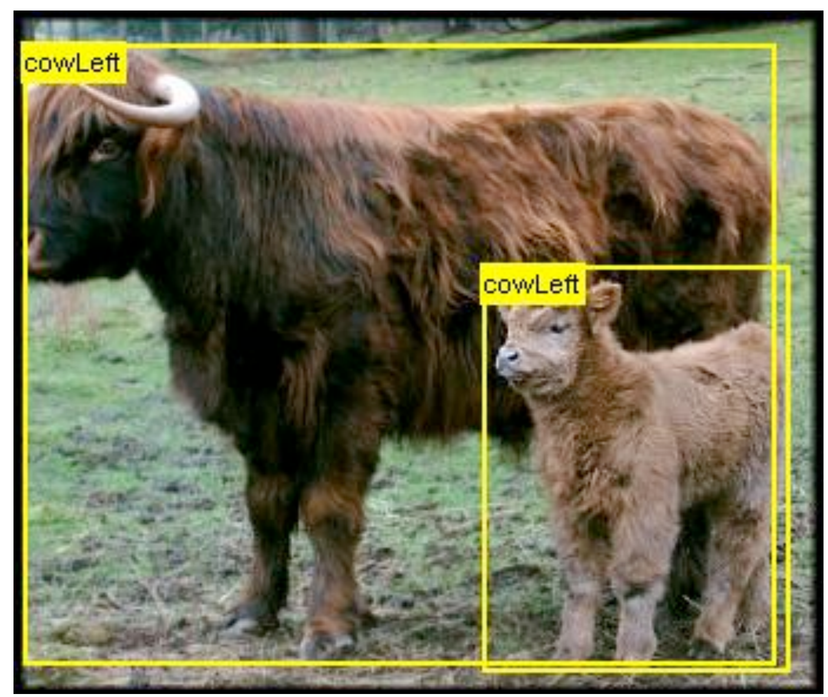
Object Detection with Discriminatively Trained Part Based Models

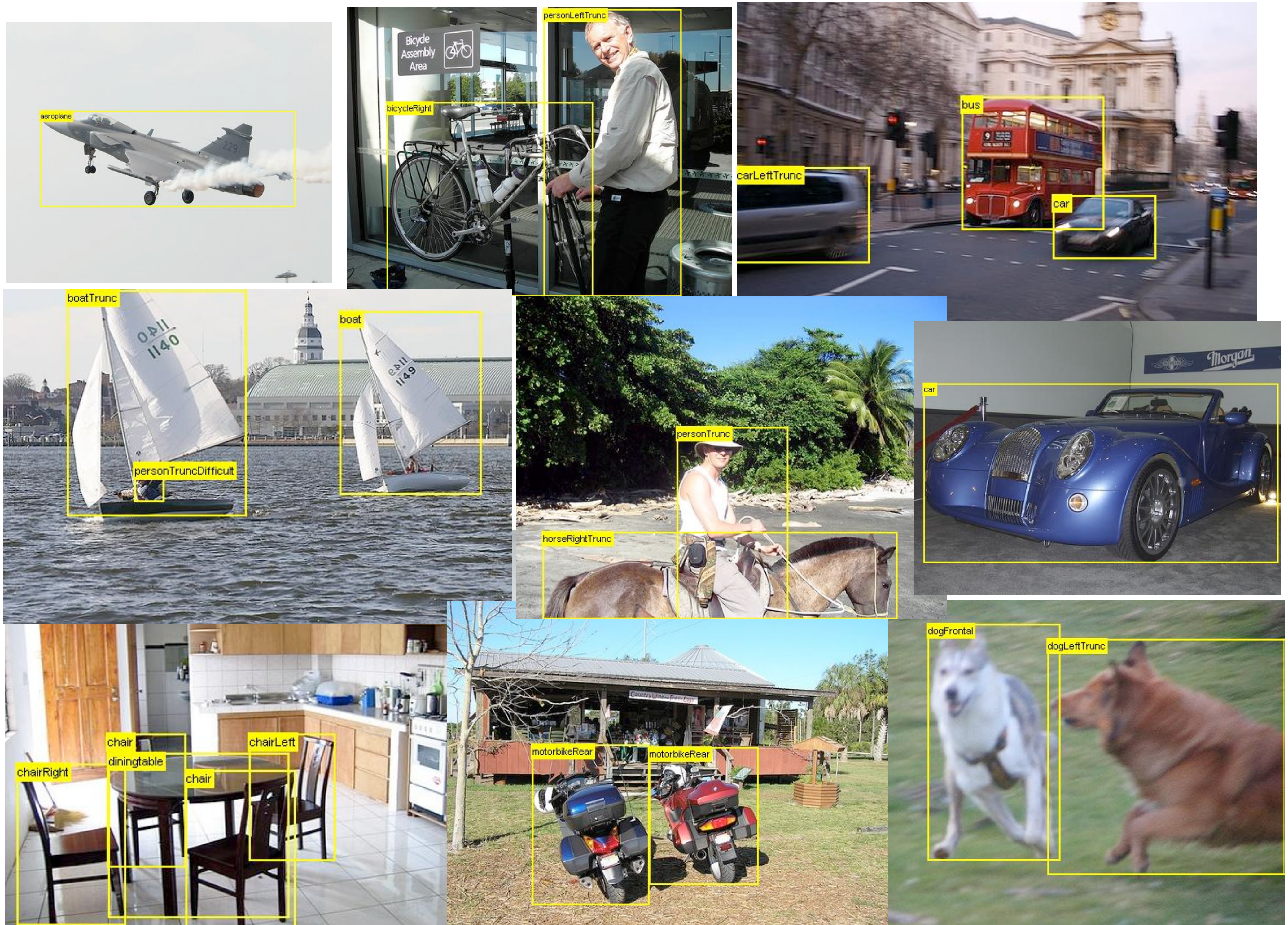
Pedro F. Felzenszwalb
Department of Computer Science
University of Chicago

Joint with David Mcallester, Deva Ramanan, Ross Girshick

PASCAL Challenge

- ~10,000 images, with ~25,000 target objects
  - Objects from 20 categories (person, car, bicycle, cow, table...)
  - Objects are annotated with labeled bounding boxes



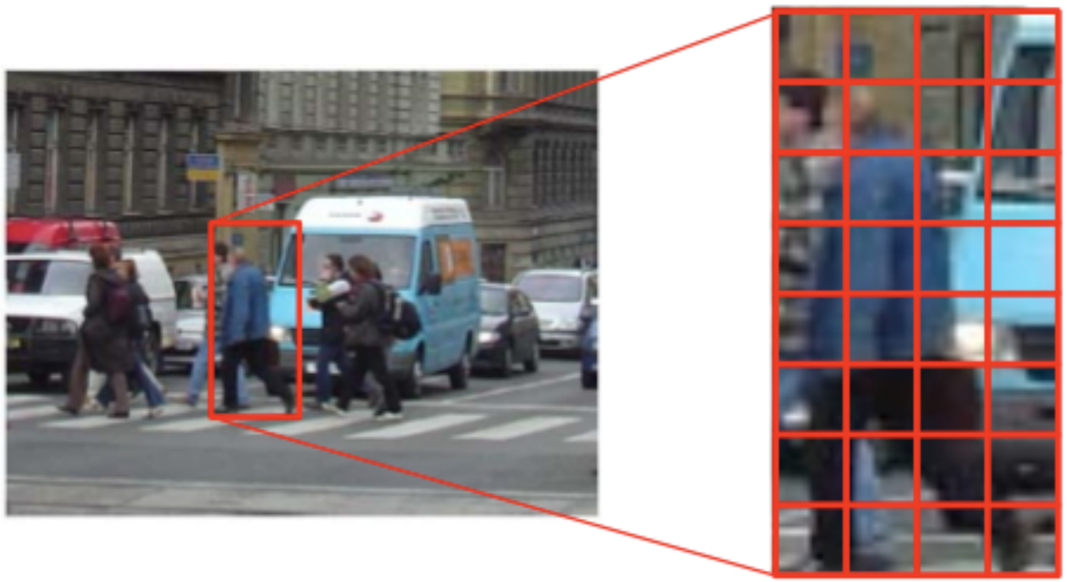

aeroplane
Bicycle Assembly Area
bicycleRight
personLeftTrunc
carLeftTrunc
bus
car
boatTrunc
1140
personTruncDifficult
boat
1149
personTrunc
horseRightTrunc
Morgan
car
chair
diningtable
chairRight
chair
chairLeft
motorbikeRear
motorbikeRear
dogFrontal
dogLeftTrunc

Why is it hard?

- Objects in rich categories exhibit significant variability
  - Photometric variation
  - Viewpoint variation
  - Intra-class variability
    - Cars come in a variety of shapes (sedan, minivan, etc)
    - People wear different clothes and take different poses

We need rich object models
But this leads to difficult matching and training problems

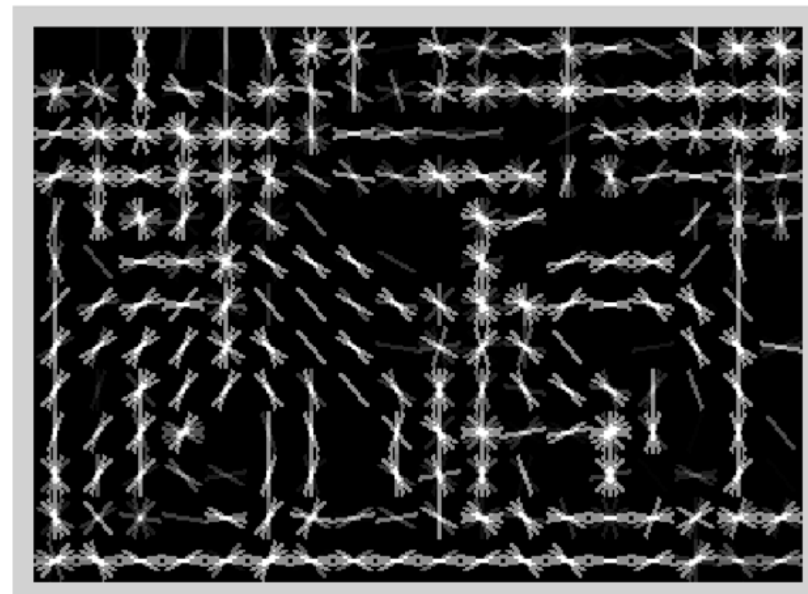
Starting point: sliding window classifiers



Feature vector
$x = [\ldots, \ldots, \ldots, \ldots]$

- Detect objects by testing each subwindow
  - Reduces object detection to binary classification
  - Dalal & Triggs: HOG features + linear SVM classifier
  - Previous state of the art for detecting people

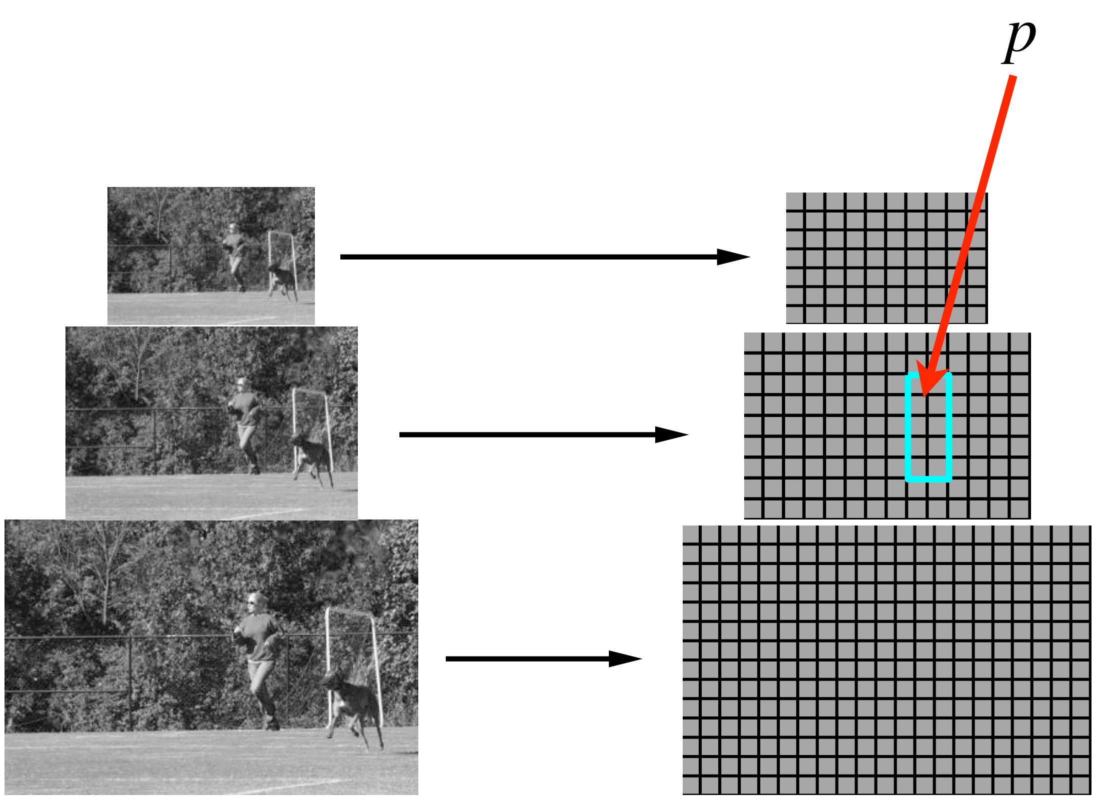
Histogram of Gradient (HOG) features

- Image is partitioned into 8x8 pixel blocks
- In each block we compute a histogram of gradient orientations
  - Invariant to changes in lighting, small deformations, etc.
- Compute features at different resolutions (pyramid)

HOG Filters

- Array of weights for features in subwindow of HOG pyramid
- Score is dot product of filter and feature vector



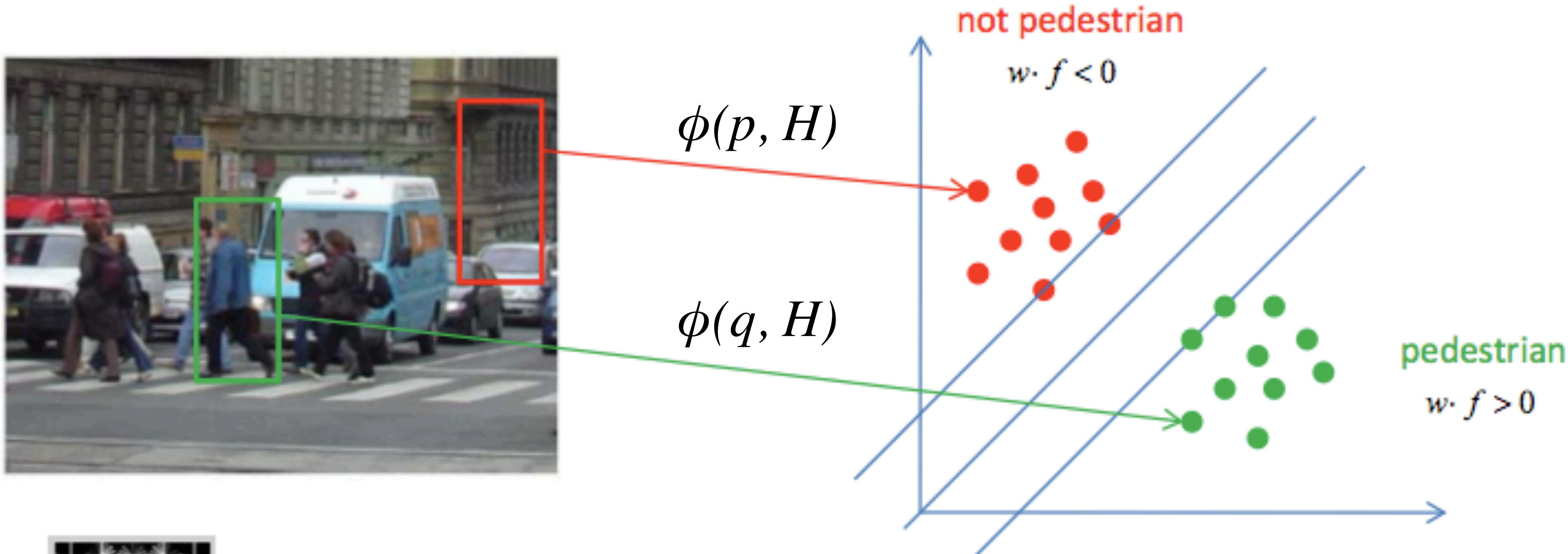
HOG pyramid H

Filter F

Score of F at position p is
$F \cdot \phi(p, H)$

$\phi(p, H)$ = concatenation of HOG features from subwindow specified by p

Dalal & Triggs: HOG + linear SVMs



Typical form of a model

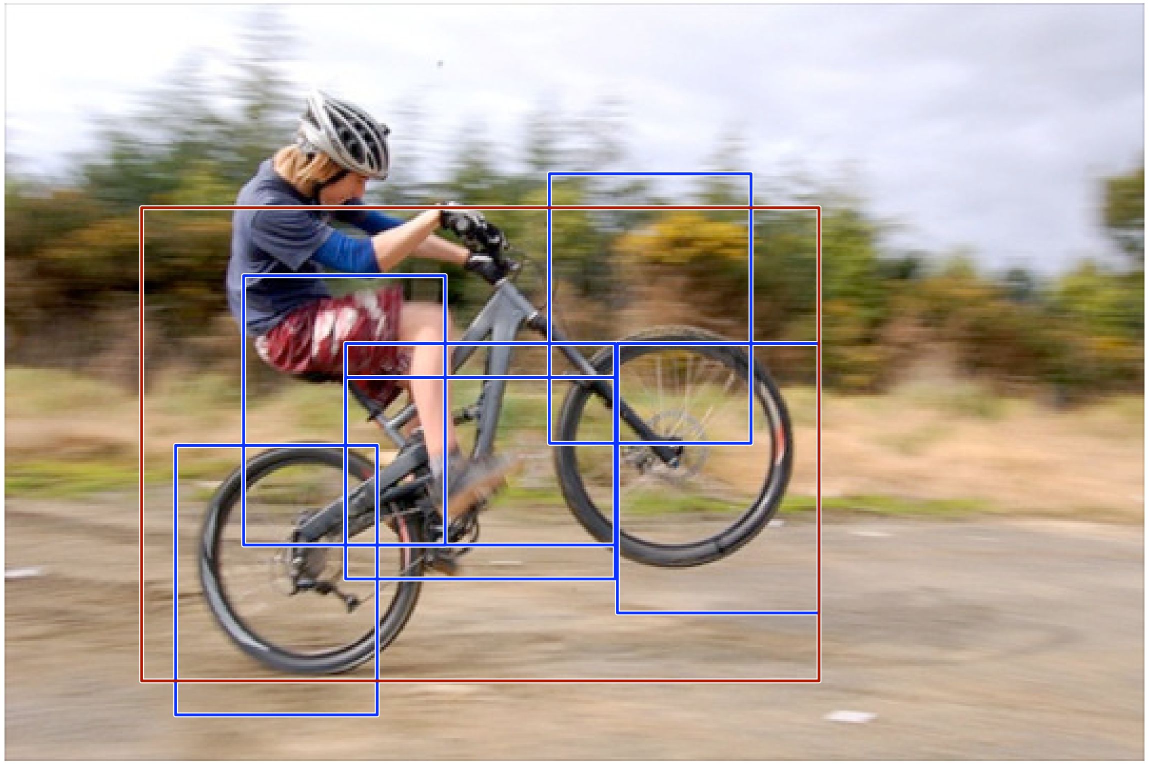
There is much more background than objects

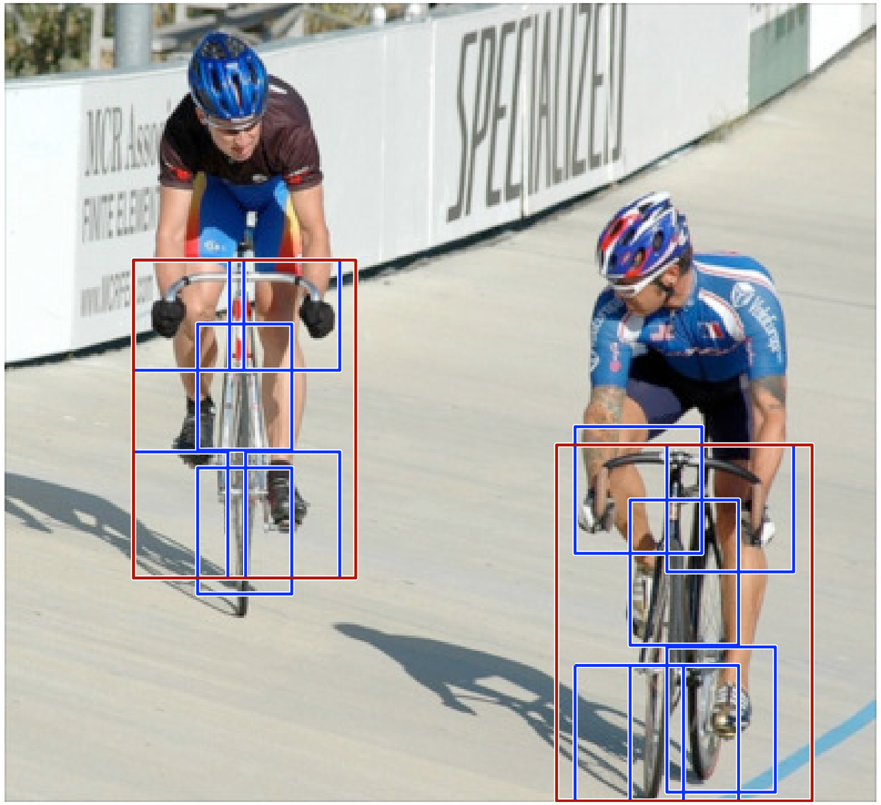
Start with random negatives and repeat:

1) Train a model

2) Harvest false positives to define "hard negatives"

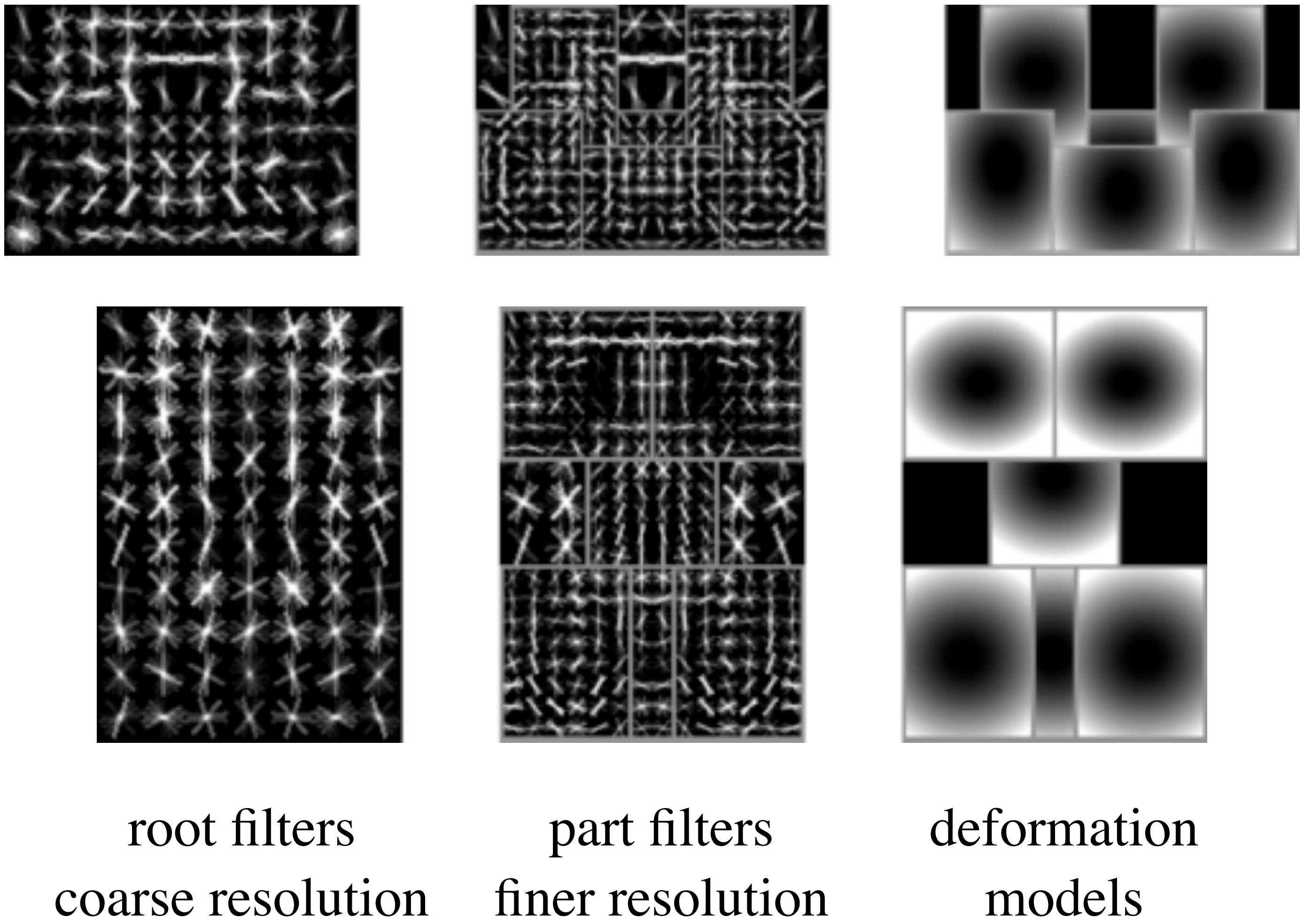
Overview of our models





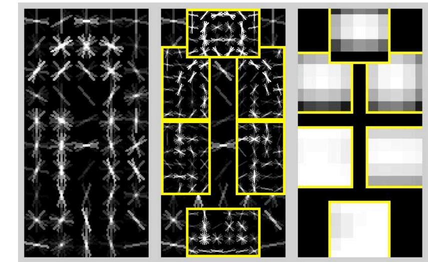
- Mixture of deformable part models
- Each component has global template + deformable parts
- Fully trained from bounding boxes alone

2 component bicycle model



Each component has a root filter F_0
and n part models (F_i, v_i, d_i)

Object hypothesis



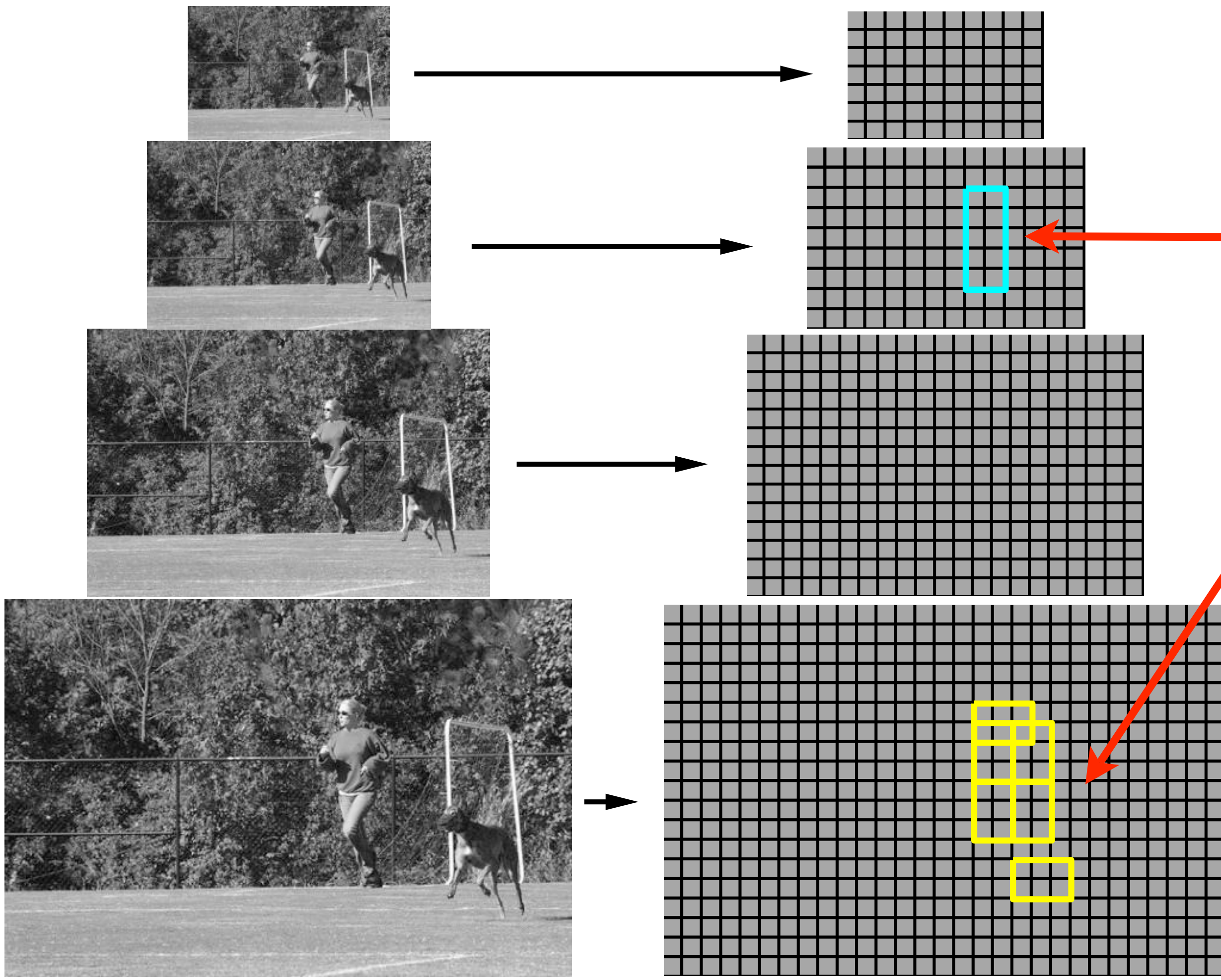


Image pyramid HOG feature pyramid

$z = (p_0, \ldots, p_n)$

p_0 : location of root

$p_1, \ldots, p_n$: location of parts

Score is sum of filter scores minus deformation costs

Multiscale model captures features at two-resolutions

Score of a hypothesis

"data term" "spatial prior"

$$\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)$$

displacements

filters

deformation parameters

$$\text{score}(z) = \beta \cdot \Psi(H, z)$$

concatenation filters and deformation parameters

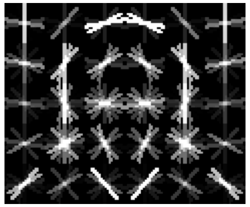
concatenation of HOG features and part displacement features

Matching

- Define an overall score for each root location
  - Based on best placement of parts

$$\text{score}(p_0) = \max_{p_1,\ldots,p_n} \text{score}(p_0, \ldots, p_n).$$

- High scoring root locations define detections
  - "sliding window approach"
- Efficient computation: dynamic programming + generalized distance transforms (max-convolution)



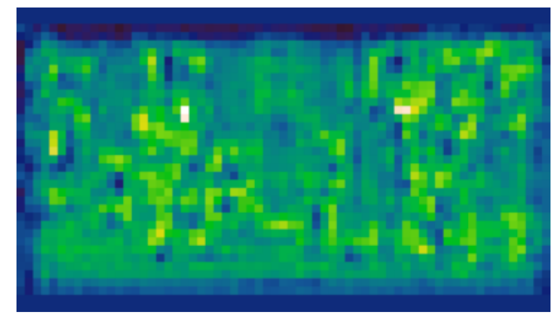
head filter

input image

Response of filter in l-th pyramid level

$$R_l(x, y) = F \cdot \phi(H, (x, y, l))$$

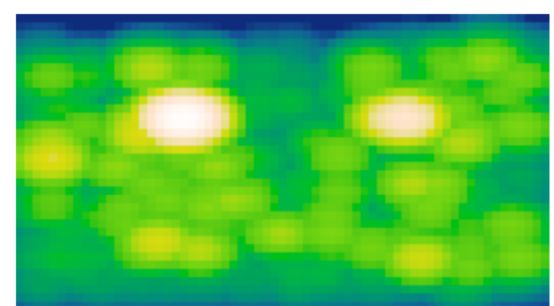
cross-correlation

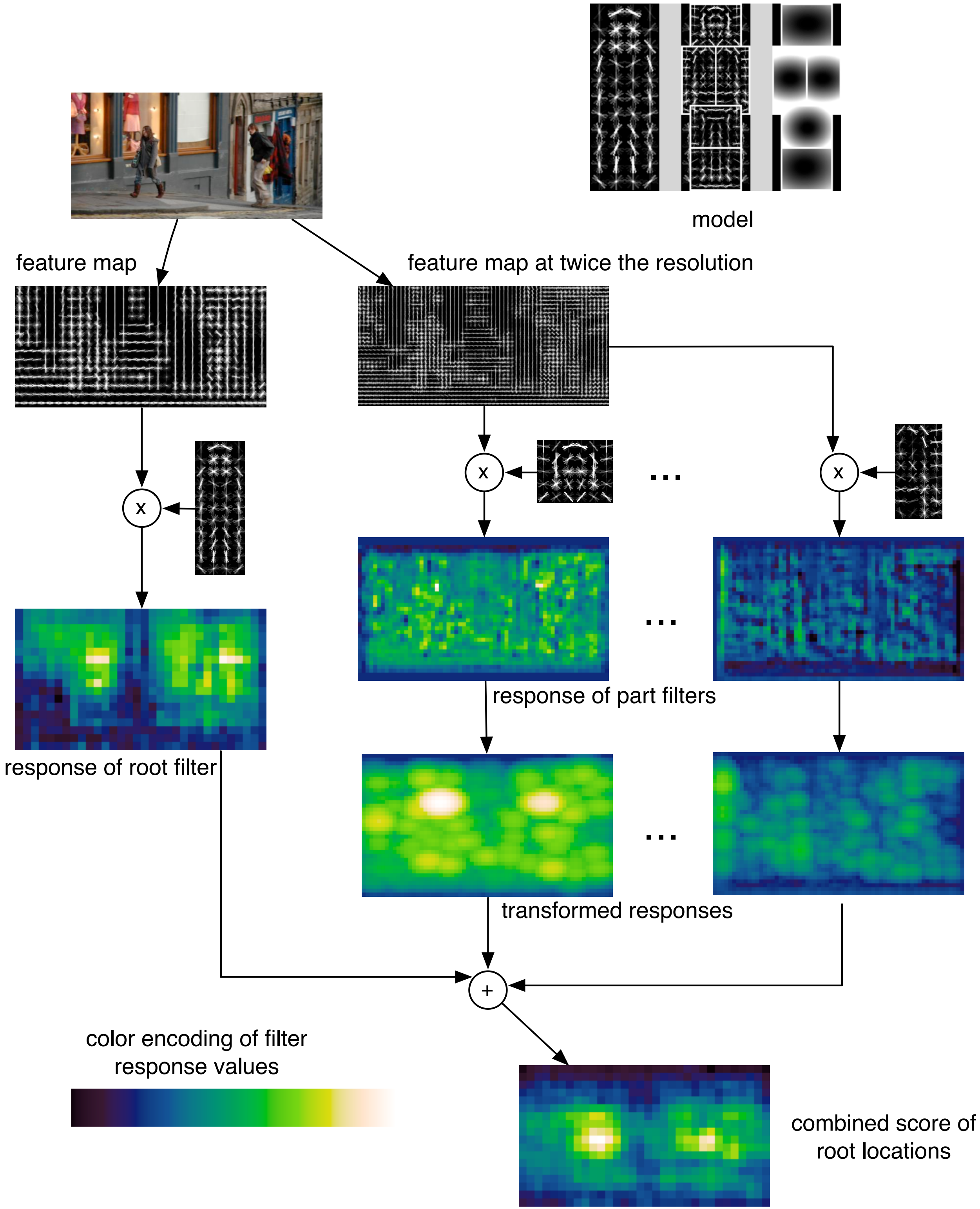


Transformed response

$$D_l(x, y) = \max_{dx, dy} \left(R_l(x + dx, y + dy) - d_i \cdot (dx^2, dy^2) \right)$$

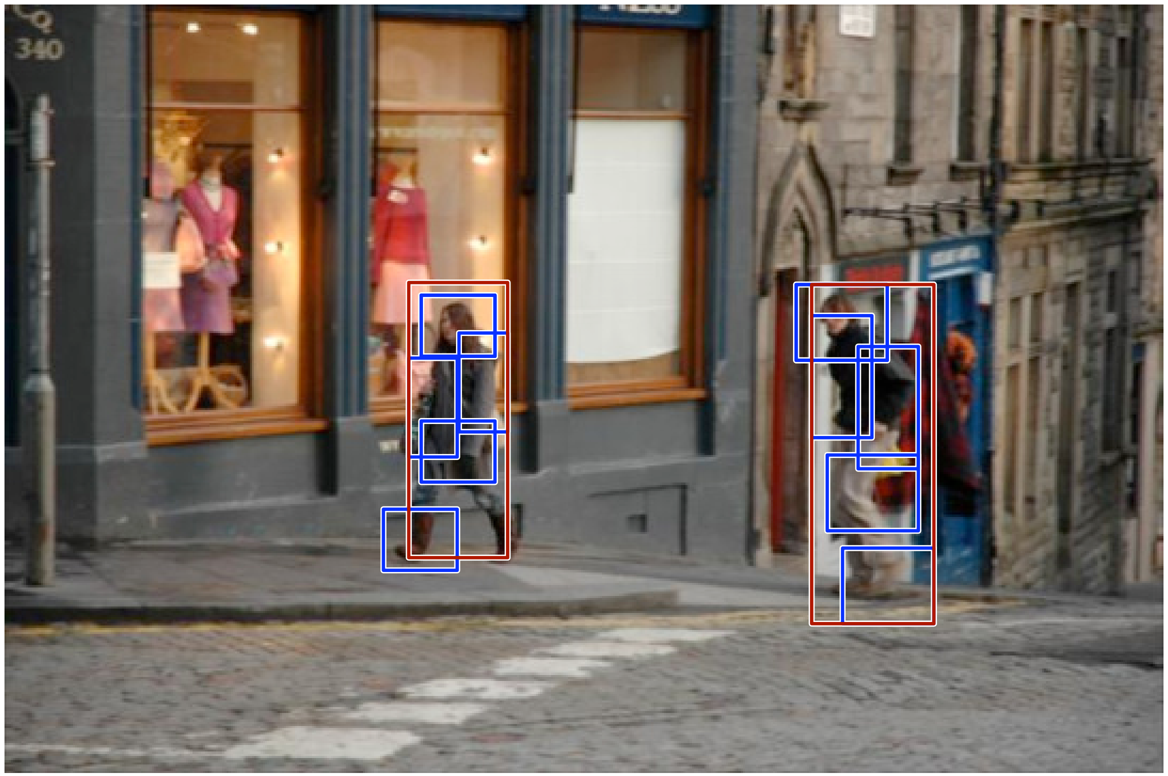
max-convolution, computed in linear time
(spreading, local max, etc)

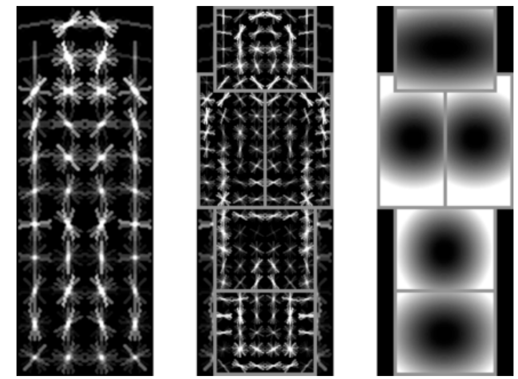




model
feature map
feature map at twice the resolution
response of part filters
response of root filter
transformed responses
color encoding of filter response values
combined score of root locations

Matching results



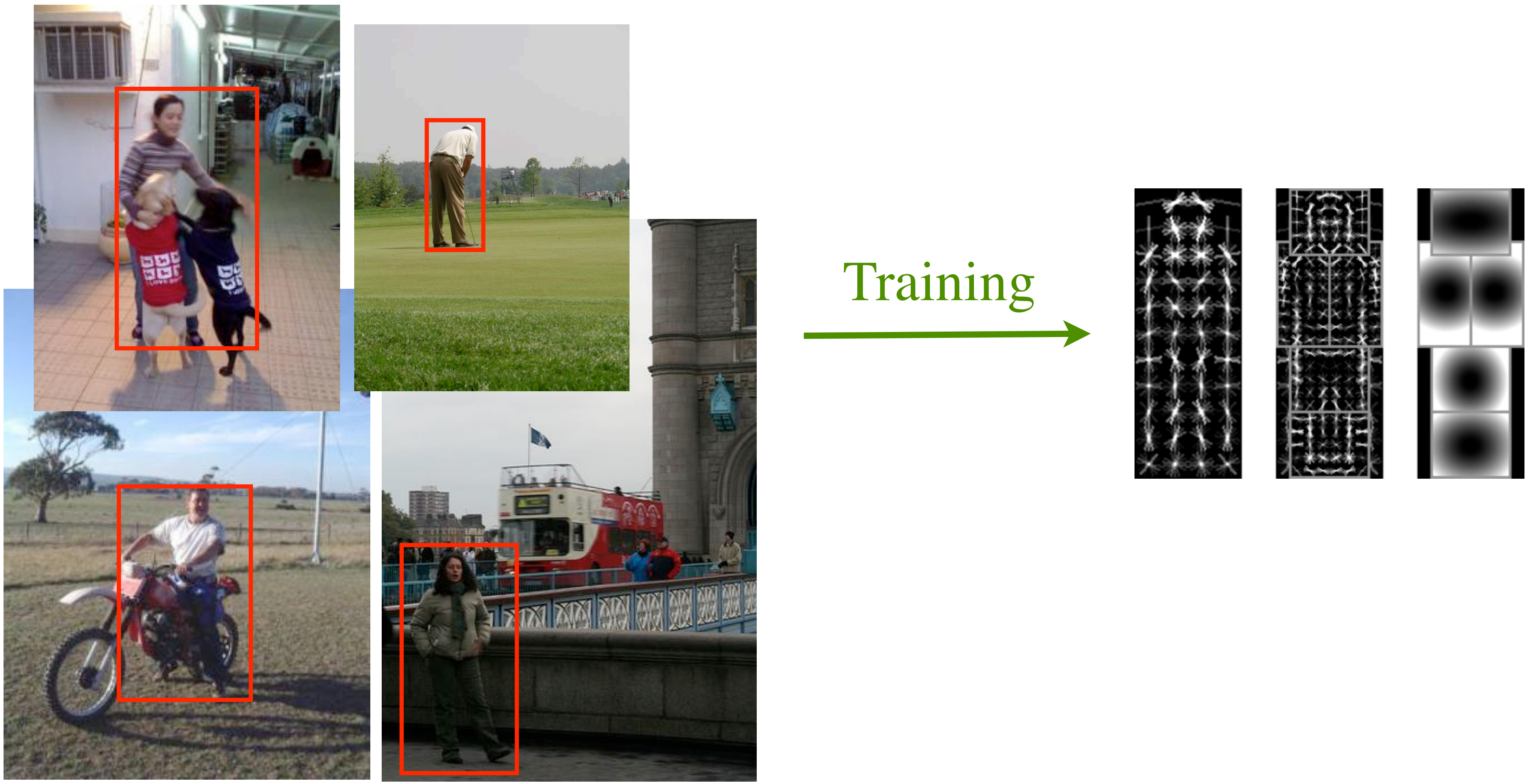


(after non-maximum suppression)

~1 second to search all scales

Training

- Training data consists of images with labeled bounding boxes.
- Need to learn the model structure, filters and deformation costs.



Latent SVM (MI-SVM)

Classifiers that score an example *x* using

$$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z)$$

β are model parameters
z are latent values

Training data $D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle)$ $\quad y_i \in \{-1, 1\}$

We would like to find β such that: $y_i f_\beta(x_i) > 0$

Minimize

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$

Semi-convexity

- Maximum of convex functions is convex
- $f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z)$ is convex in β
- $\max(0, 1 - y_i f_\beta(x_i))$ is convex for negative examples

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$

Convex if latent values for positive examples are fixed

Latent SVM training

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{n}\max(0, 1 - y_i f_\beta(x_i))$$

- Convex if we fix z for positive examples
- Optimization:
  - Initialize β and iterate:
    - Pick best z for each positive example
    - Optimize β via gradient descent with data-mining

Training Models

- Reduce to Latent SVM training problem
- Positive example specifies some z should have high score
- Bounding box defines range of root locations
  - Parts can be anywhere
  - This defines $Z(x)$

Background

- Negative example specifies no z should have high score
- One negative example per root location in a background image
  - Huge number of negative examples
  - Consistent with requiring low false-positive rate

Training algorithm, nested iterations

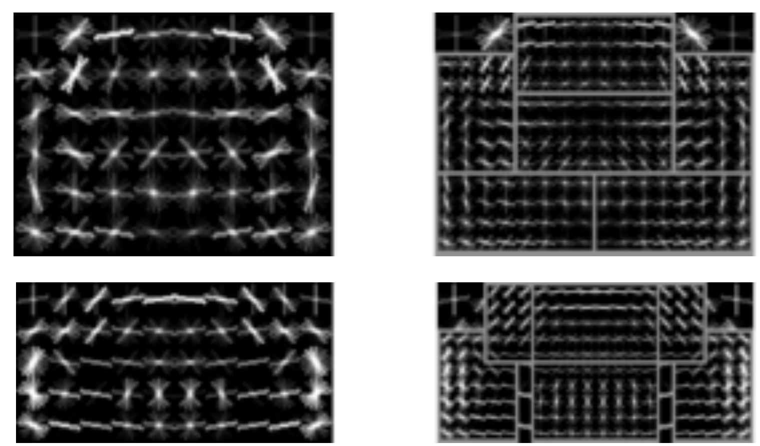
Fix “best” positive latent values for positives

Harvest high scoring (x,z) pairs from background images

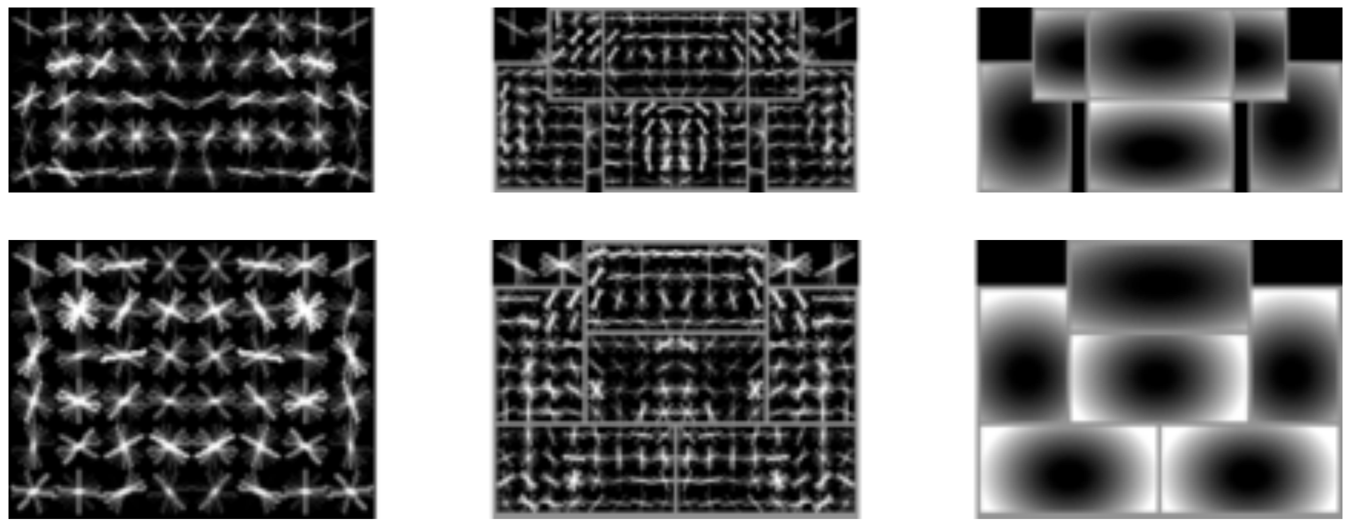
Update model using gradient descent

Trow away (x,z) pairs with low score

- Sequence of training rounds
  - Train root filters
  - Initialize parts from root
  - Train final model



Car model

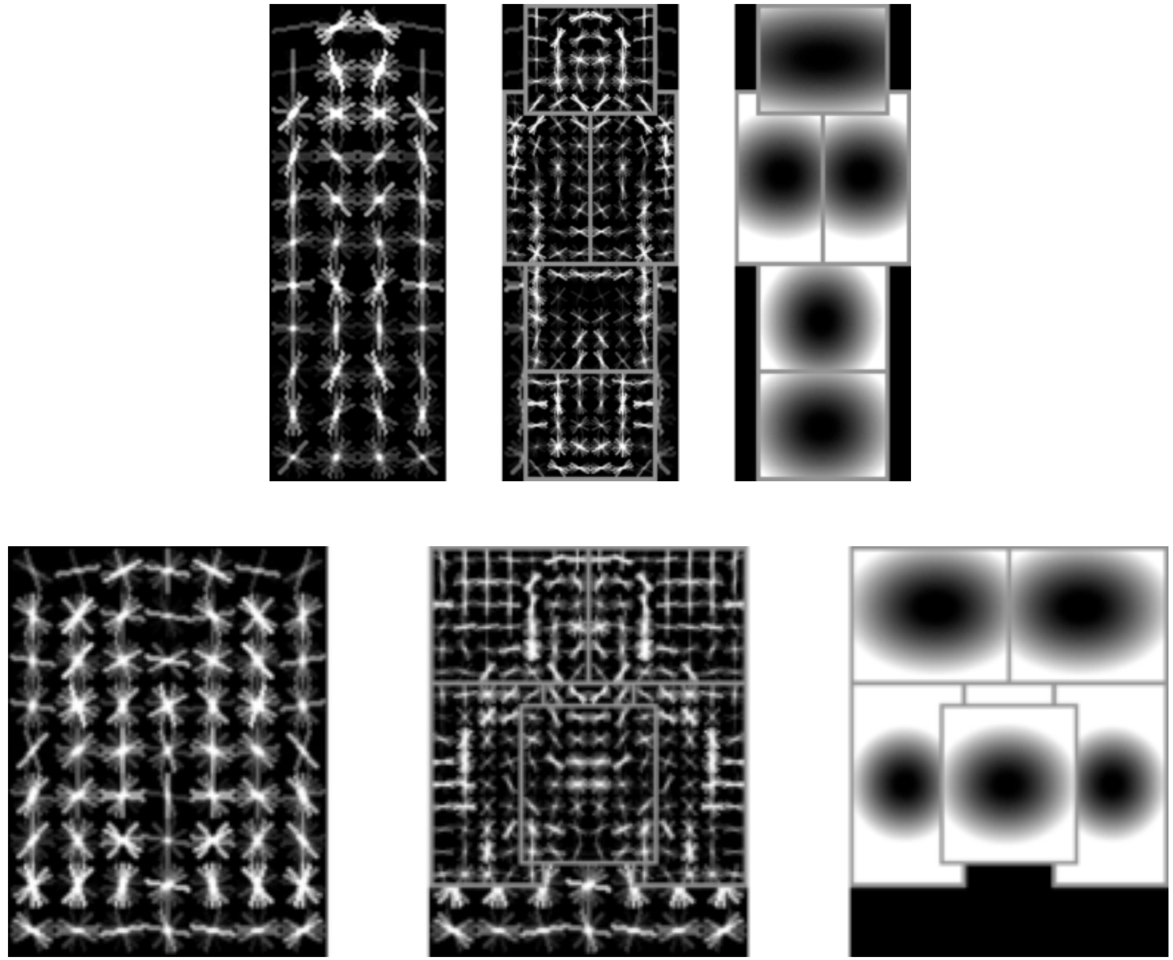


root filters
coarse resolution

part filters
finer resolution

deformation
models

Person model

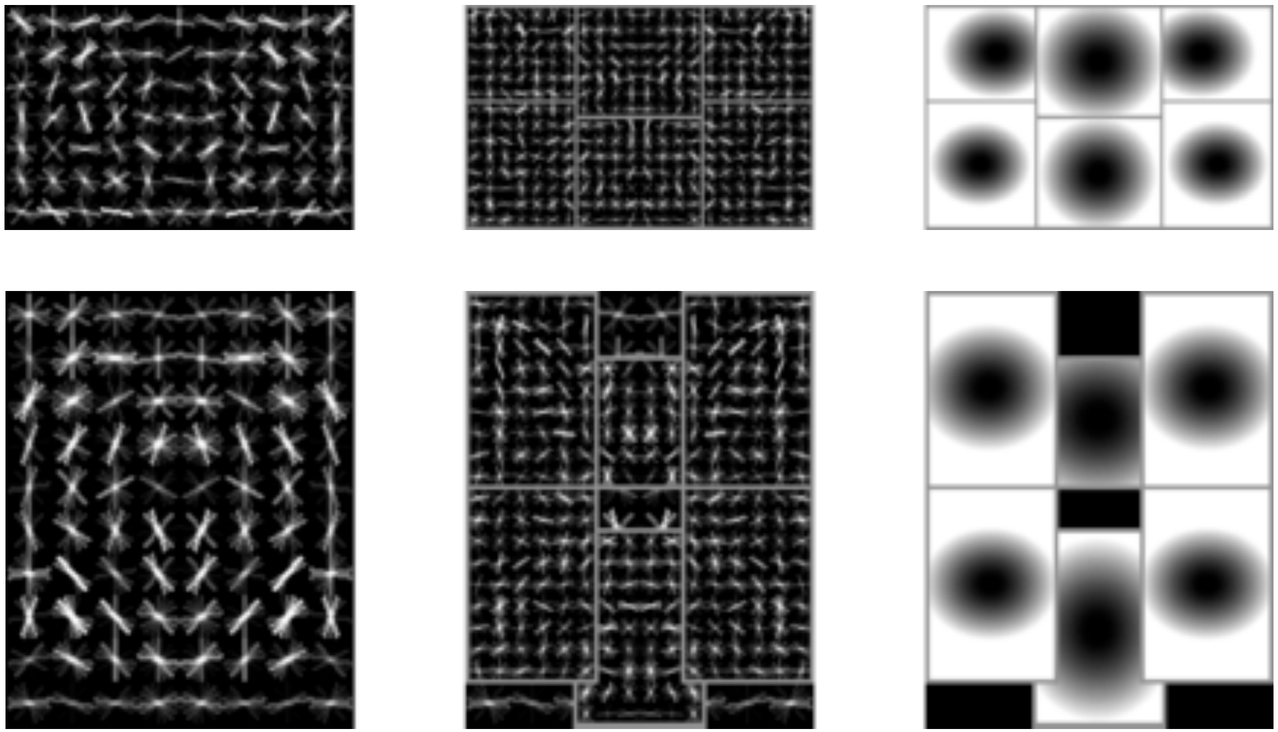


root filters
coarse resolution

part filters
finer resolution

deformation
models

Cat model

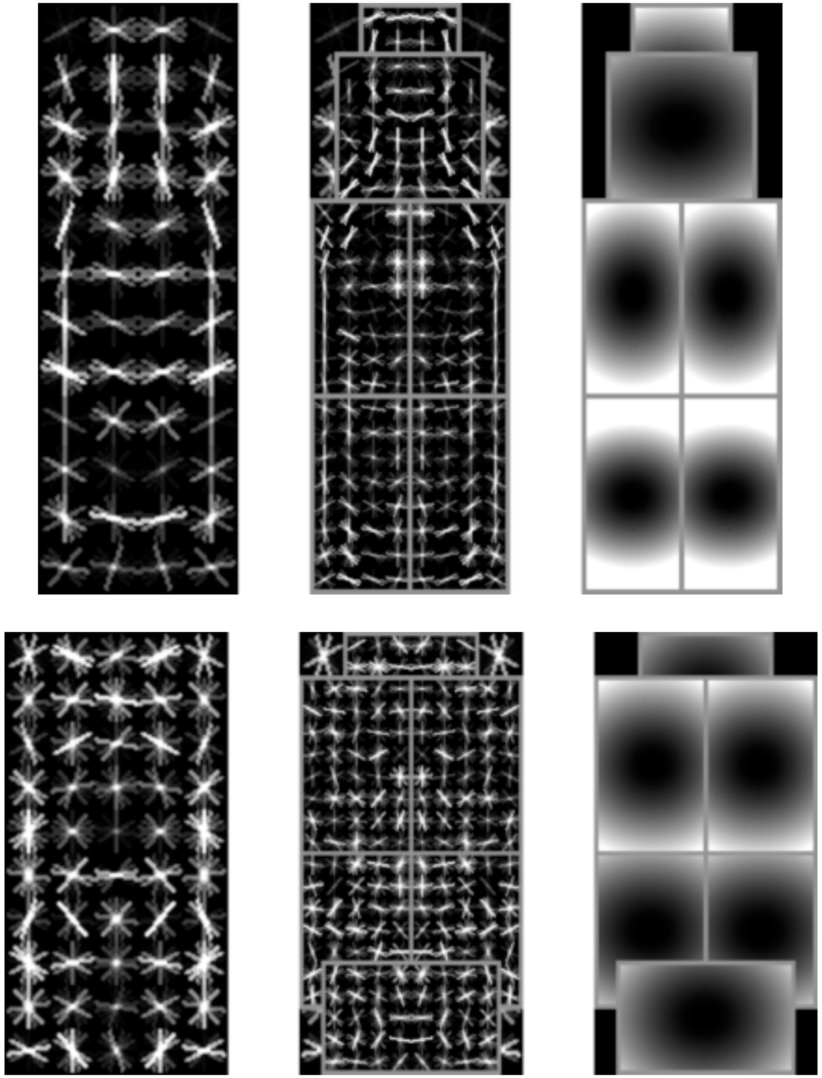


root filters
coarse resolution

part filters
finer resolution

deformation
models

Bottle model



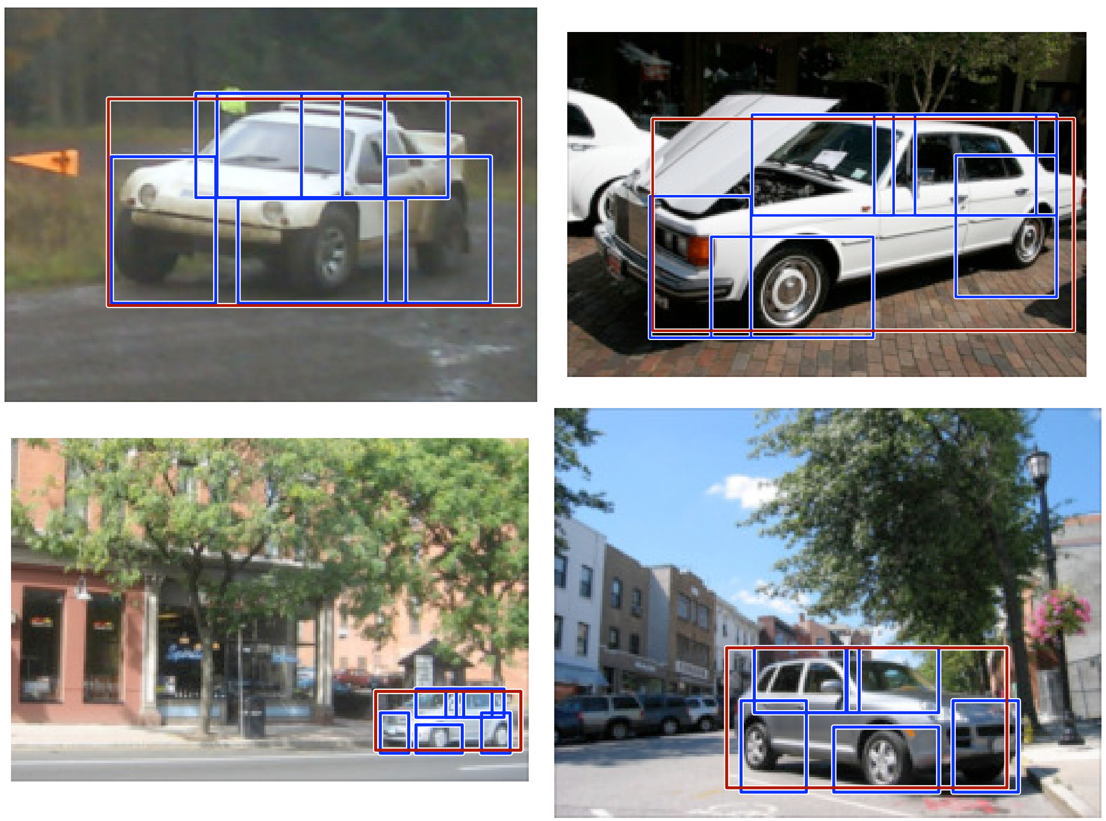
root filters
coarse resolution

part filters
finer resolution

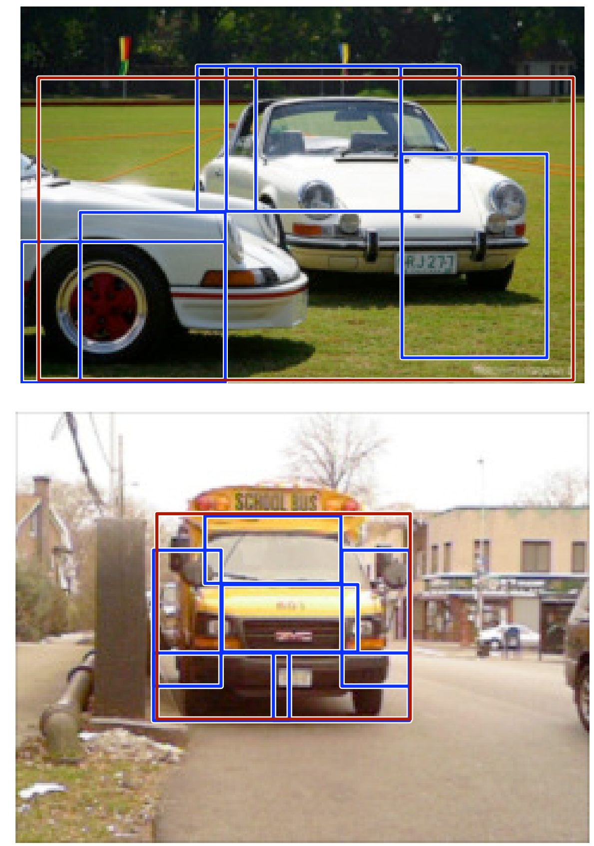
deformation
models

Car detections

high scoring true positives

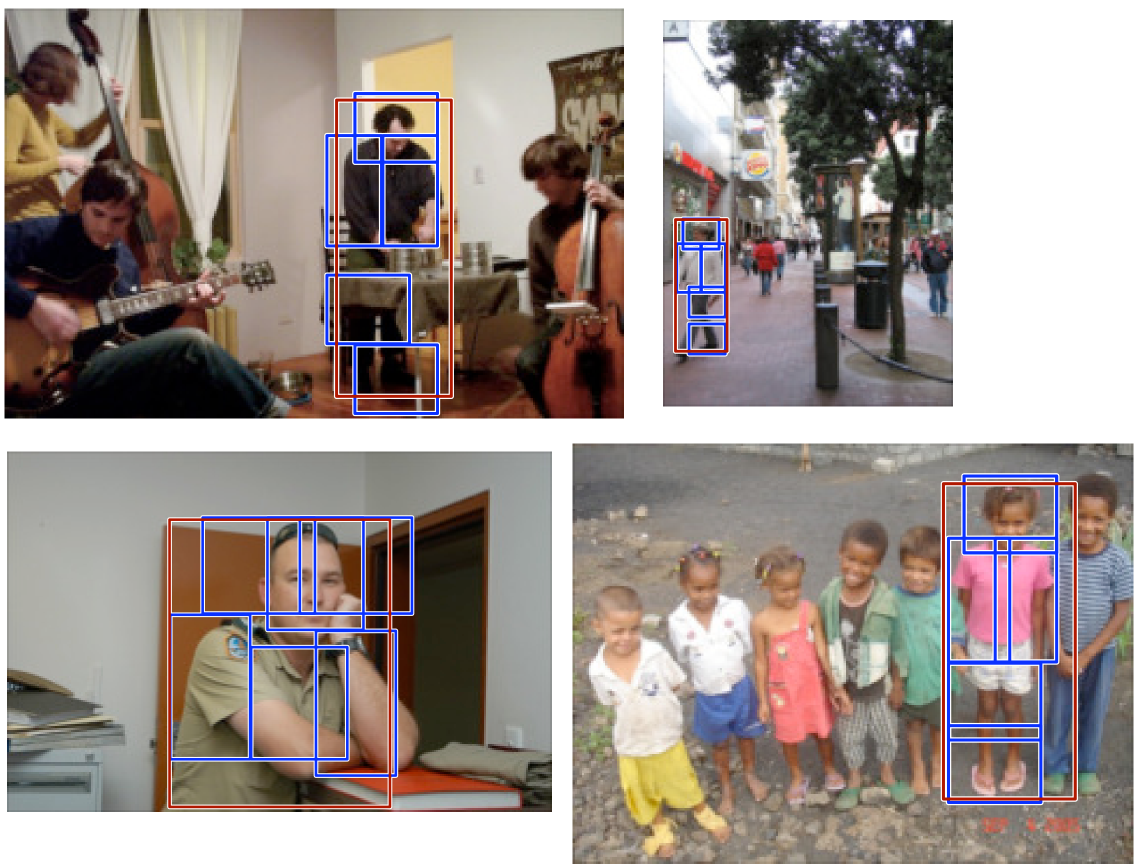


high scoring false positives

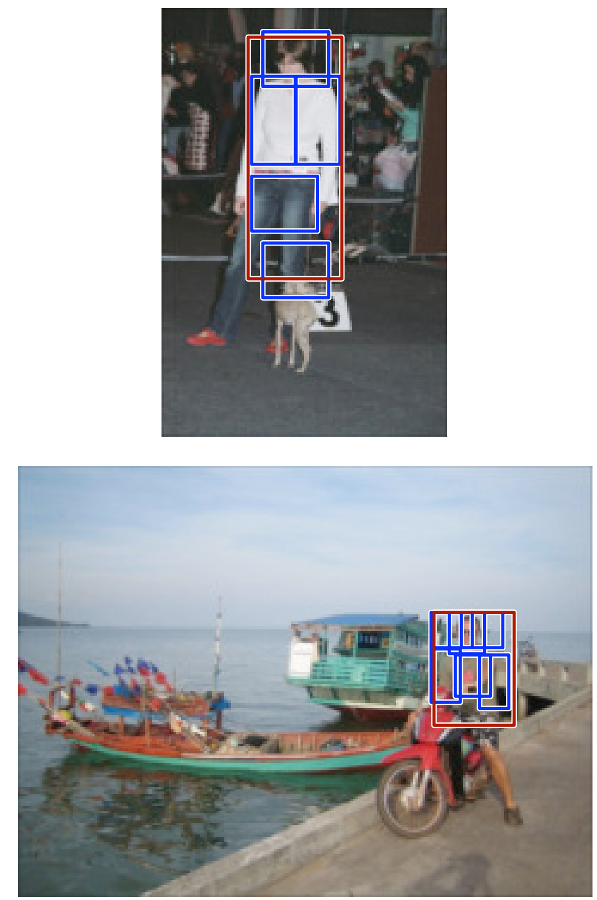


Person detections

high scoring true positives

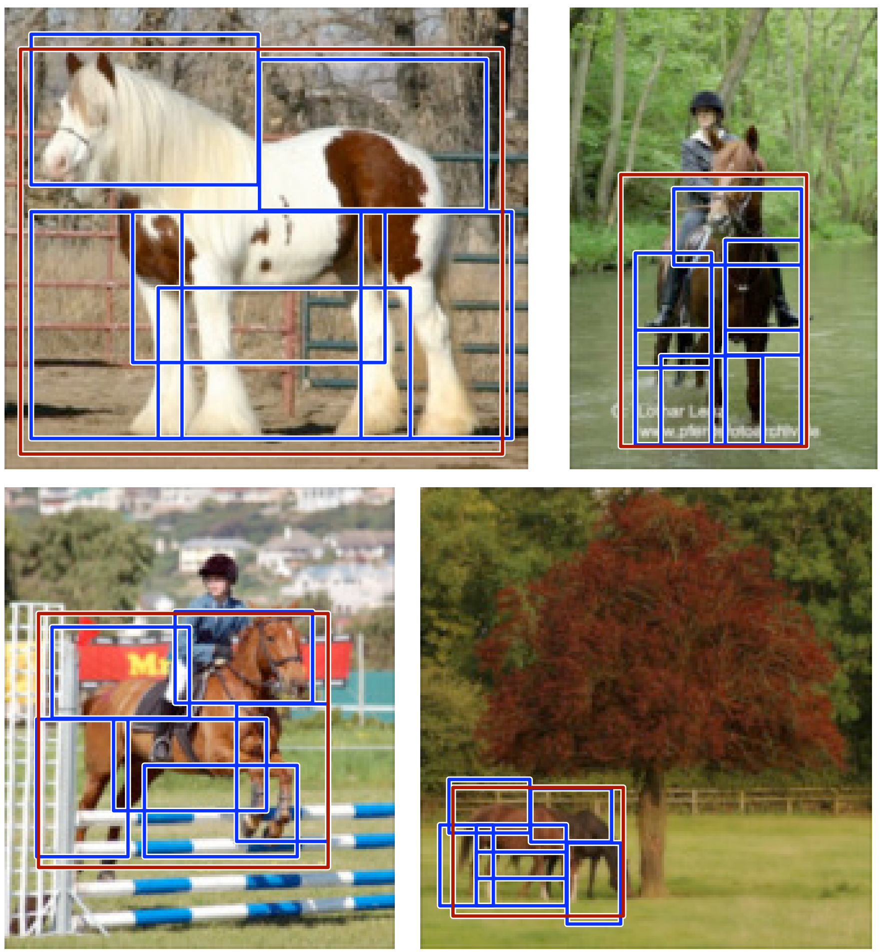


high scoring false positives
(not enough overlap)

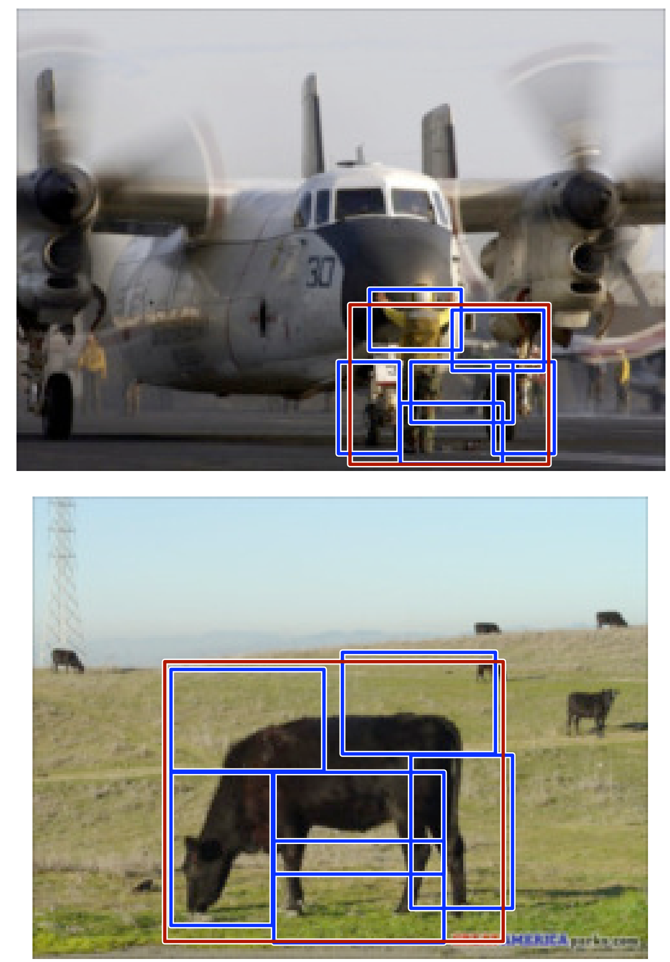


Horse detections

high scoring true positives

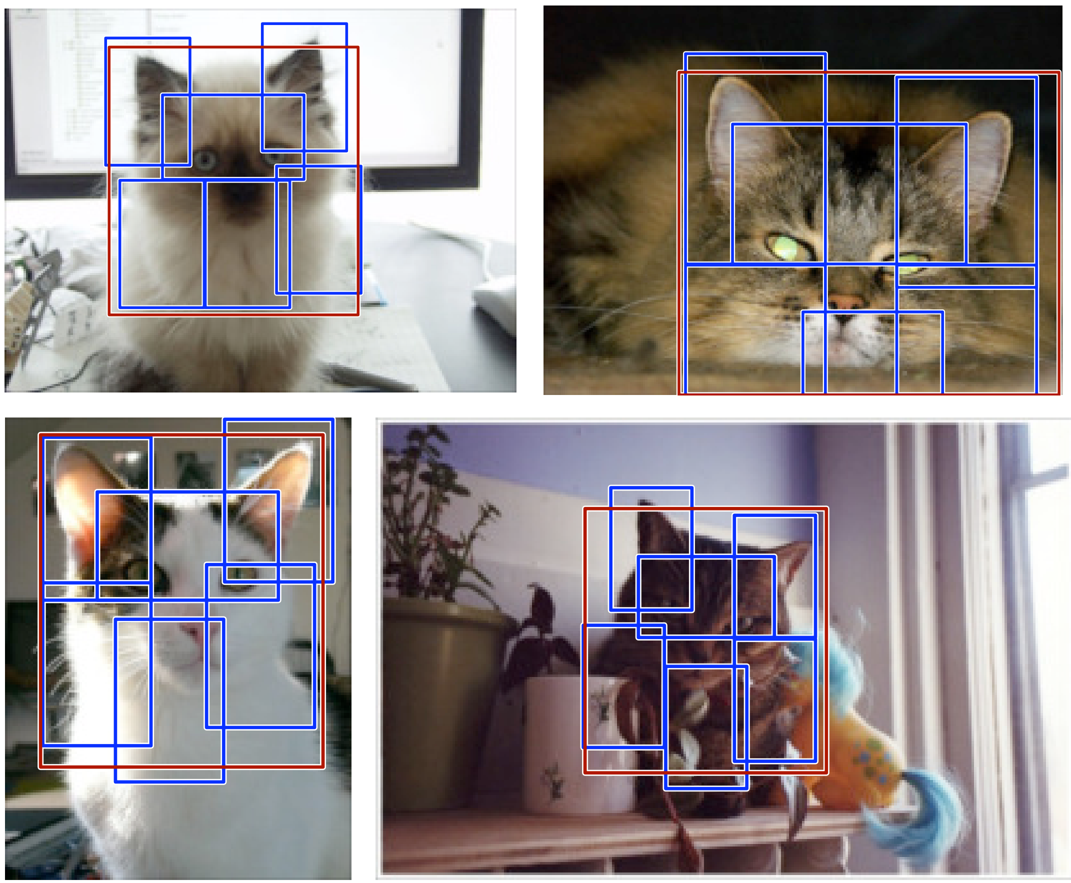


high scoring false positives

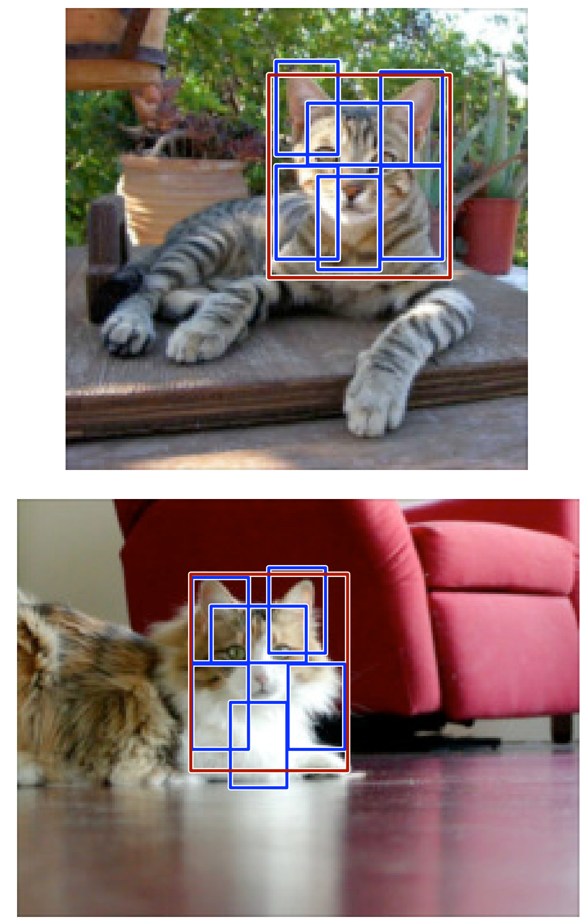


Cat detections

high scoring true positives



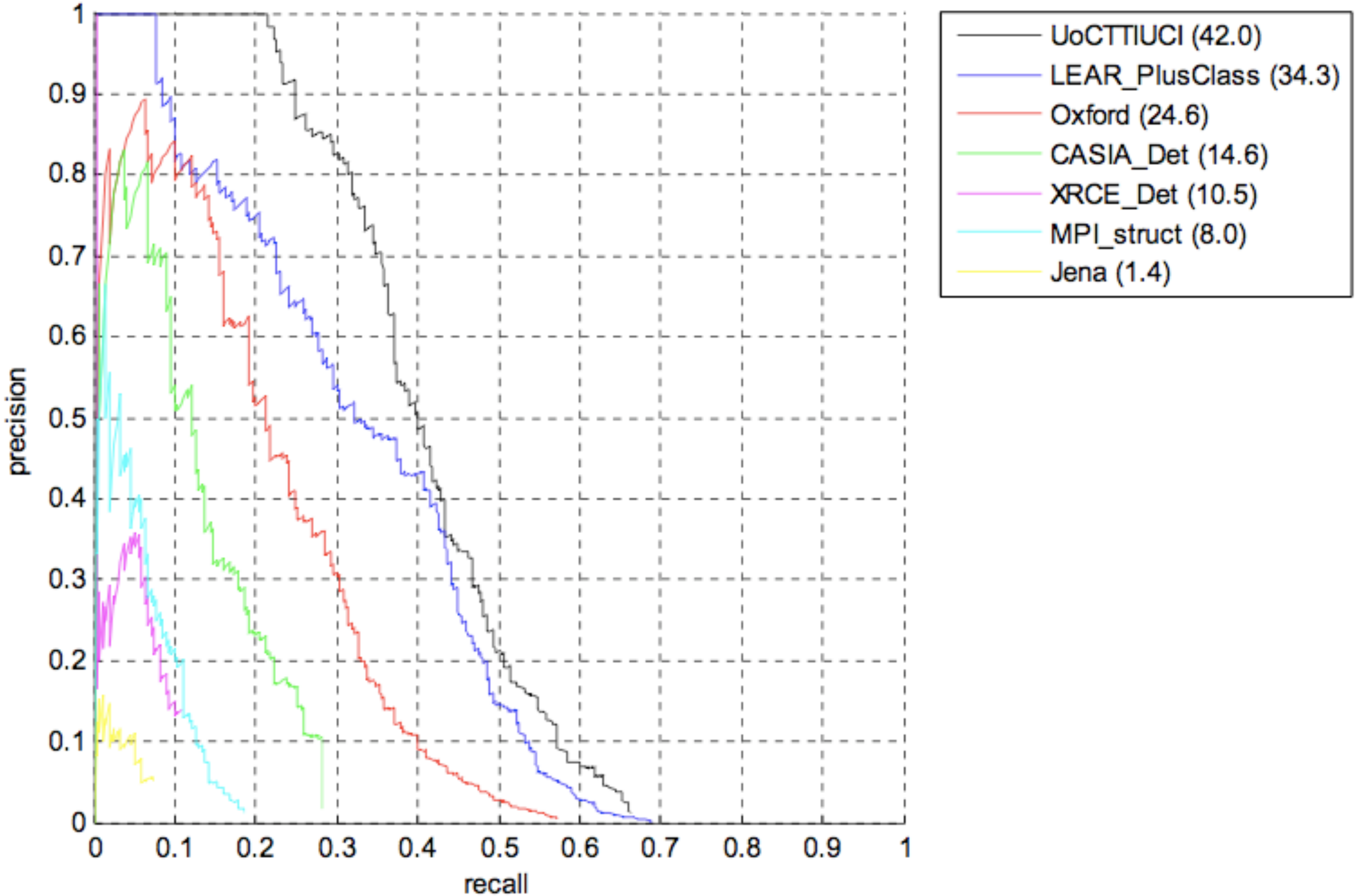
high scoring false positives
(not enough overlap)



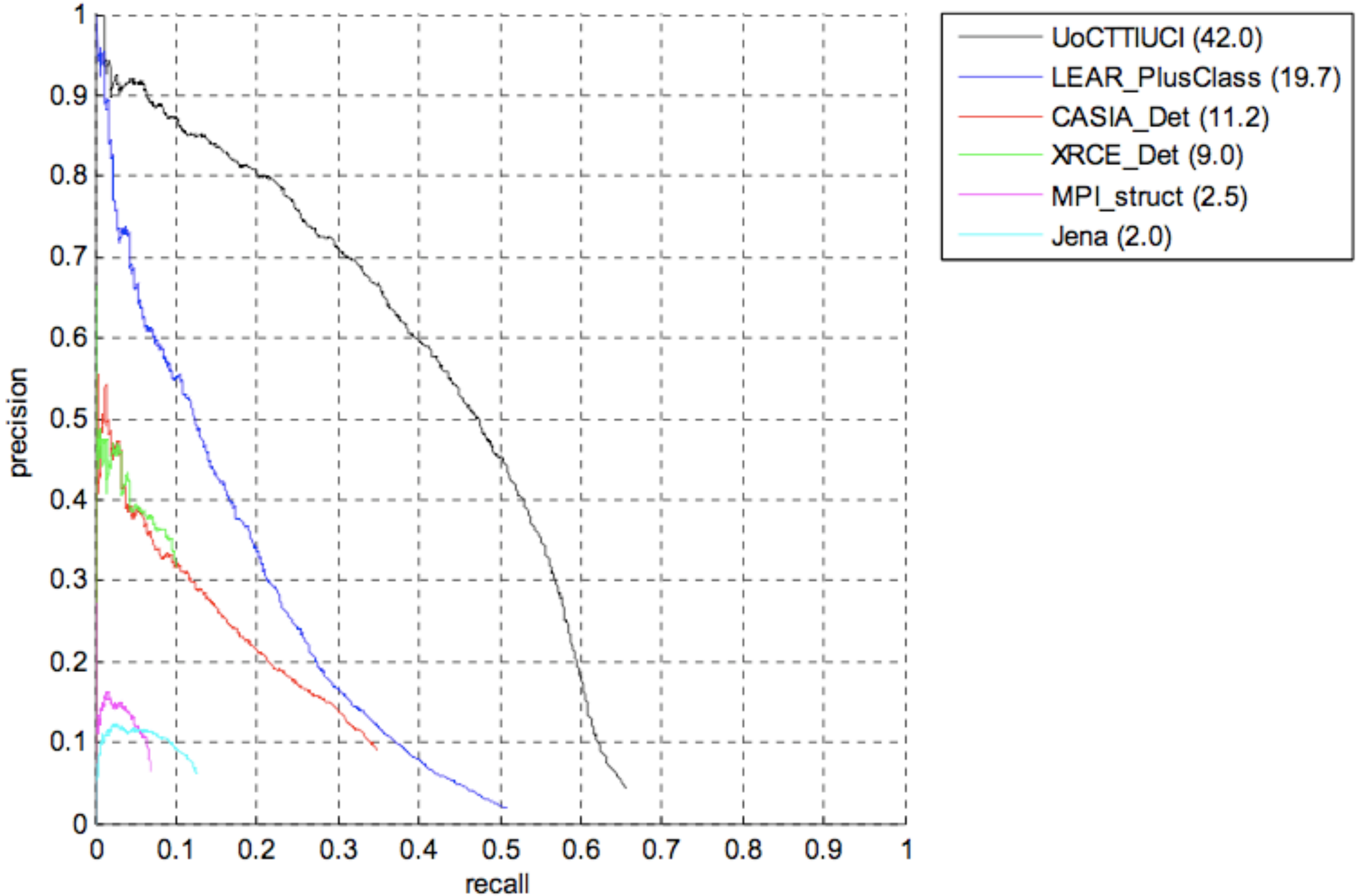
Quantitative results

- 7 systems competed in the 2008 challenge
- Out of 20 classes we got:
  - First place in 7 classes
  - Second place in 8 classes
- Some statistics:
  - It takes ~2 seconds to evaluate a model in one image
  - It takes ~4 hours to train a model
  - MUCH faster than most systems.

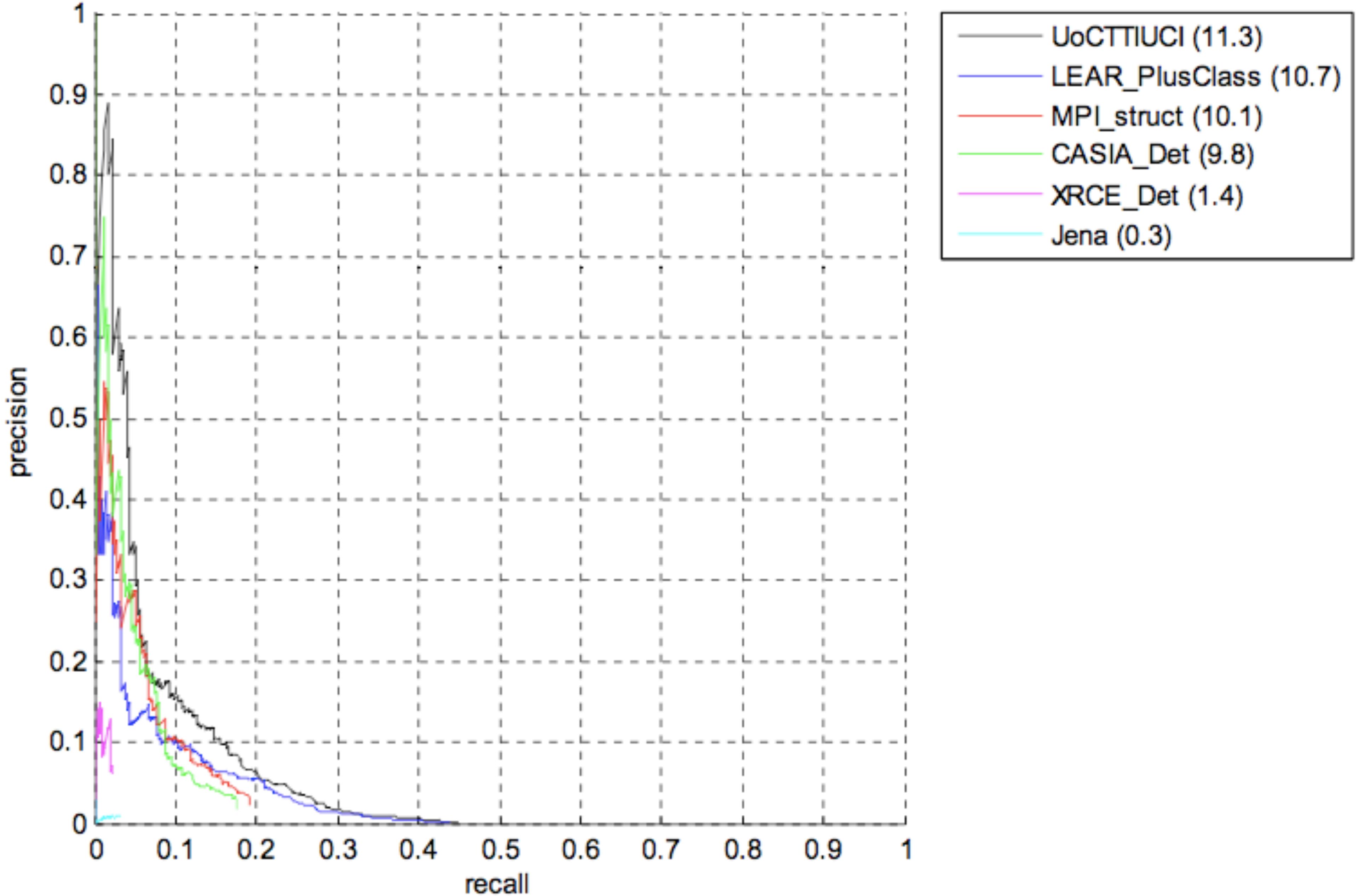
Precision/Recall results on Bicycles 2008



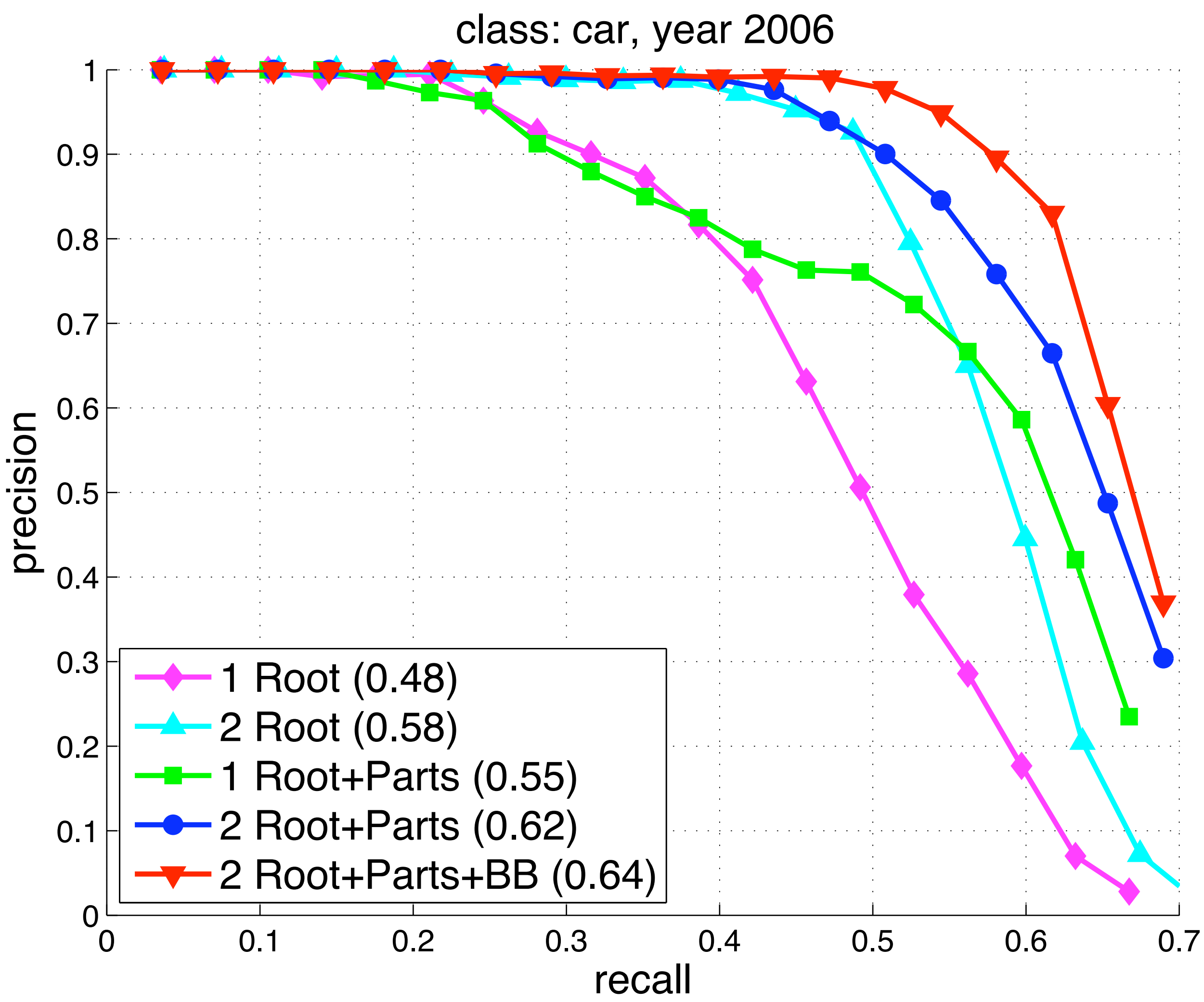
Precision/Recall results on Person 2008



Precision/Recall results on Bird 2008



Comparison of Car models on 2006 data



Summary

- Deformable models for object detection
  - Fast matching algorithms
  - Learning from weakly-labeled data
  - Leads to state-of-the-art results in PASCAL challenge
- Future work:
  - Hierarchical models
  - Visual grammars
  - AO* search (coarse-to-fine)