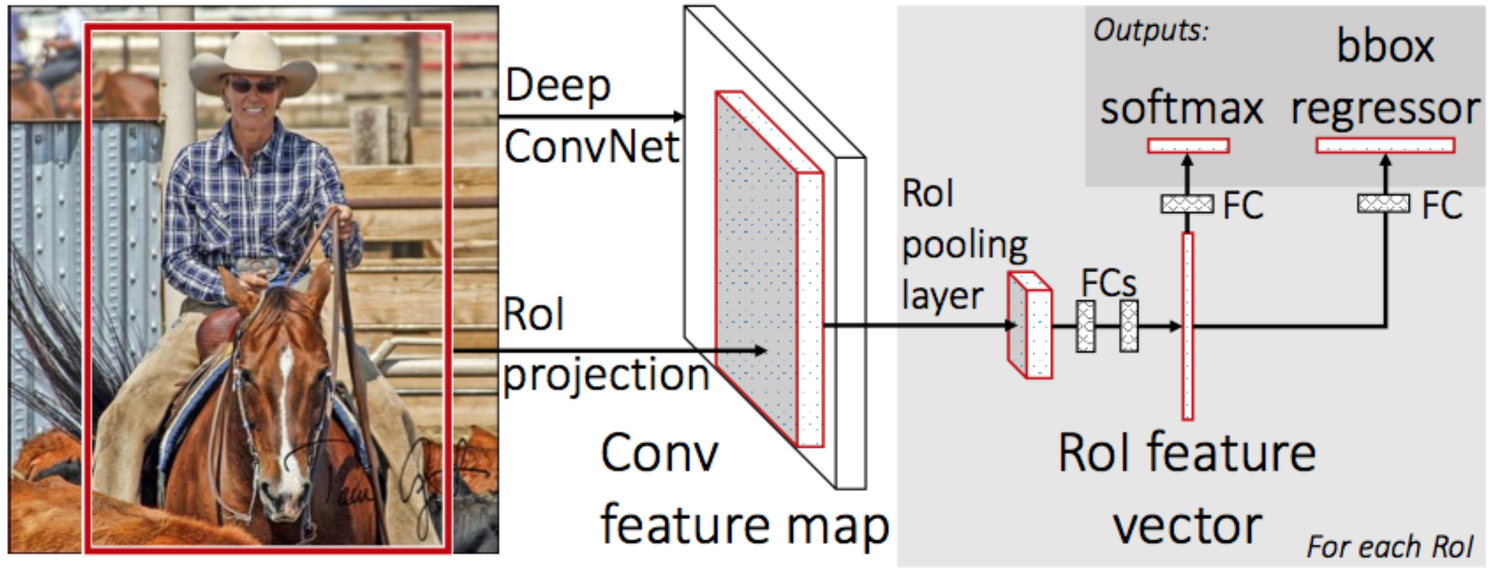
# AlexNet Architecture - 7 hidden weight layers




The output of the last fully-connected layer is fed to a 1000-way softmax which produces a distribution over the 1000 class labels
The ReLU non-linearity is applied to the output of every convolutional and fully-connected layer.

# Detection

Fast R-CNN
- convolve once
- project + detect




*Faster* R-CNN
- end-to-end proposals and detection
- 200 ms / image inference
- fully convolutional Region Proposal Net
  + Fast R-CNN

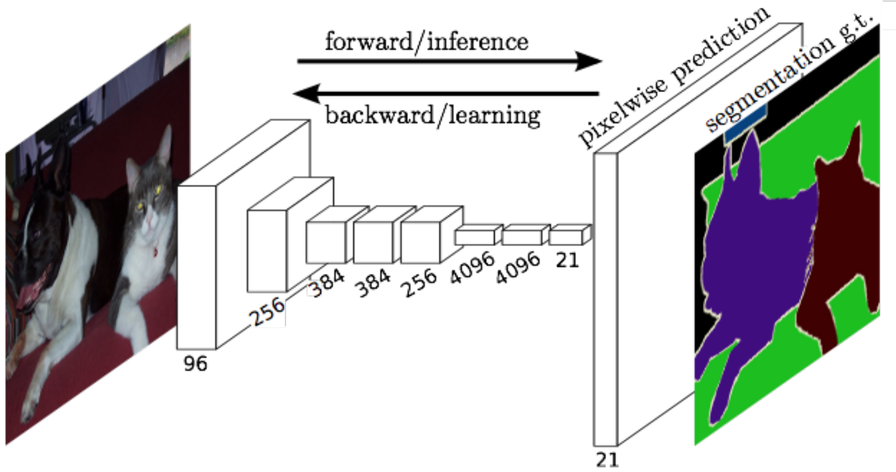
arXiv and code for Fast R-CNN

**Ross Girshick, Shaoqing Ren,**
**Kaiming He, Jian Sun**

# Pixelwise Prediction

Fully convolutional networks for pixel prediction applied to semantic segmentation

- end-to-end learning
- efficient inference and learning
  150 ms per-image prediction
- multi-modal, multi-task




Further applications

- depth
- boundaries
- flow + more

CVPR15 arXiv and pre-release

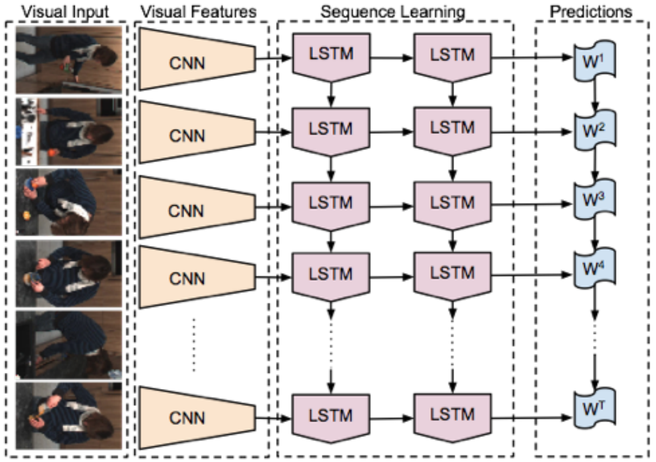
**Jon Long* & Evan Shelhamer*, Trevor Darrell**

# Sequences

Recurrent Net and Long Short Term Memory LSTM are sequential models

- video
- language
- dynamics

learned by backpropagation through time.




LRCN: Long-term Recurrent Convolutional Network

- activity recognition
- image captioning
- video captioning

CVPR15 arXiv and project site

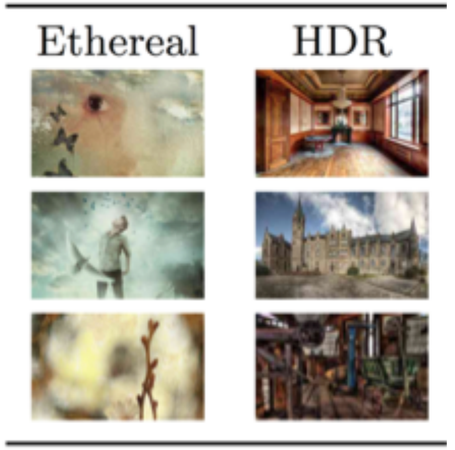
A group of young men playing a game of soccer.

**Jeff Donahue et al.**

# Pre-trained Models

Lots of Data


image by Andrej Karpathy



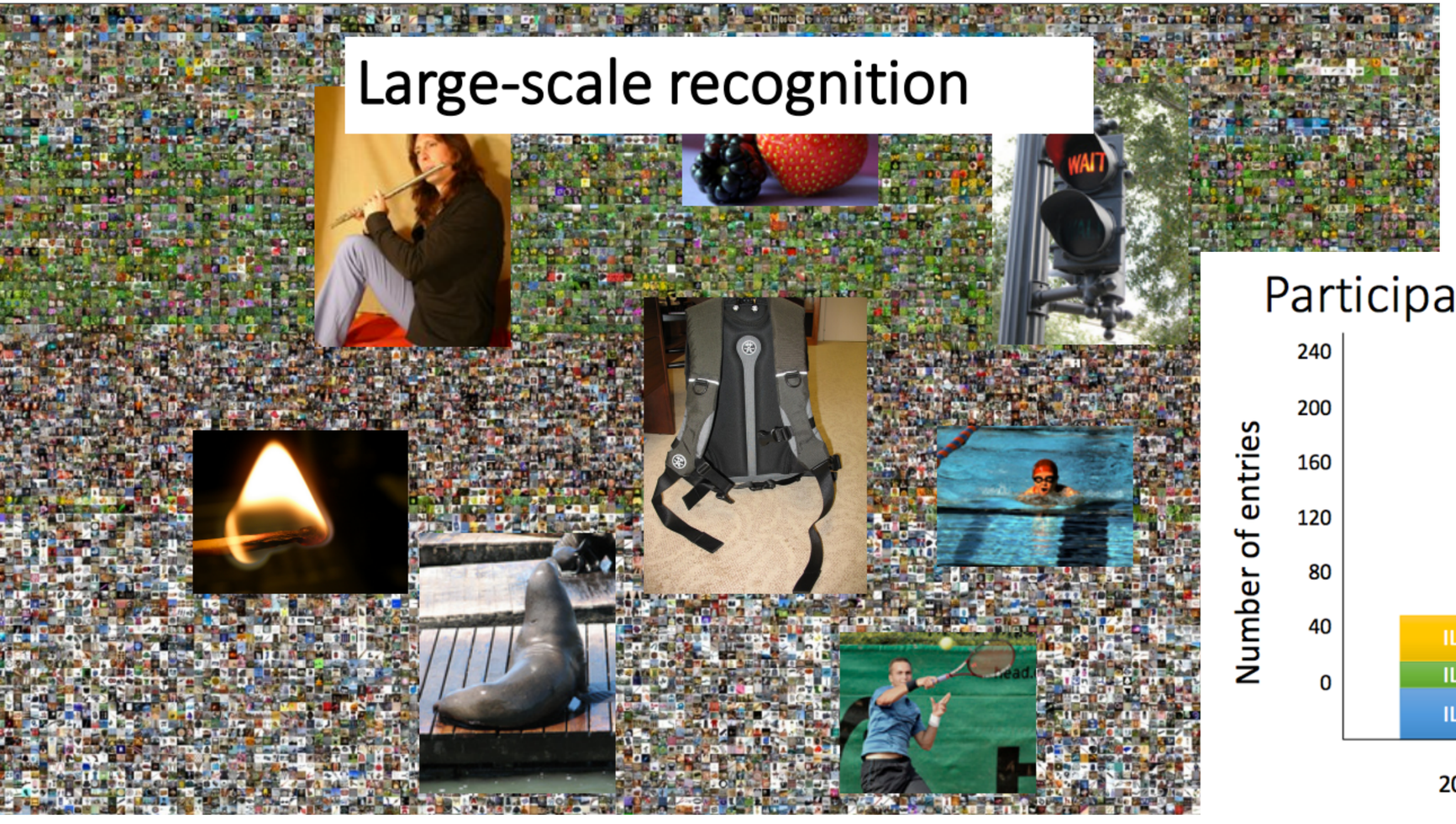

**Style Recognition**

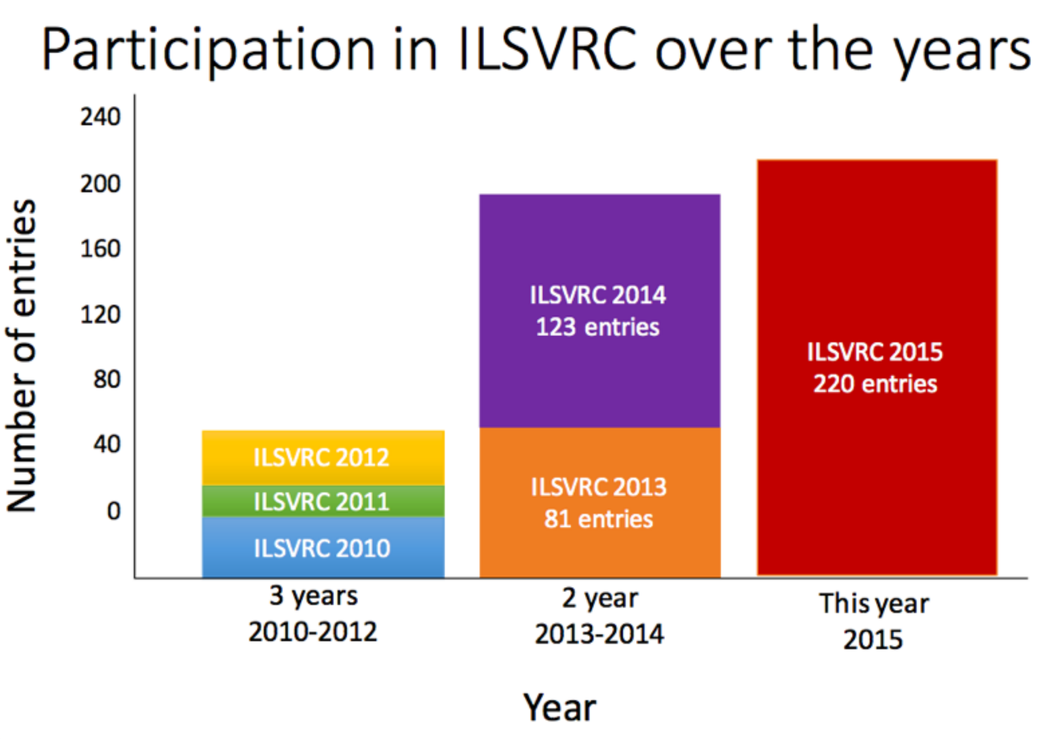
© kaggle.com

**Dogs vs. Cats**
top 10 in 10 minutes

Your Task

# IMAGENET Large Scale Visual Recognition Challenge (ILSVRC) 2015



# IMAGENET Large Scale Visual Recognition Challenge

## Year 2010

NEC-UIUC

Dense grid descriptor: HOG, LBP

Coding: local coordinate, super-vector

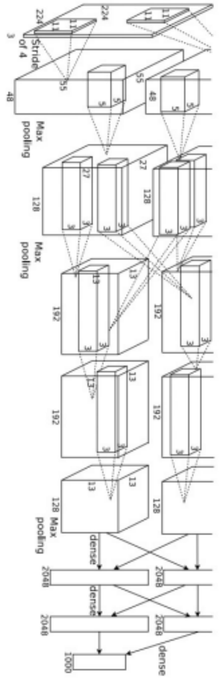
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

## Year 2012

SuperVision



[Krizhevsky NIPS 2012]

## Year 2014

GoogLeNet

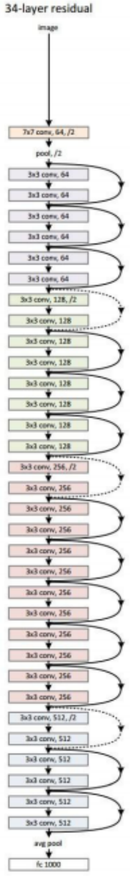

[Szegedy arxiv 2014]

VGG

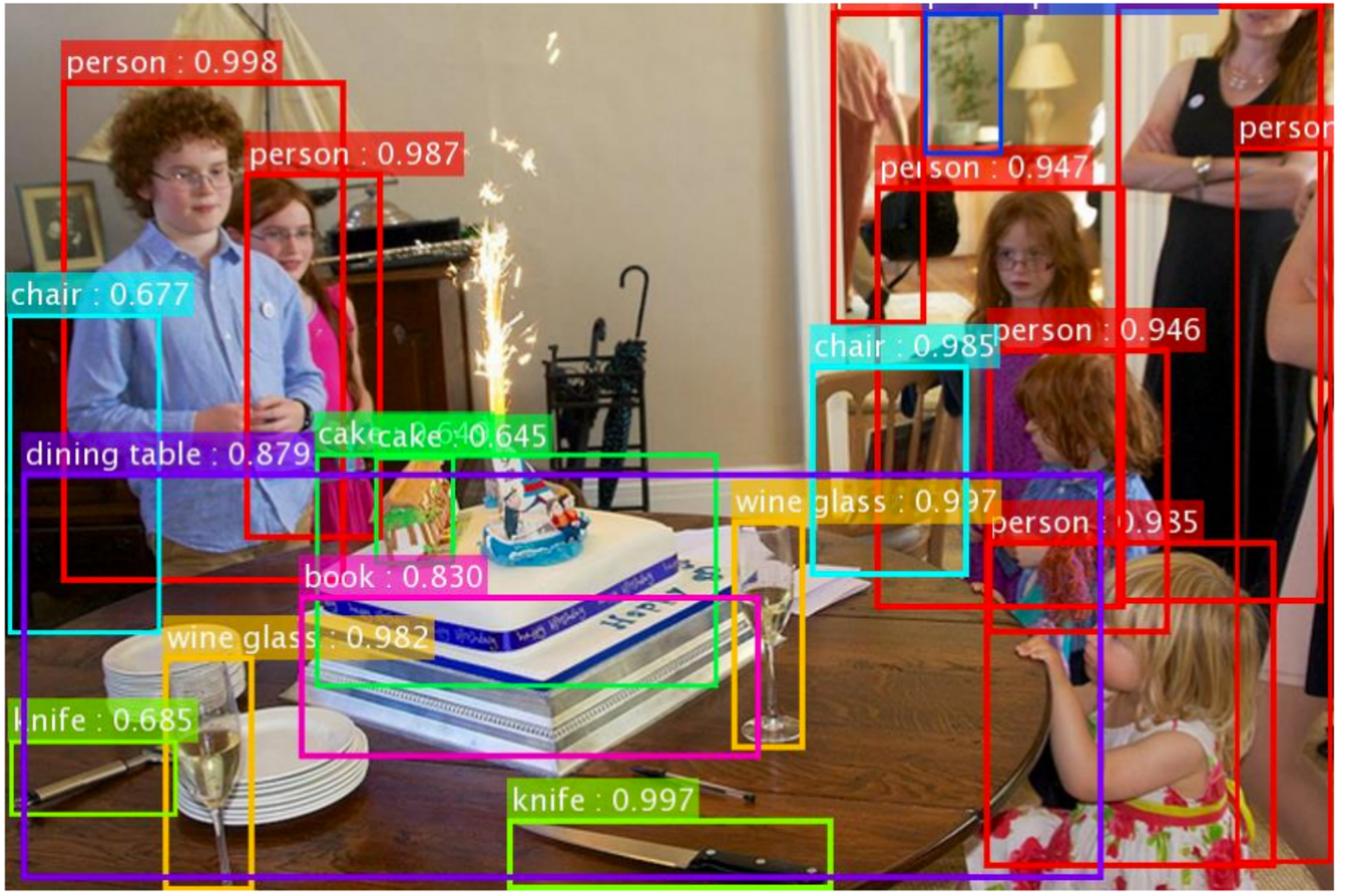

[Simonyan arxiv 2014]

## Year 2015

MSRA



Microsoft Research




# Our results on COCO – too many objects, let's check carefully!

*the original image is from the COCO dataset

ICCV15 International Conference on Computer Vision

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

Shaoqing Ren, Kaiming He, Ross Girshick, & Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". NIPS 2015.

# Visual Turing test for computer vision systems

Donald Geman[a], Stuart Geman[b,1], Neil Hallonquist[a], and Laurent Younes[a]

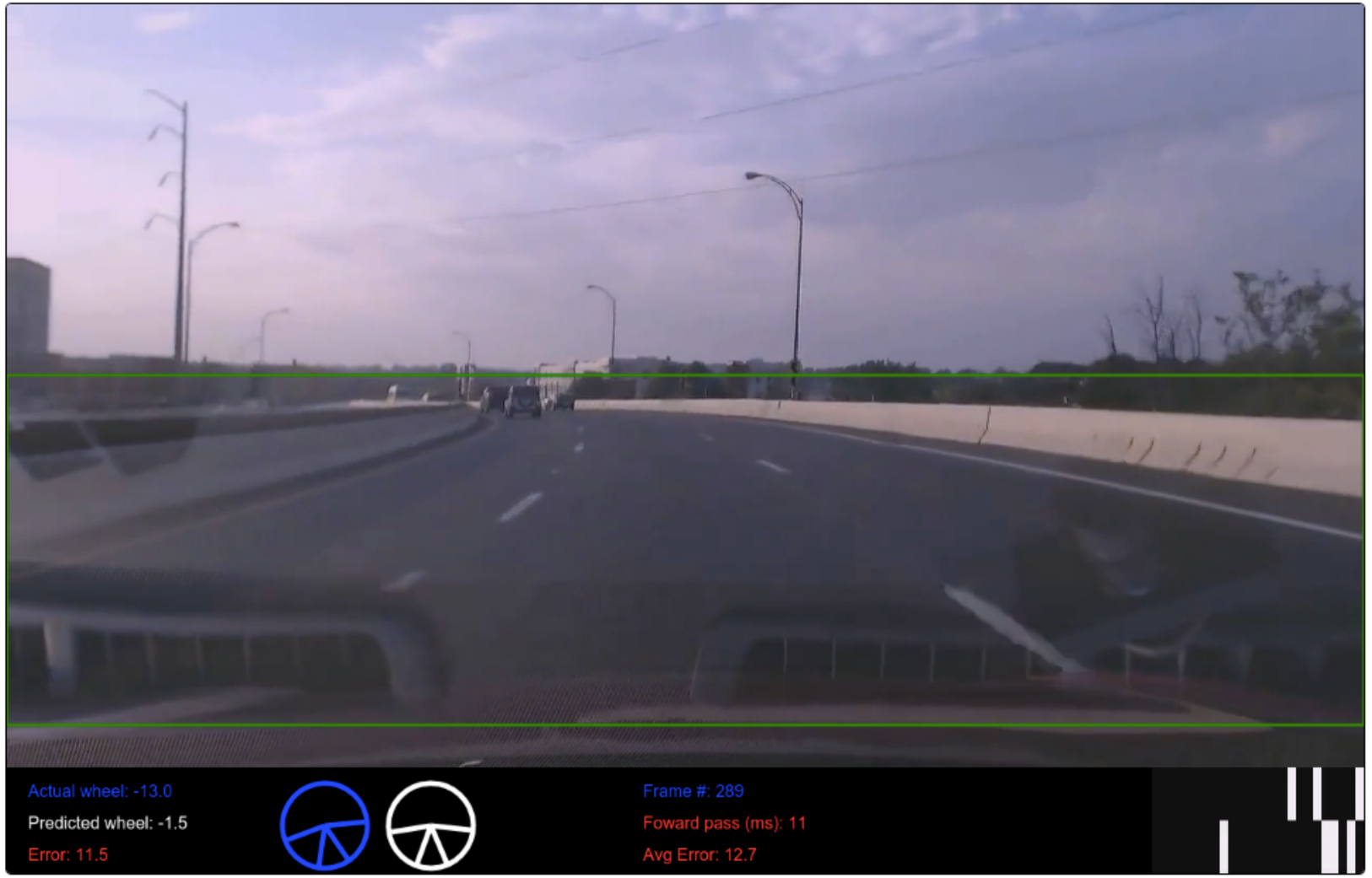
3618–3623 | PNAS | March 24, 2015 | vol. 112 | no. 12


| | |
|---|---|
| 1. Q: Is there a person in the blue region? | A: yes |
| 2. Q: Is there a unique person in the blue region? (Label this person 1) | A: yes |
| 3. Q: Is person 1 carrying something? | A: yes |
| 4. Q: Is person 1 female? | A: yes |
| 5. Q: Is person 1 walking on a sidewalk? | A: yes |
| 6. Q: Is person 1 interacting with any other object? | A: no |
| ⋮ | |
| 9. Q: Is there a unique vehicle in the yellow region? ( Label this vehicle 1) | A: yes |
| 10. Q: Is vehicle 1 light-colored? | A: yes |
| 11. Q: Is vehicle 1 moving? | A: no |
| 12. Q: Is vehicle 1 parked and a car? | A: yes |
| ⋮ | |
| 14. Q: Does vehicle 1 have exactly one visible tire? | A: no |
| 15. Q: Is vehicle 1 interacting with any other object? | A: no |
| 17. Q: Is there a unique person in the red region? | A: no |
| 18. Q: Is there a unique person that is female in the red region? | A: no |
| 19. Q: Is there a person that is standing still in the red region? | A: yes |
| 20. Q: Is there a unique person standing still in the red region? (Label this person 2) | A: yes |
| ⋮ | |
| 23. Q: Is person 2 interacting with any other object? | A: yes |
| 24. Q: Is person 1 taller than person 2? | A: amb. |
| 25. Q: Is person 1 closer (to the camera) than person 2? | A: no |
| 26. Q: Is there a person in the red region? | A: yes |
| 27. Q: Is there a unique person in the red region? (Label this person 3) | A: yes |
| ⋮ | |
| 36. Q: Is there an interaction between person 2 and person 3? | A: yes |
| 37. Q: Are person 2 and person 3 talking? | A: yes |


SEEK
TRUTH
LEGEA

# Deep Learning IRL

# Self-Driving Cars



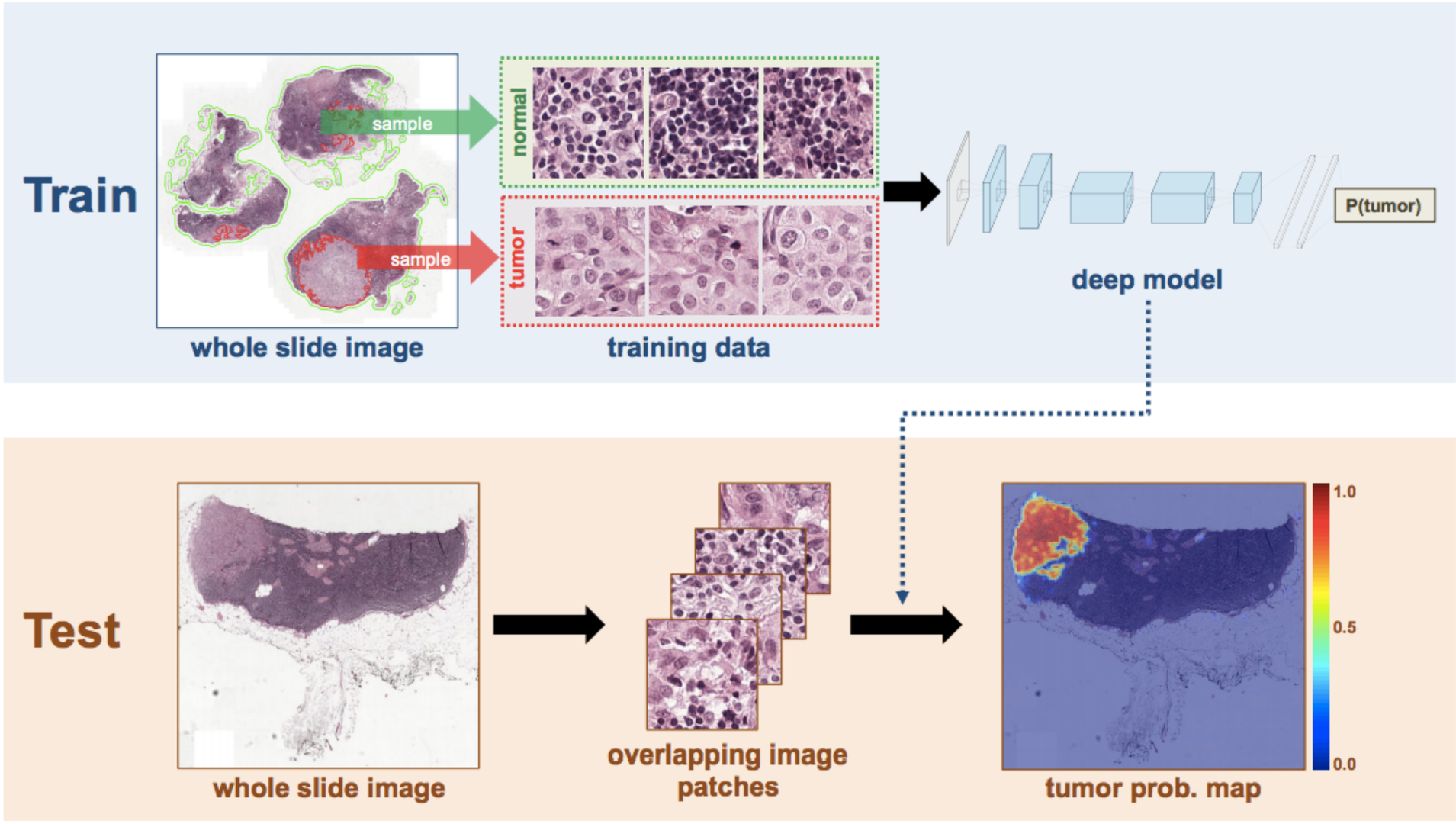
# Product Search

# Auto-tagging

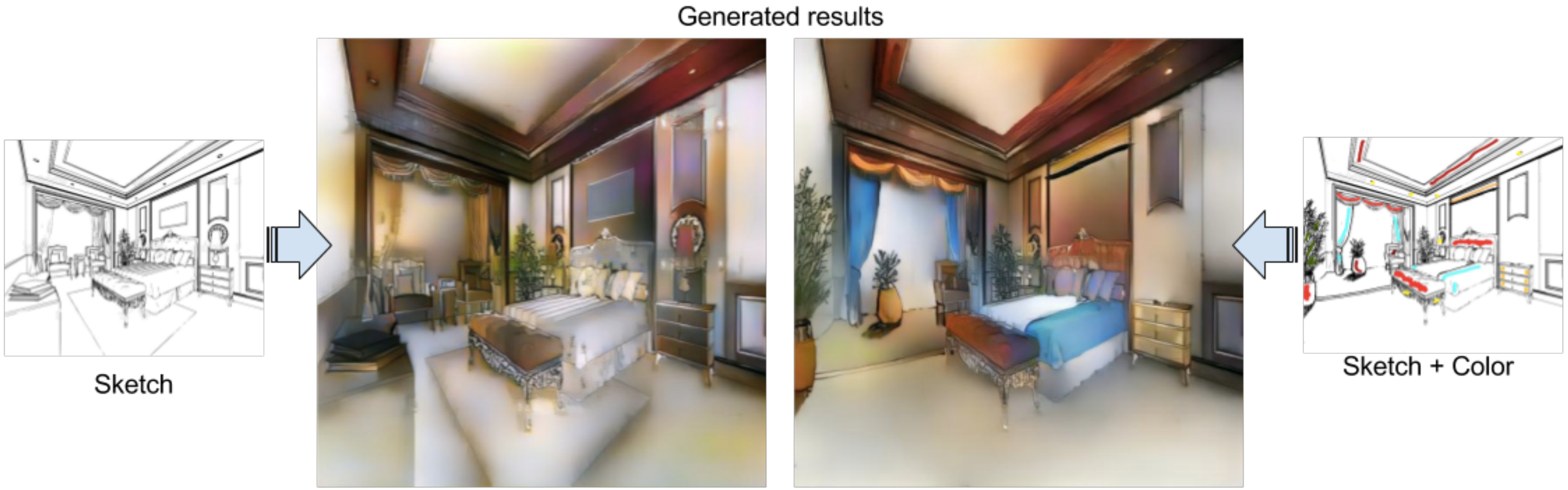
Predicted Tags

north america | politics | business | leader | finance | commerce | government | meeting | conference | group

# Medical Research



# Image Generation



# What is this class about?

# Course Description

- Learned Representations
- Object Proposals
- CNN detection and segmentation
- Weakly Supervised and Unsupervised CNNs
- Recurrent Neural Nets and Long-Short Term Memory Networks
- Generative Networks
- Siamese / Ranking / Triplet Networks
- Co-attention models
- Residual Nets
- Ensemble methods
- Reinforcement Learning

# Preparation

- Programming Experience
  - Python, Matlab
- Math
  - Linear Algebra, Basic Calculus, Probability
- Machine Learning
- Computer Vision

# Step 1: Datasets

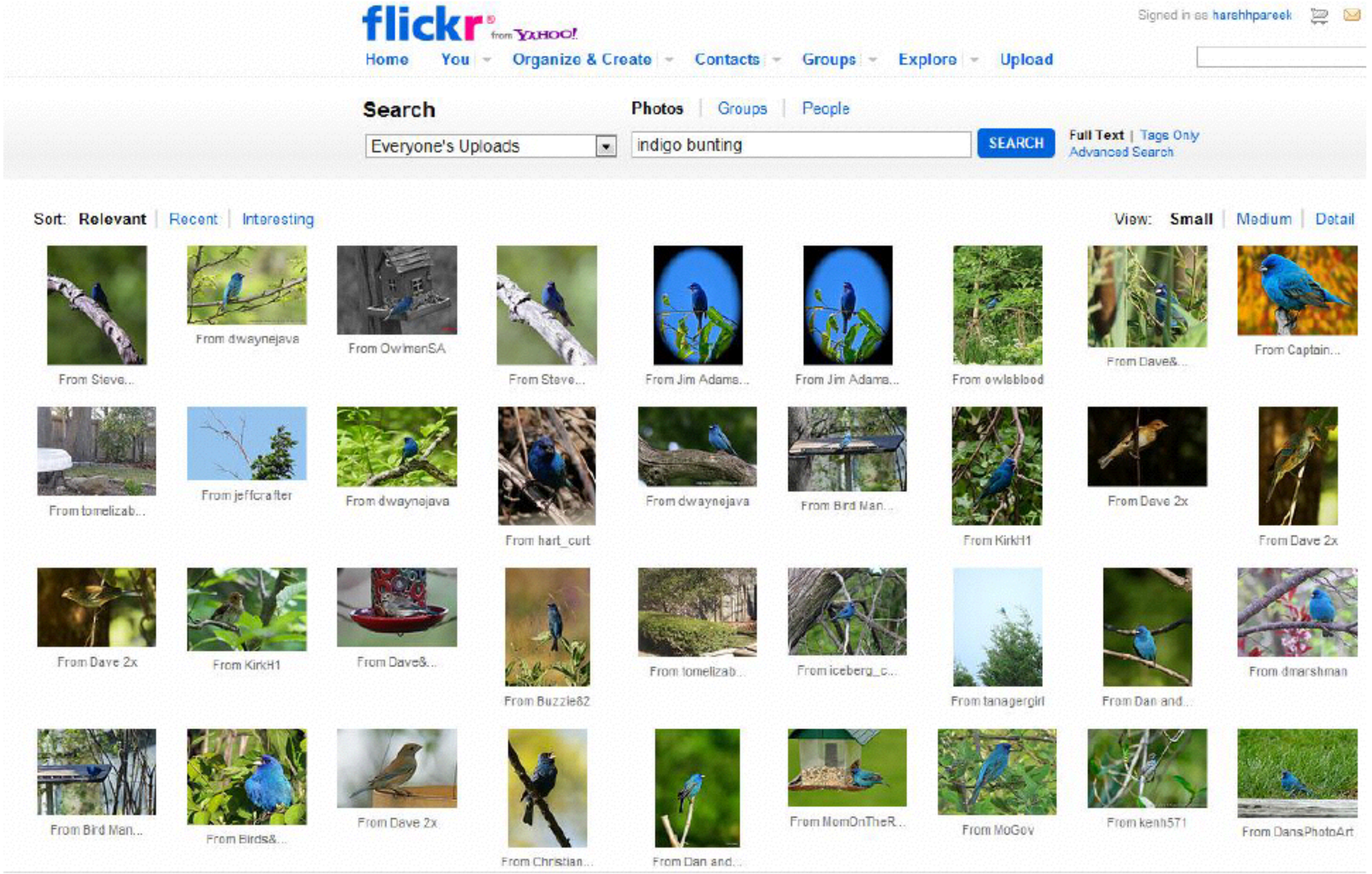
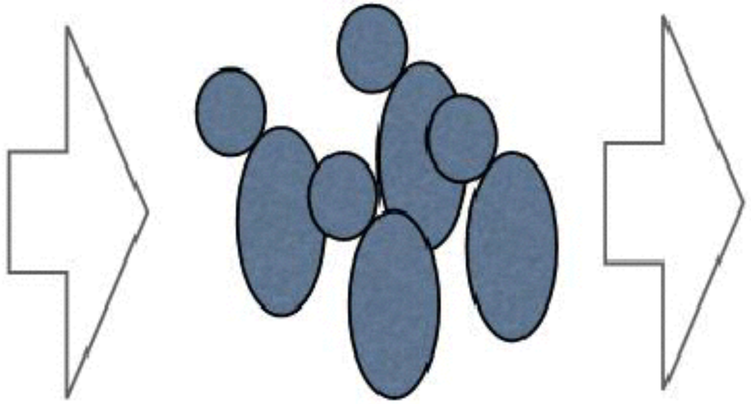



Image credit: Flickr.com

# Building datasets

6000 images from flickr.com

Annotators



amazonmechanical turk
Artificial Artificial Intelligence

Is there an Indigo bunting in the image?

100s of training images

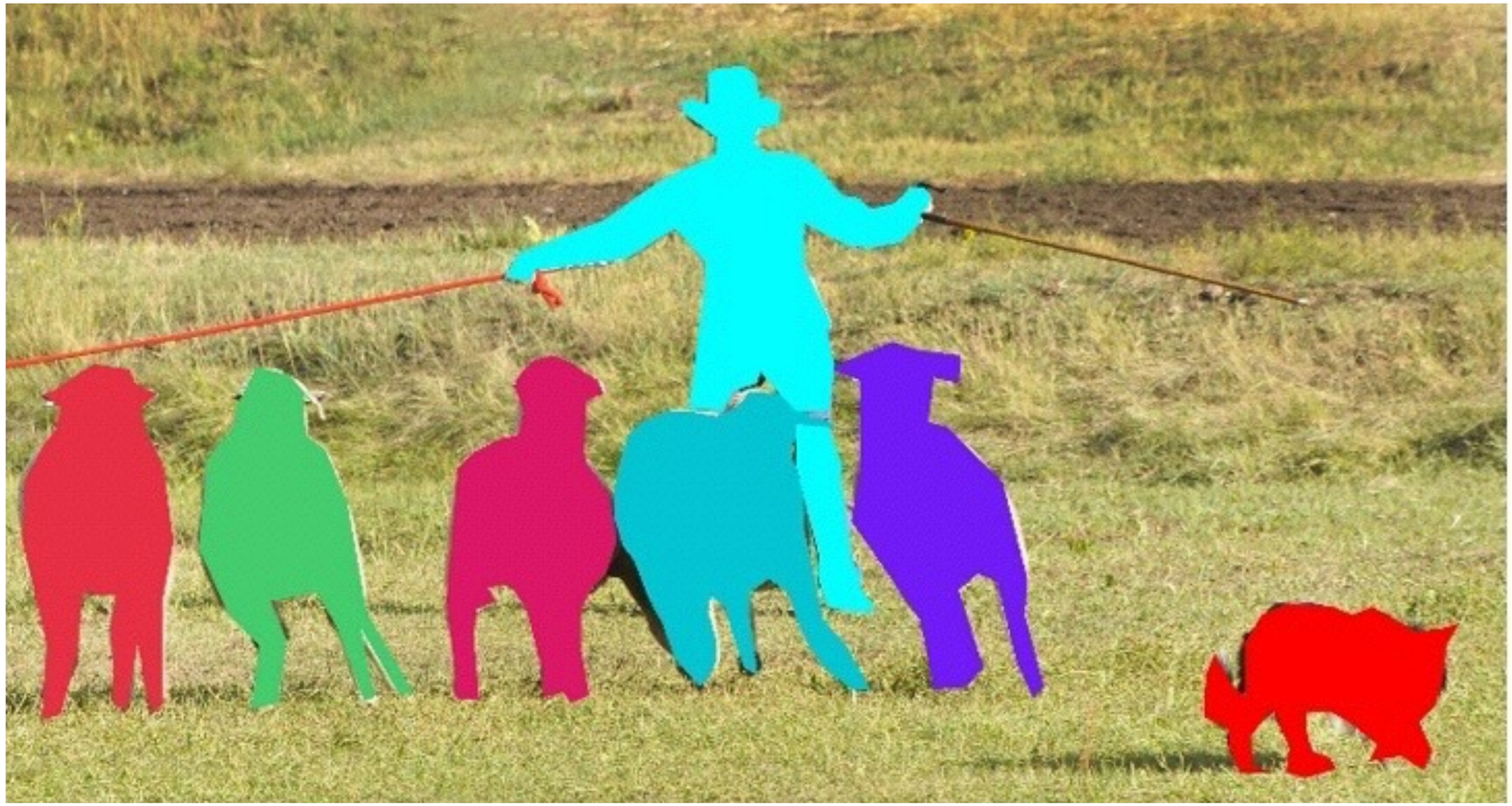
Slide credit: Welinder et al


http://mscoco.org


http://mscoco.org

✓ Instance segmentation

✓ Non-iconic Images



http://mscoco.org

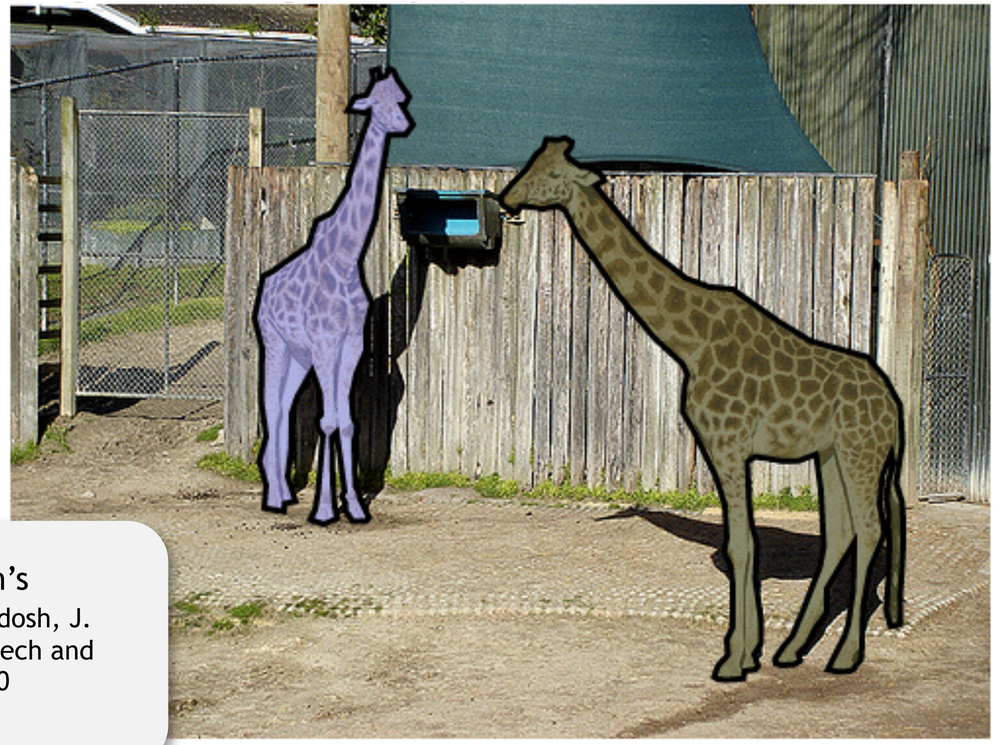
- 330,000 images
- >2 million instances (700k people)
- Every instance is segmented
- 7.7 instances per image (3.5 categories)


http://mscoco.org

# Beyond detection

✓ Sentences

two giraffe standing next to each other in front of a wooden fence.
two giraffes standing in the dirt near a gate.
two giraffes stand by a food box awaiting the goods.
two giraffes are standing next to a wooden fence.
two giraffes standing alone by a picket fence.

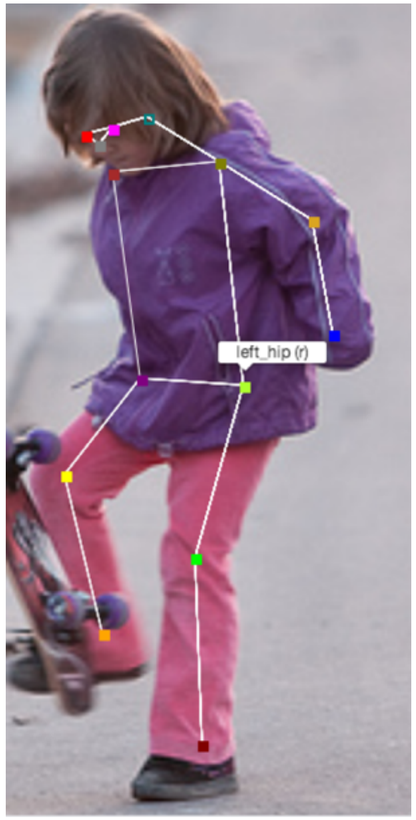


Collecting Image Annotations Using Amazon's Mechanical Turk, C. Rashtchian, P. Young, M. Hodosh, J. Hockenmaier, NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, 2010

http://mscoco.org

# Beyond detection

✓ Keypoints
(provided by Facebook)

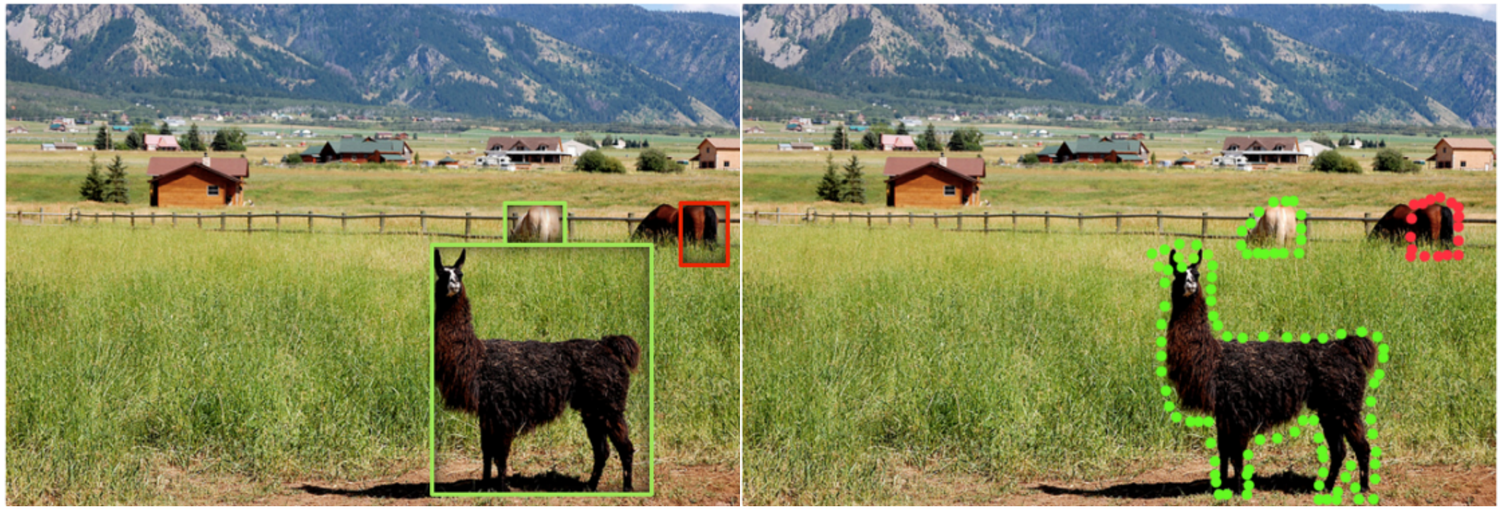



http://mscoco.org

# MS COCO Challenges at ICCV 2015

Microsoft COCO
Common Objects in Context


- Detection
- Segmentation



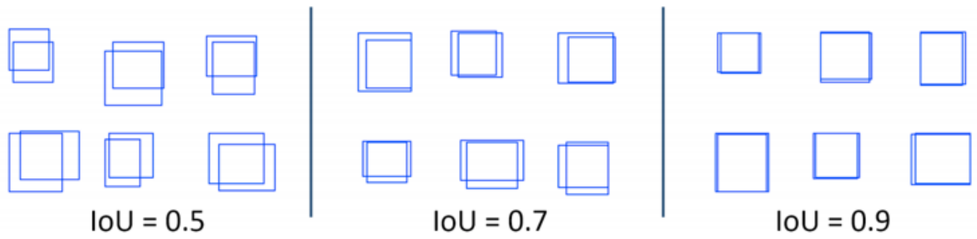
# Evaluation Metrics

**Average Precision (AP):**

| | |
|---|---|
| AP | % AP at IoU=.50:.05:.95 **(determines challenge winner)** |
| $AP^{IoU=.50}$ | % AP at IoU=.50 (PASCAL VOC metric) |
| $AP^{IoU=.75}$ | % AP at IoU=.75 (strict metric) |

## Challenges Score: AP

- AP is averaged over multiple IoU values between 0.5 and 0.95 (and categories, size).
- More comprehensive metric than the traditional AP at a fixed IoU value (0.5 for Pascal ).



# Evaluation Metrics

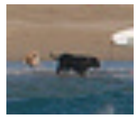
**AP Across Scales:**

| | |
|---|---|
| $AP^{small}$ | % AP for small objects: area < $32^2$ |
| $AP^{medium}$ | % AP for medium objects: $32^2$ < area < $96^2$ |
| $AP^{large}$ | % AP for large objects: area > $96^2$ |

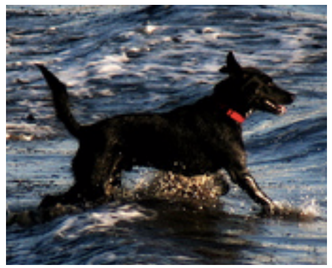
## Other Scores: Size AP

- AP is averaged over small (A < 32 x 32), medium (32x 32 < A < 96 x 96) and large (A > 96 x 96) instances of objects.

< 32x32



32x32 < A < 96x96



> 96x96

# Evaluation Metrics

**Average Recall (AR):**

| | |
|---|---|
| $AR^{max=1}$ | % AR given 1 detection per image |
| $AR^{max=10}$ | % AR given 10 detections per image |
| $AR^{max=100}$ | % AR given 100 detections per image |

**AR Across Scales:**

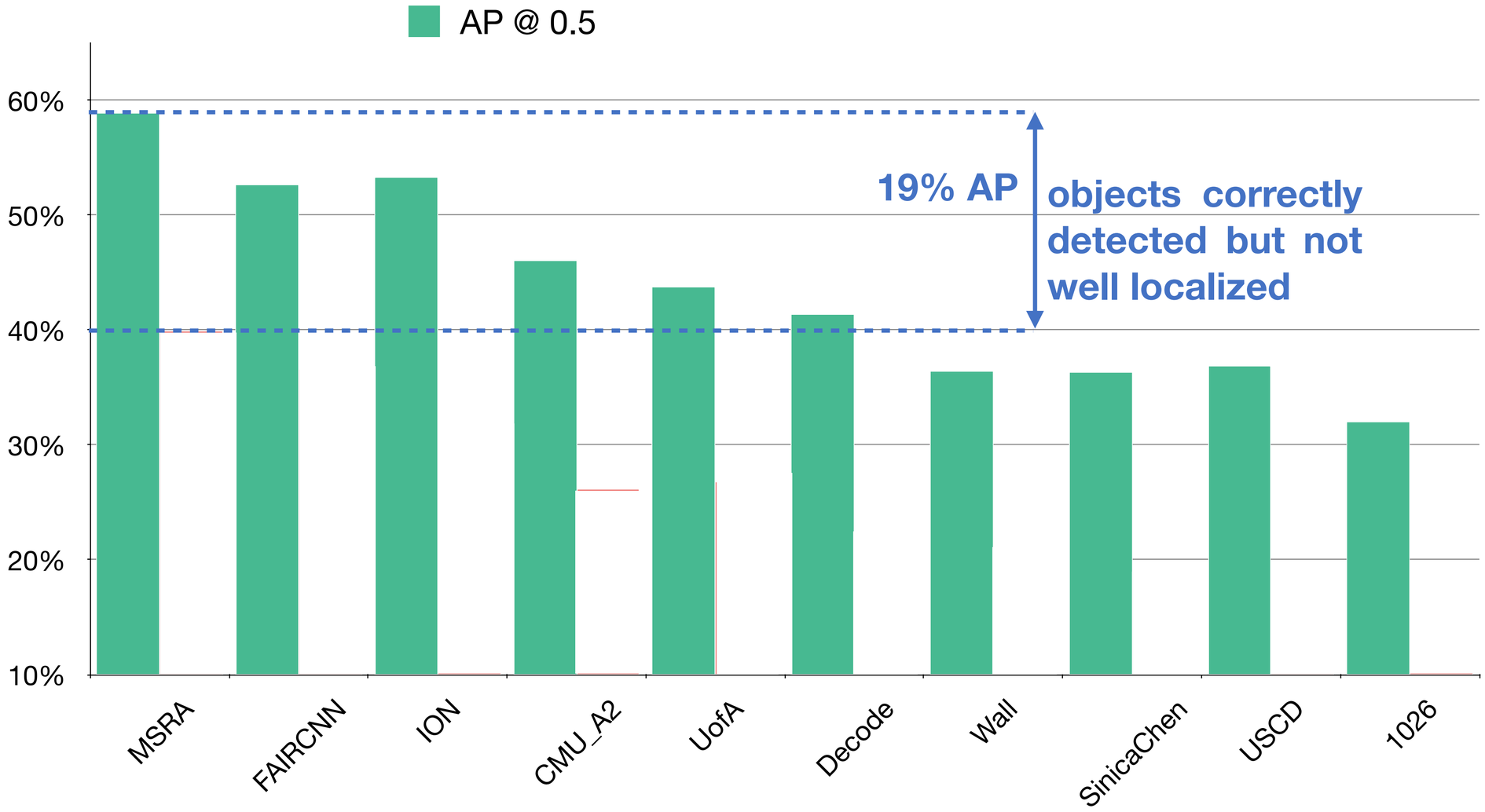
| | |
|---|---|
| $AR^{small}$ | % AR for small objects: area < $32^2$ |
| $AR^{medium}$ | % AR for medium objects: $32^2$ < area < $96^2$ |
| $AR^{large}$ | % AR for large objects: area > $96^2$ |

## Other Scores: AR

- Measures the maximum recall over a fixed number of detections allowed in the image of 1, 10, 100.
- AR is averaged over small ($A < 32 \times 32$), medium ($32 \times 32 < A < 96 \times 96$) and large ($A > 96 \times 96$) instances of objects.

# Detection Leaderboard (II)

## Object Localization can improve

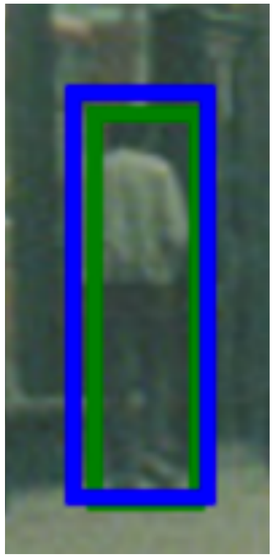


# BBox detections and IoU

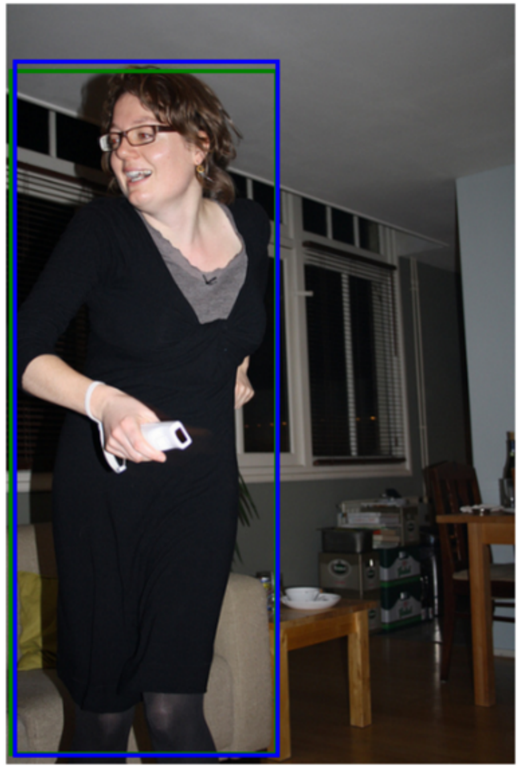
## Also hard for humans

IoU = 0.5

IoU = 0.7



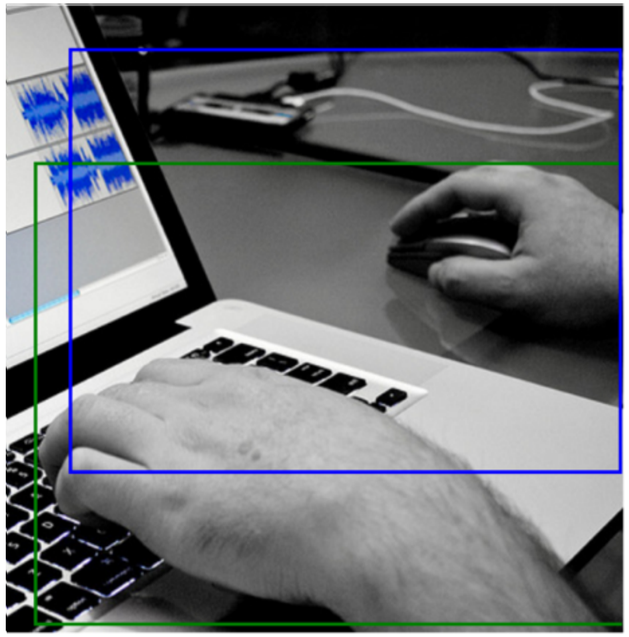
IoU = 0.95



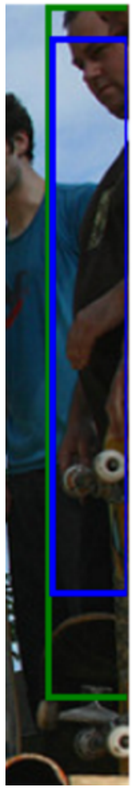
# BBox detections and IoU

## Also hard for humans

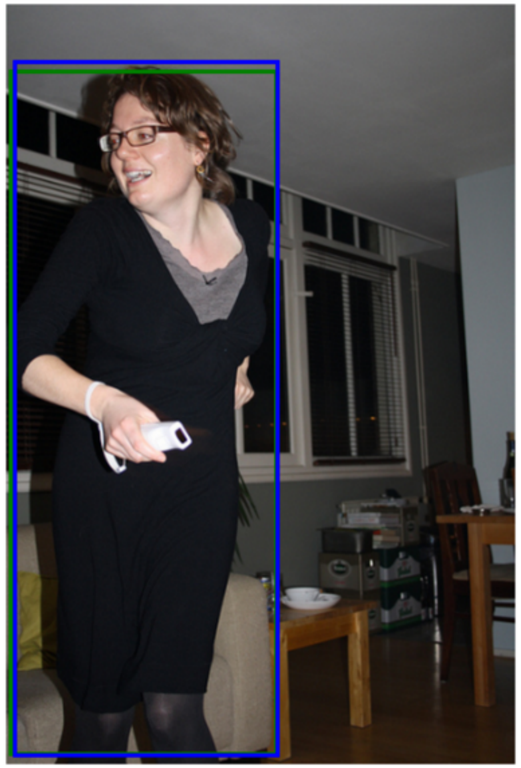
IoU = 0.5



IoU = 0.75

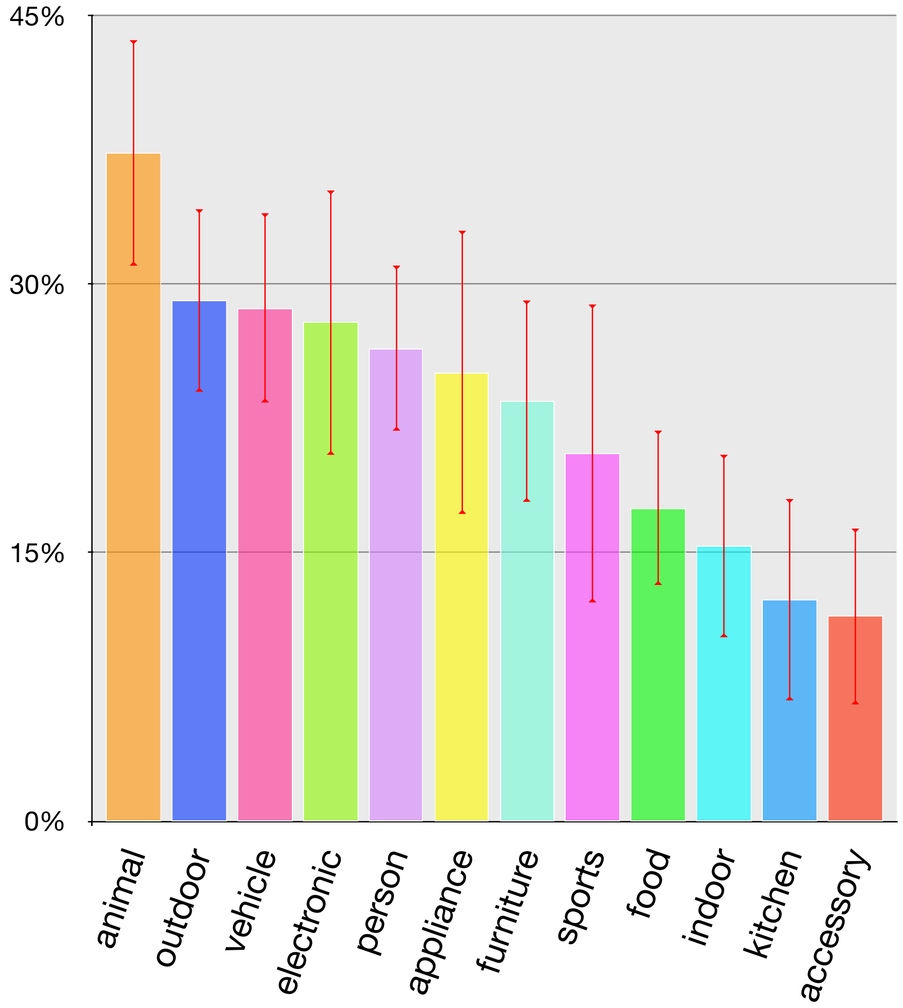


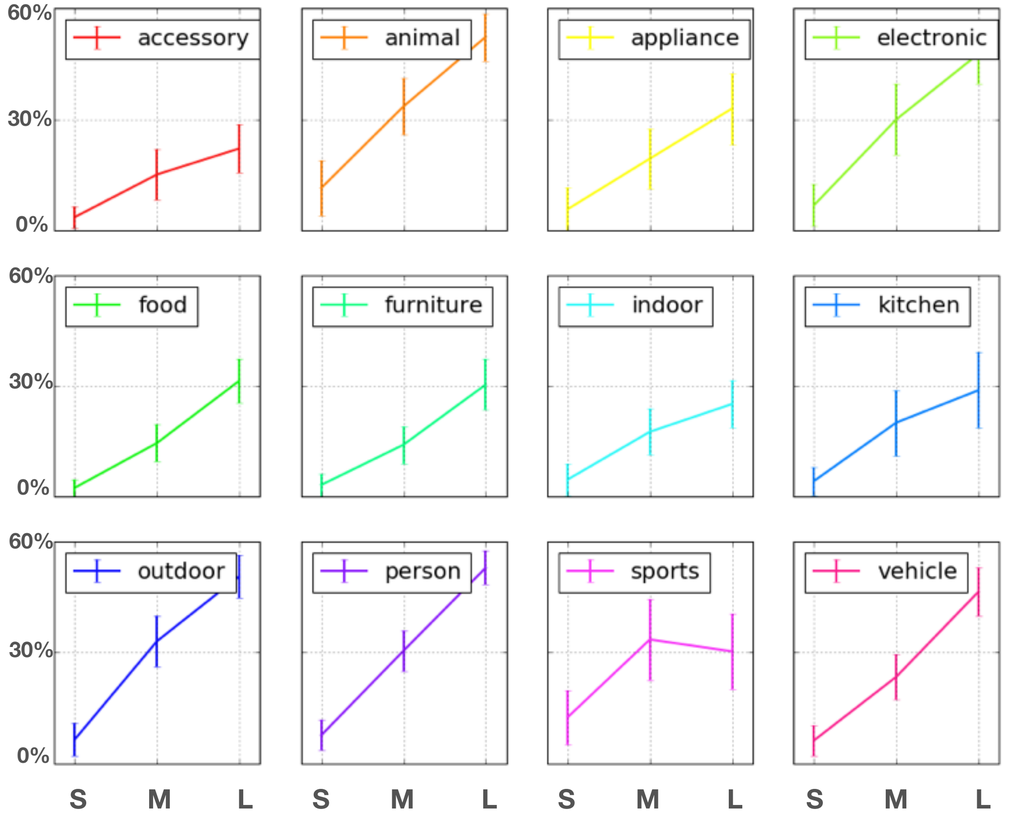
IoU = 0.95



# Performance Breakdown (I)

## COCO AP varies across supercategories and size



# Bounding Box Detection Errors (I)

## What type of errors are algorithms making?

AP @ IoU = [0.5; 0.75]

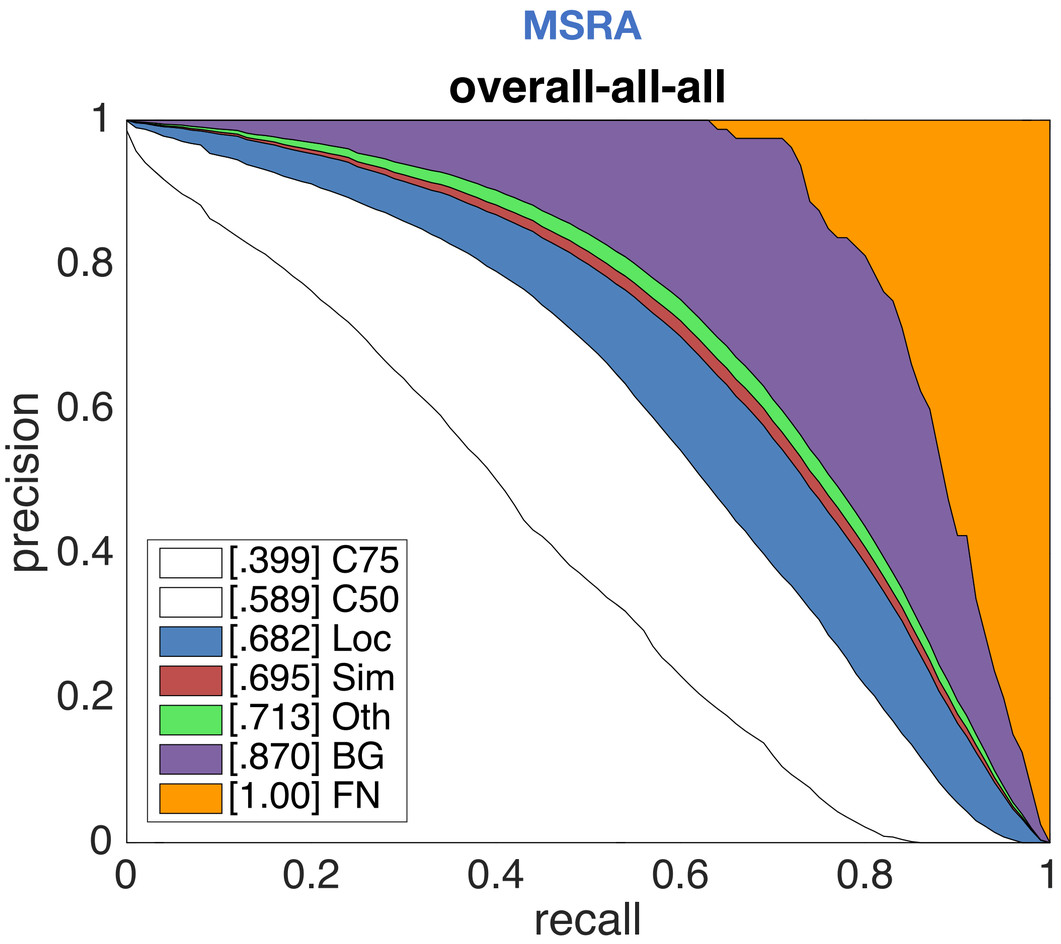
AP @ IoU = 0.1

Super-category FP removed

Category FP removed

Background FP removed

FN errors are removed




[1] - Hoiem, Derek et al., “Diagnosing error in object detectors.”, ECCV 2012. [2] - Dollár, Piotr - https://github.com/pdollar/coco

# Bounding Box Detection Errors (II)

AP @ IoU = [0.5; 0.75]

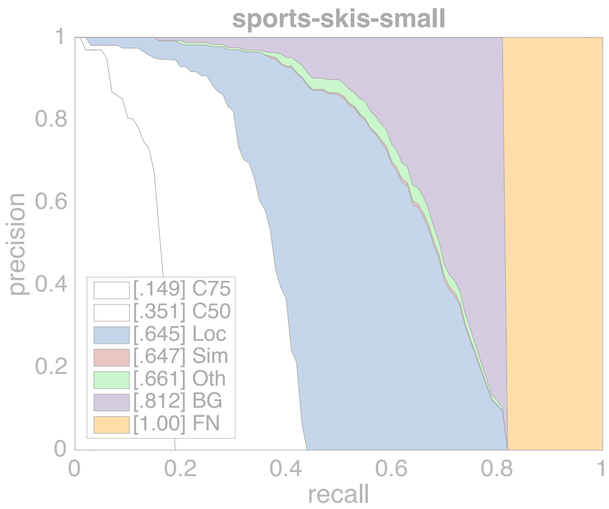
AP @ IoU = 0.1

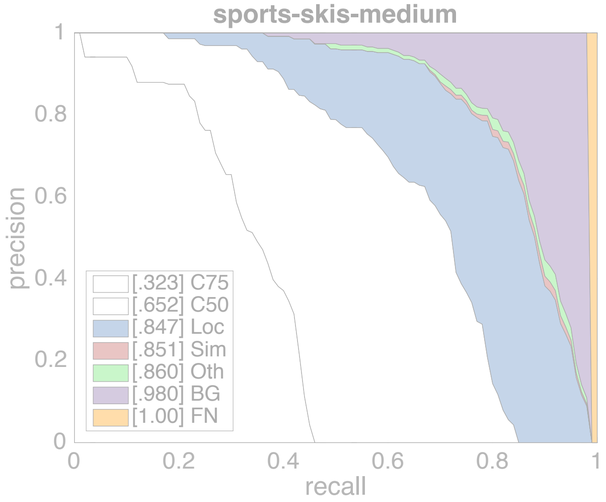
Super-category FP removed

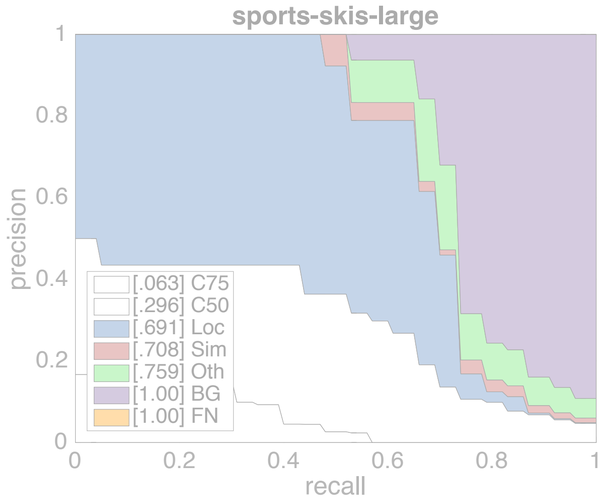
Category FP removed

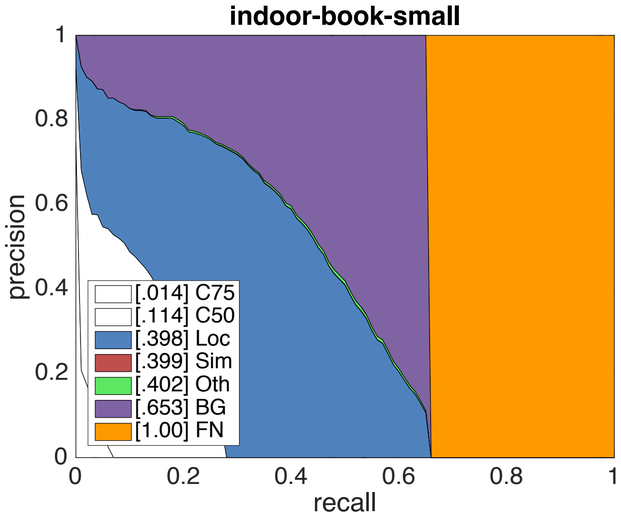
Background FP removed

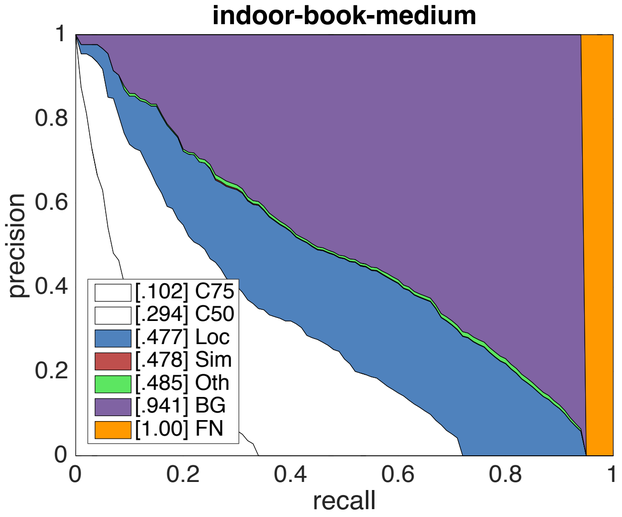
FN errors are removed

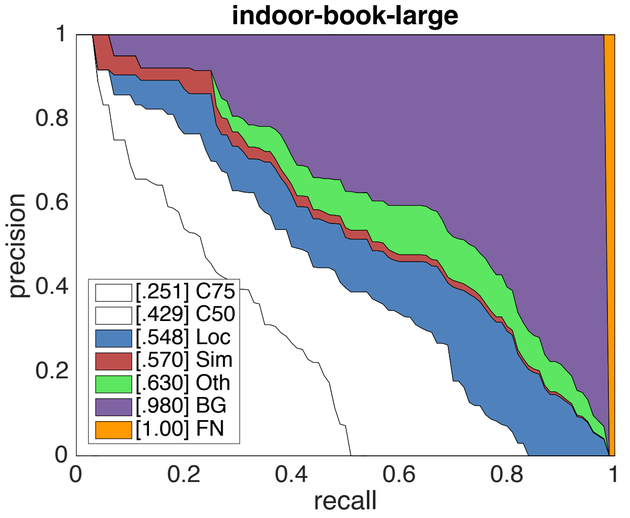


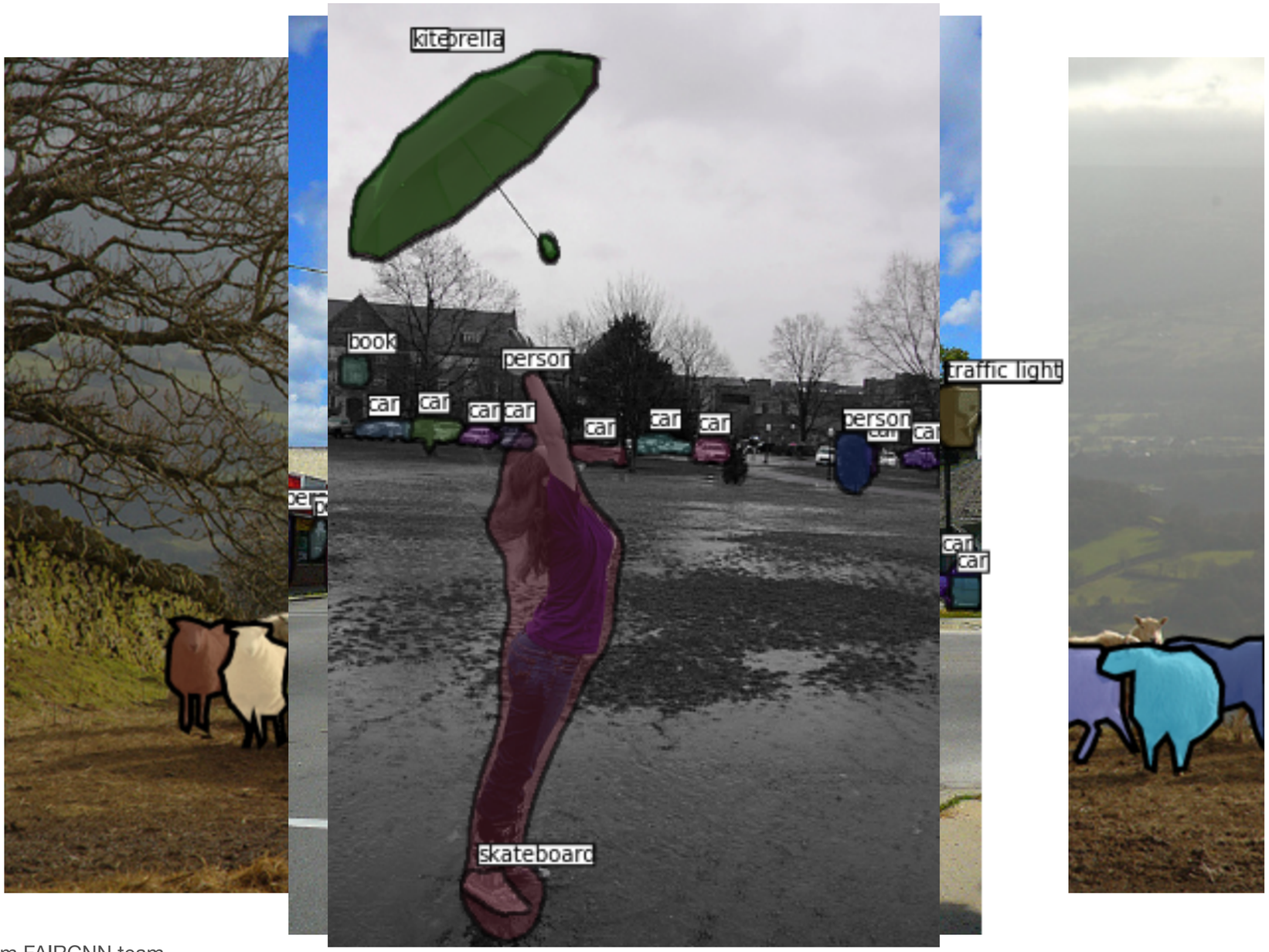



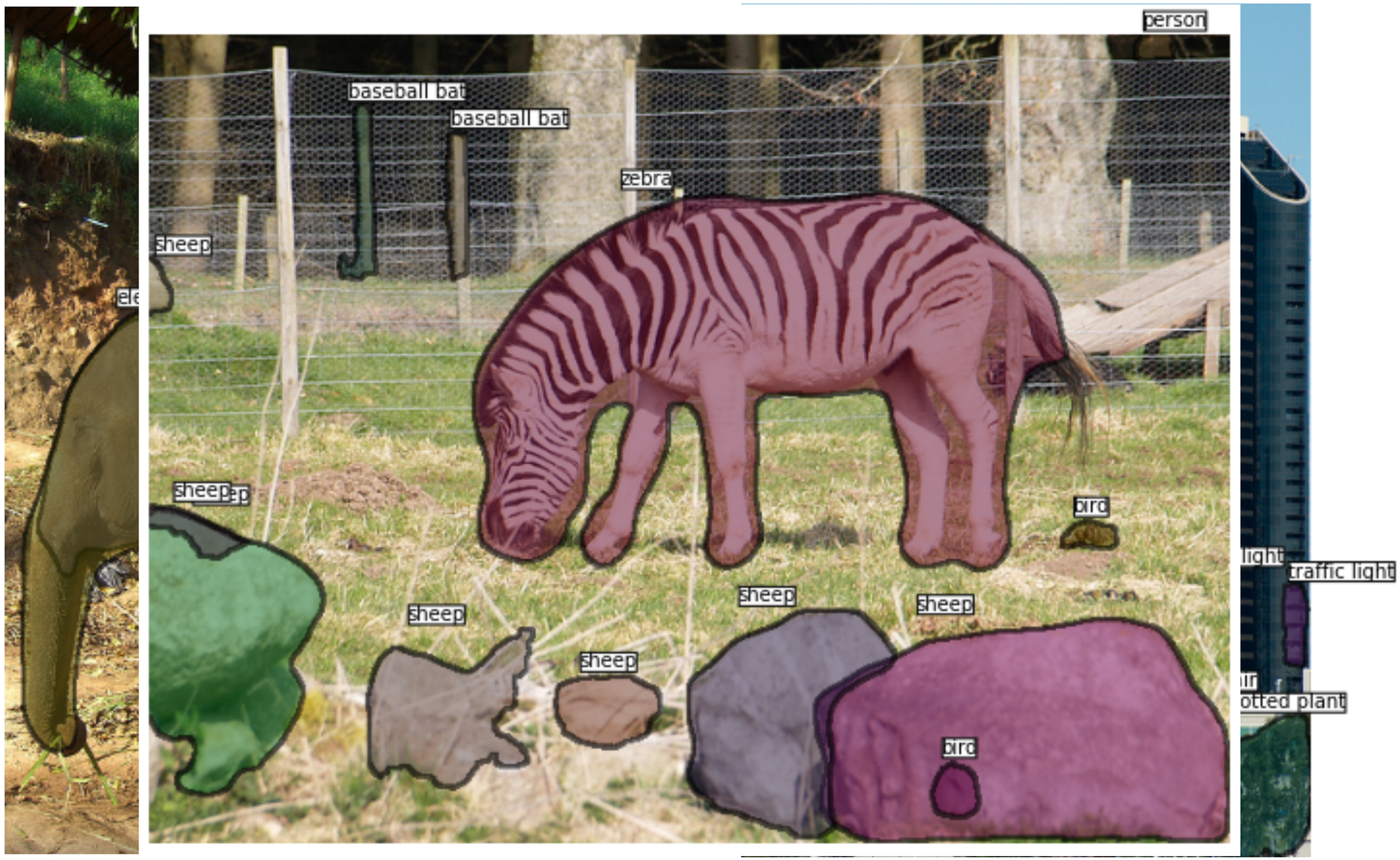

Results from MSRA team.

# Some success cases …




Results from FAIRCNN team.

# … and some failures




Results from FAIRCNN team.