Object Recognition II

Linda Shapiro

ECE/CSE 576

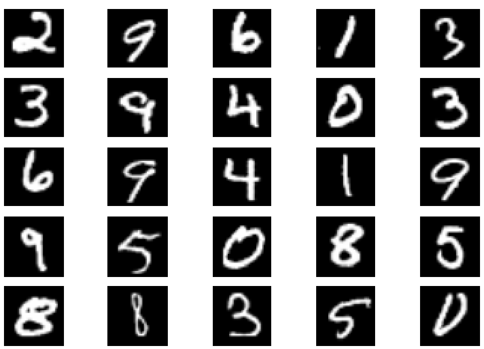
with CNN slides from Ross Girshick

Outline

- Object detection
  - the task, evaluation, datasets
- Convolutional Neural Networks (CNNs)
  - overview and history
- Region-based Convolutional Networks (R-CNNs)
- You Only Look Once (YOLO)

Image classification

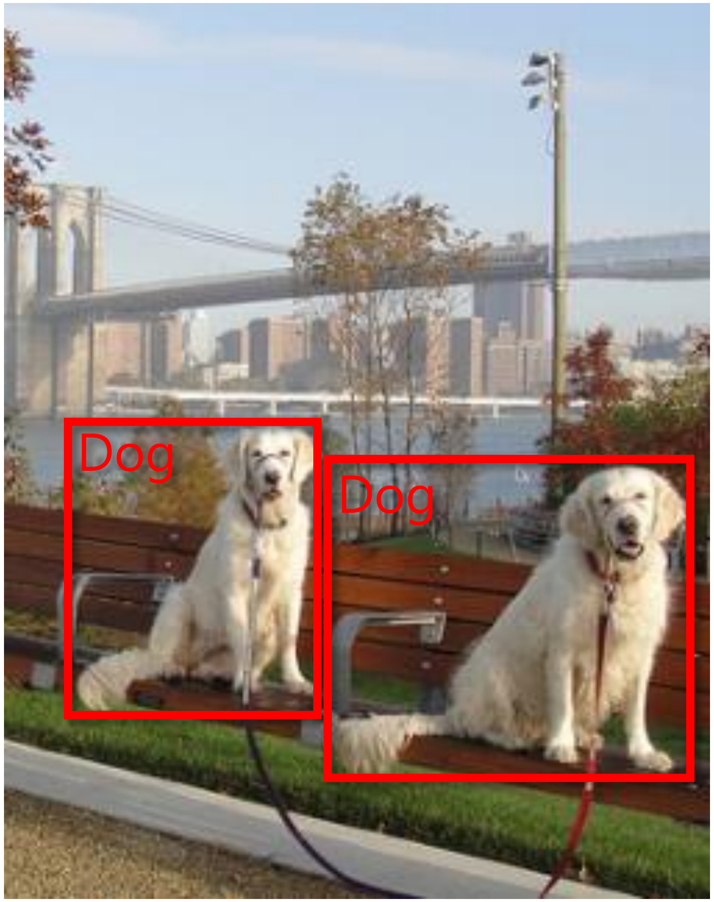
- K classes
- Task: assign correct class label to the whole image



Digit classification (MNIST)

Object recognition (Caltech-101)

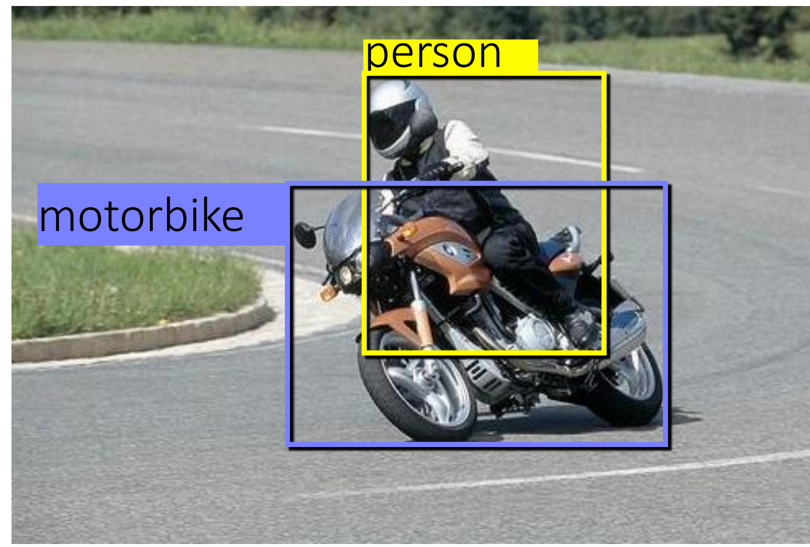
Classification vs. Detection



Problem formulation

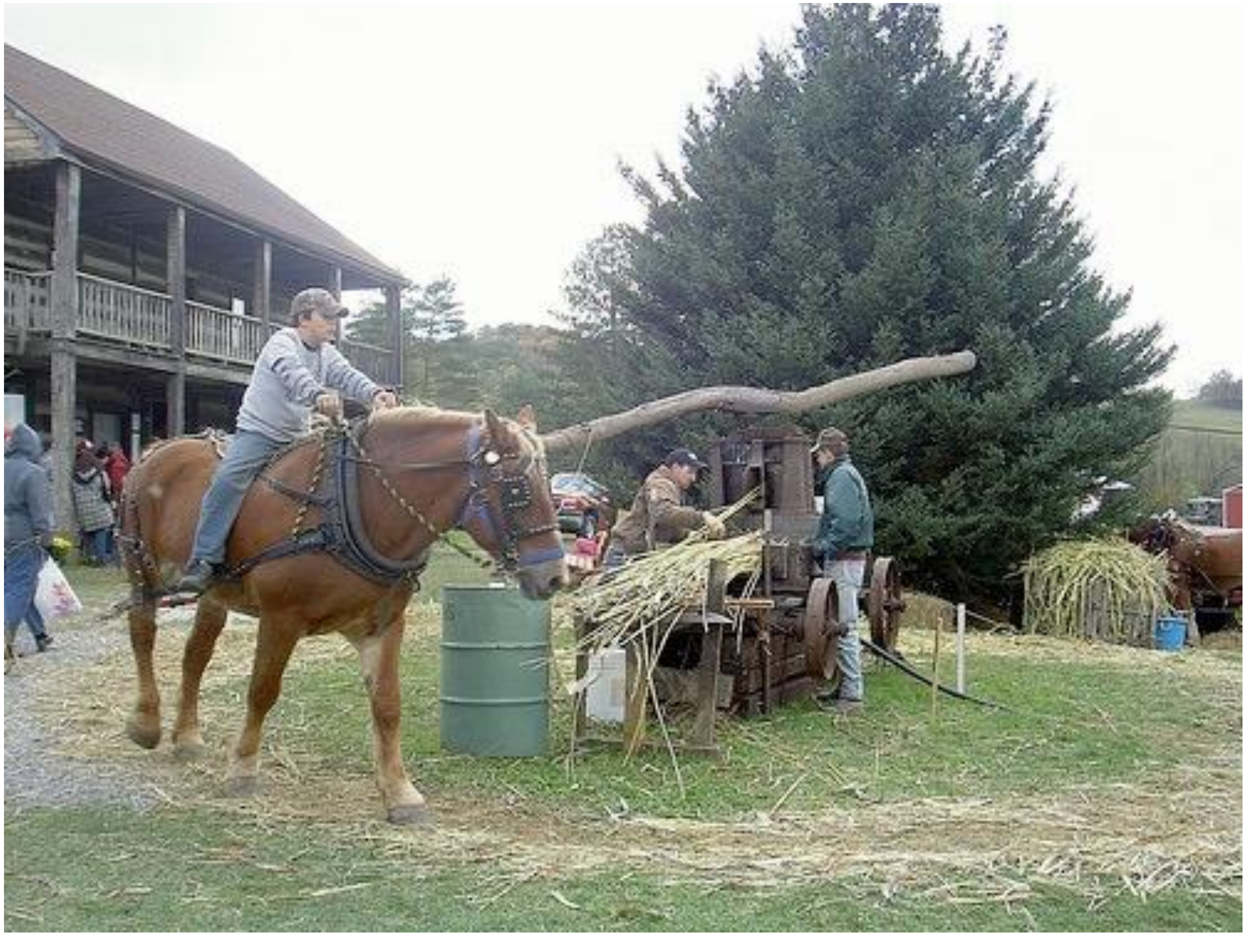
{ airplane, bird, motorbike, person, sofa }

Input



Desired output

Evaluating a detector

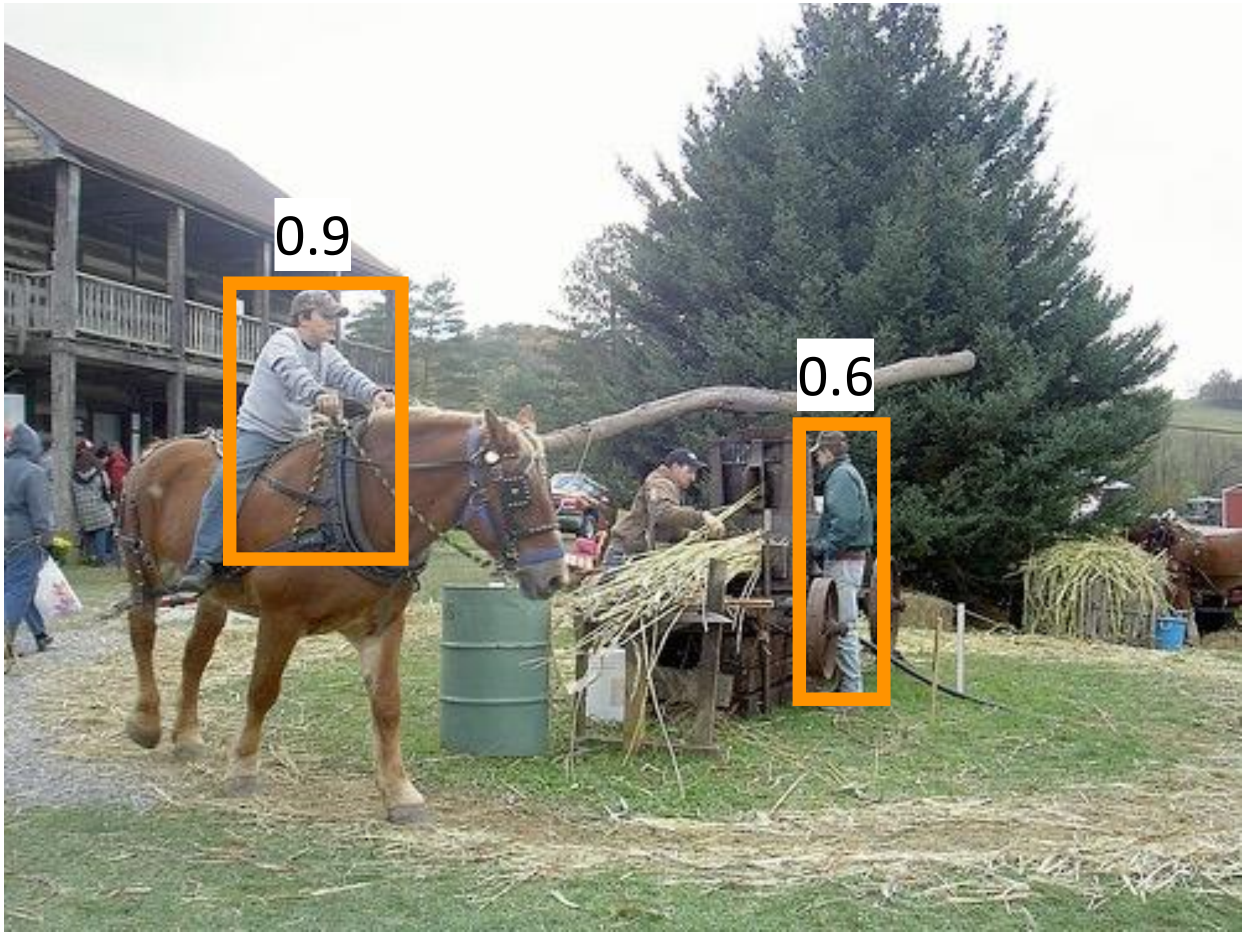


Test image (previously unseen)

First detection ...

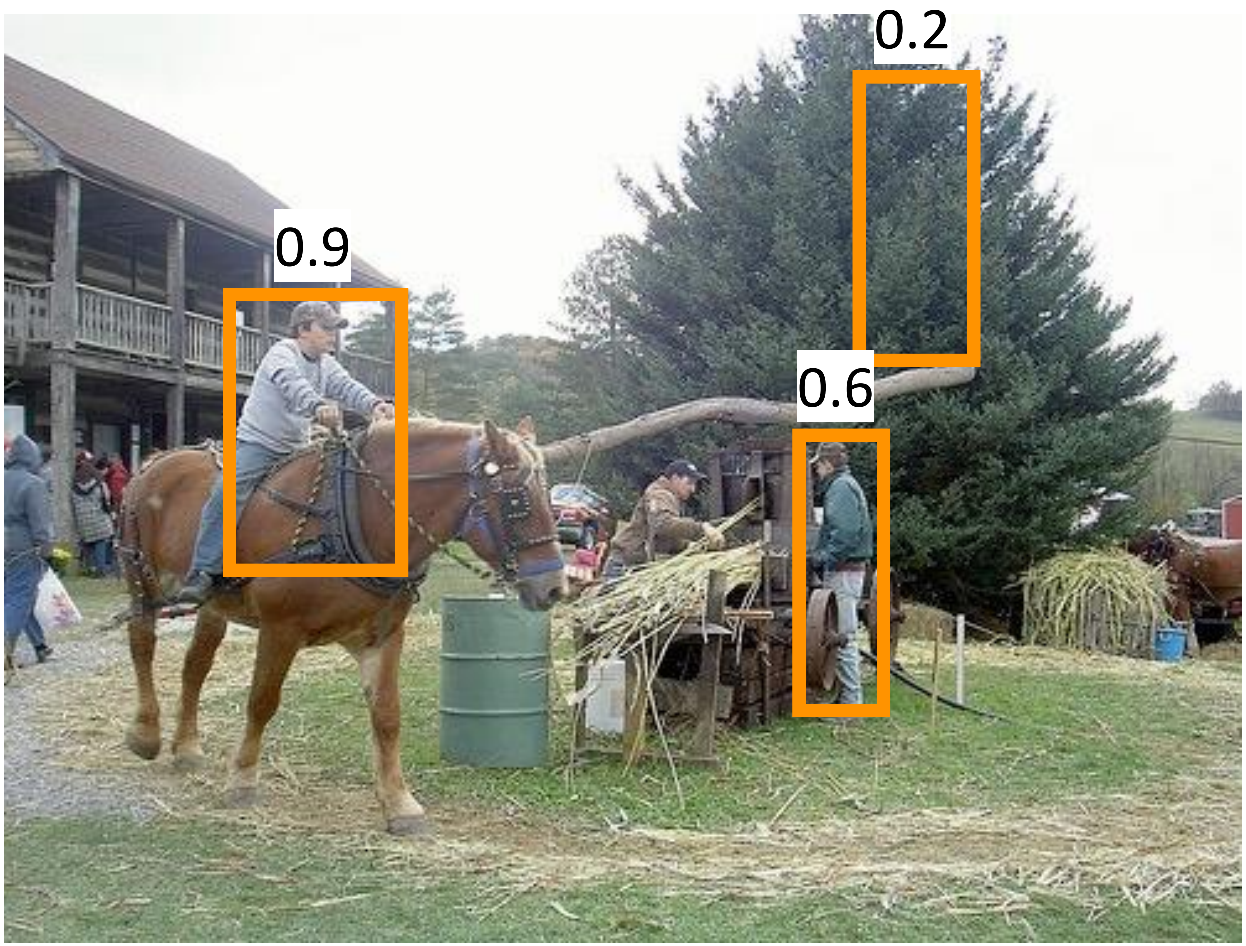
'person' detector predictions

Second detection ...



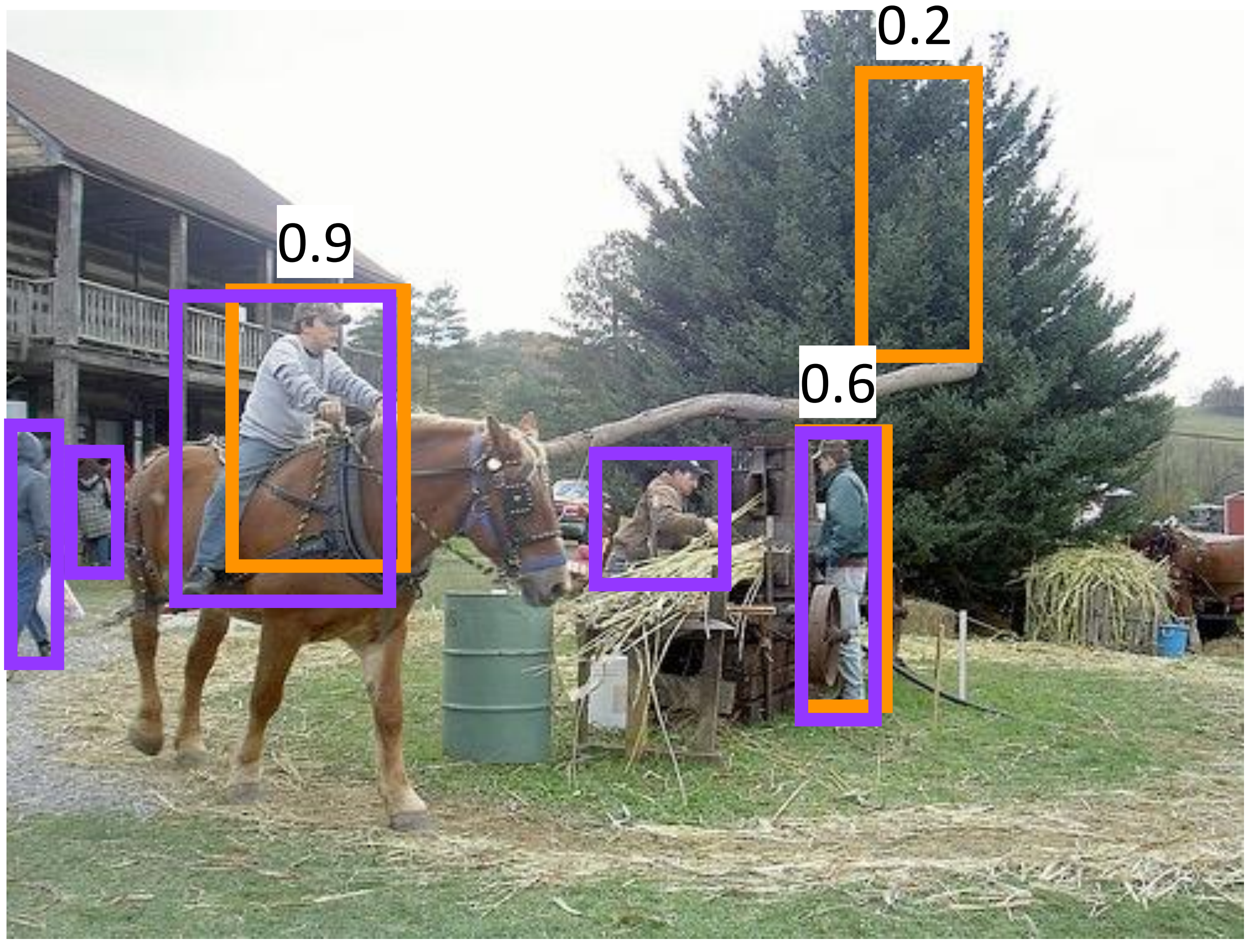
'person' detector predictions

Third detection ...



'person' detector predictions

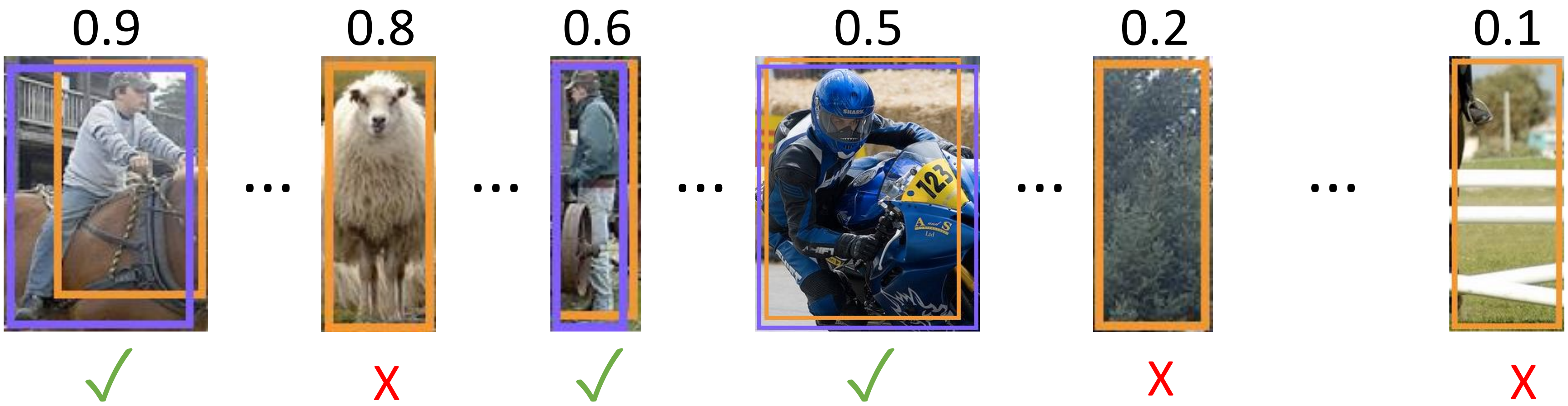
Compare to ground truth



'person' detector predictions

ground truth 'person' boxes

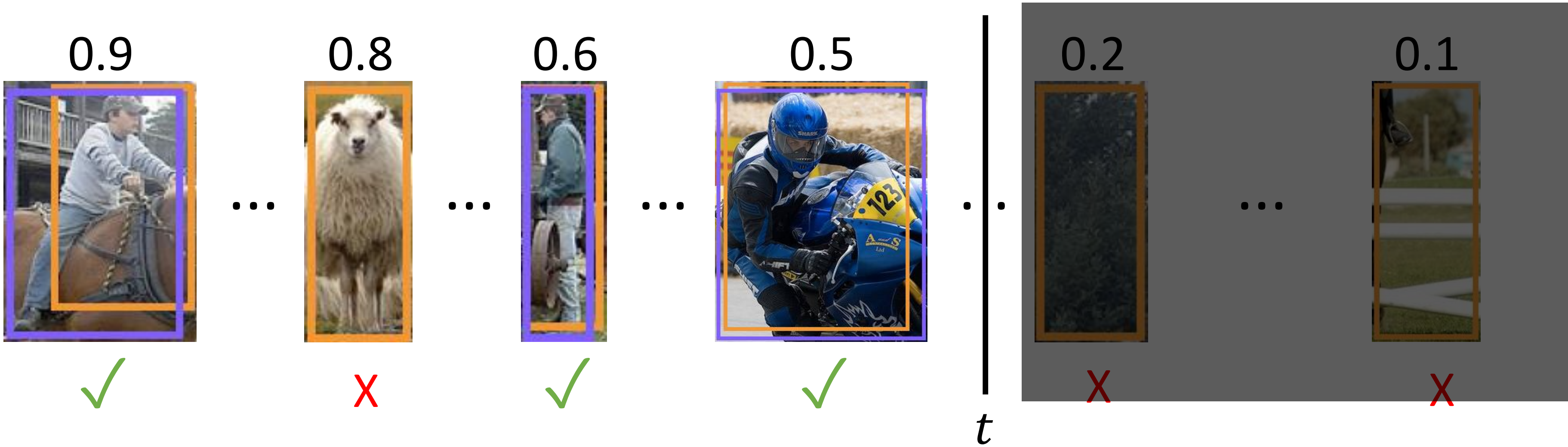
Sort by confidence



true positive (high overlap)

false positive (no overlap, low overlap, or duplicate)

Evaluation metric

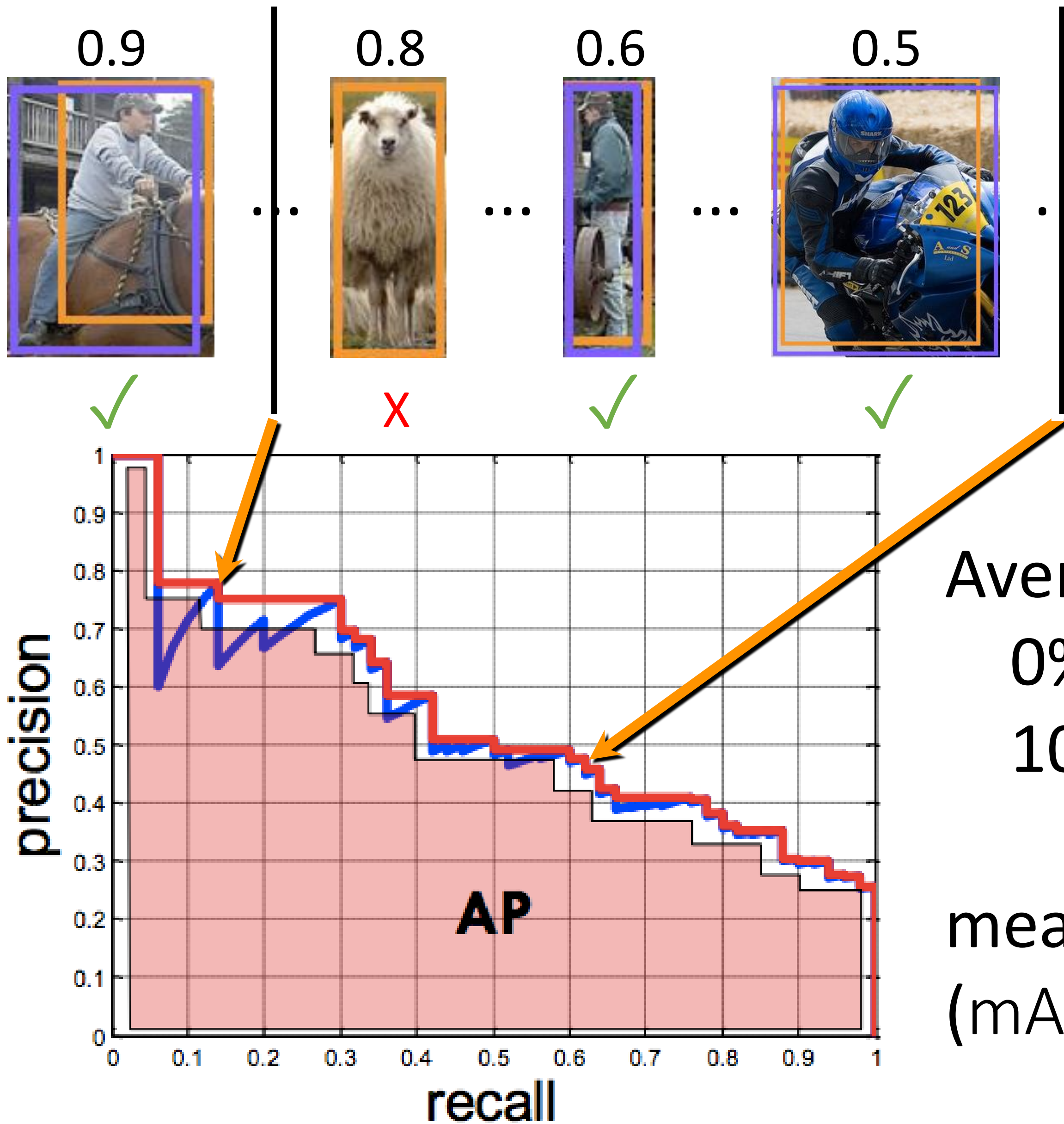


$$precision@t = \frac{\#true\ positives@t}{\#true\ positives@t + \#false\ positives@t}$$

$$\frac{✓}{✓ + X}$$

$$recall@t = \frac{\#true\ positives@t}{\#ground\ truth\ objects}$$

Evaluation metric



Average Precision (AP)

0% is worst

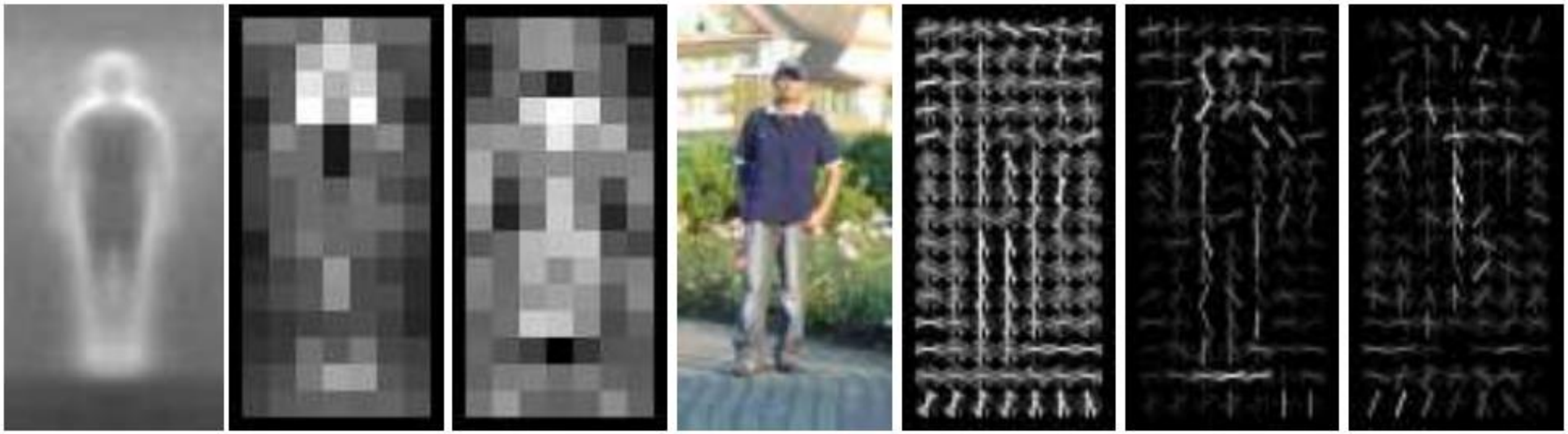
100% is best

mean AP over classes (mAP)

Pedestrians

Histograms of Oriented Gradients for Human Detection, Dalal and Triggs, CVPR 2005

AP ~77%
More sophisticated methods: AP ~90%



(a) (b) (c) (d) (e) (f) (g)

(a) average gradient image over training examples
(b) each “pixel” shows max positive SVM weight in the block centered on that pixel
(c) same as (b) for negative SVM weights
(d) test image
(e) its R-HOG descriptor
(f) R-HOG descriptor weighted by positive SVM weights
(g) R-HOG descriptor weighted by negative SVM weights

Overview of HOG Method

1. Compute gradients in the region to be described
2. Put them in bins according to orientation
3. Group the cells into large blocks
4. Normalize each block
5. Train classifiers to decide if these are parts of a human

Details

- Gradients

  [-1 0 1] and $[-1\ 0\ 1]^T$ were good enough filters.

- Cell Histograms

  Each pixel within the cell casts a weighted vote for an orientation-based histogram channel based on the values found in the gradient computation. (9 channels worked)

- Blocks

  Group the cells together into larger blocks, either R-HOG blocks (rectangular) or C-HOG blocks (circular).

More Details

- Block Normalization

  They tried 4 different kinds of normalization.

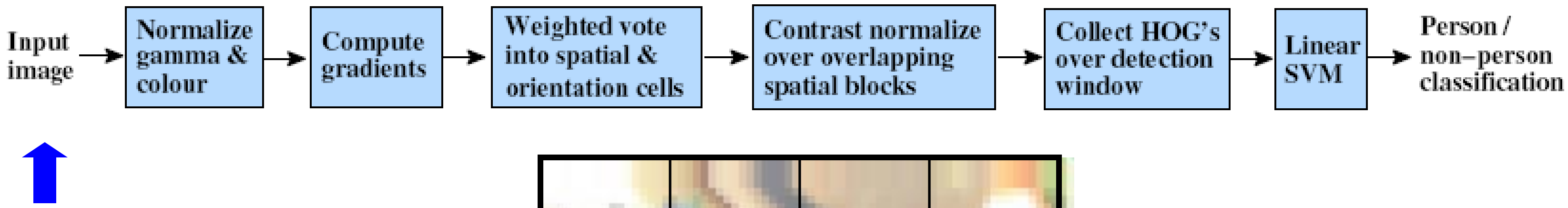
  - L1-norm
  - sqrt of L1-norm
  - L2 norm
  - L2-norm followed by clipping

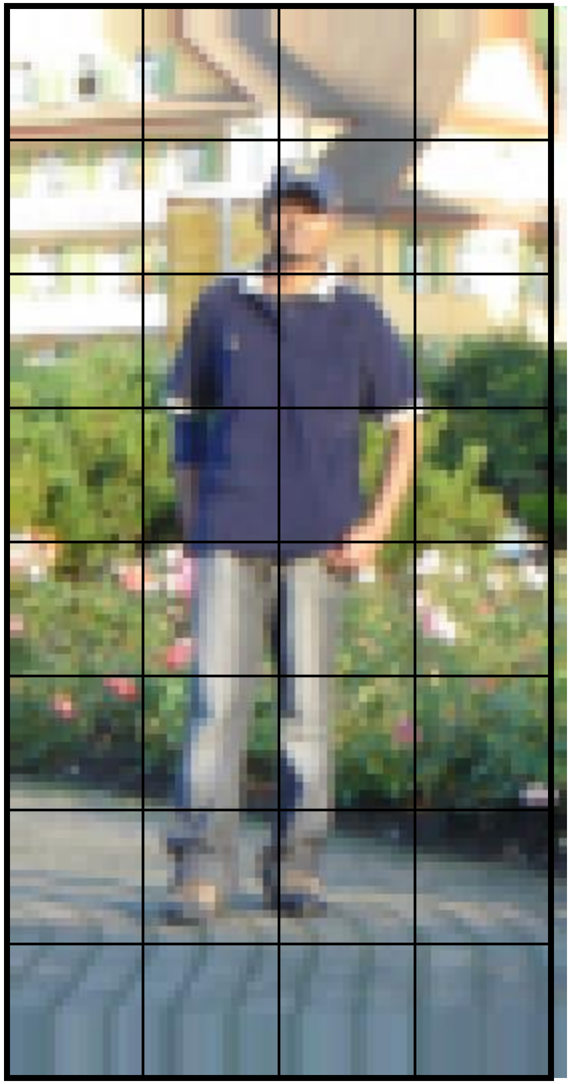
- If you think of the block as a vector **v**, then the normalized block is **v**/norm(**v**)

Example: Dalal-Triggs pedestrian

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

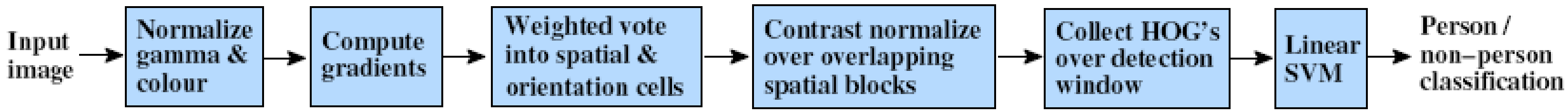
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



Outperforms

-1	0	1

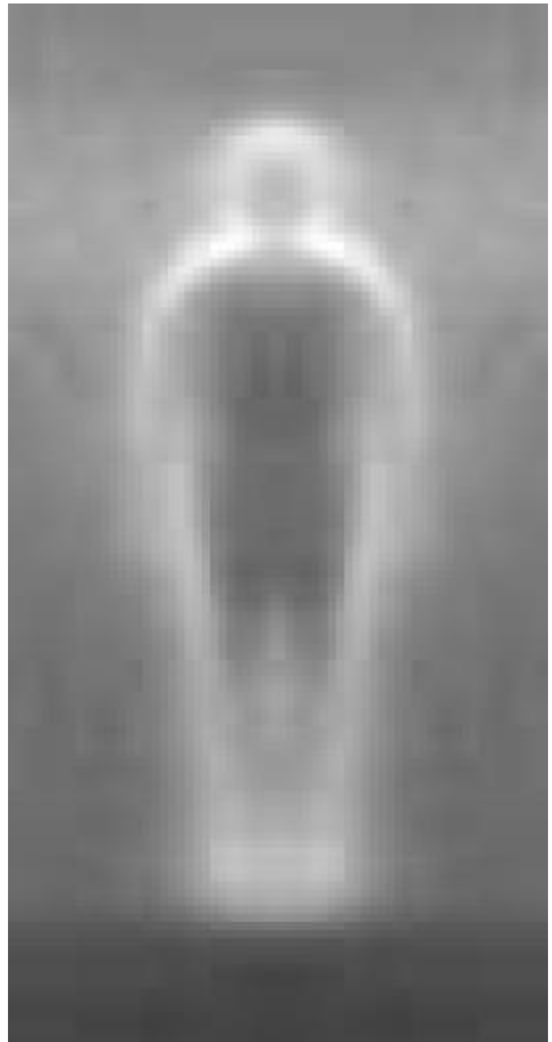
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



0	1
-1	0

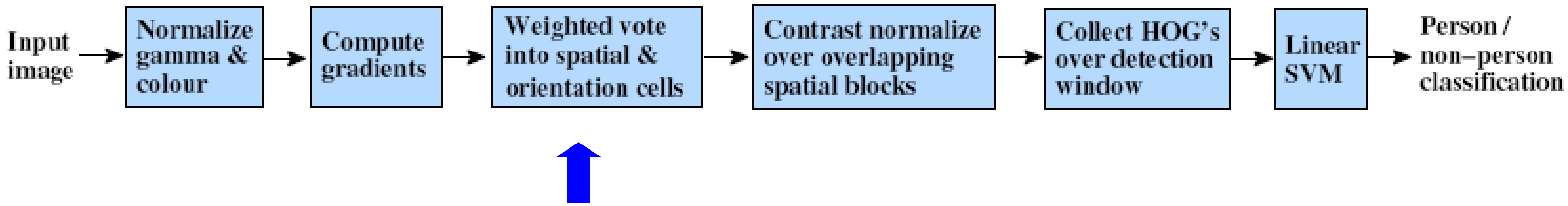
diagonal

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

Sobel

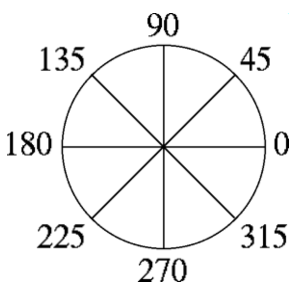
Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

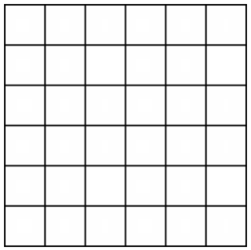


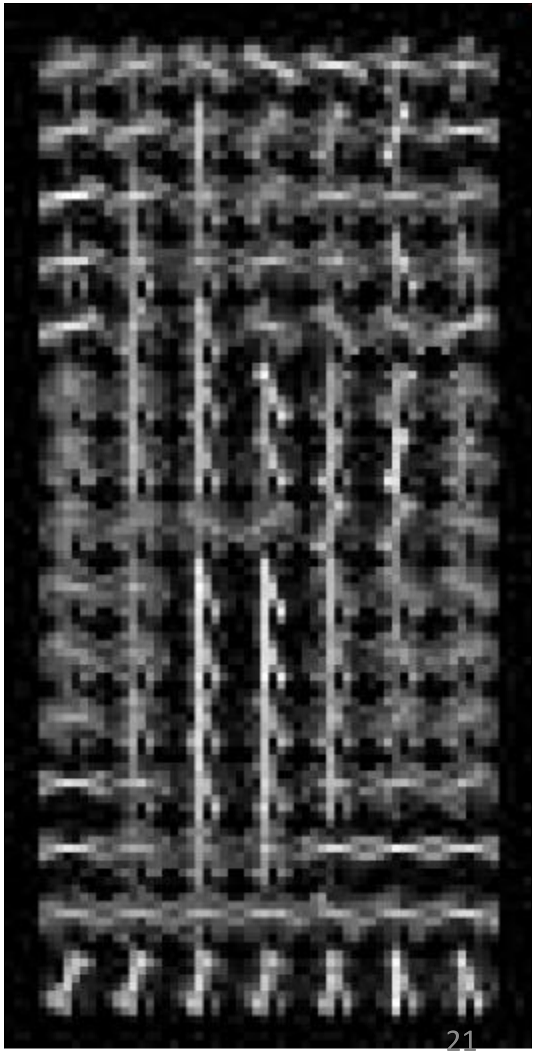
- Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles)



Histograms in 8x8 pixel cells

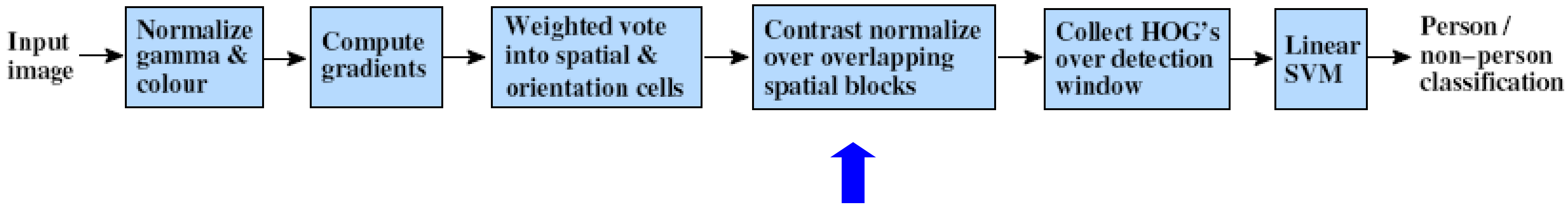




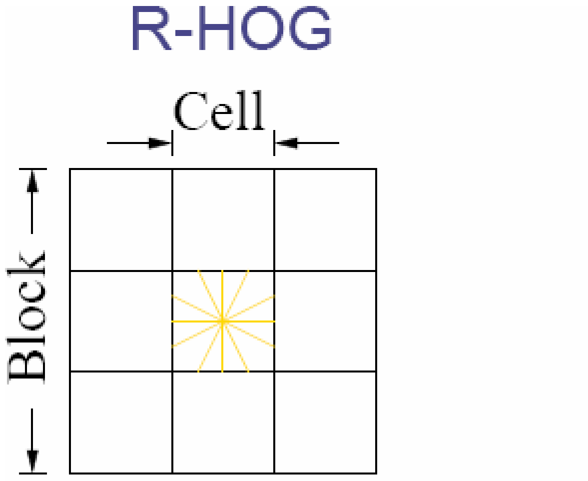
- Votes weighted by magnitude
- Bilinear interpolation between cells

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



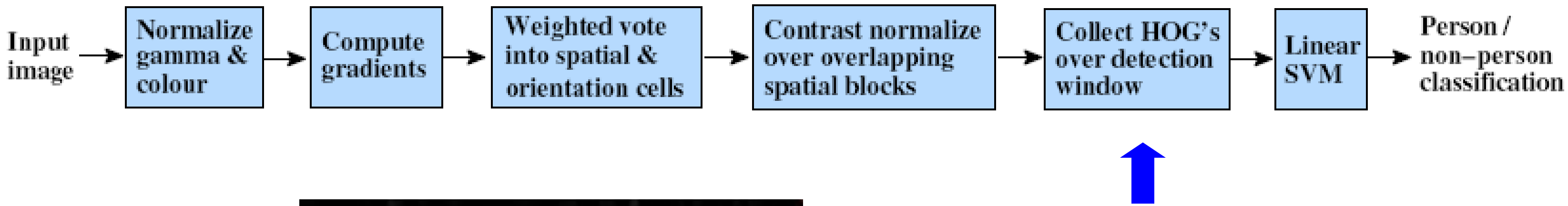
Normalize with respect to surrounding cells



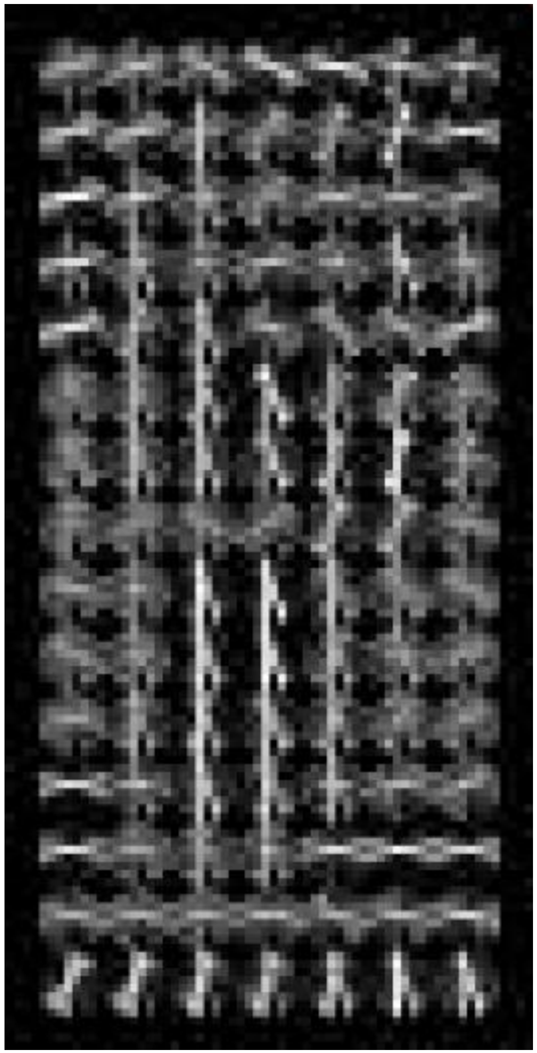
$$L2-norm : v \longrightarrow v/\sqrt{||v||_2^2+\epsilon^2}$$

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



X=



orientations

features = 15 x 7 x 9 x 4 = 3780

cells

normalizations by neighboring cells

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

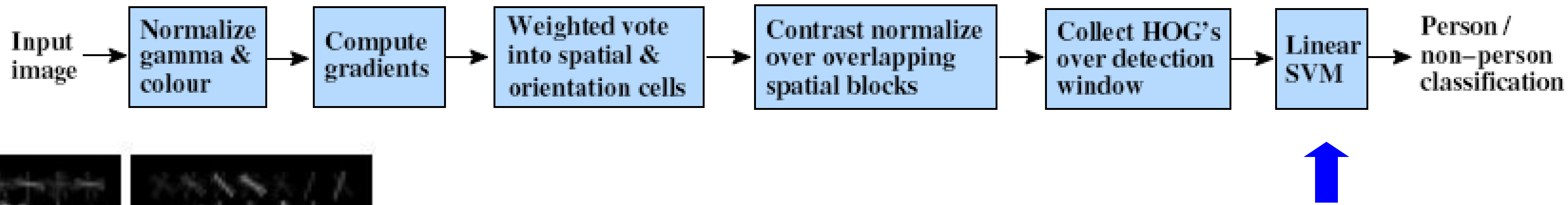
Training set

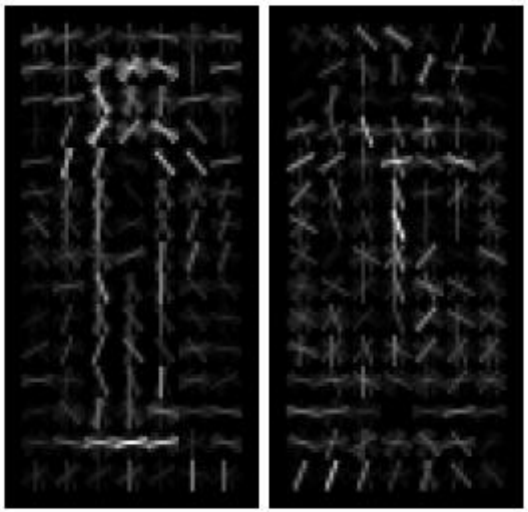
+

–

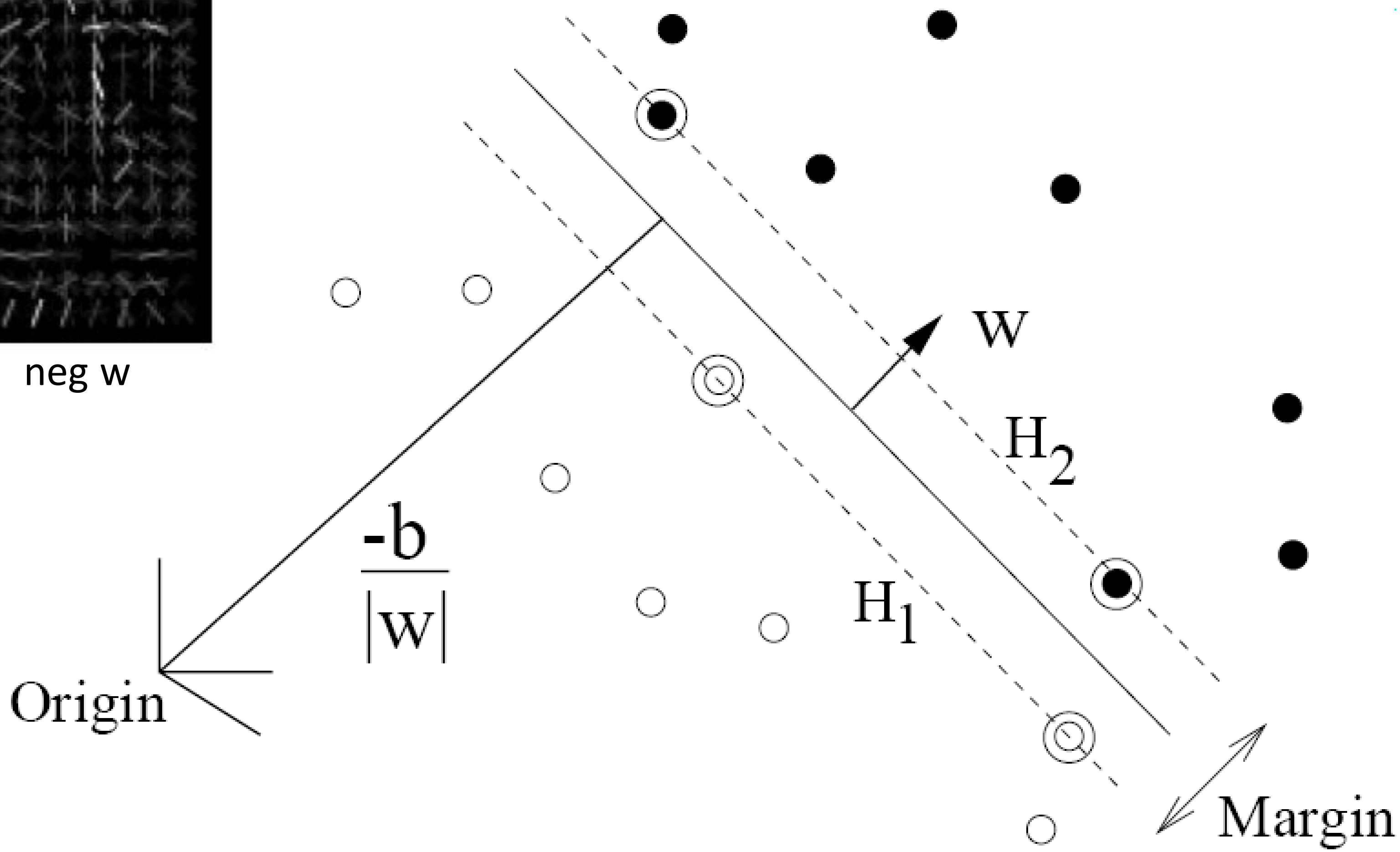




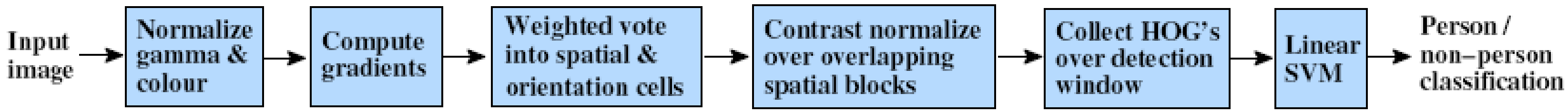


pos w   neg w



Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



$$0.16 = w^T x - b$$

$$sign(0.16) = 1$$

$=>$ pedestrian

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Detection examples

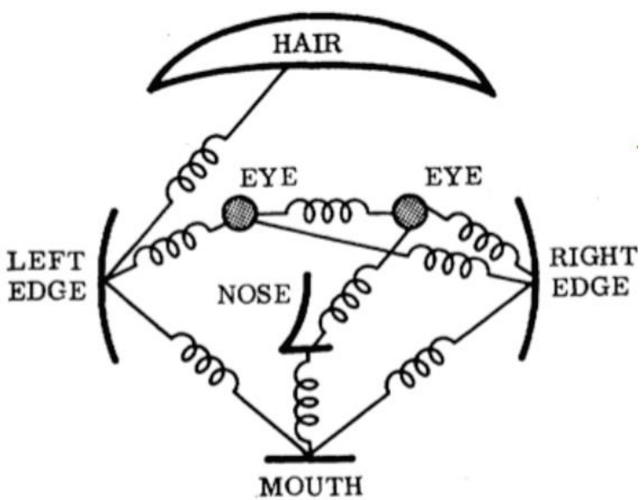
Deformable Parts Model

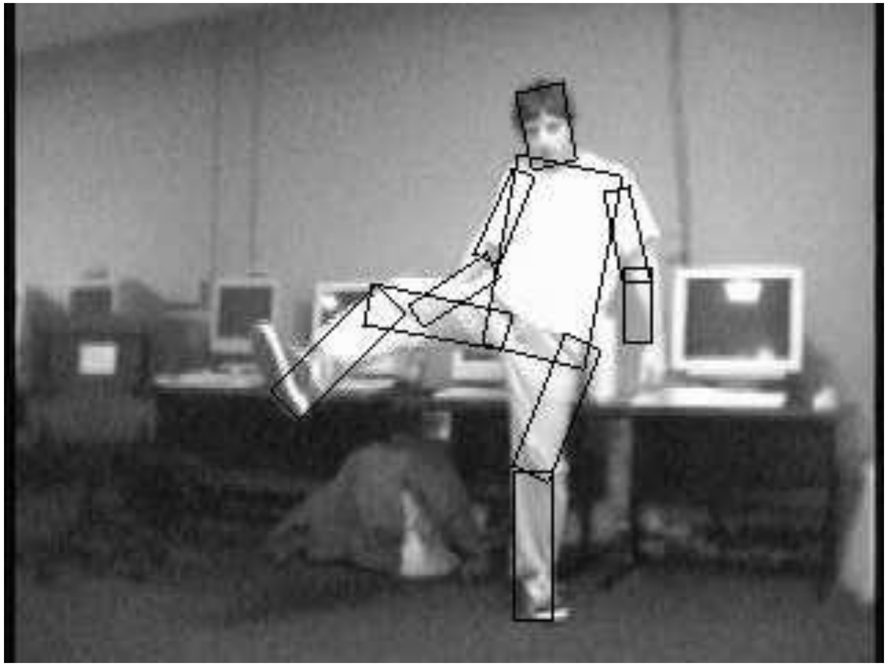
- Takes the idea a little further
- Instead of one rigid HOG model, we have multiple HOG models in a spatial arrangement
- One root part to find first and multiple other parts in a tree structure.

The Idea

Articulated parts model

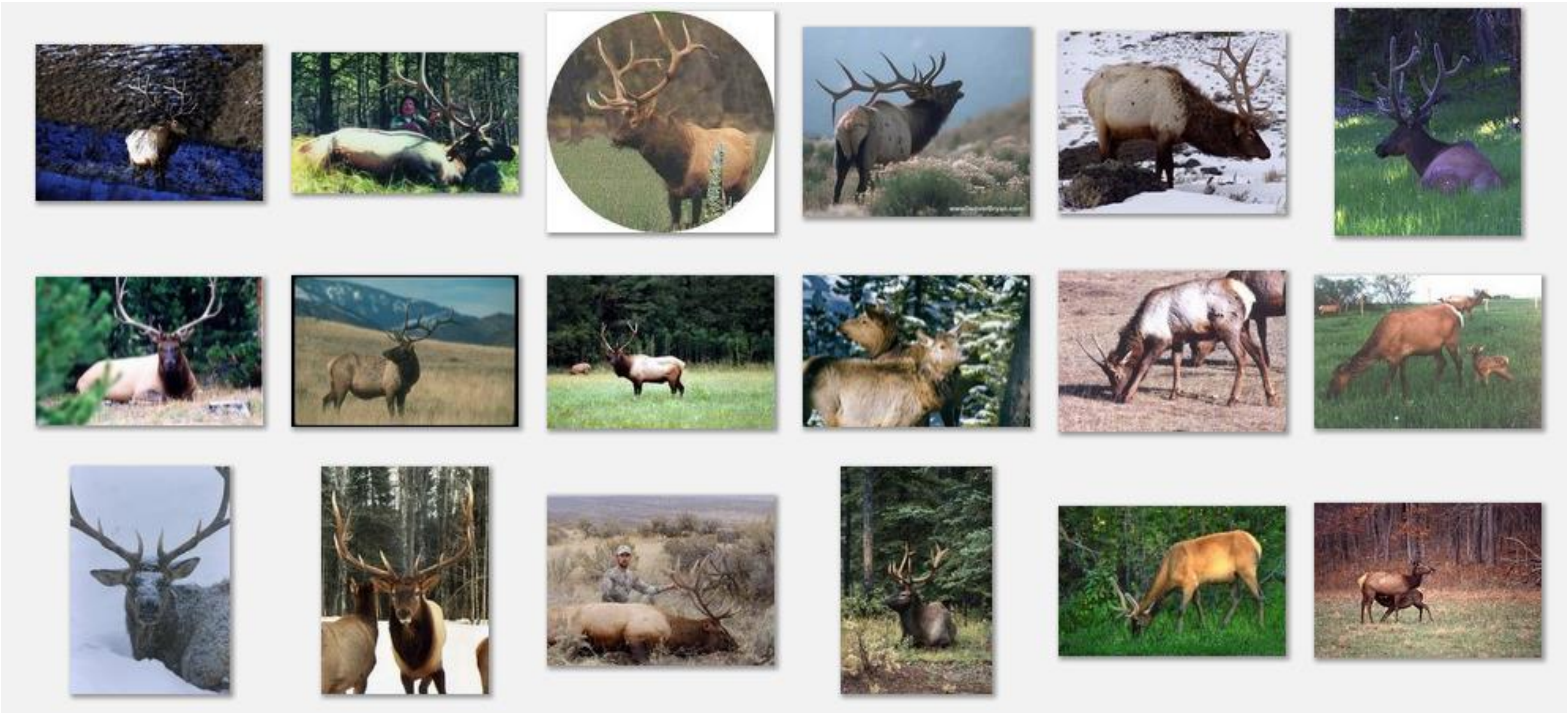
- Object is configuration of parts
- Each part is detectable





Images from Felzenszwalb

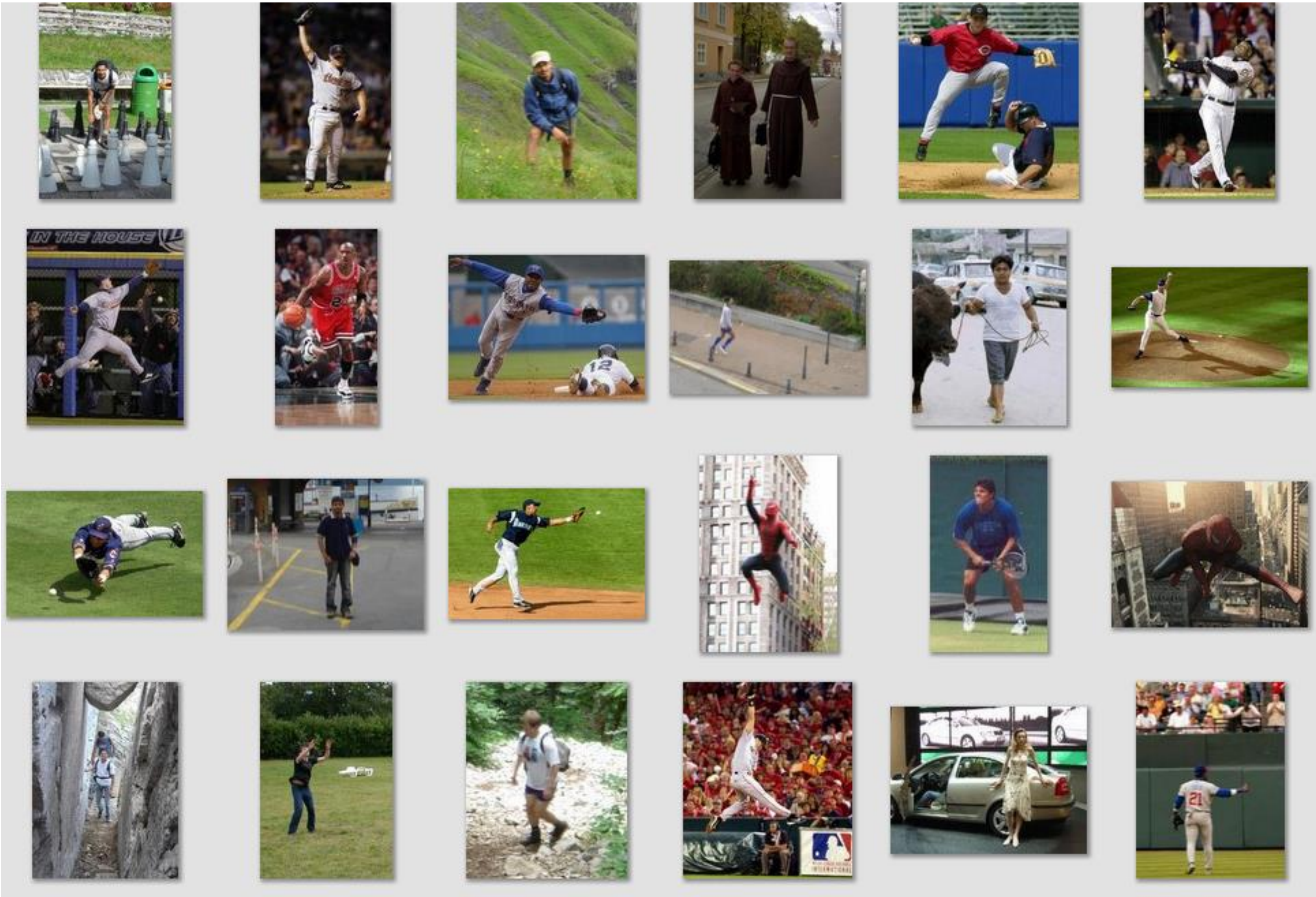
Deformable objects



Images from Caltech-256

Slide Credit: Duan Tran

Deformable objects

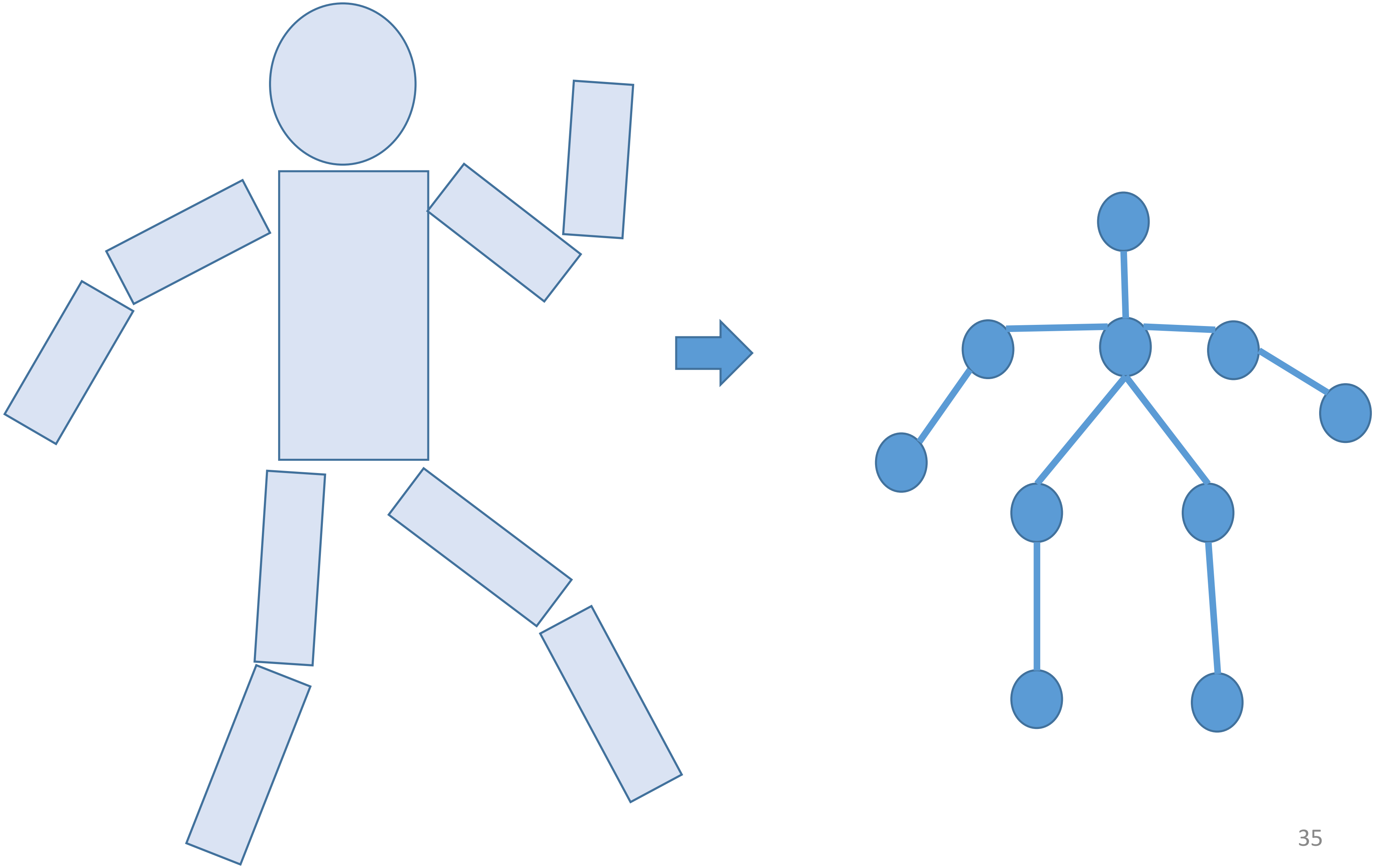


Images from D. Ramanan’s dataset

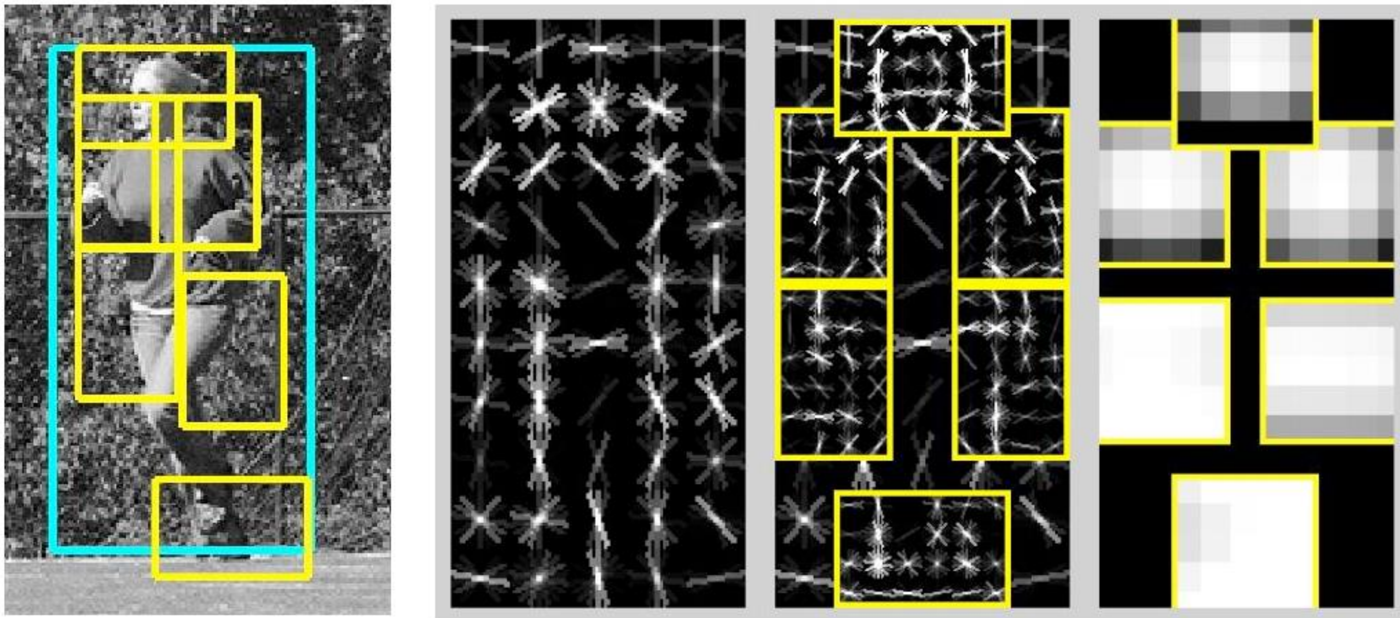
Slide Credit: Duan Tran

How to model spatial relations?

- Tree-shaped model



Model Overview

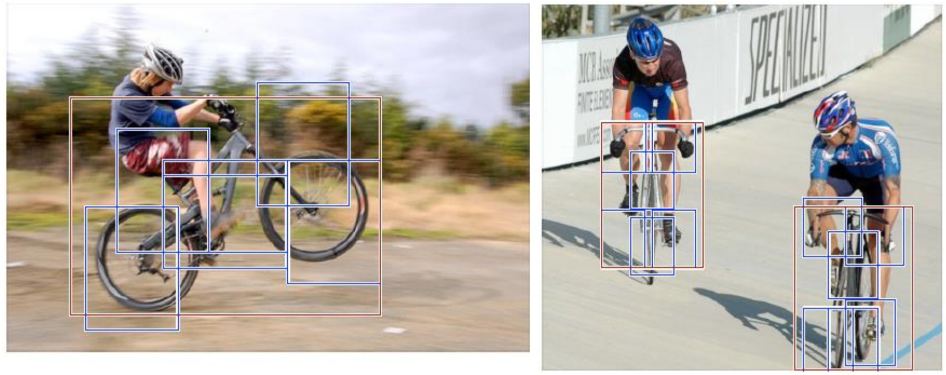


detection root filter part filters deformation models

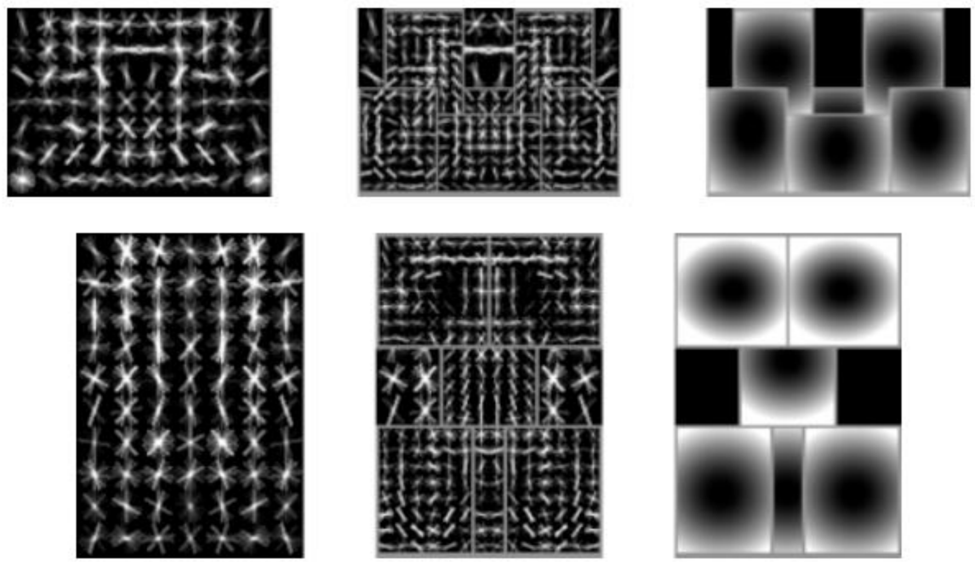
Model has a root filter plus deformable parts

Hybrid template/parts model

Detections



Template Visualization



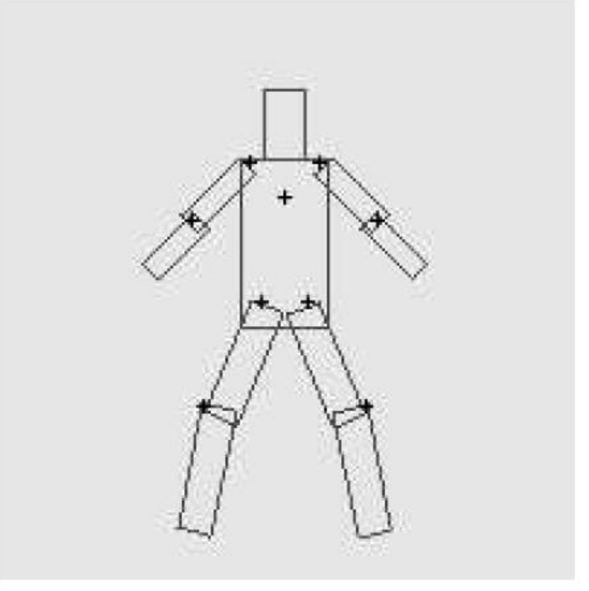
root filters coarse resolution

part filters finer resolution

deformation models

Felzenszwalb et al. 2008

Pictorial Structures Model

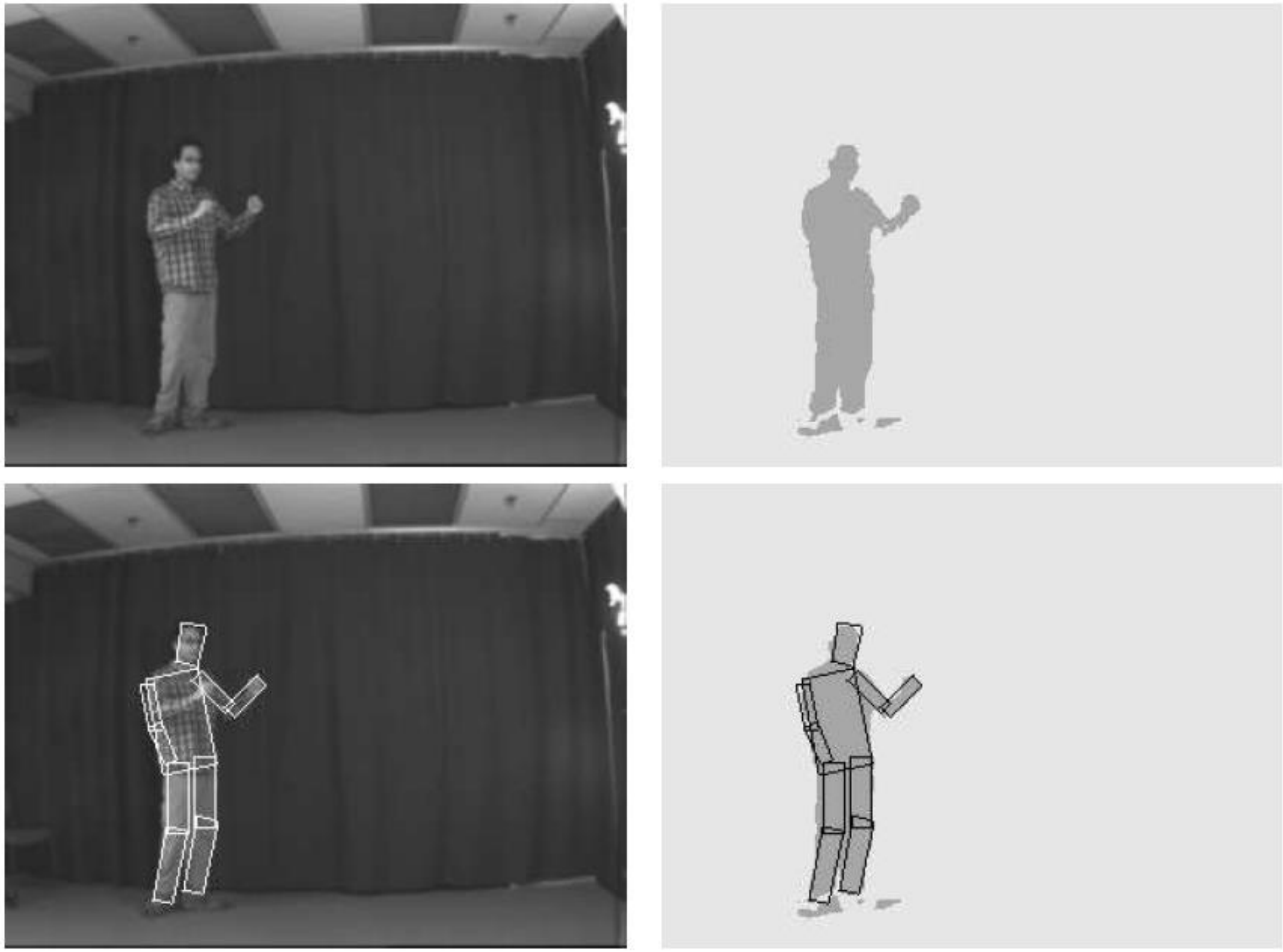


$$P(L|I,\theta) \propto \left(\prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i,v_j)\in E} p(l_i, l_j|c_{ij}) \right)$$

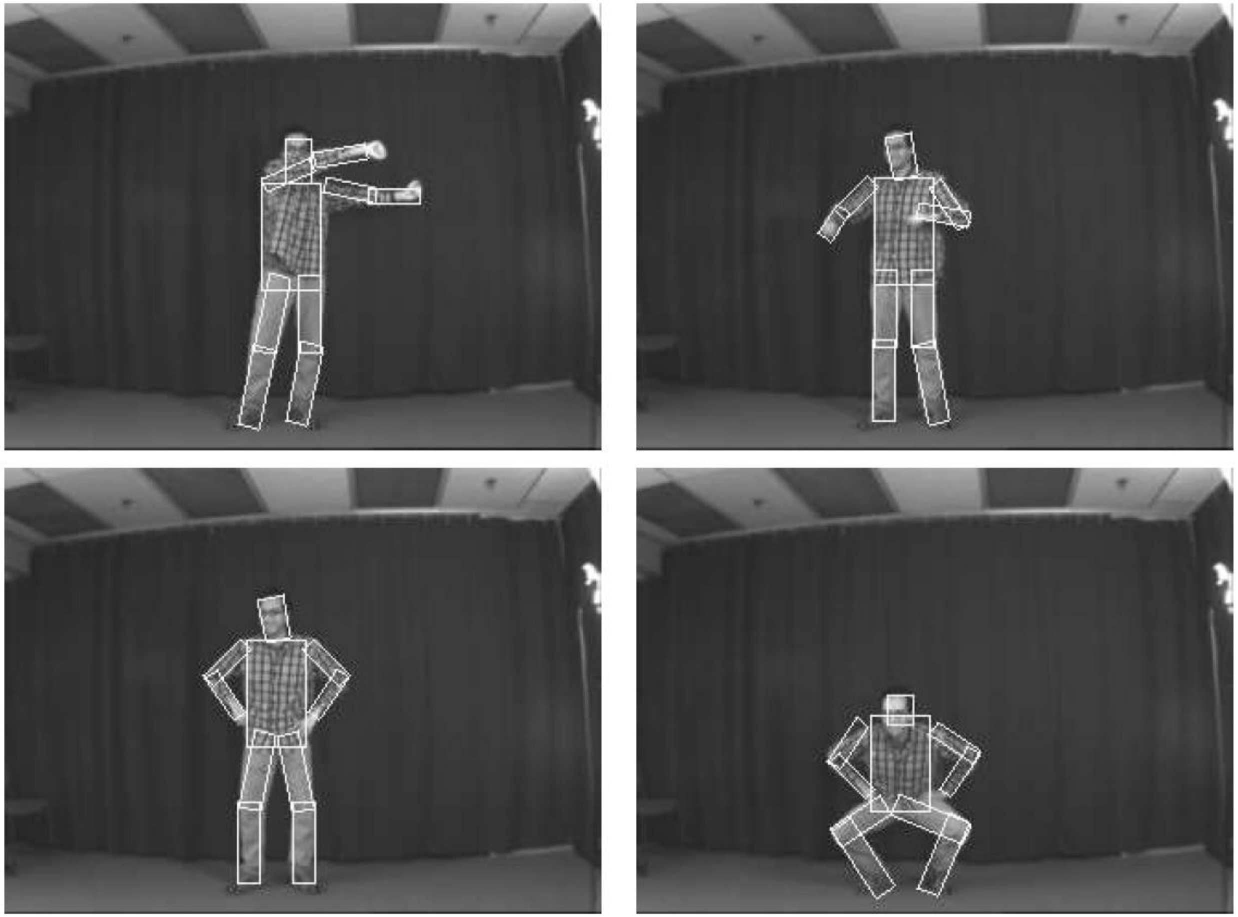
Appearance likelihood

Geometry likelihood

Results for person matching

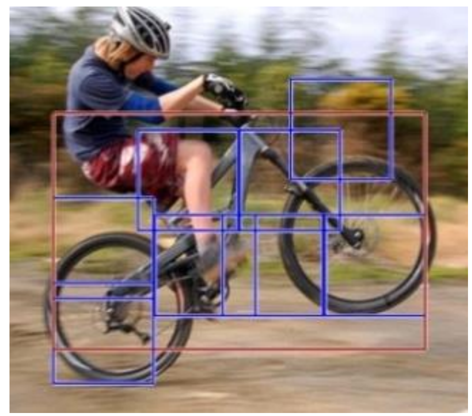


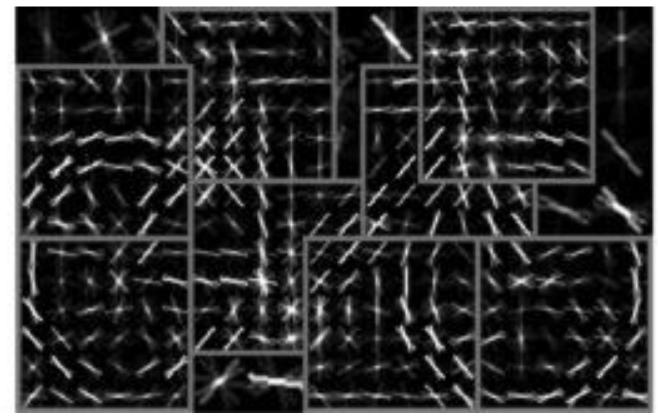
Results for person matching



BMVC 2009

2012 State-of-the-art Detector: Deformable Parts Model (DPM)





1. Strong low-level features based on HOG
2. Efficient matching algorithms for deformable part-based models (pictorial structures)
3. Discriminative learning with latent variables (latent SVM)

Felzenszwalb et al., 2008, 2010, 2011, 2012

Why did gradient-based models work?

Average gradient image

Generic categories

Can we detect people, chairs, horses, cars, dogs, buses, bottles, sheep …?

PASCAL Visual Object Categories (VOC) dataset

Generic categories

Why doesn't this work (as well)?

Can we detect people, chairs, horses, cars, dogs, buses, bottles, sheep …?

PASCAL Visual Object Categories (VOC) dataset

Quiz time
(Back to Girshick)

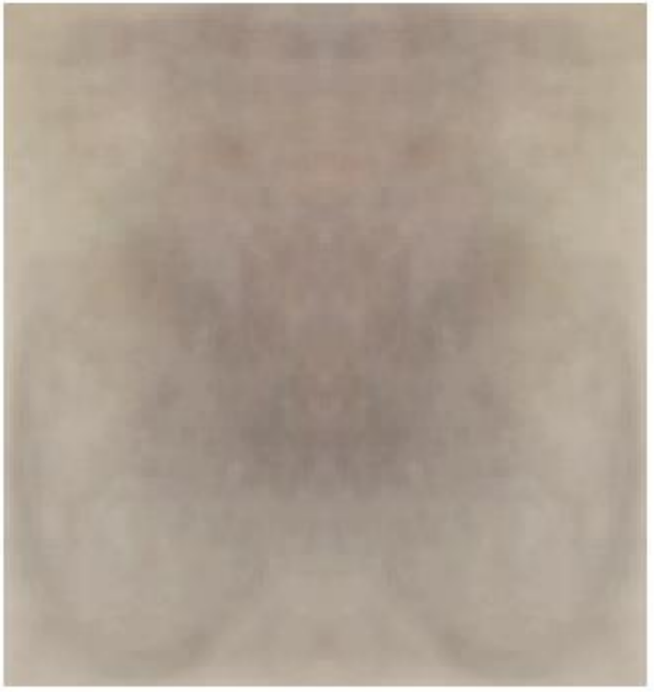
Warm up

This is an average image of which object class?

Warm up

pedestrian

A little harder



?

A little harder

?

Hint: airplane, bicycle, bus, car, cat, chair, cow, dog, dining table

A little harder

bicycle (PASCAL)

A little harder, yet

?

A little harder, yet

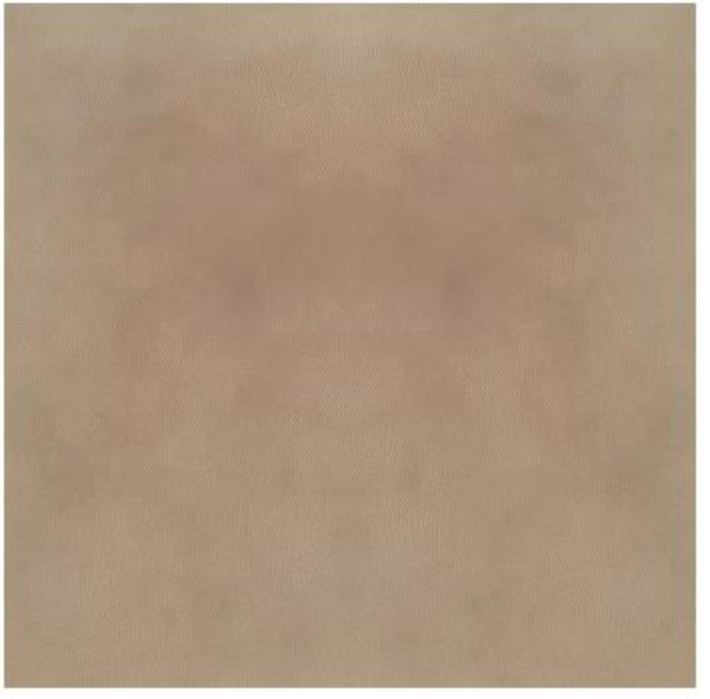
?

Hint: white blob on a green background

A little harder, yet

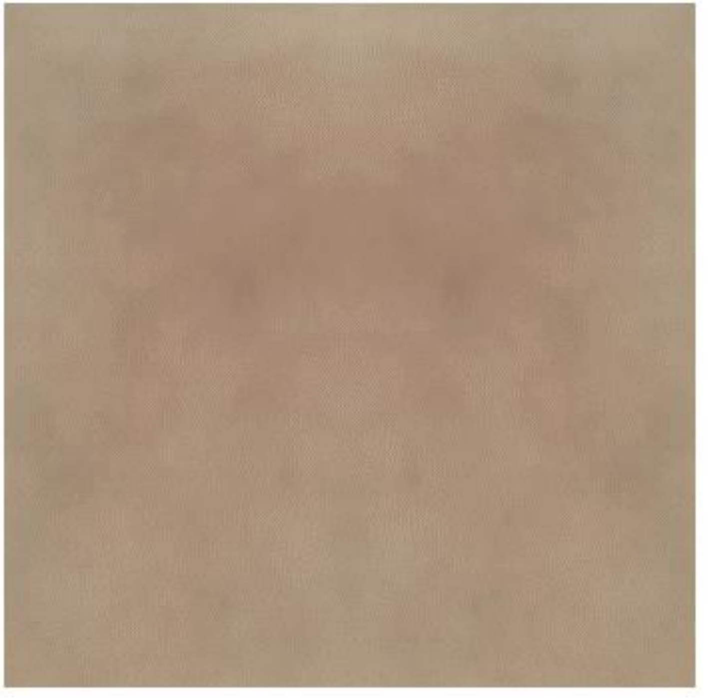
sheep (PASCAL)

Impossible?



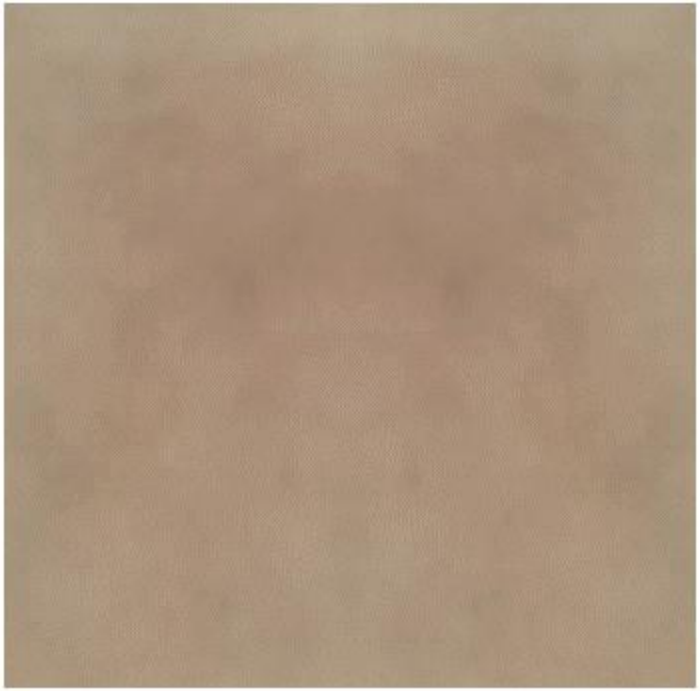
?

Impossible?



dog (PASCAL)

Impossible?



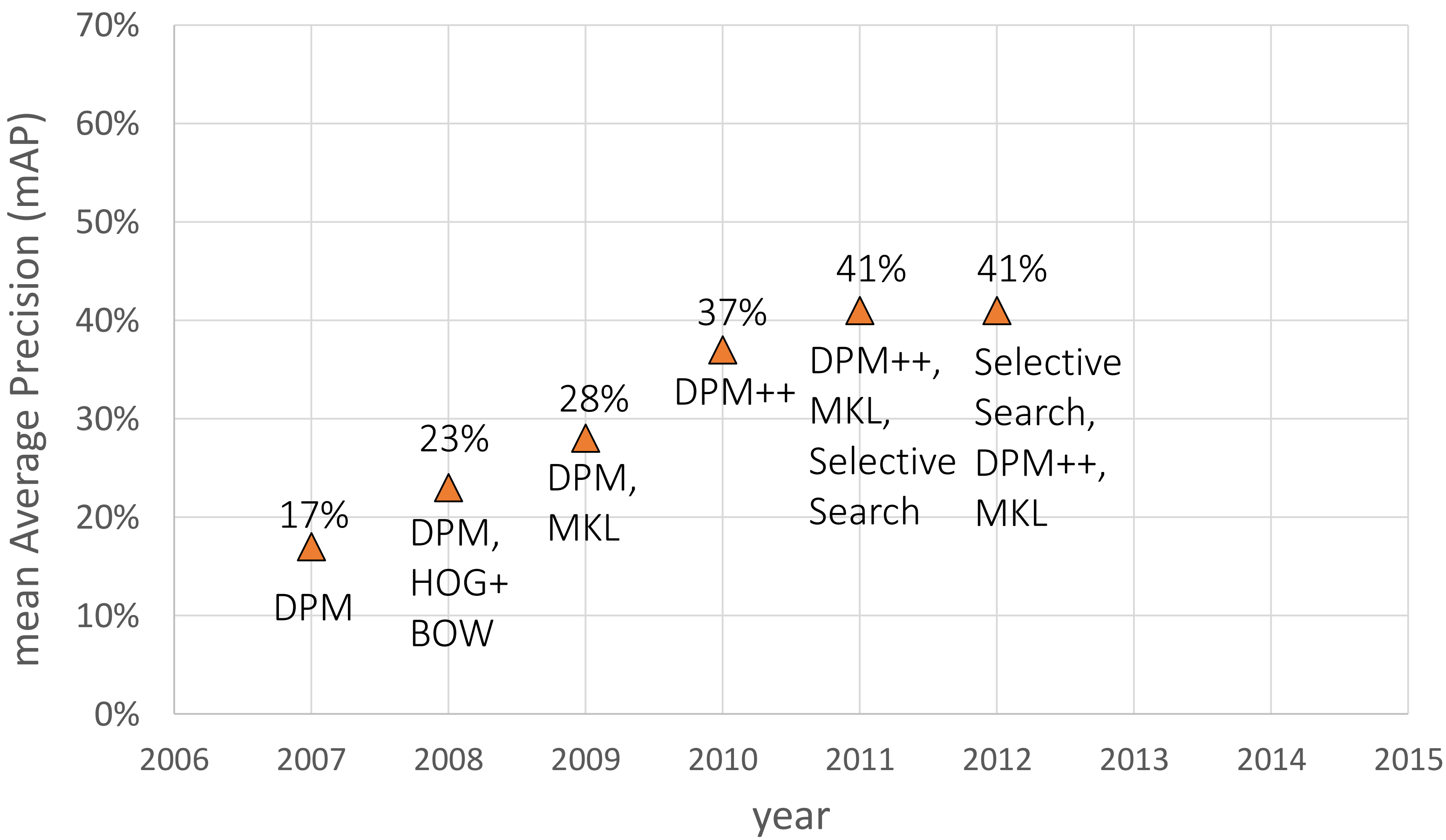
dog (PASCAL)

Why does the mean look like this?

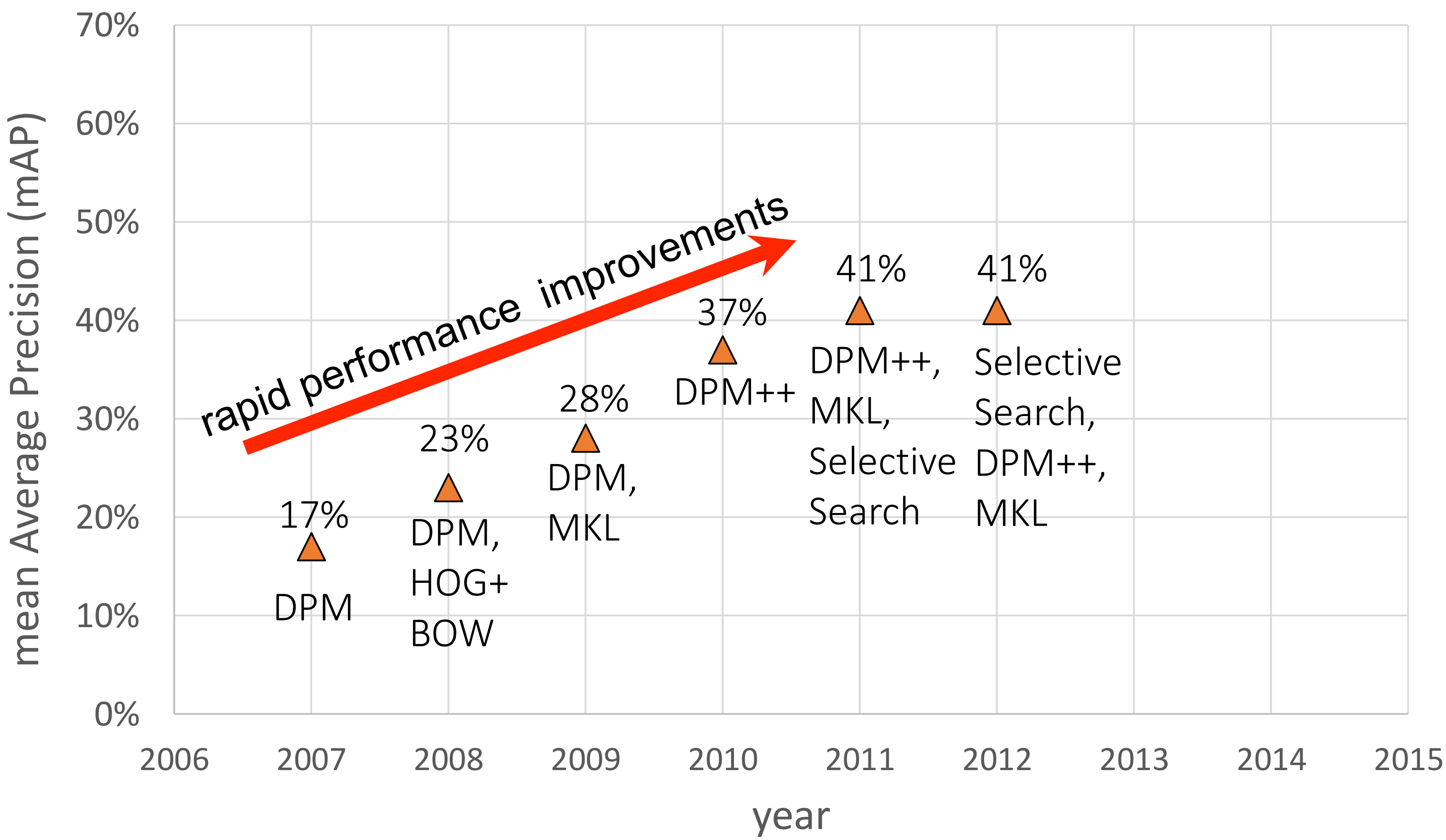
There's no alignment between the examples!

How do we combat this?

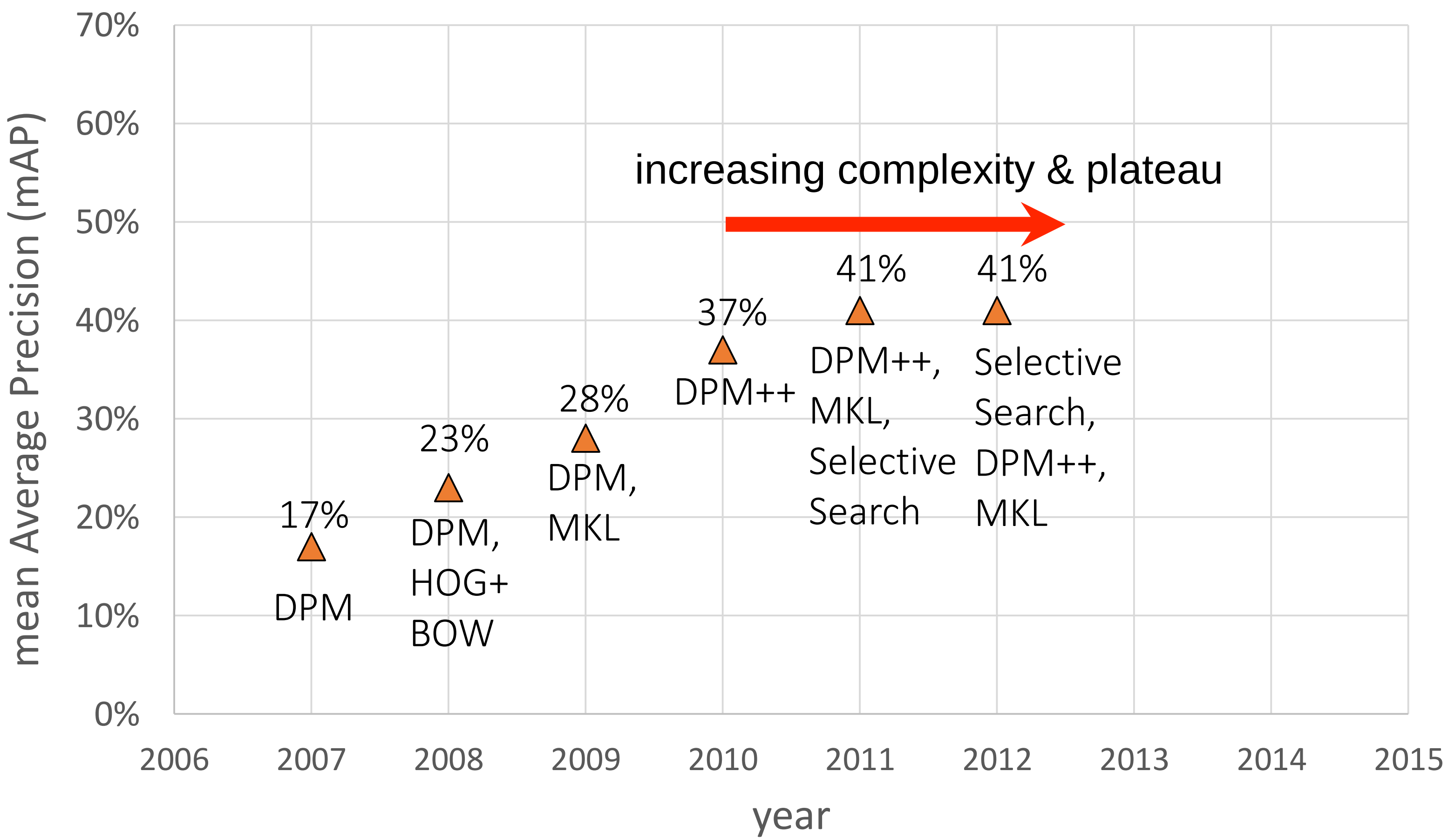
PASCAL VOC detection history



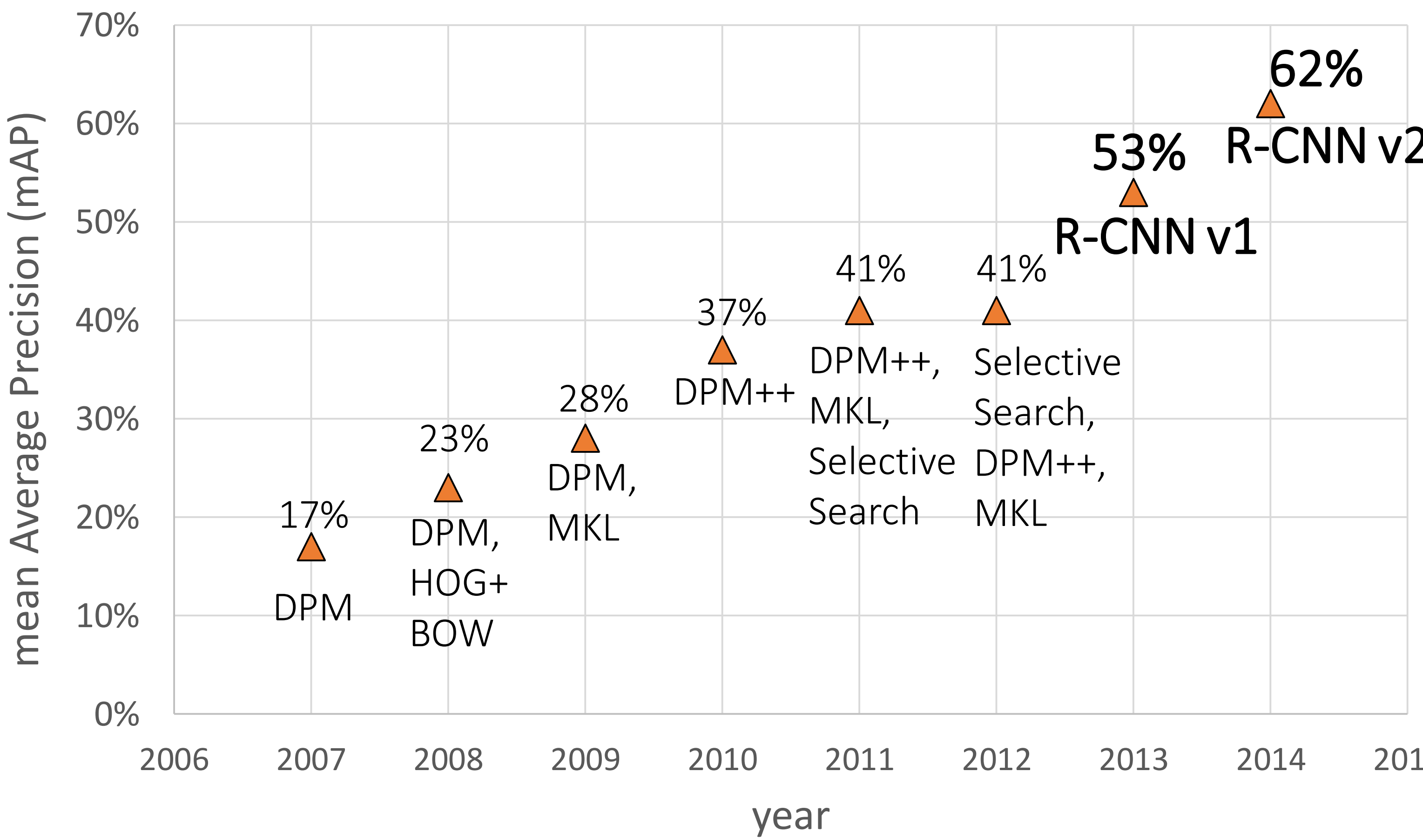
Part-based models & multiple features (MKL)



Kitchen-sink approaches

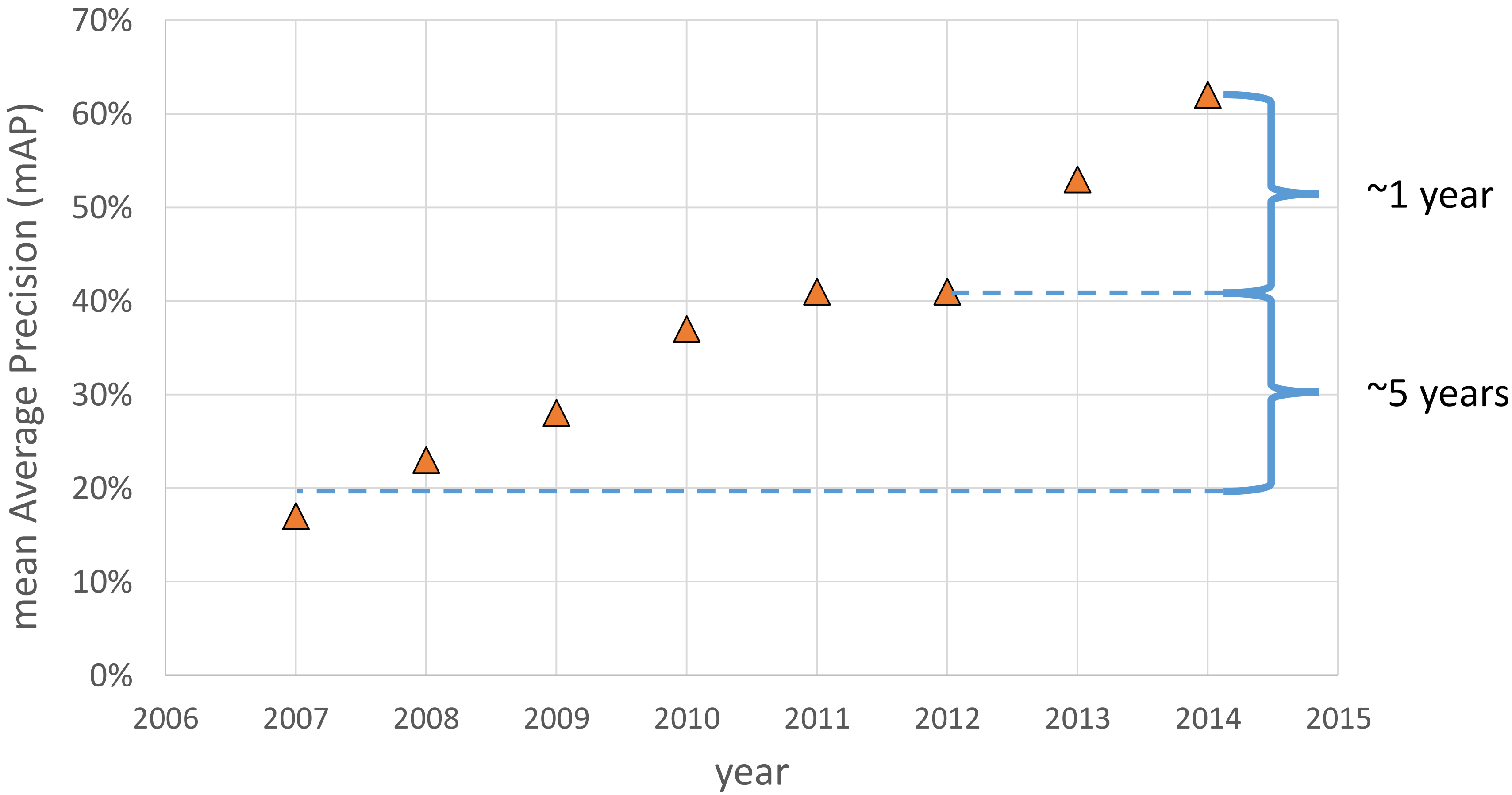


Region-based Convolutional Networks (R-CNNs)



[R-CNN. Girshick et al. CVPR 2014]

Region-based Convolutional Networks (R-CNNs)

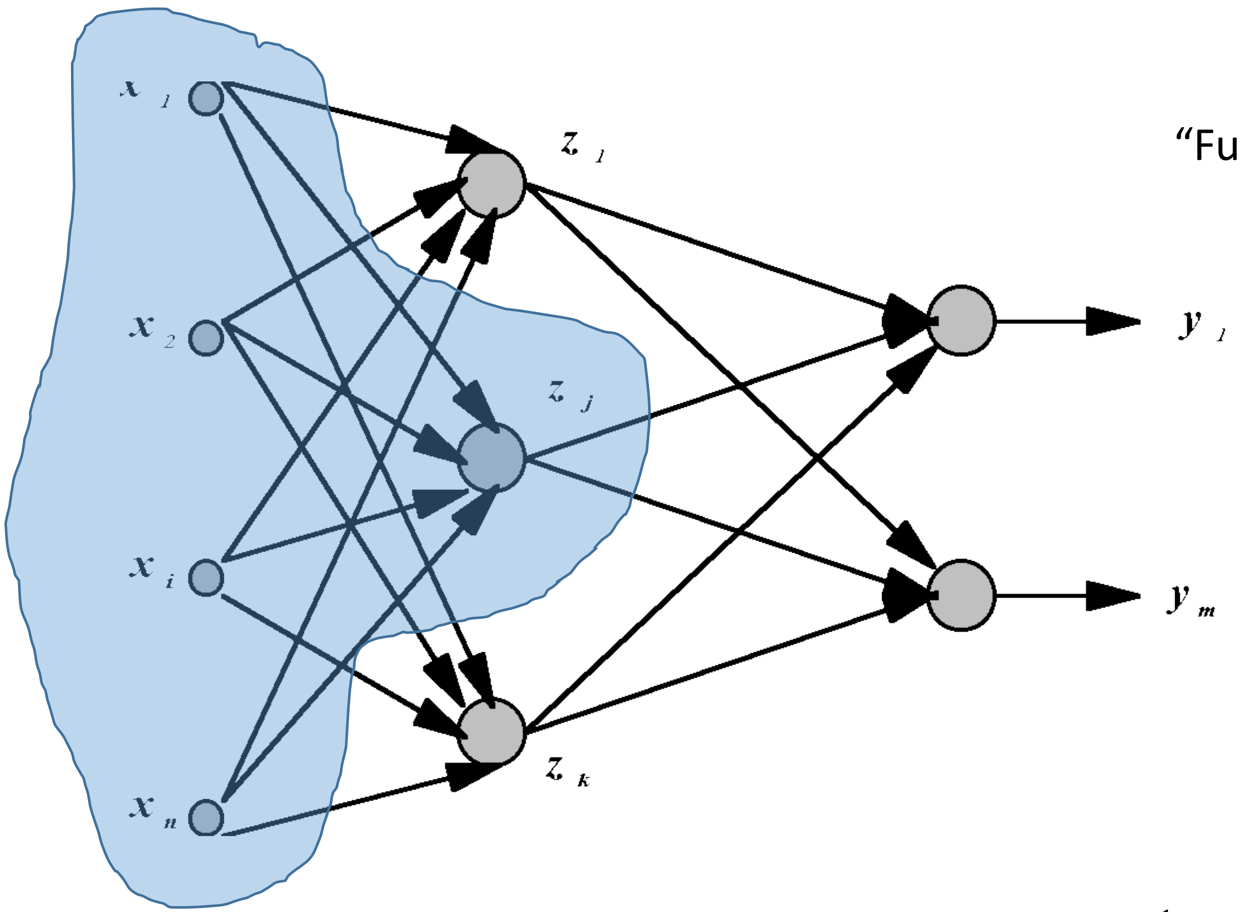


[R-CNN. Girshick et al. CVPR 2014]

Convolutional Neural Networks

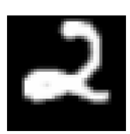
- Overview

Standard Neural Networks



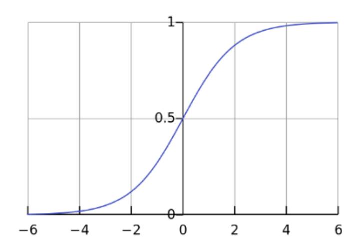
"Fully connected"

$\boldsymbol{x} = (x_1, \ldots, x_{784})^T$



$z_j = g(\boldsymbol{w}_j^T \boldsymbol{x})$

$g(t) = \dfrac{1}{1 + e^{-t}}$

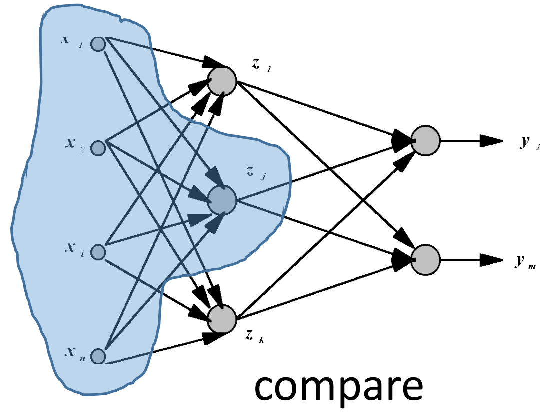


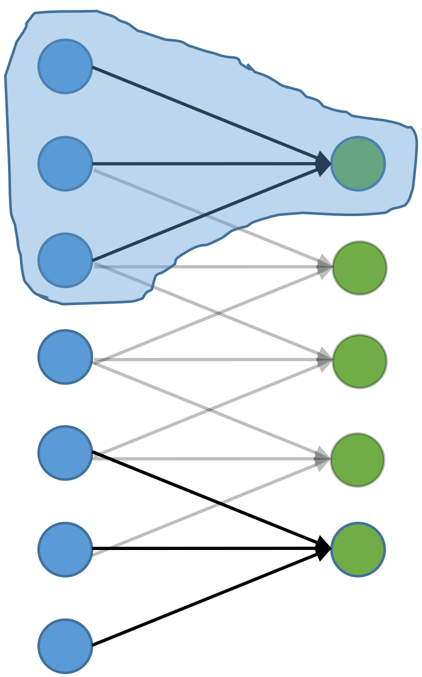
From NNs to Convolutional NNs

- Local connectivity
- Shared (“tied”) weights
- Multiple feature maps
- Pooling

Convolutional NNs

- Local connectivity

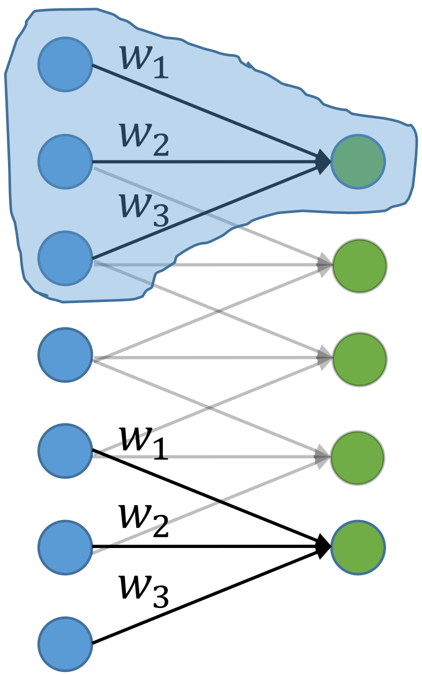




- Each green unit is only connected to (3) **neighboring** blue units

Convolutional NNs

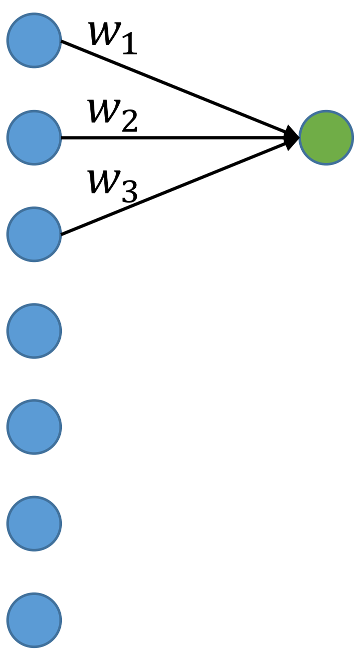
- Shared ("tied") weights



- All green units **share** the same parameters $\boldsymbol{w}$
- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

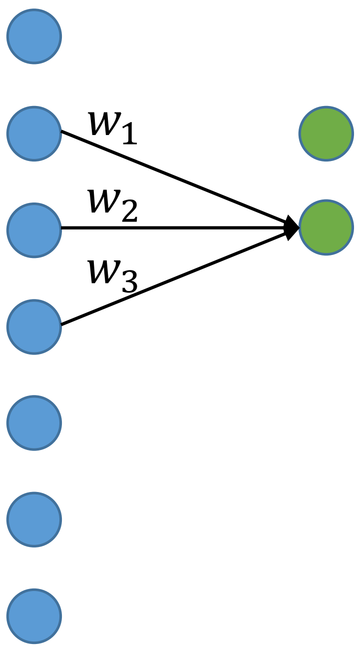
- Convolution with 1-D filter: $[w_3, w_2, w_1]$



- All green units **share** the same parameters $\boldsymbol{w}$
- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

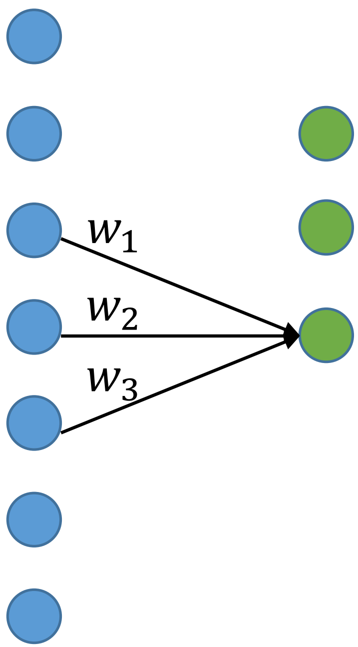
- Convolution with 1-D filter: $[w_3, w_2, w_1]$



- All green units **share** the same parameters $\boldsymbol{w}$
- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

- Convolution with 1-D filter: $[w_3, w_2, w_1]$

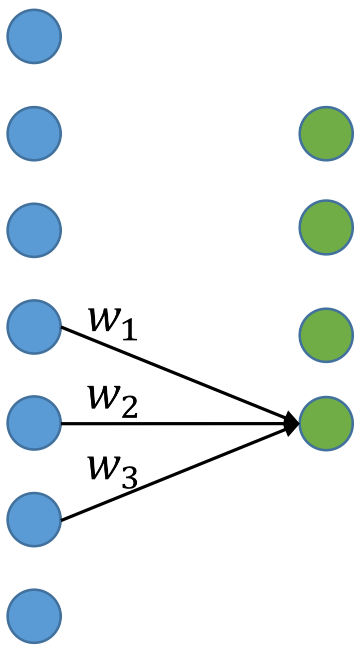


- All green units **share** the same parameters $\boldsymbol{w}$

- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

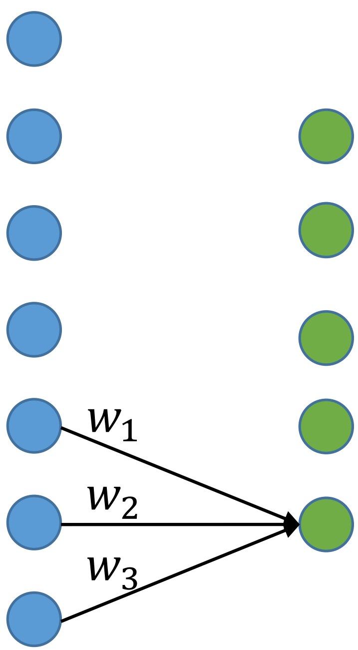
- Convolution with 1-D filter: $[w_3, w_2, w_1]$



- All green units **share** the same parameters $\boldsymbol{w}$
- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

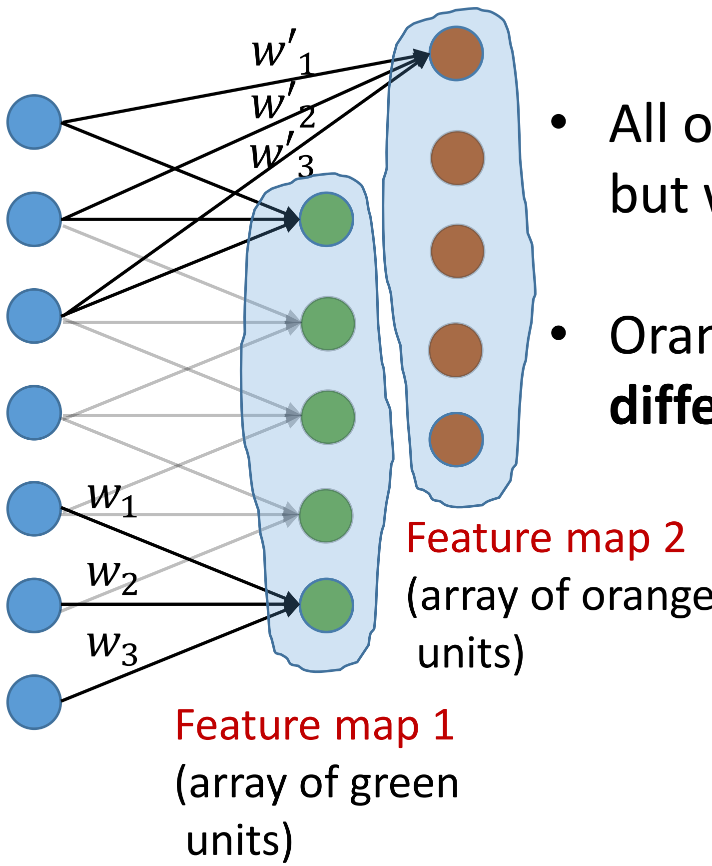
- Convolution with 1-D filter: $[w_3, w_2, w_1]$



- All green units **share** the same parameters $\boldsymbol{w}$
- Each green unit computes the **same function,** but with a **different input window**

Convolutional NNs

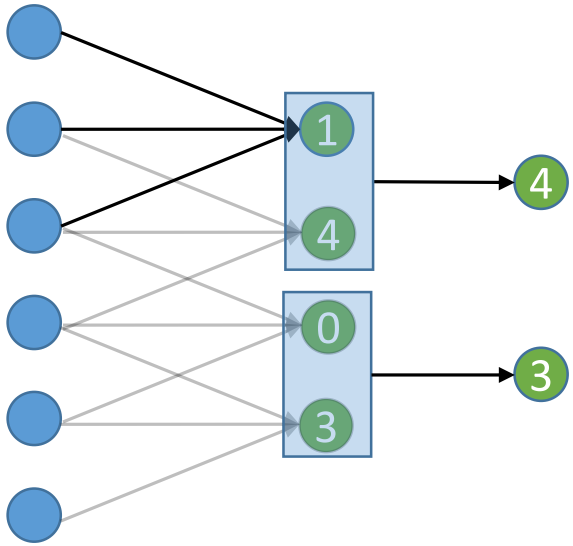
- Multiple feature maps



- All orange units compute the **same function** but with a **different input windows**
- Orange and green units **compute different functions**

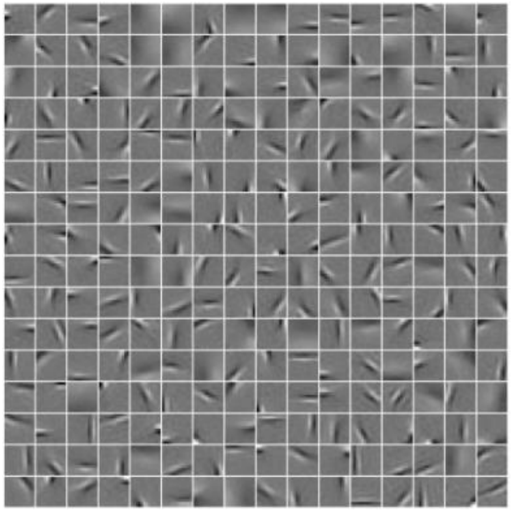
Convolutional NNs

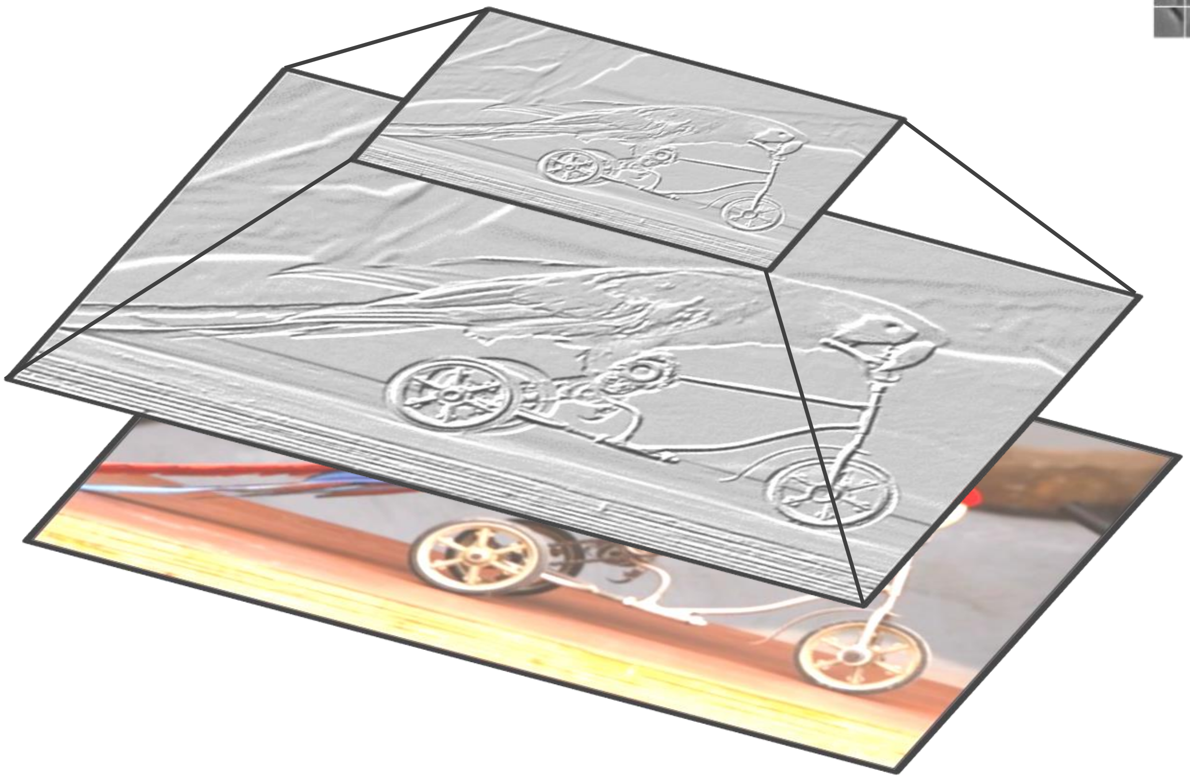
- Pooling (max, average)

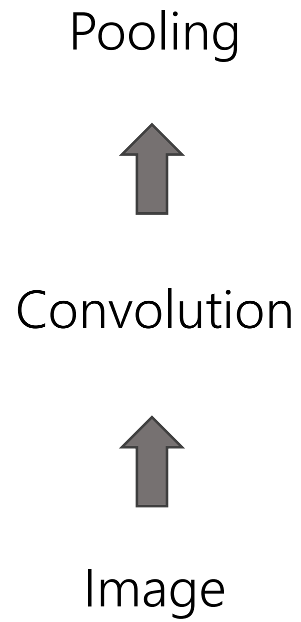


- Pooling area (how much to pool): 2 units
- Pooling stride (how far to move): 2 units
- **Subsamples** feature maps

2D input







Historical perspective – 1980

Hubel and Wiesel
1962

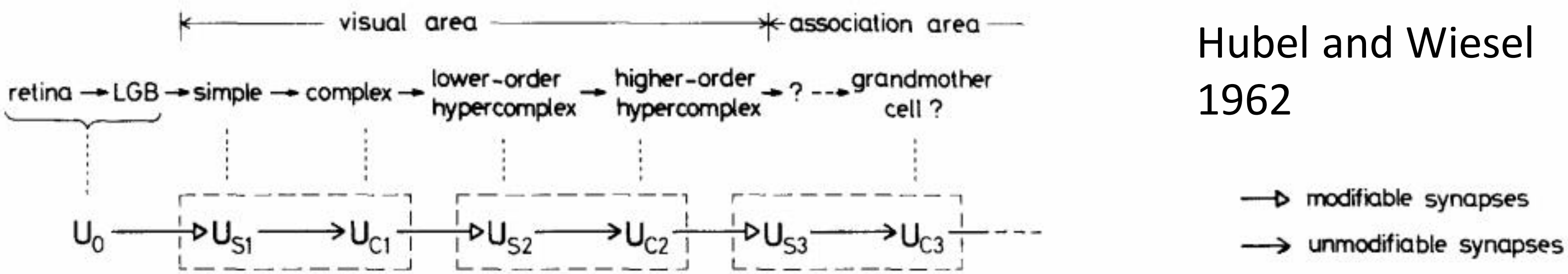


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

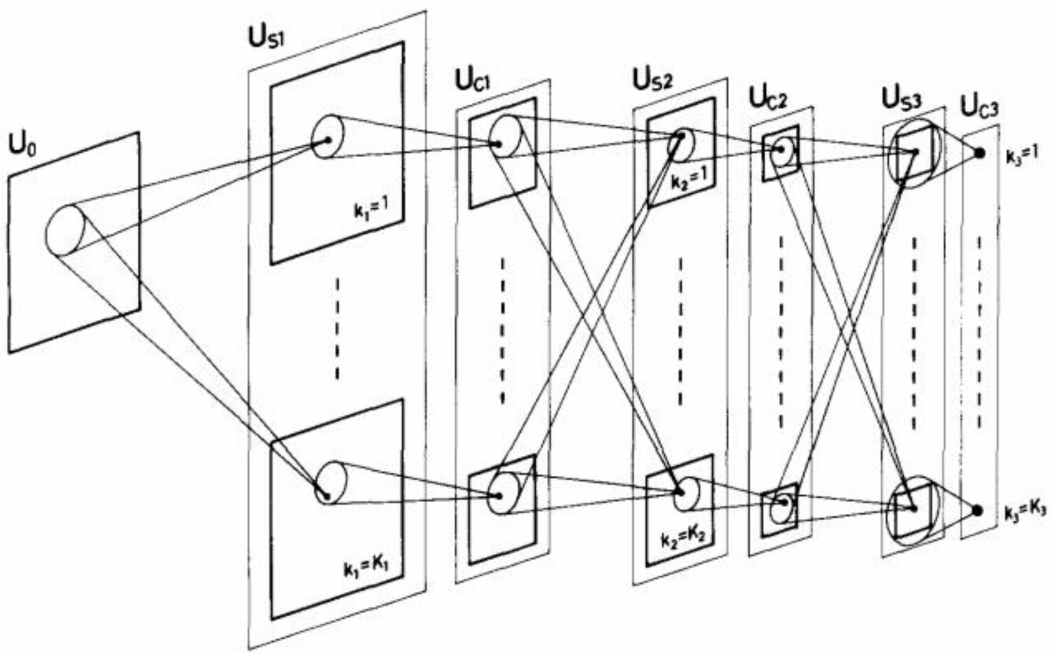


Fig. 2. Schematic diagram illustrating the interconnections between layers in the neocognitron

Included basic ingredients of ConvNets, but no supervised learning algorithm

Supervised learning – 1986

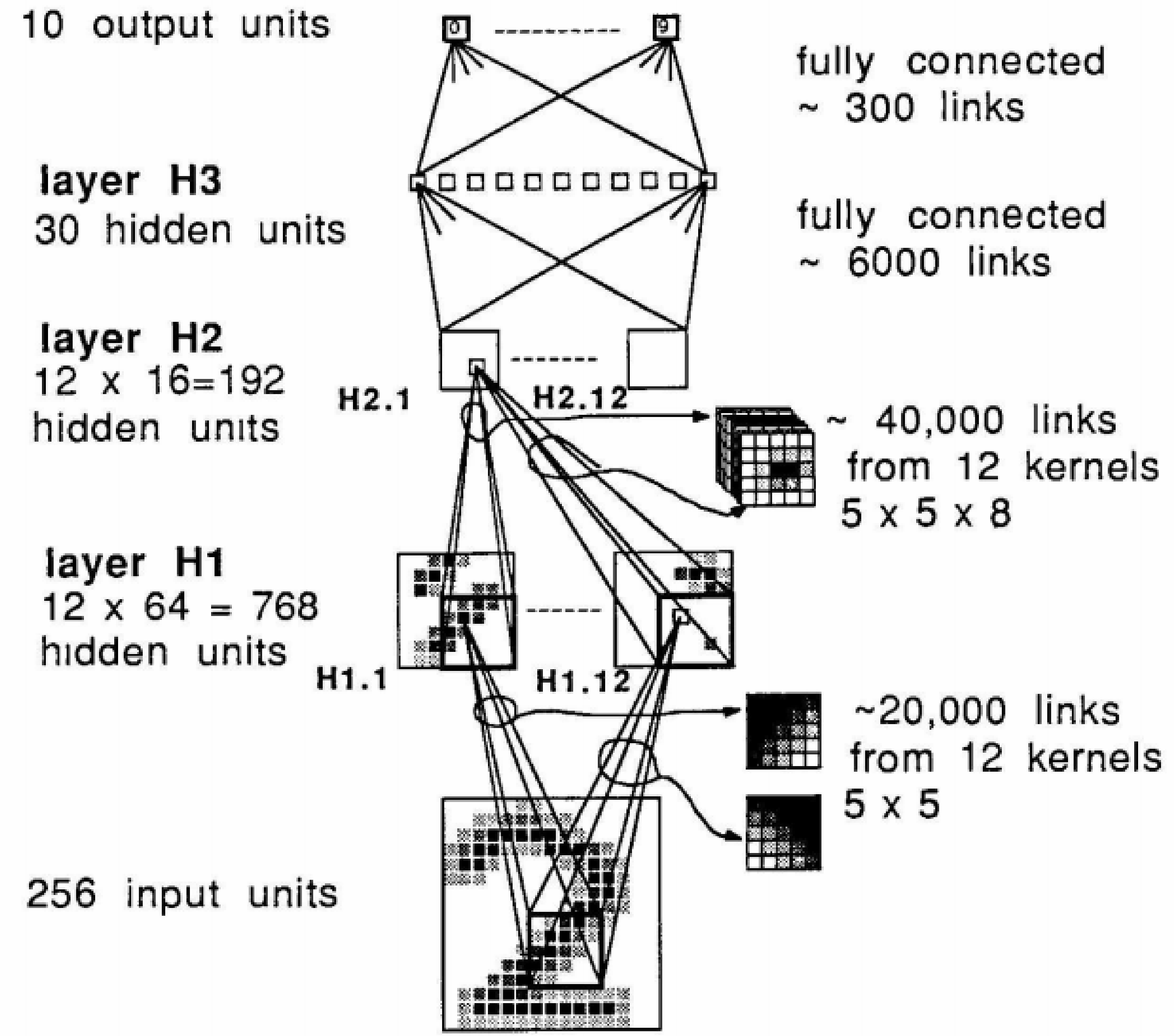
Gradient descent training with error backpropagation

Learning Internal Representations by Error Propagation

D. E. RUMELHART, G. E. HINTON, and R. J. WILLIAMS

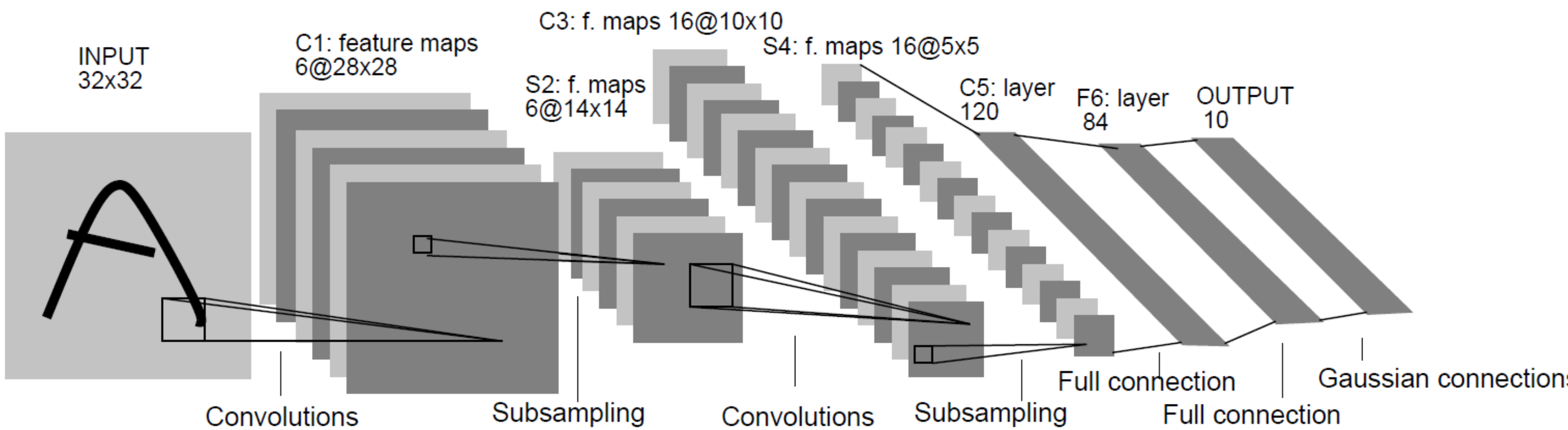
Early demonstration that error backpropagation can be used for supervised training of neural nets (including ConvNets)

1989



Backpropagation applied to handwritten zip code recognition, Lecun et al., 1989

Practical ConvNets



Gradient-Based Learning Applied to Document Recognition,
Lecun et al., 1998

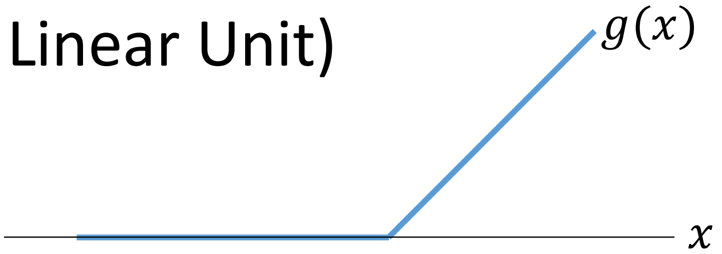
Core idea of “deep learning”

- Input: the “*raw*” signal (image, waveform, …)

- Features: hierarchy of features is *learned* from the raw input

What’s new since the 1980s?

- More layers
  - LeNet-3 and LeNet-5 had 3 and 5 learnable layers
  - Current models have 8 – 20+
- “ReLU” non-linearities (Rectified Linear Unit)
  - $g(x) = \max(0, x)$
  - Gradient doesn’t vanish



- “Dropout” regularization (randomly selects neurons to remove during training epochs, reduces overfitting)
- Fast GPU implementations
- More data

Demo

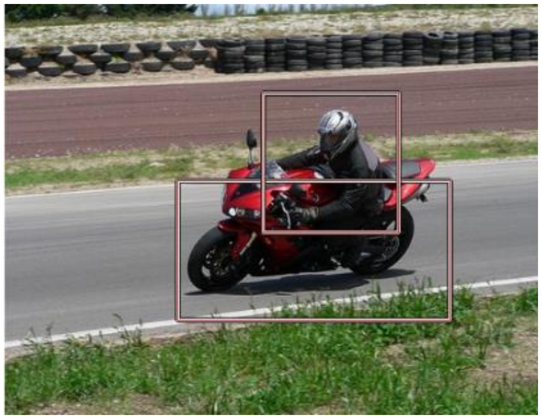
- http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html
- ConvNetJS by Andrej Karpathy (Ph.D. student at Stanford)

Software libraries

- Caffe (C++, python, matlab)
- Torch7 (C++, lua)
- Theano (python)

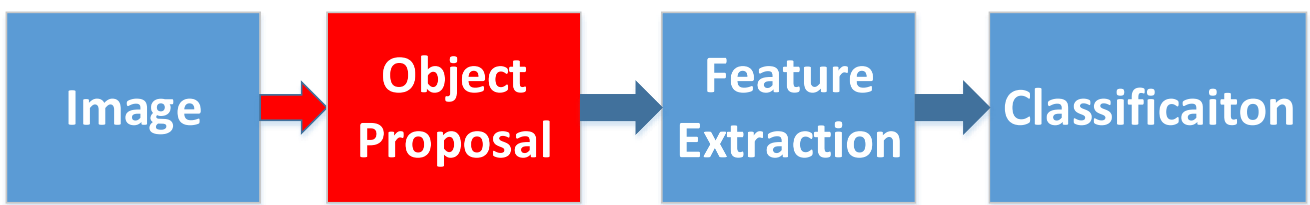
What else? Object Proposals

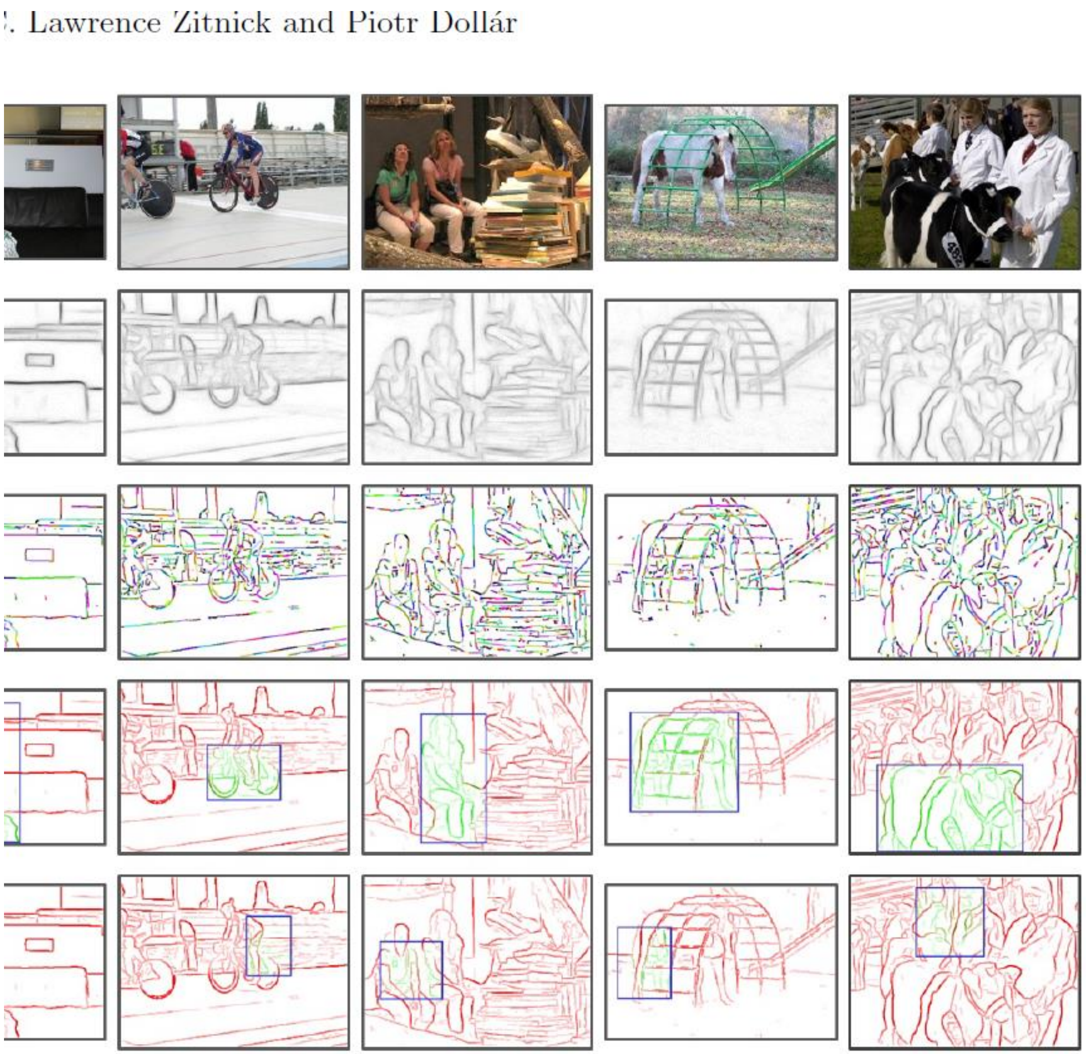
- Sliding window based object detection



Iterate over window size, aspect ratio, and location

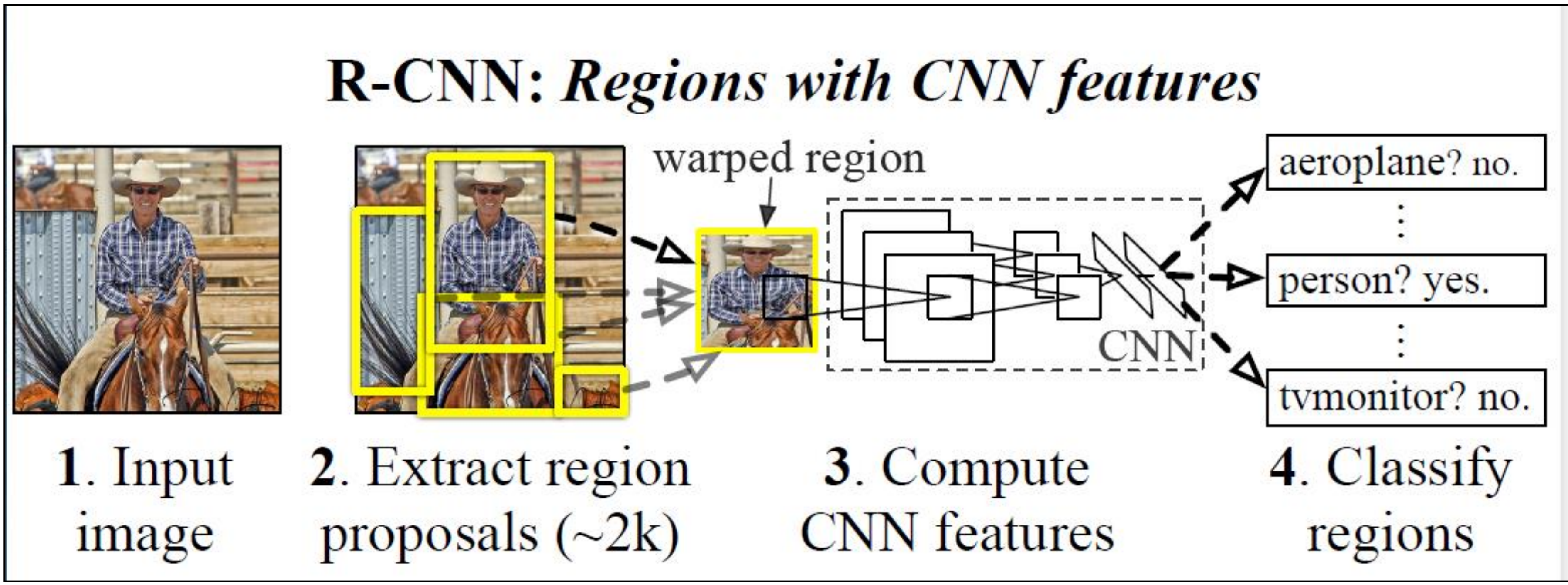
- Object proposals
  - Fast execution
  - High recall with low # of candidate boxes





The number of contours wholly enclosed by a bounding box is indicative of the likelihood of the box containing an object.

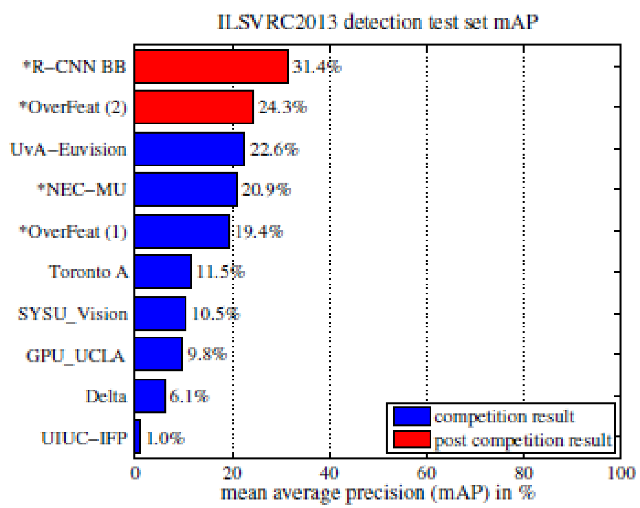
Ross's Own System: Region CNNs



Competitive Results

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [20]†	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [39]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [41]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [18]†	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	**71.8**	**65.8**	**53.0**	**36.8**	**35.9**	**59.7**	**60.0**	**69.9**	**27.9**	**50.6**	**41.4**	**70.0**	**62.0**	**69.0**	**58.1**	**29.5**	**59.4**	**39.3**	**61.2**	**52.4**	**53.7**

Table 1: Detection average precision (%) on VOC 2010 test. R-CNN is most directly comparable to UVA and Regionlets since all methods use selective search region proposals. Bounding-box regression (BB) is described in Section C. At publication time, SegDPM was the top-performer on the PASCAL VOC leaderboard. †DPM and SegDPM use context rescoring not used by the other methods.



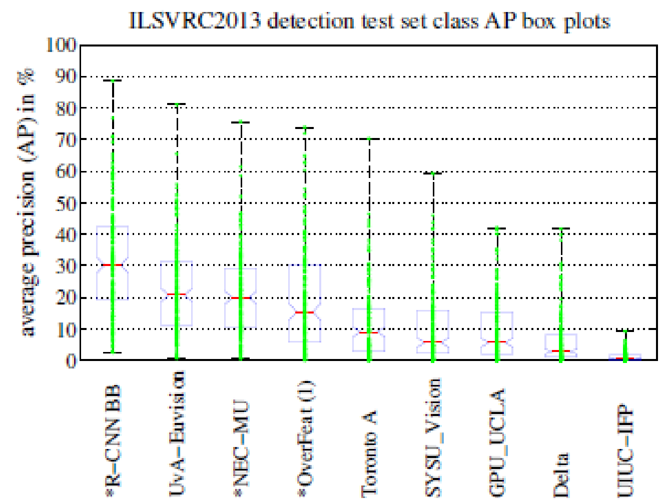


Figure 3: (Left) Mean average precision on the ILSVRC2013 detection test set. Methods preceeded by * use outside training data (images and labels from the ILSVRC classification dataset in all cases). **(Right) Box plots for the 200 average precision values per method.** A box plot for the post-competition OverFeat result is not shown because per-class APs are not yet available (per-class APs for R-CNN are in Table 8 and also included in the tech report source uploaded to arXiv.org; see `R-CNN-ILSVRC2013-APs.txt`). The red line marks the median AP, the box bottom and top are the 25th and 75th percentiles. The whiskers extend to the min and max AP of each method. Each AP is plotted as a green dot over the whiskers (best viewed digitally with zoom).

Top Regions for Six Object Classes

Figure 4: Top regions for six pool$_5$ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

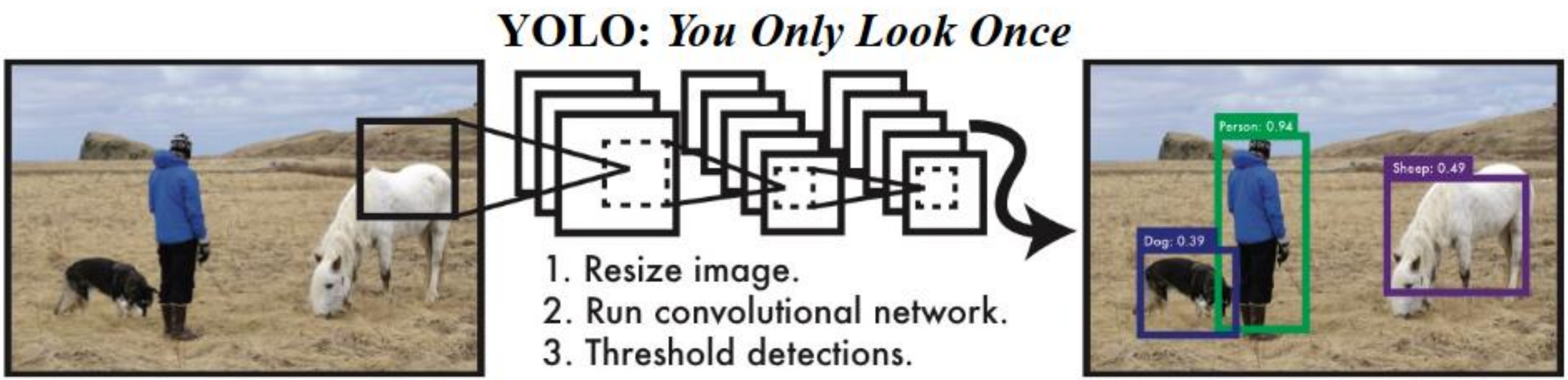
What came Next?

- Faster R-CNN (Girshick, 2016)
- YOLO (Redmon, Girshick, Divvala, Farhadi, 2016)
  You Only Look Once (People's Choice Award)
- YOLO9000: Better, Faster, Stronger (Redmon, Farhadi, CVPR 2017) Runner up for Best Paper
- YOLOv3: more improvements

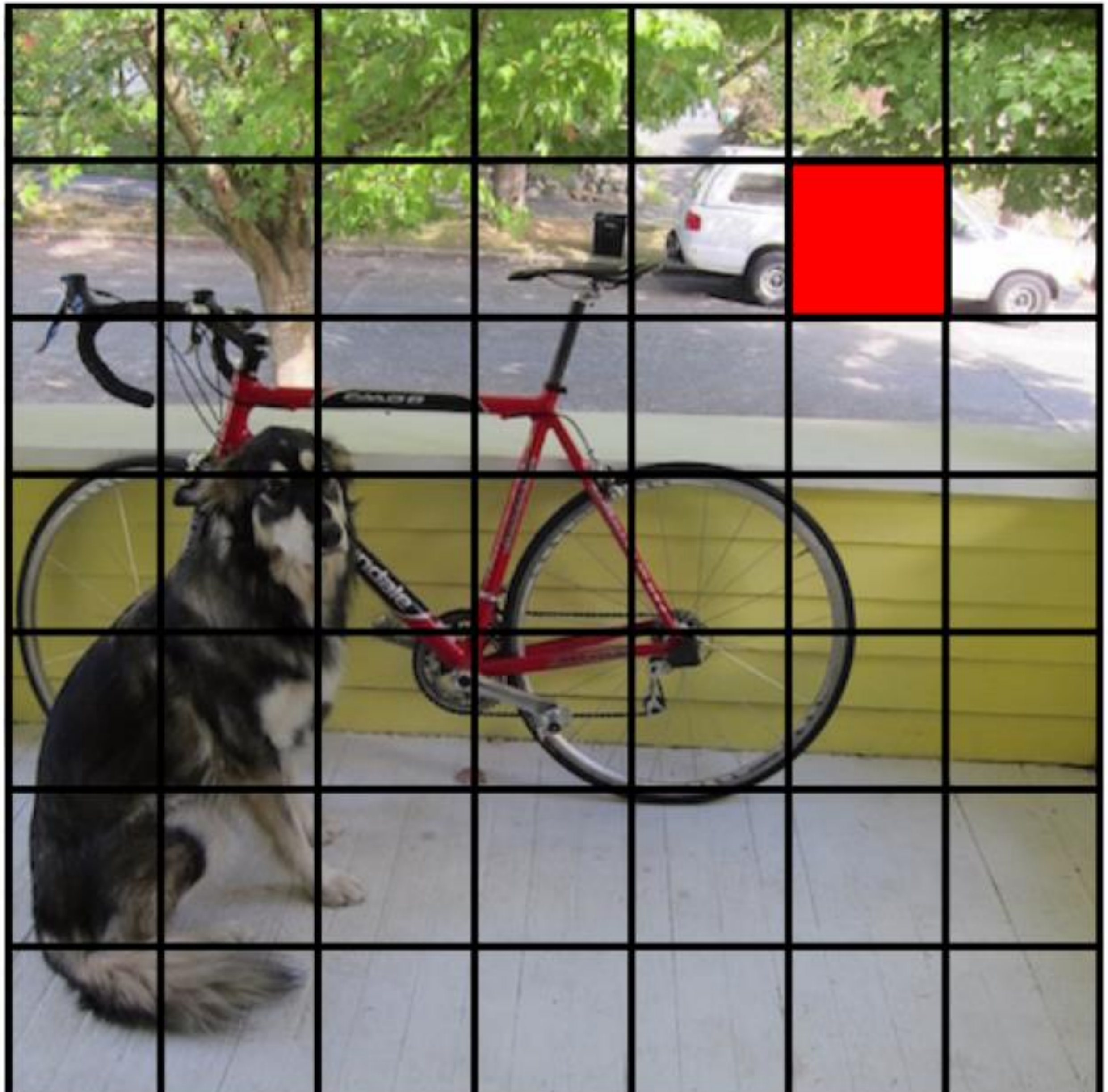
Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	~~63.4~~ 69.0	45 FPS	22 ms/img

With YOLO, you only look once at an image to perform detection

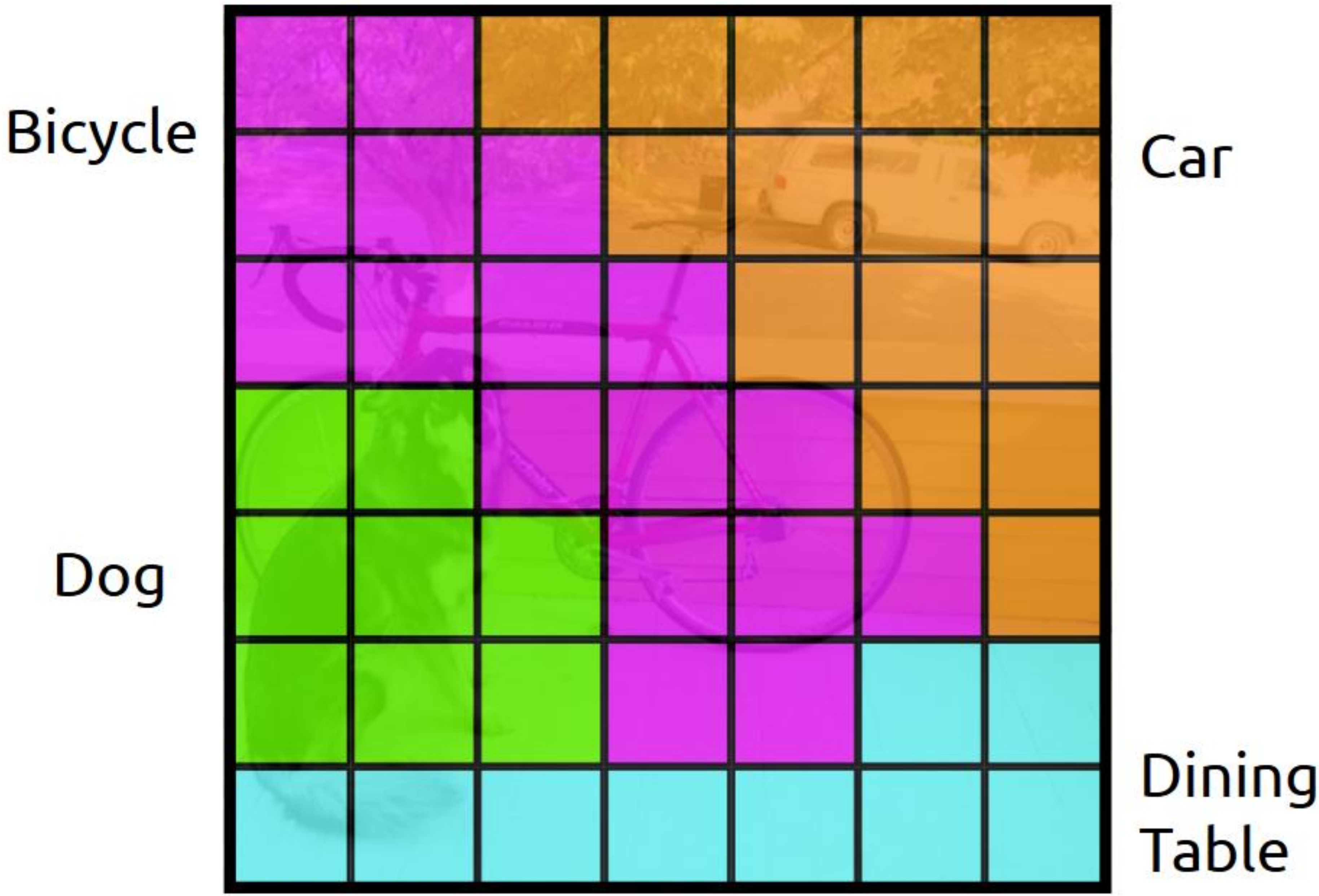


Each cell predicts boxes and confidences: P(Object)

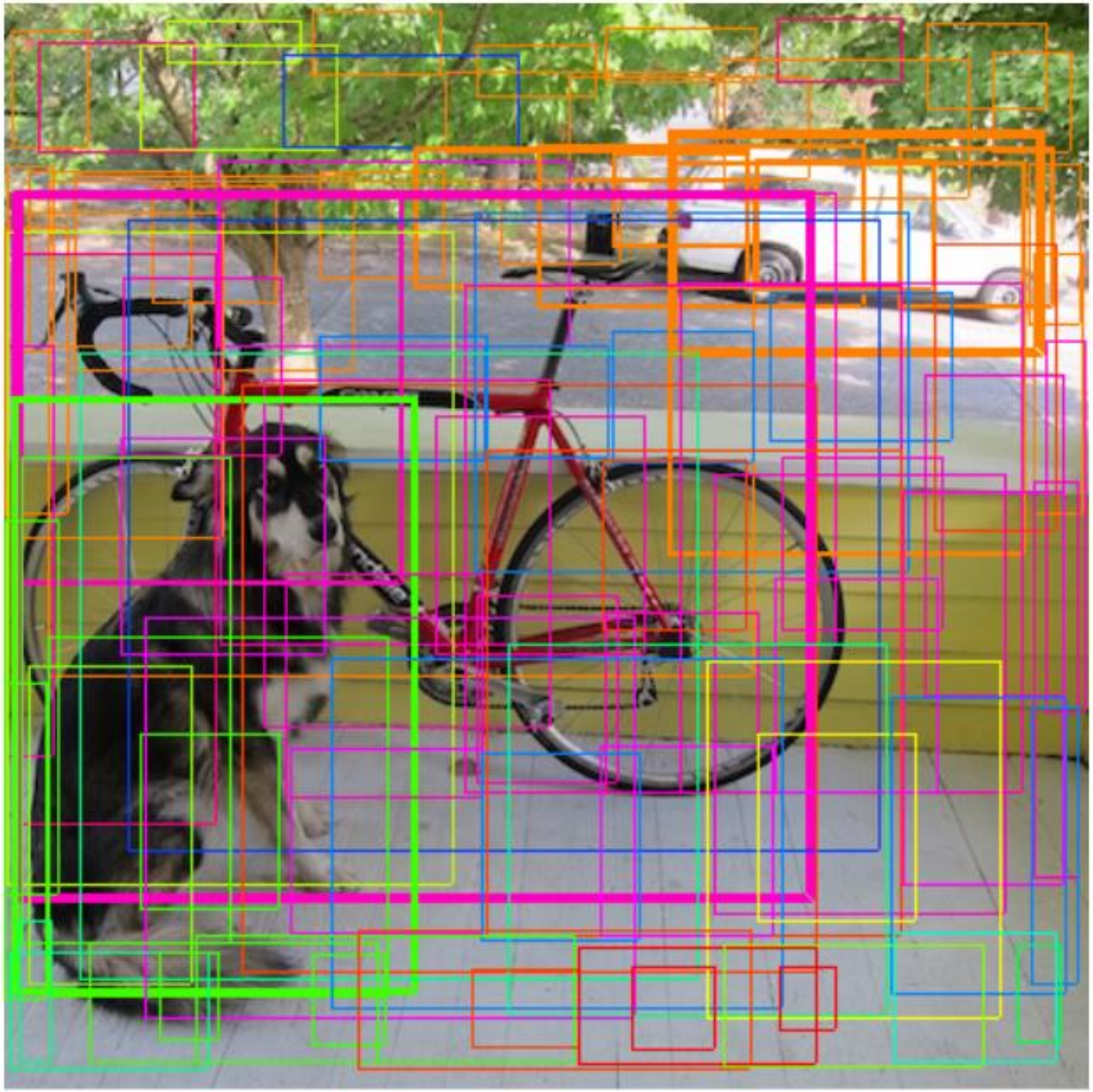


Each cell predicts boxes and confidences: P(Object)

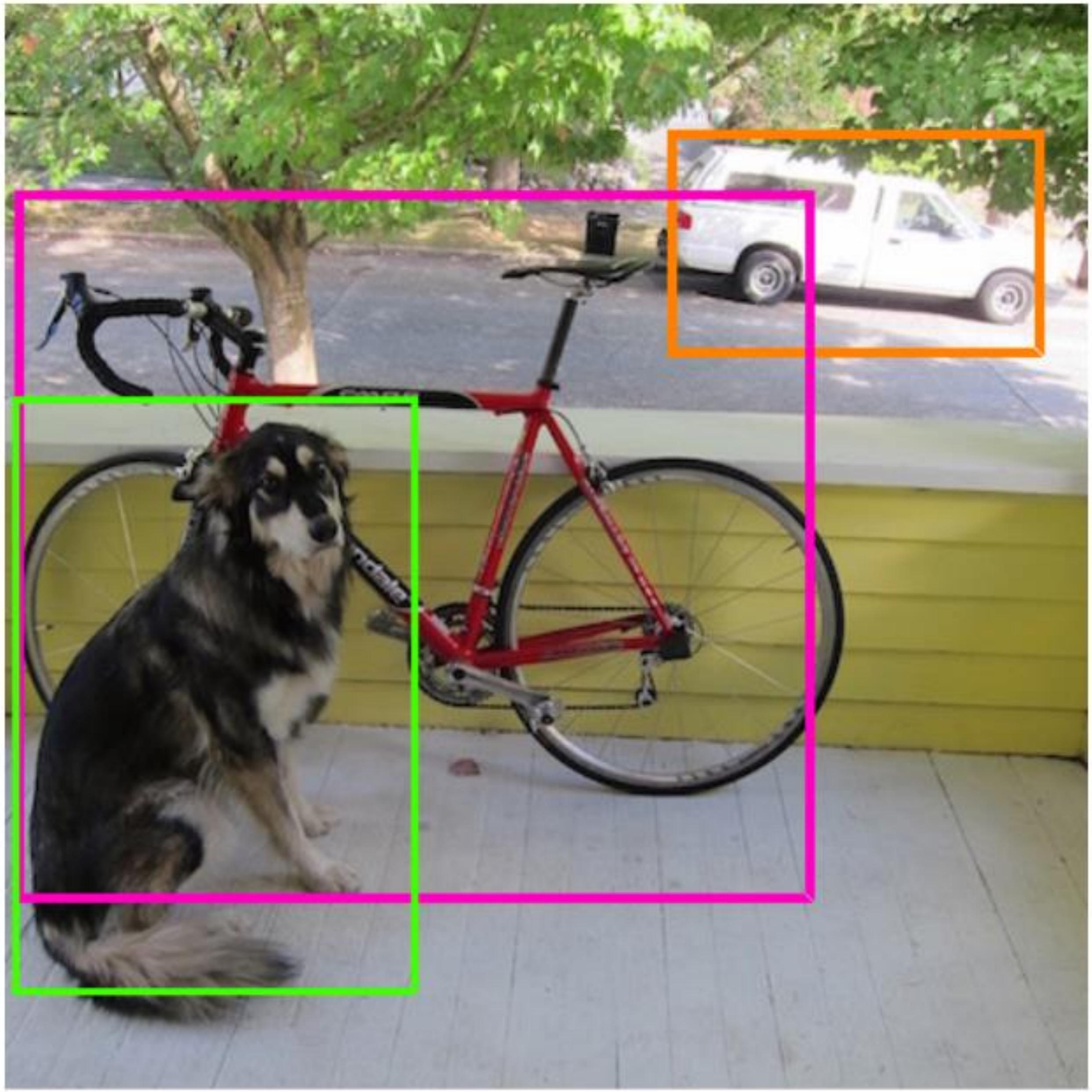
Each cell also predicts a class probability.



Then we combine the box and class predictions.



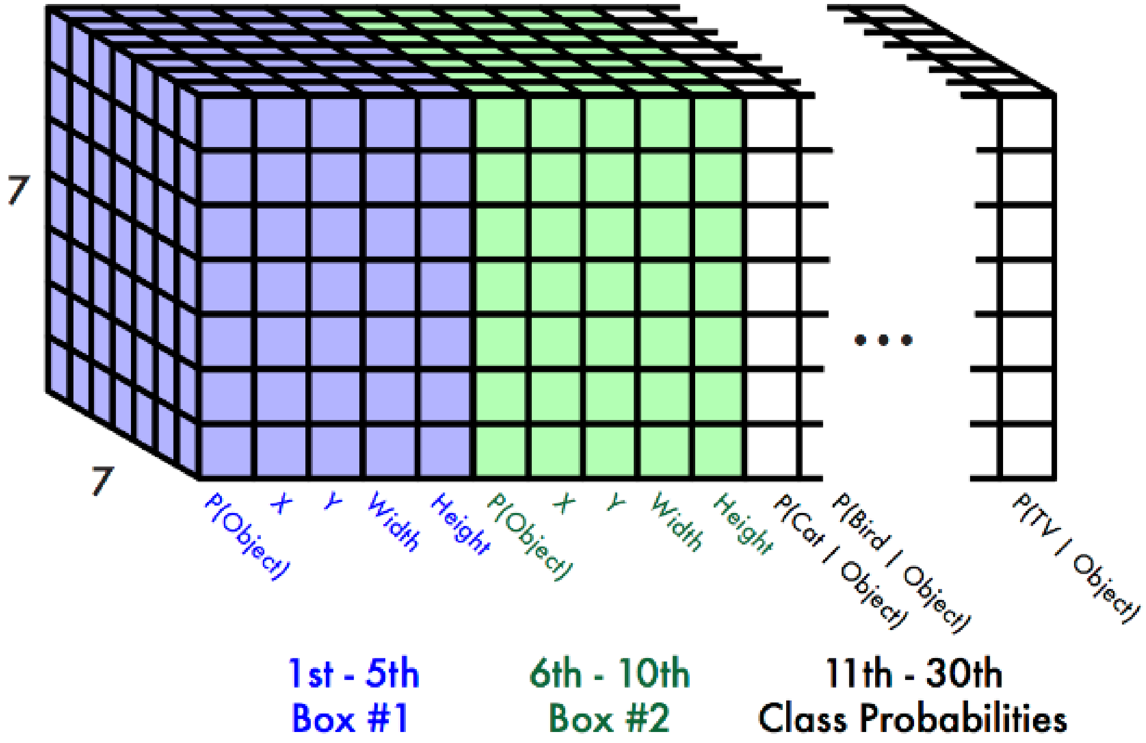
Finally we do NMS and threshold detections



This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities

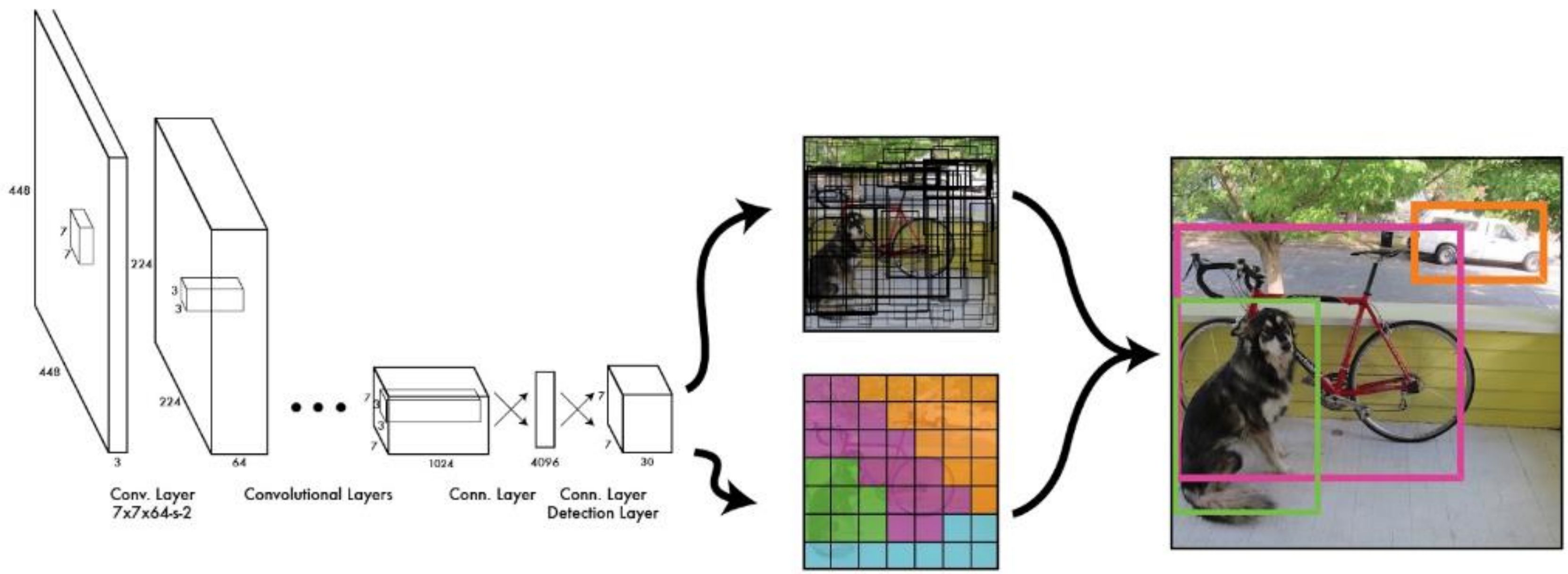


For Pascal VOC:

- 7x7 grid
- 2 bounding boxes / cell
- 20 classes

7 x 7 x (2 x 5 + 20) = 7 x 7 x 30 tensor = **1470 outputs**

Thus we can train one neural network to be a whole detection pipeline



More YOLO (second paper)

- At 67 FPS, YOLOv2 gets76.8 mAP on VOC 2007.
- At 40 FPS, YOLOv2 gets 78.6mAP, outperforming state-of-the-art methods like Faster R-CNN with ResNet and SSD while still running significantly faster
- A new methodology that jointly trains for detection and classification produced YOLO9000.
- YOLO9000 can detect more than 9000 object categories in real time.
- And YOLOv3 has come out.

Finale

- Object recognition has moved rapidly in the last 12 years to becoming very appearance based.
- The HOG descriptor lead to fast recognition of specific views of generic objects, starting with pedestrians and using SVMs.
- Deformable parts models extended that to allow more objects with articulated limbs, but still specific views.
- CNNs have become the method of choice; they learn from huge amounts of data and can learn multiple views of each object class.
- YOLO is the current winner (I think). CVPR 19 is coming up in June.