Automatic Object Detection

Jianping Fan
Dept of Computer Science
UNC-Charlotte

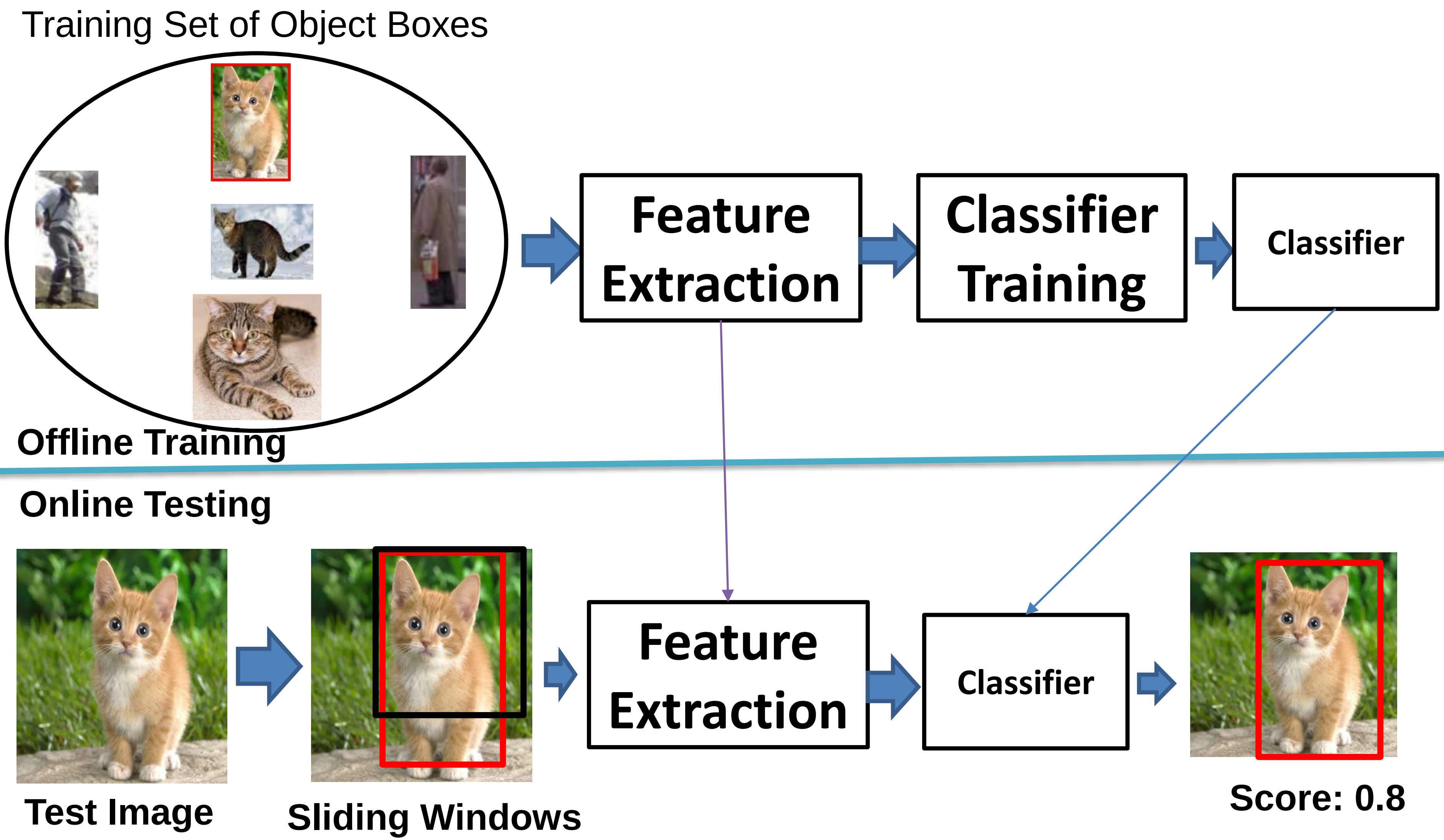
Course Website:
http://webpages.uncc.edu/jfan/itcs5152.html

Object Detection

- Overview
- Viola-Jones
- Dalal-Triggs

- Later classes:
  - Deformable models
  - Deep learning

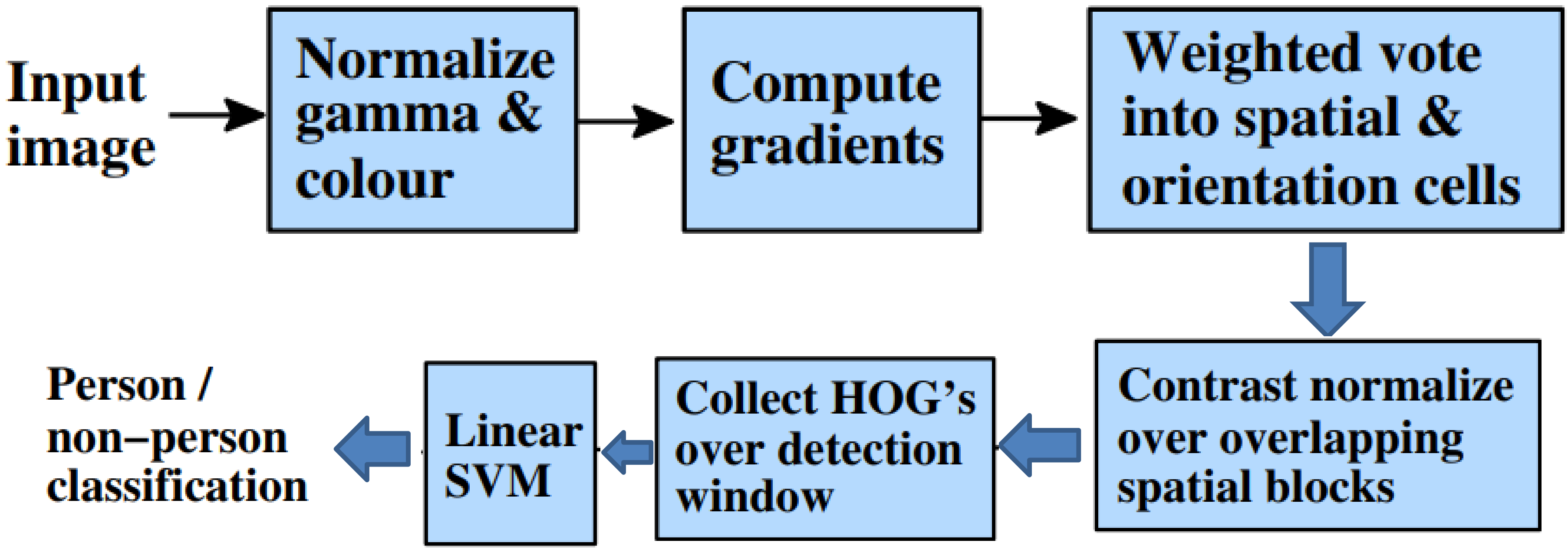
Pipeline for Object Detection System



Person detection with HoG's & linear SVM's

- Histograms of Oriented Gradients for Human Detection, Navneet Dalal, Bill Triggs, International Conference on Computer Vision & Pattern Recognition - June 2005
- http://lear.inrialpes.fr/pubs/2005/DT05/

Person Detection via HOG & SVM

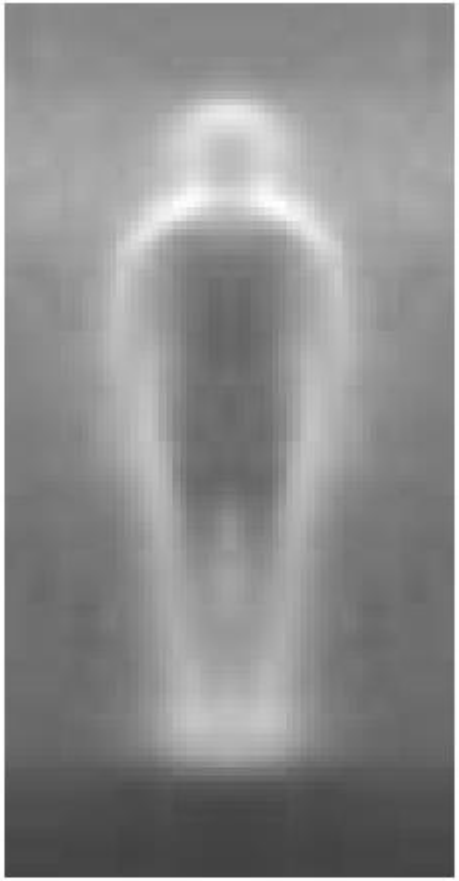


Grids of Histograms of Oriented Gradient (HOG) Descriptors

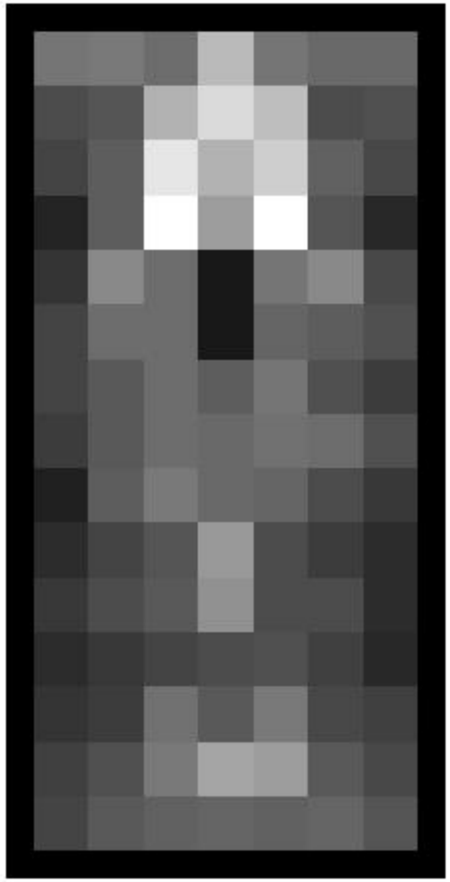
fine-scale gradients, fine orientation binning, relatively coarse spatial binning, and high-quality local contrast normalization in overlapping descriptor blocks are all important for good results.

R-HOG: HOG from Rectangular Cells
C-HOG: HOG from Circular Cells

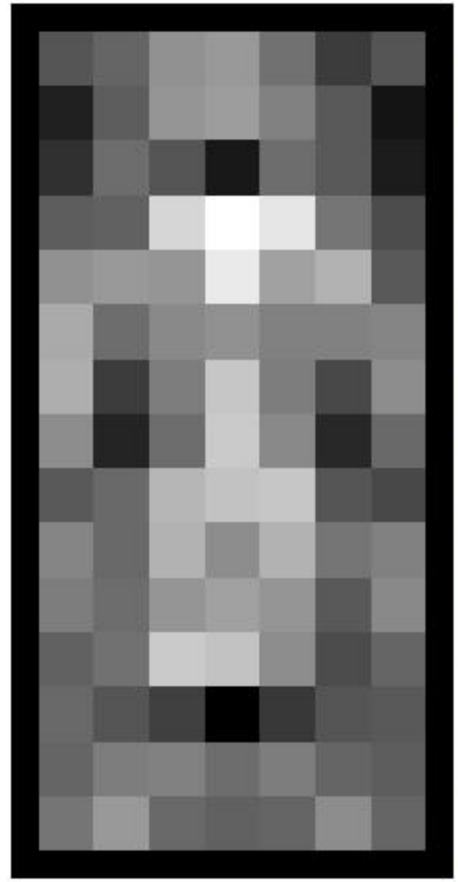
HOG Features



(a) average gradient image over all the training examples



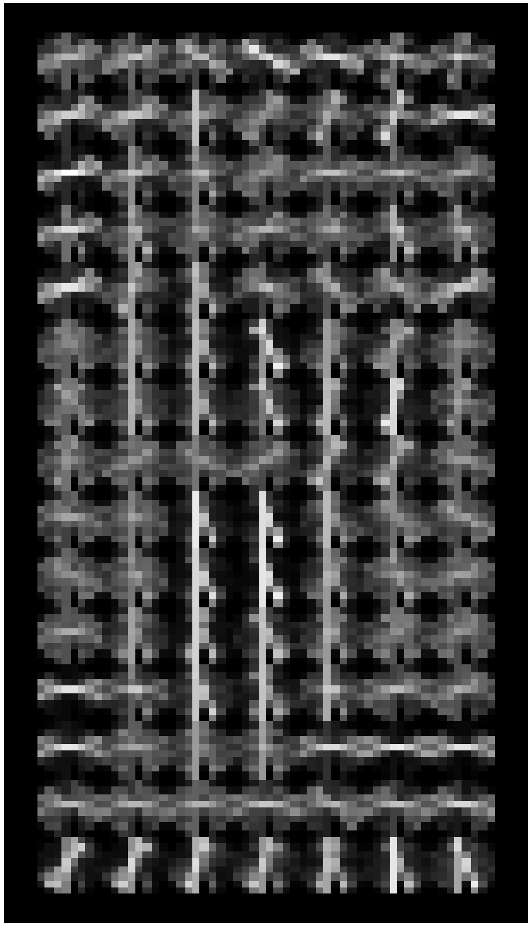
(b) Each "pixel" shows maximum positive SVM weight in pixel-centered block



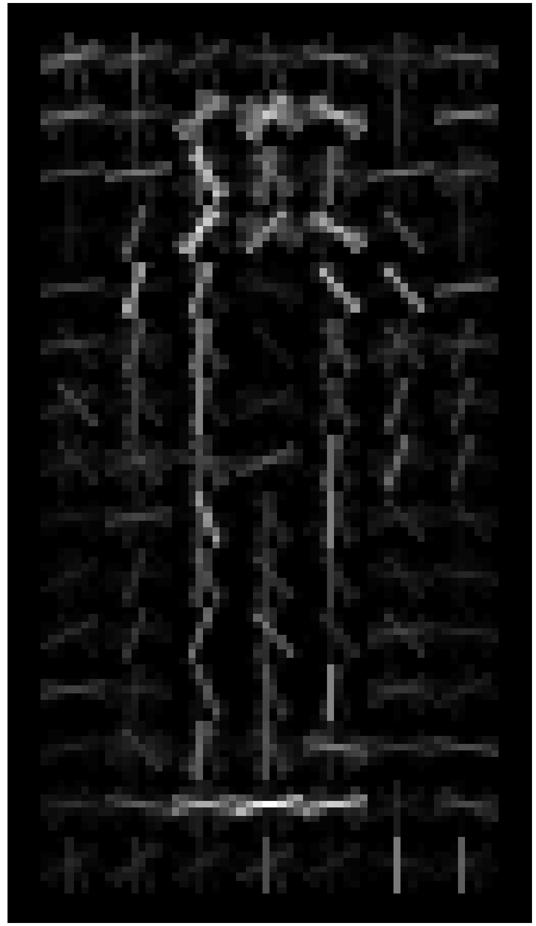
© Each "pixel" shows maximum negative SVM weight in pixel centered block

(d) A test image

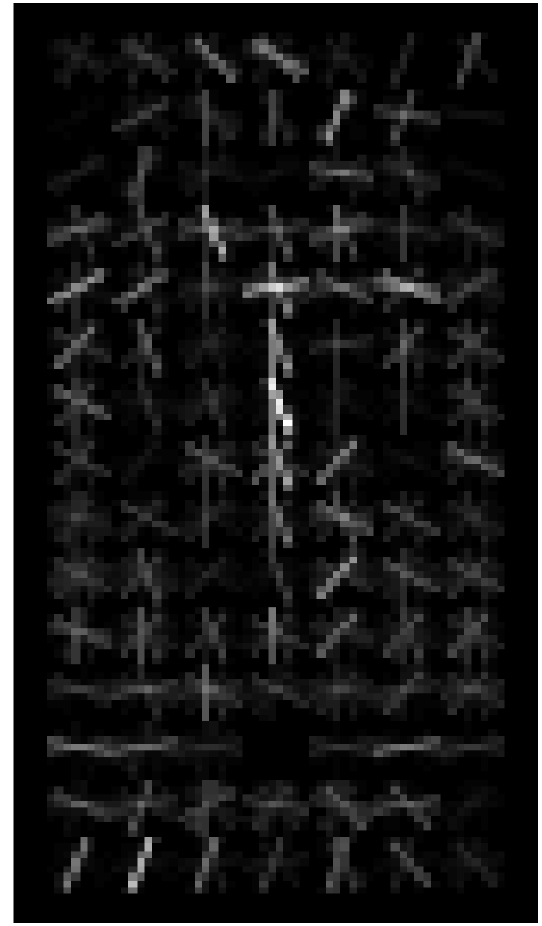
HOG Features



(e) computed R-HOG descriptor of test image



(f) R-HOG descriptor weighted by the positive SVM weights



(g) R-HOG descriptor weighted by the negative SVM weights

Object Detection *vs.* Scene Recognition

- What's the difference?

- Objects (even if deformable and articulated) probably have more consistent shapes than scenes.

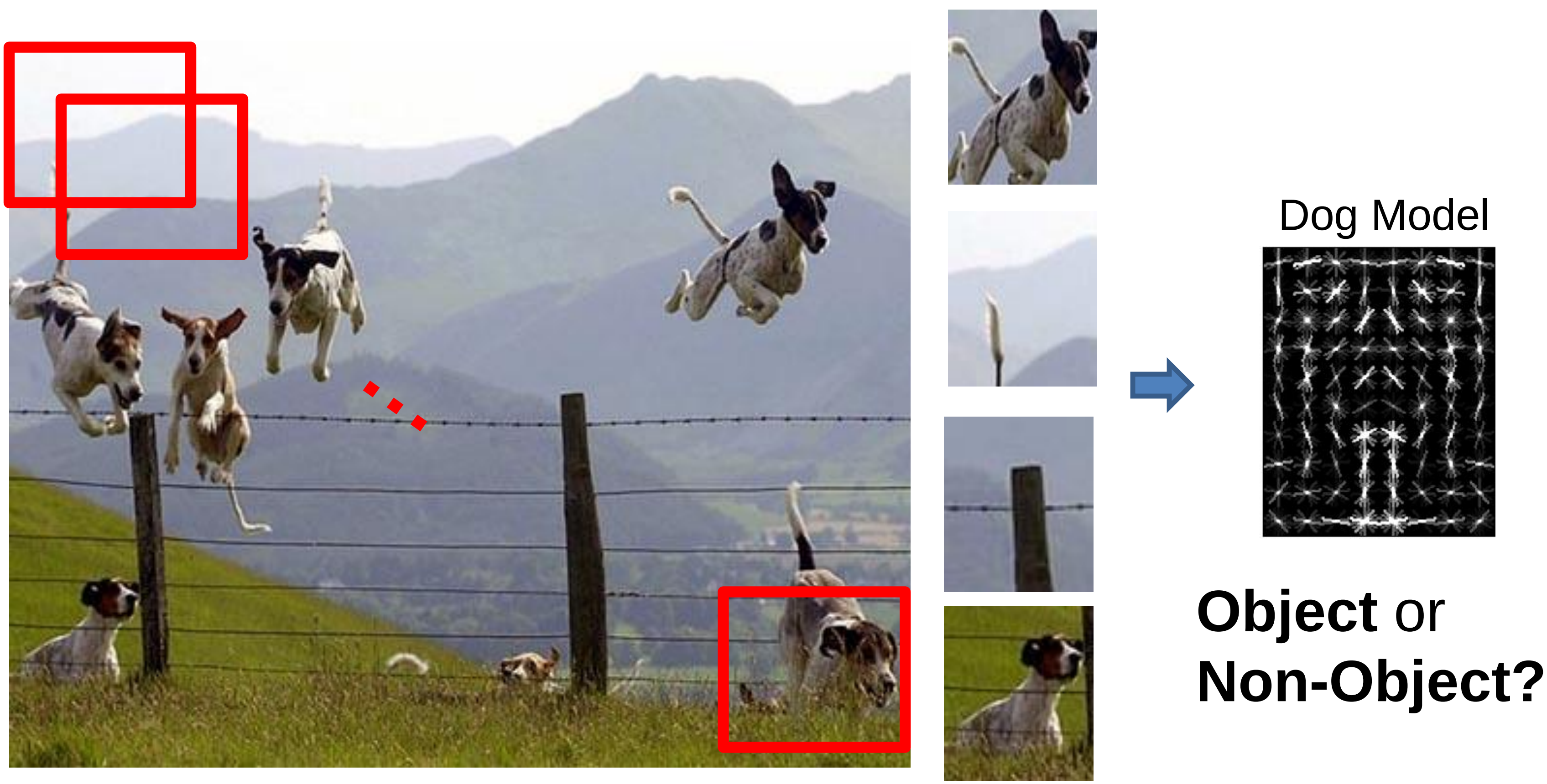
- Scenes can be defined by **distribution of "stuff"** – materials and surfaces with arbitrary shape.

- Objects are "things" that own their boundaries

- Bag of words models were less popular for object detection because they throw away shape info.

Object Category Detection

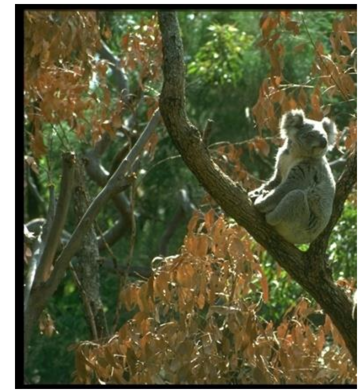
- Focus on object search: “Where is it?”
- Build templates that quickly differentiate object patch from background patch



Challenges in modeling the object class

Illumination

Object pose



Clutter

Occlusions

Intra-class appearance

Viewpoint

Slide from K. Grauman, B. Leibe

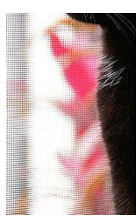
Challenges in modeling the non-object class

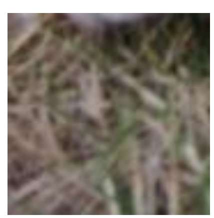
True Detections

Bad Localization

Confused with Similar Object

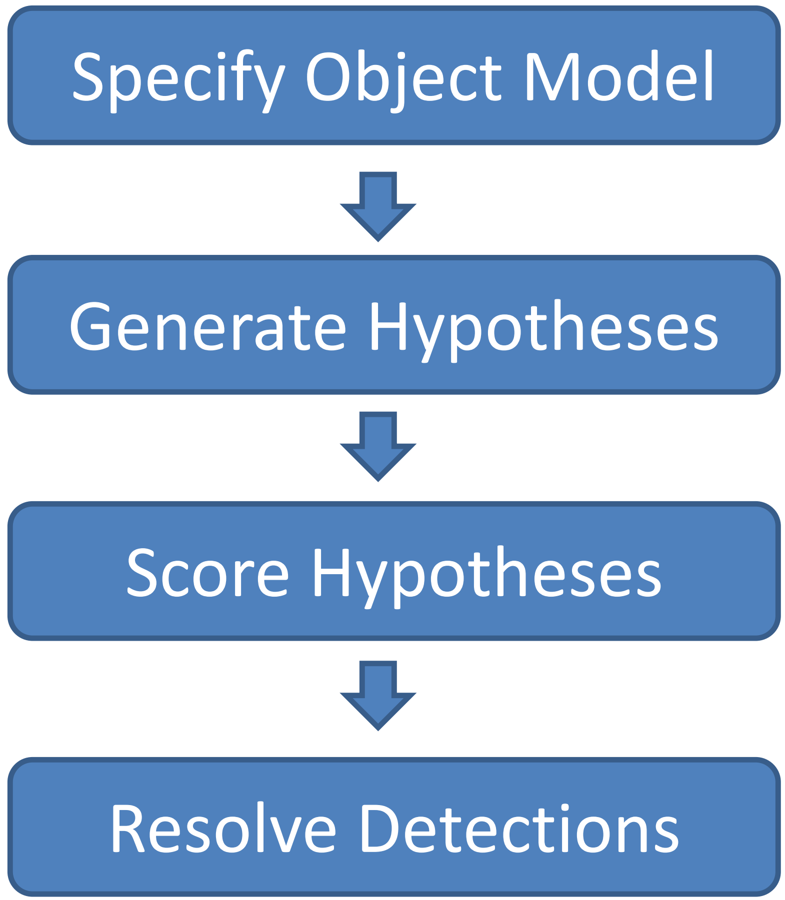
Misc. Background





Confused with Dissimilar Objects

General Process of Object Recognition



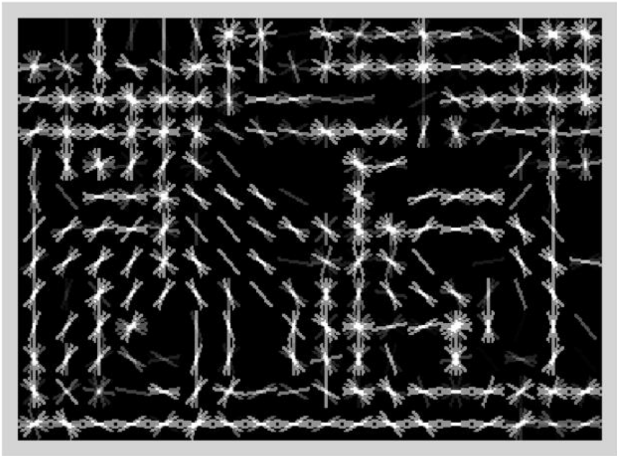
What are the object parameters?

Specifying an object model

1. Statistical Template in Bounding Box

- Object is some (x,y,w,h) in image
- Features defined wrt bounding box coordinates

Image

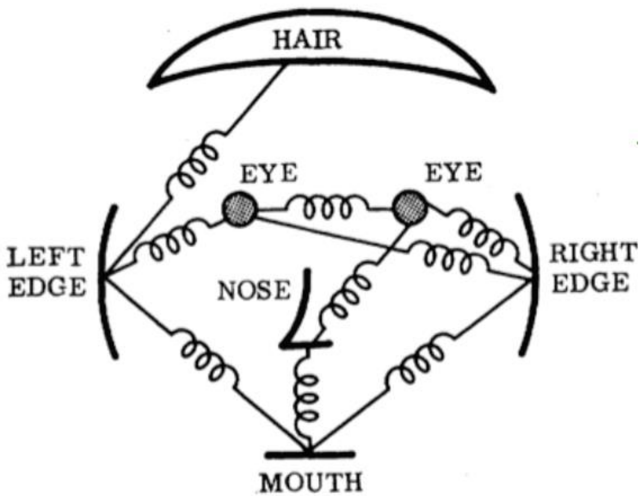


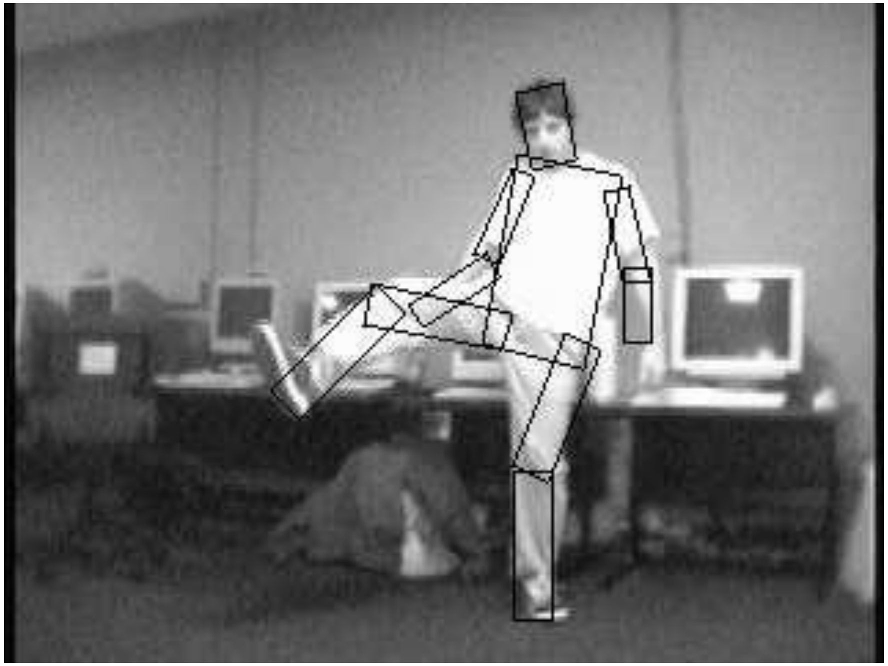
Template Visualization

Images from Felzenszwalb

Specifying an object model

2. **Articulated parts model**
   – Object is configuration of parts
   – Each part is detectable



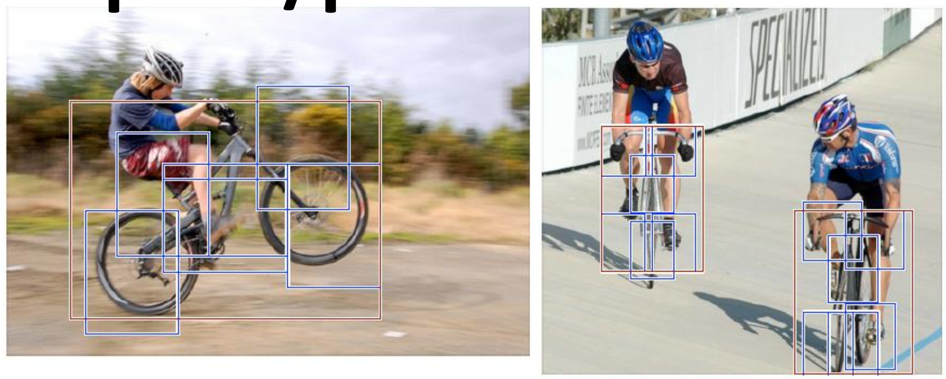


Images from Felzenszwalb

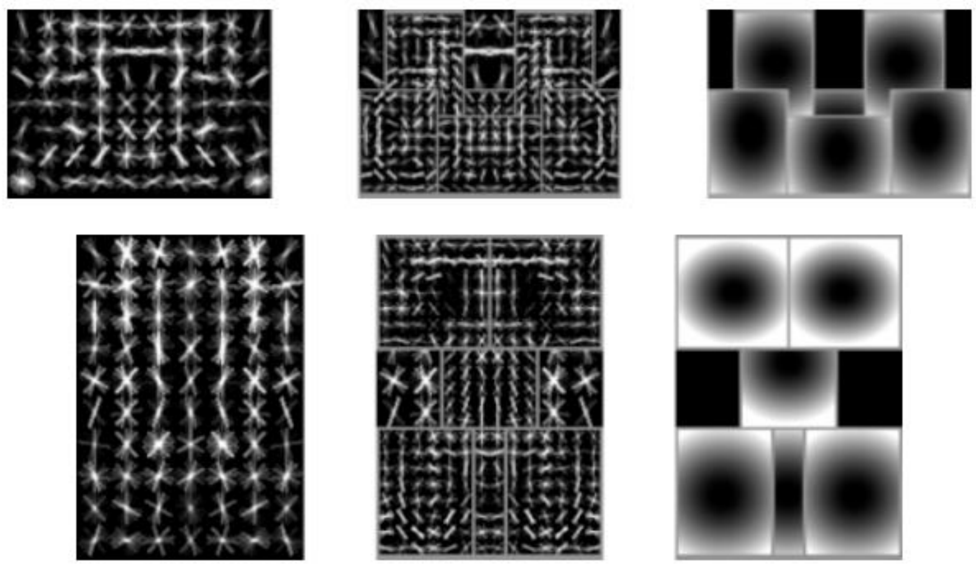
Specifying an object model

3. Hybrid template/parts model

Detections



Template Visualization



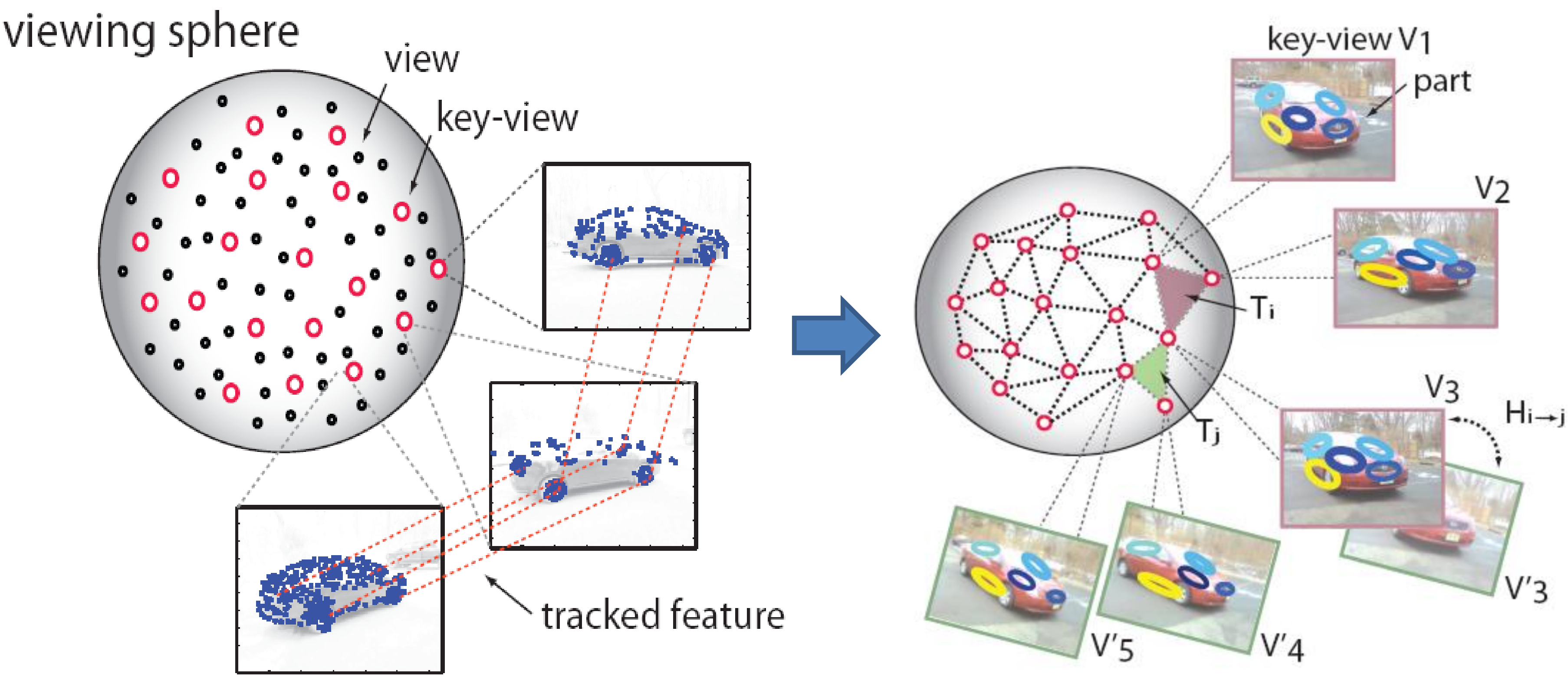
root filters coarse resolution | part filters finer resolution | deformation models

Felzenszwalb et al. 2008

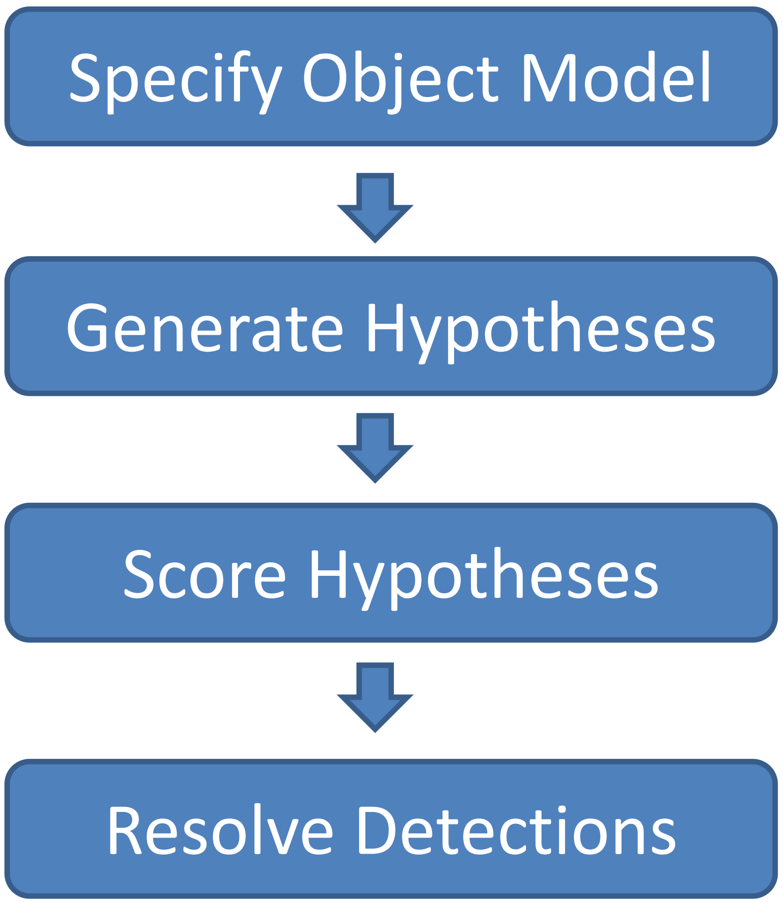
Specifying an object model

4. 3D-ish model

- Object is collection of 3D planar patches under affine transformation



General Process of Object Recognition



Propose an alignment of the model to the image

Generating hypotheses

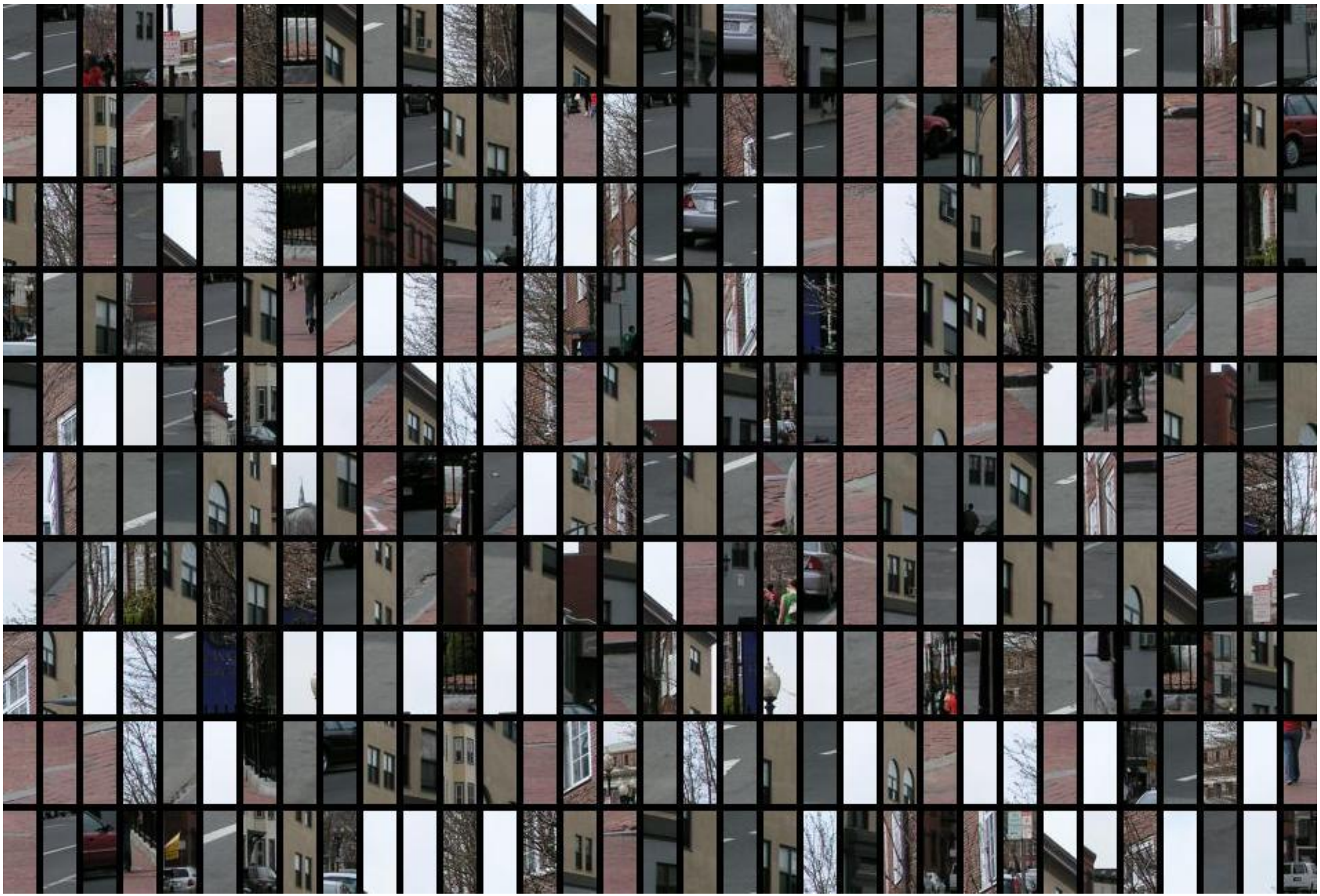
1. **Sliding window**
   – Test patch at each location and scale

Generating hypotheses

1. **Sliding window**
   – Test patch at each location and scale

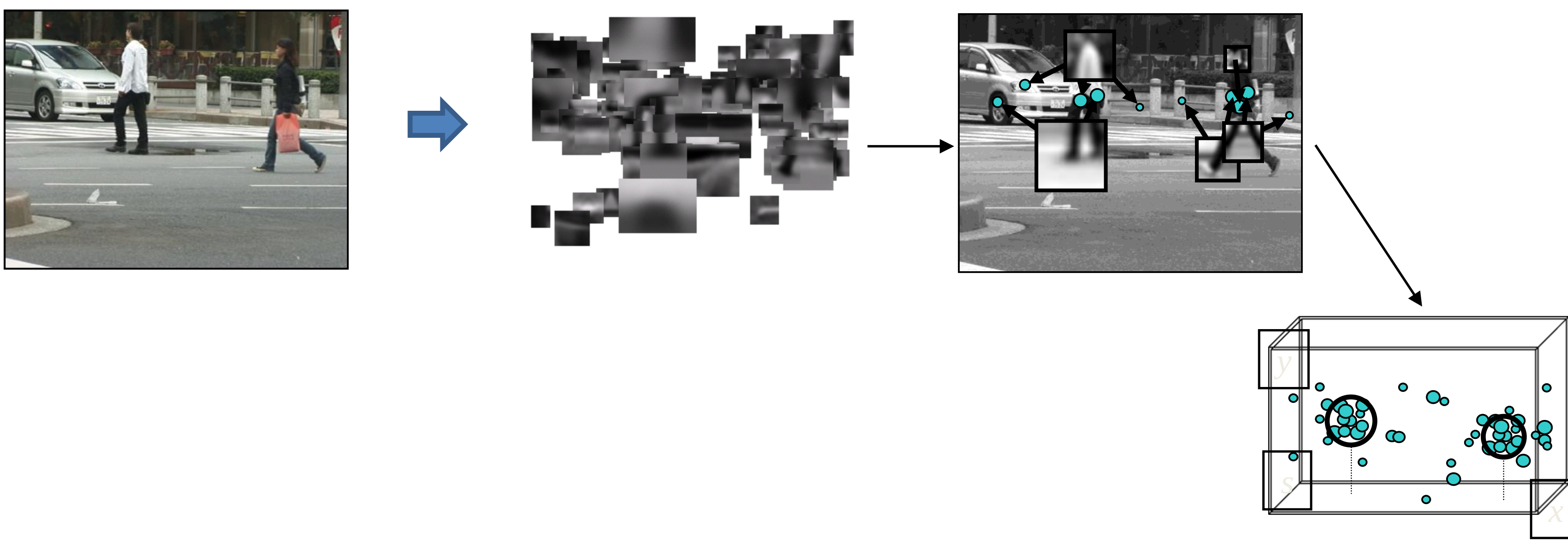
Note – Template did not change size

Each window is separately classified



Generating hypotheses

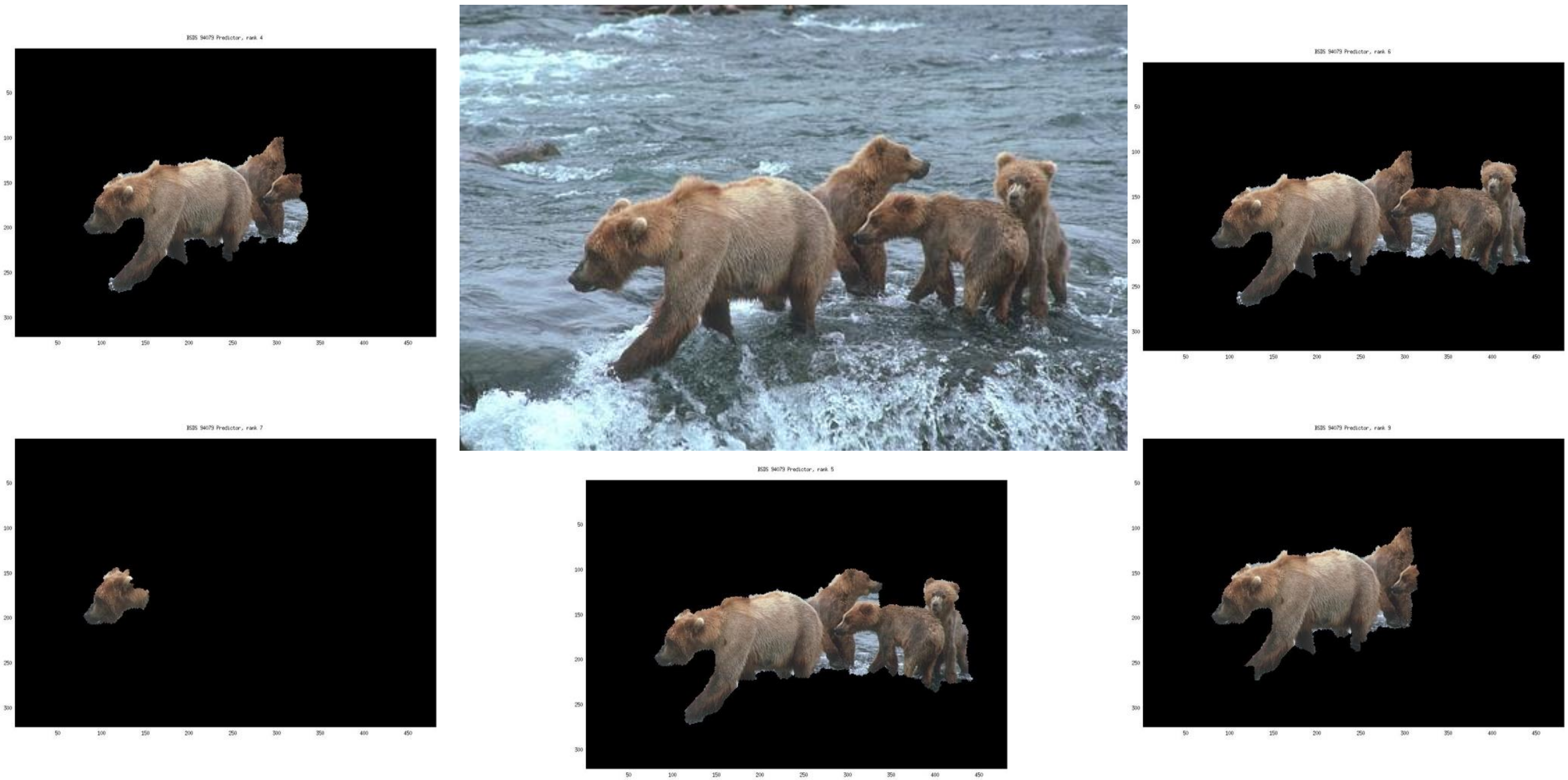
2. **Voting from patches/keypoints**



ISM model by Leibe et al.

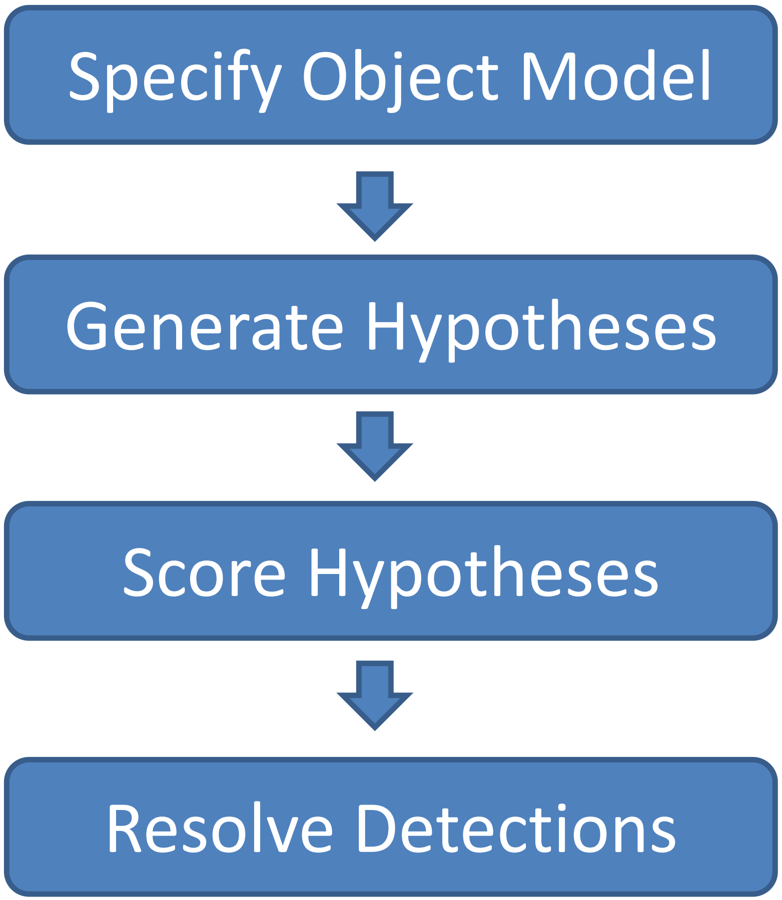
Generating hypotheses

3. **Region-based proposal**



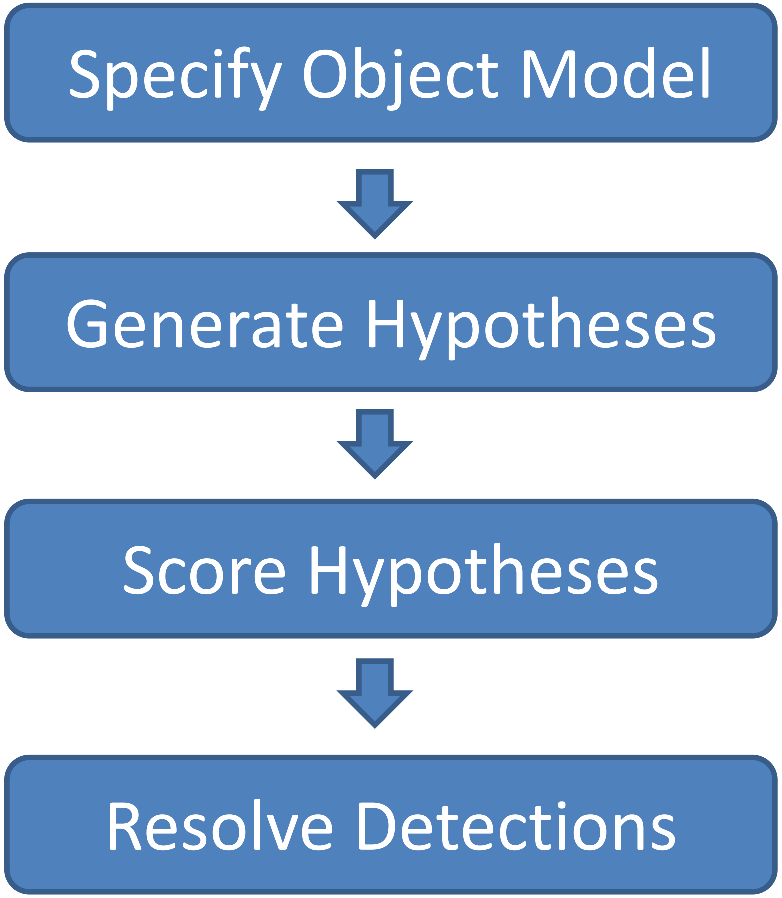
Endres Hoiem 2010

General Process of Object Recognition



Mainly-gradient based features, usually based on summary representation, many classifiers

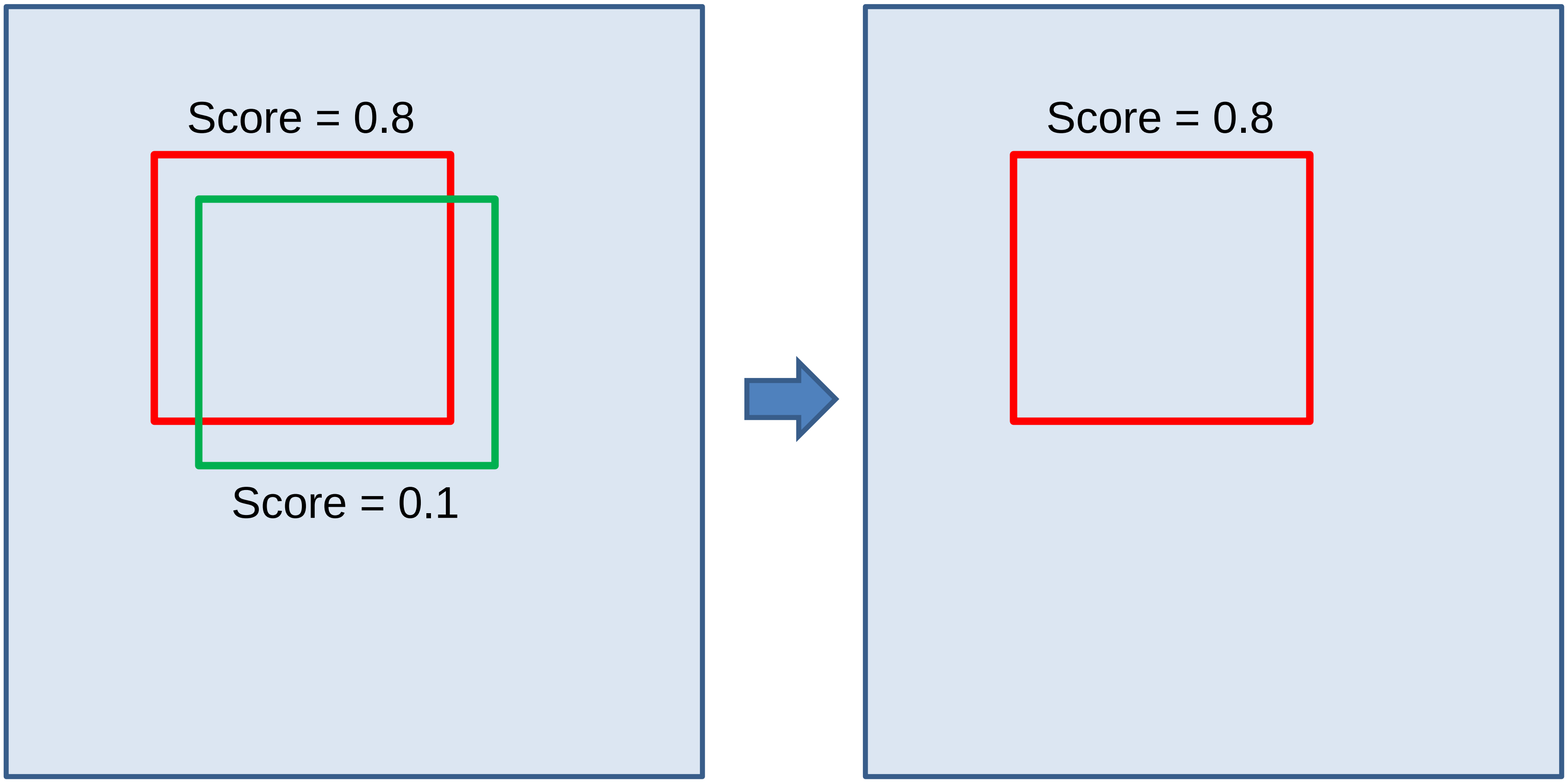
General Process of Object Recognition



Rescore each proposed object based on whole set

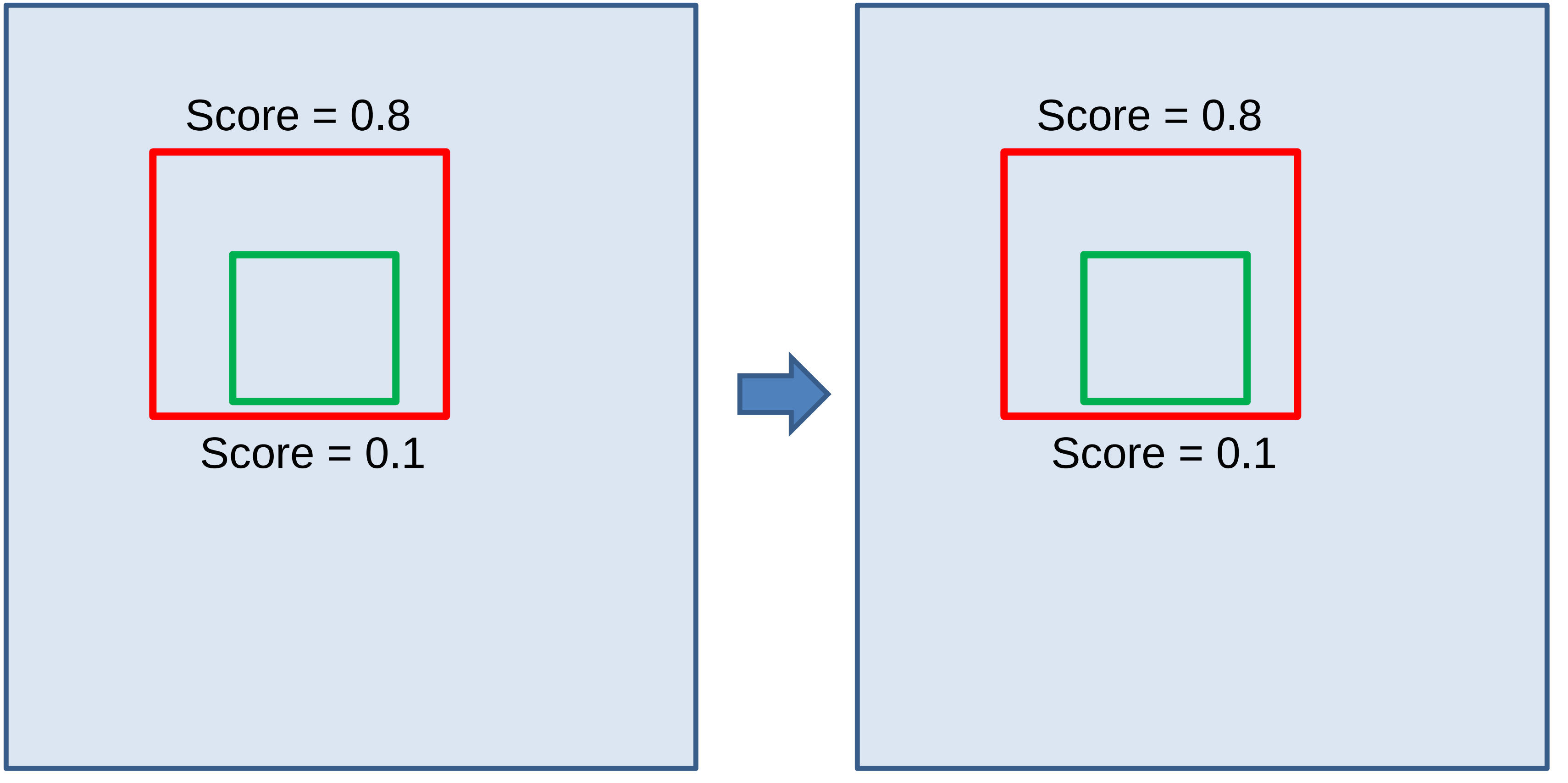
Resolving detection scores

1. Non-max suppression



Resolving detection scores

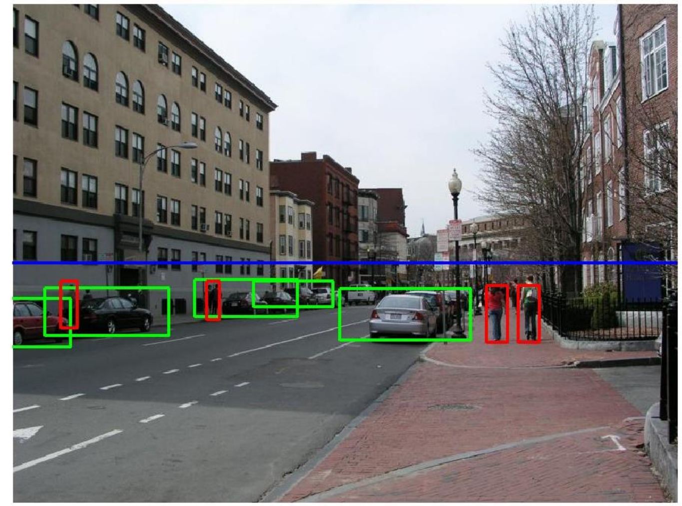
1. Non-max suppression

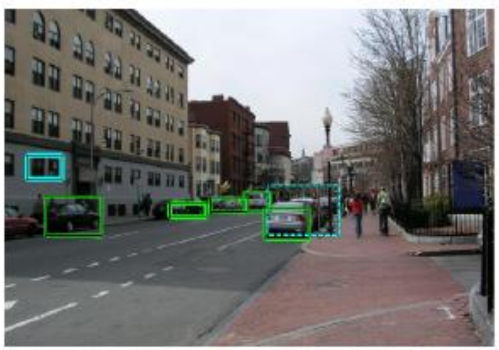


"Overlap" score is below some threshold

Resolving detection scores

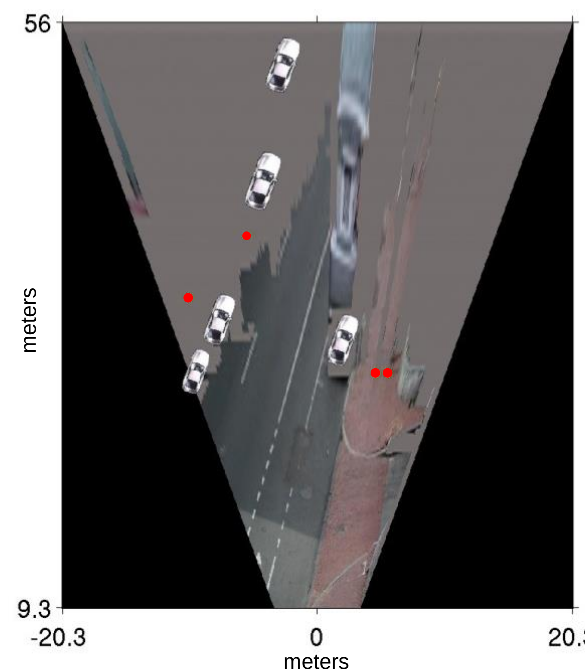
2. **Context/reasoning**





(g) Car Detections: Local

(h) Ped Detections: Local



Hoiem et al. 2006

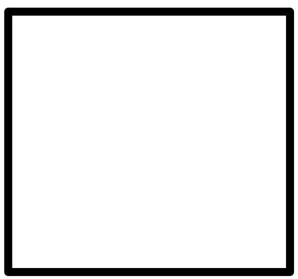
Bounding Box Regression (BBR)

We may have many “good” candidates

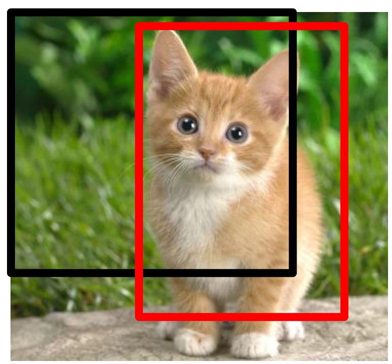
Many “good” candidates with close confidence scores

var voting

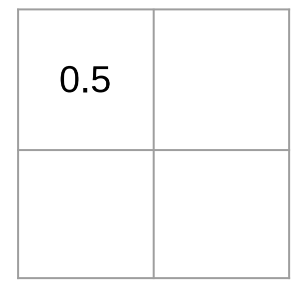
Sliding Window: Overfeat



Network input:
3 x 221 x 221



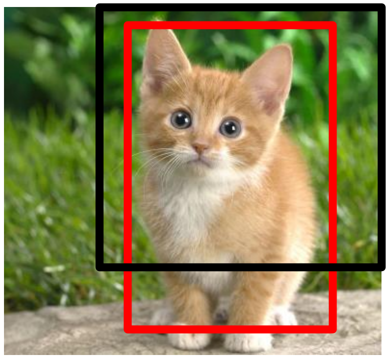
Larger image:
3 x 257 x 257



Classification scores:
P(cat)

Sliding Window: Overfeat

Network input:
3 x 221 x 221

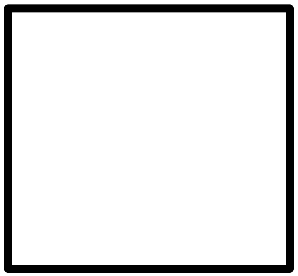


Larger image:
3 x 257 x 257

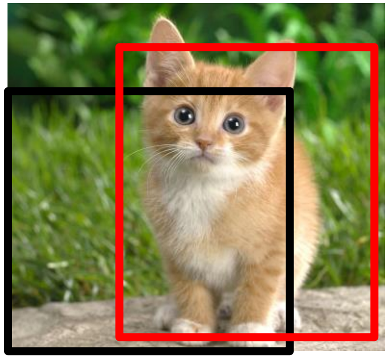
0.5	0.75

Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221

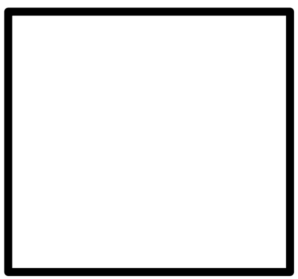


Larger image:
3 x 257 x 257

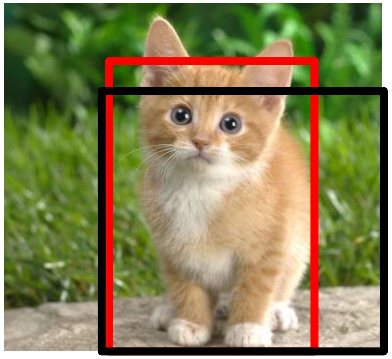
0.5	0.75
0.6	

Classification scores:
P(cat)

Sliding Window: Overfeat



Network input:
3 x 221 x 221



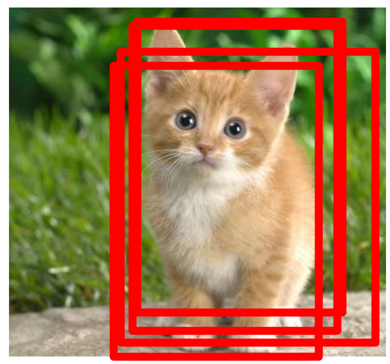
Larger image:
3 x 257 x 257

0.5	0.75
0.6	0.8

Classification scores:
P(cat)

Sliding Window: Overfeat

Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

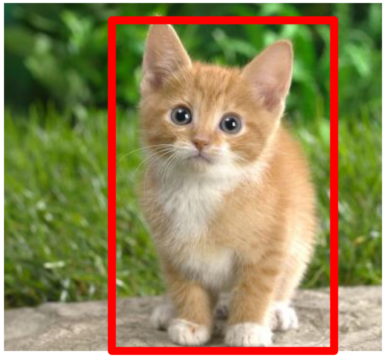
0.5	0.75
0.6	0.8

Classification scores:
P(cat)

Sliding Window: Overfeat

Greedily merge boxes and scores (details in paper)

Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

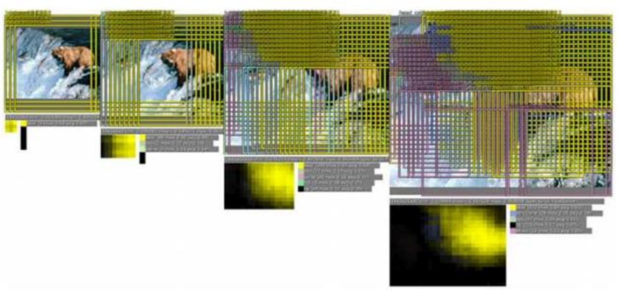
0.8

Classification score: P (cat)

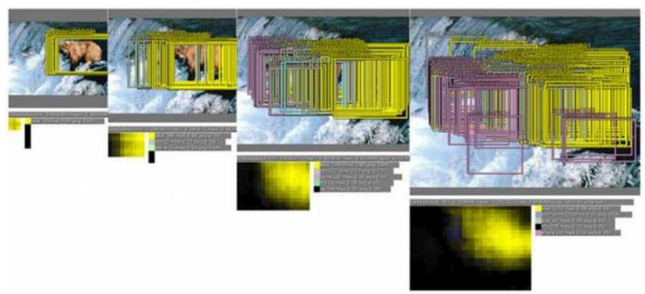
Sliding Window: Overfeat

In practice use many sliding window
locations and multiple scales

Window positions + score maps



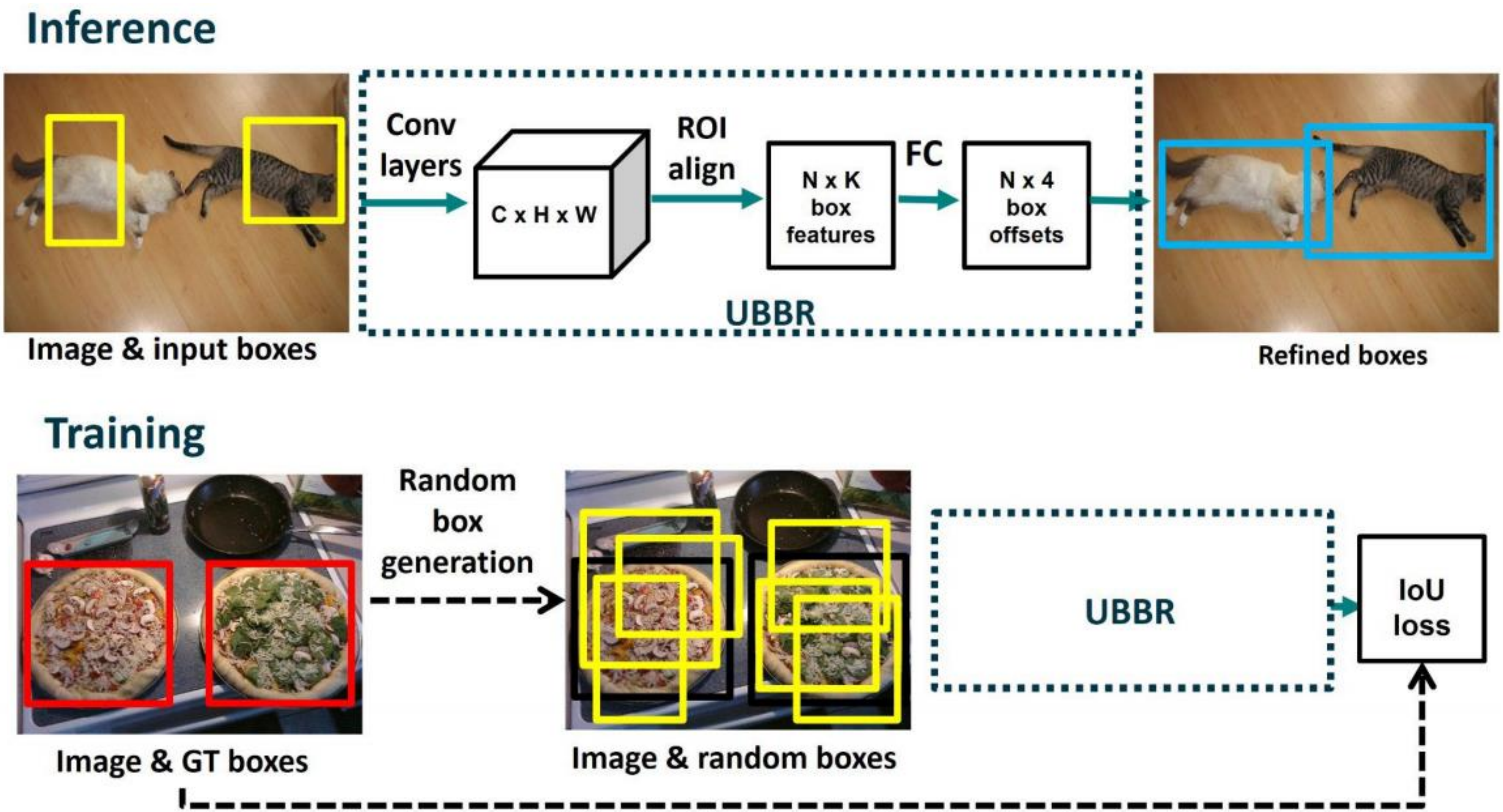
Box regression outputs



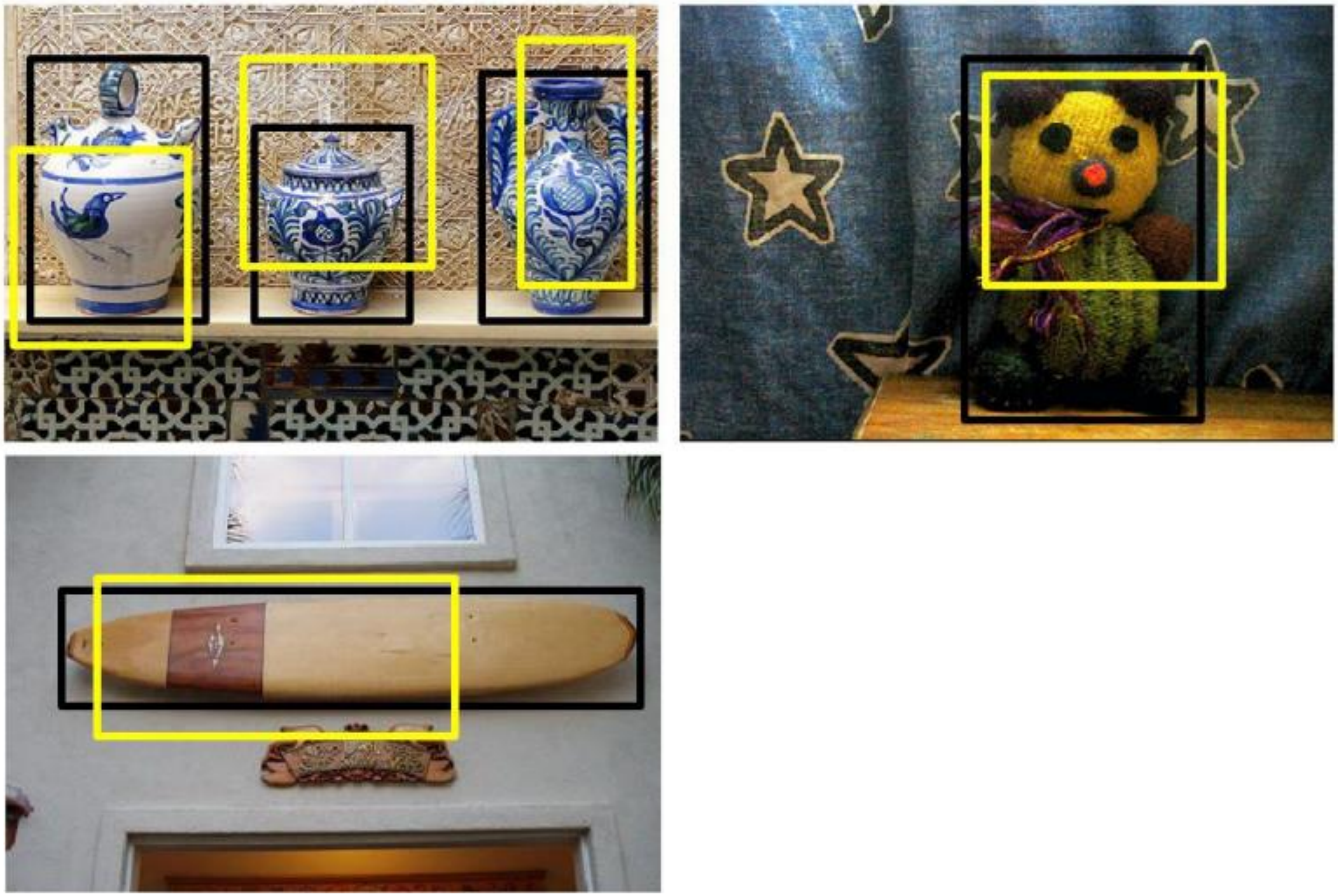
Final Predictions

Sermanet et al, "Integrated Recognition, Localization and Detection using Convolutional Networks", ICLR 2014

Universal Bounding Box Regression (UBBR)

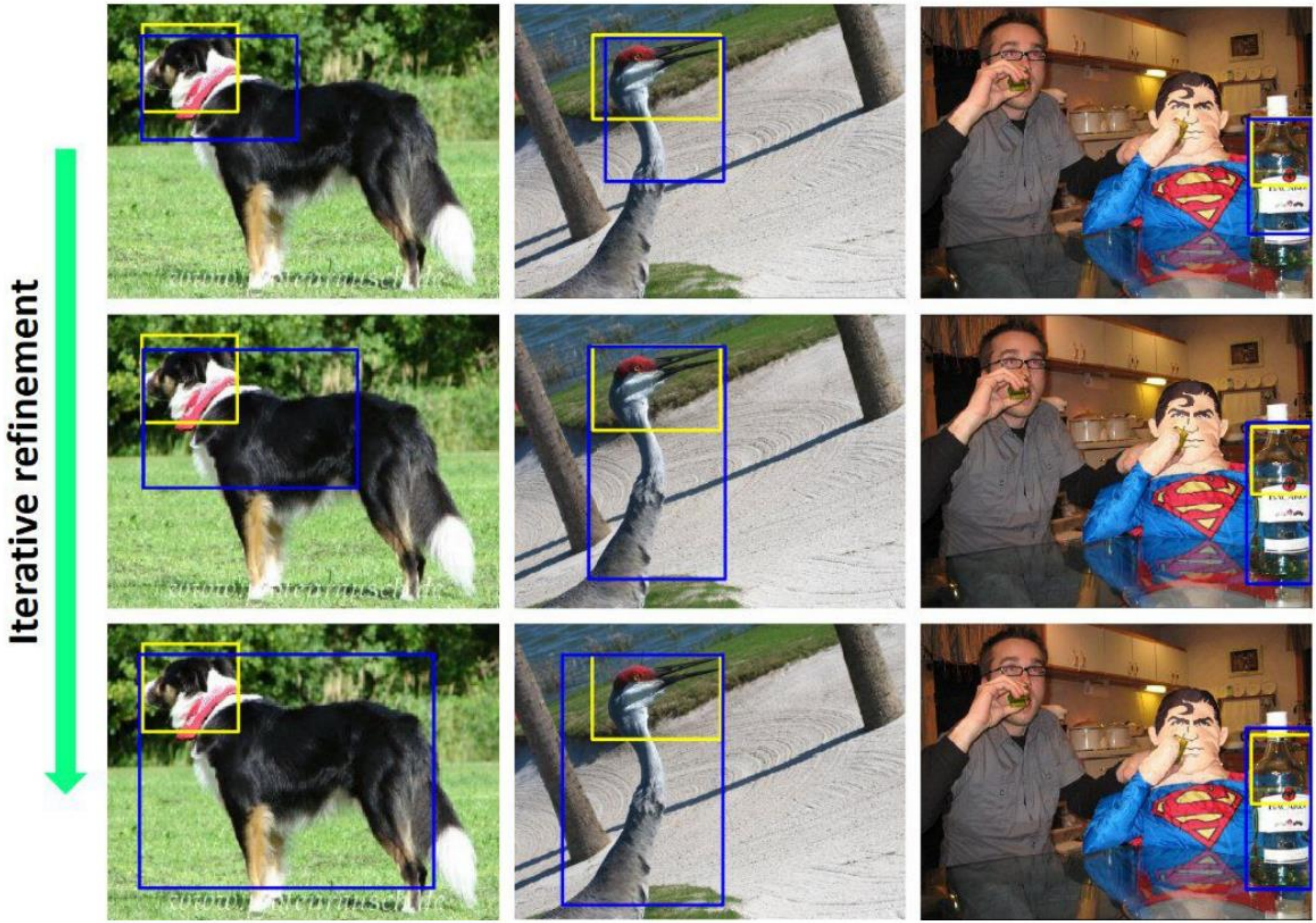


Universal Bounding Box Regression (UBBR)



Example of randomly generated bounding boxes for training UBBR. Black boxes are ground-truths and yellow ones are randomly generated boxes.

Universal Bounding Box Regression (UBBR)



Influential Works in Detection

- Sung-Poggio (1994, 1998) : ~2000 citations
  - Basic idea of statistical template detection, bootstrapping to get “face-like” negative examples, multiple whole-face prototypes (in 1994)
- Rowley-Baluja-Kanade (1996-1998) : ~3600 citations
  - “Parts” at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
- Schneiderman-Kanade (1998-2000,2004) : ~1700 citations
  - Careful feature engineering, excellent results, cascade
- Viola-Jones (2001, 2004) : ~18,000 citations
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast
- Dalal-Triggs (2005) : ~24,000 citations
  - Careful feature engineering, excellent results, HOG feature, easy to implement
- Felzenszwalb-McAllester-Ramanan (2008): ~7,300 citations
  - Template/parts-based blend
- Girshick et al. (2013): ~6500 citations
  - R-CNN / Fast R-CNN / Faster R-CNN. Deep learned models on object proposals.

Sliding Window Face Detection with Viola-Jones

Many Slides from Lana Lazebnik

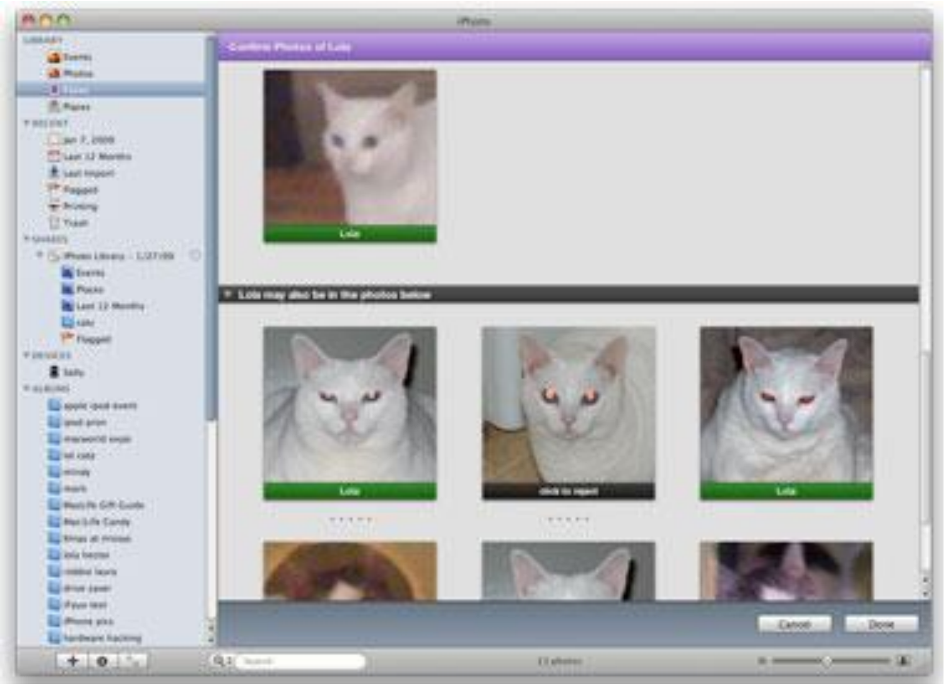
Face detection and recognition

Consumer application: Apple iPhoto

http://www.apple.com/ilife/iphoto/

Consumer application: Apple iPhoto

Can be trained to recognize pets!



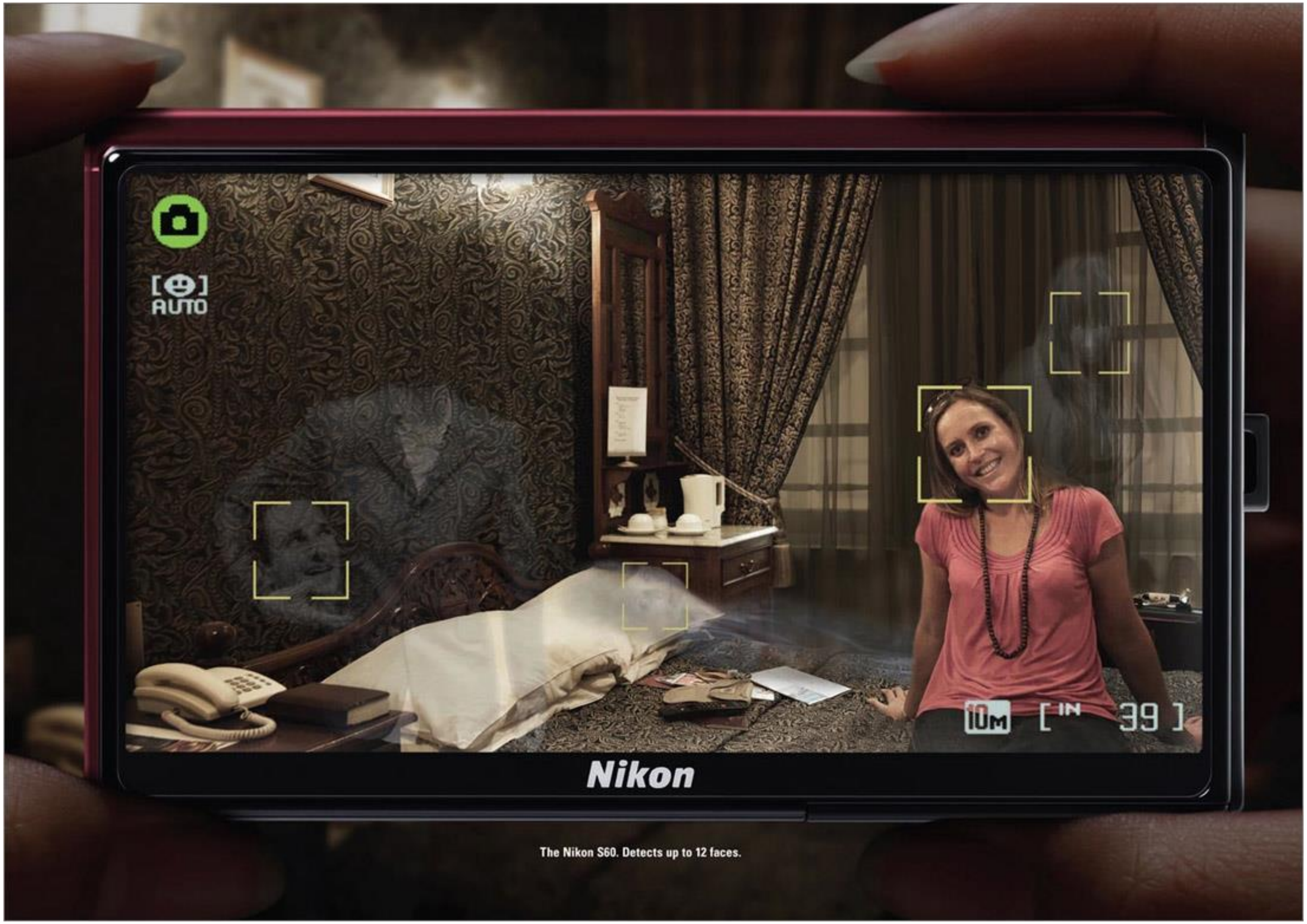
http://www.maclife.com/article/news/iphotos_faces_recognizes_cats

Consumer application: Apple iPhoto

Things iPhoto thinks are faces

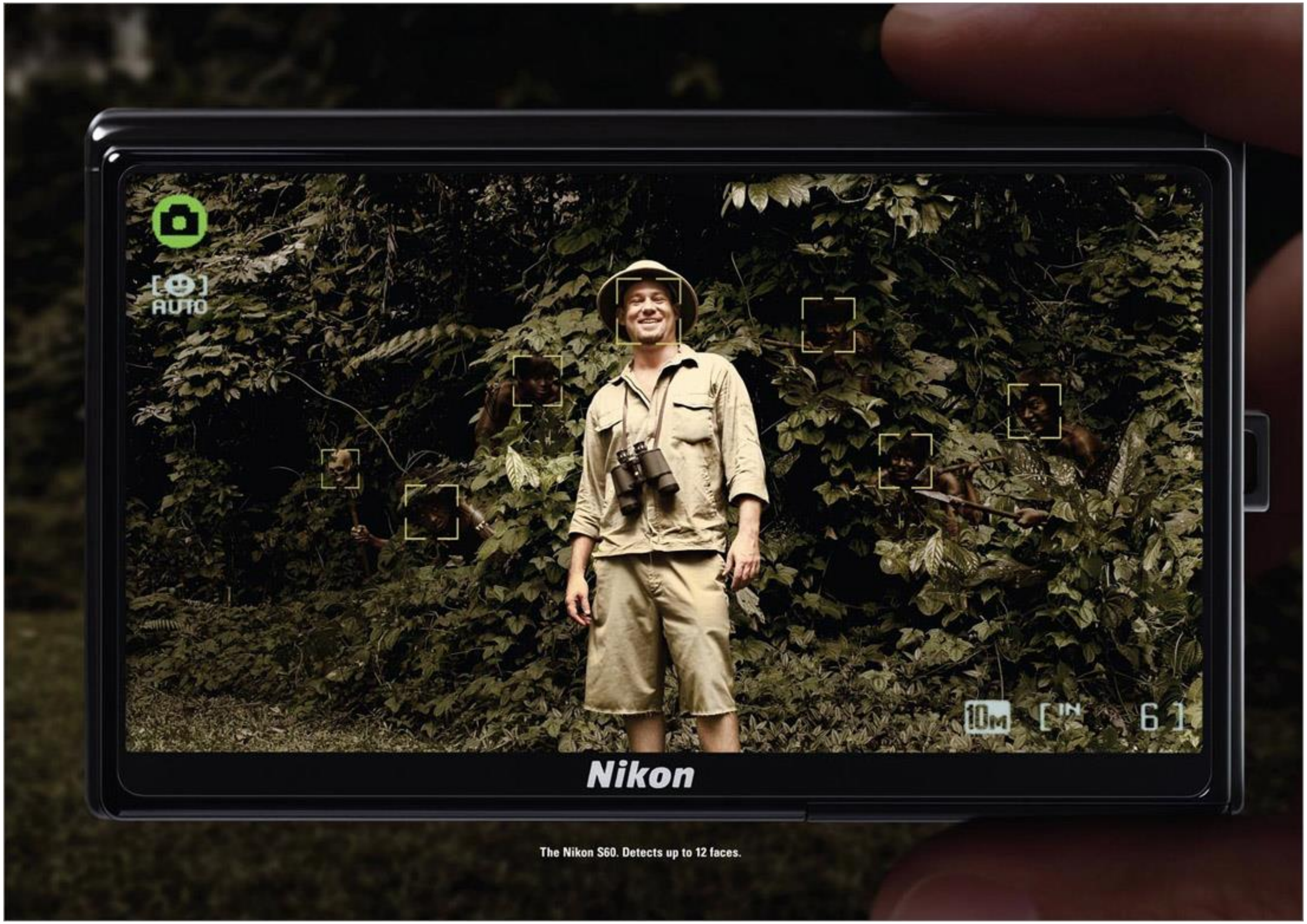
Funny Nikon ads

"The Nikon S60 detects up to 12 faces."



Funny Nikon ads

"The Nikon S60 detects up to 12 faces."



Challenges of face detection

- Sliding window detector must evaluate tens of thousands of location/scale combinations
- Faces are rare: 0–10 per image
  - For computational efficiency, we should try to spend as little time as possible on the non-face windows
  - A megapixel image has ~10^6 pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image image, our false positive rate has to be less than 10^{-6}

The Viola/Jones Face Detector

- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - *Integral images* for fast feature evaluation
  - *Boosting* for feature selection
  - *Attentional cascade* for fast rejection of non-face windows

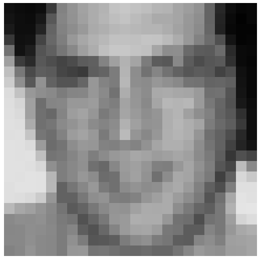
P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR 2001.

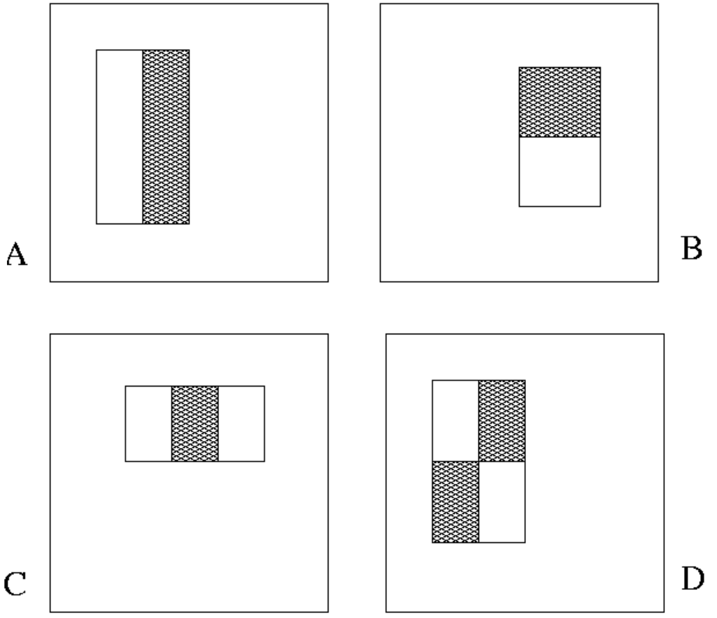
P. Viola and M. Jones. *Robust real-time face detection.* IJCV 57(2), 2004.

Image Features

Harr wavelet filter

“Rectangle filters”





Value =

∑ (pixels in white area) –
∑ (pixels in black area)

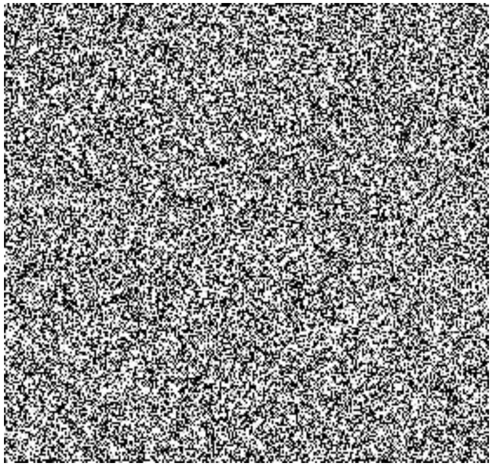
A: The value of the rectangle feature is the difference between the sum of the pixels at left side rectangular region and that of right side one

B: The value of the rectangle feature is the difference between the sum of the pixels at down rectangular region and that of up one

C: The value of the rectangle feature is the difference between the sum of the pixels at left and right rectangular regions and that of center one

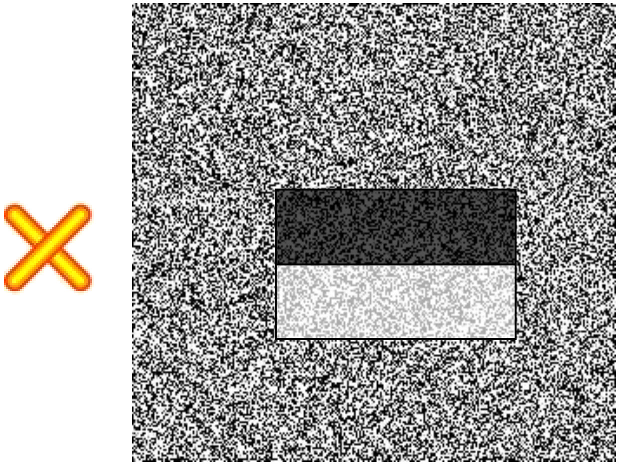
D: The value of the rectangle feature is the difference between the sum of the pixels at two northeast rectangular regions and that of two northwest ones (two diagonal pairs)

Example



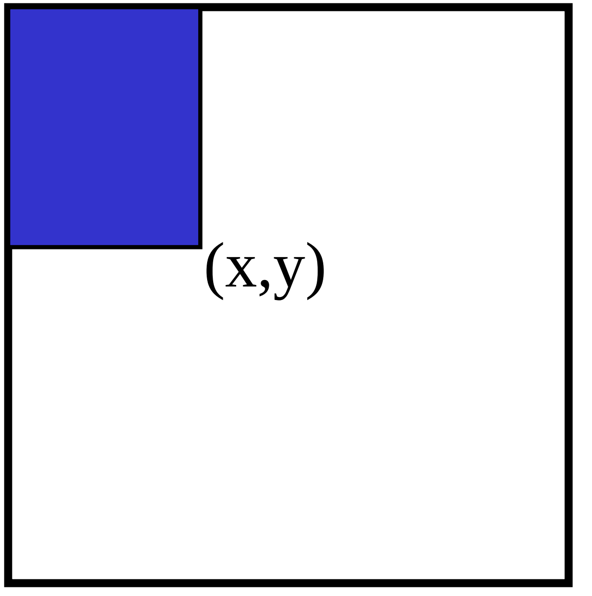
Source

Result



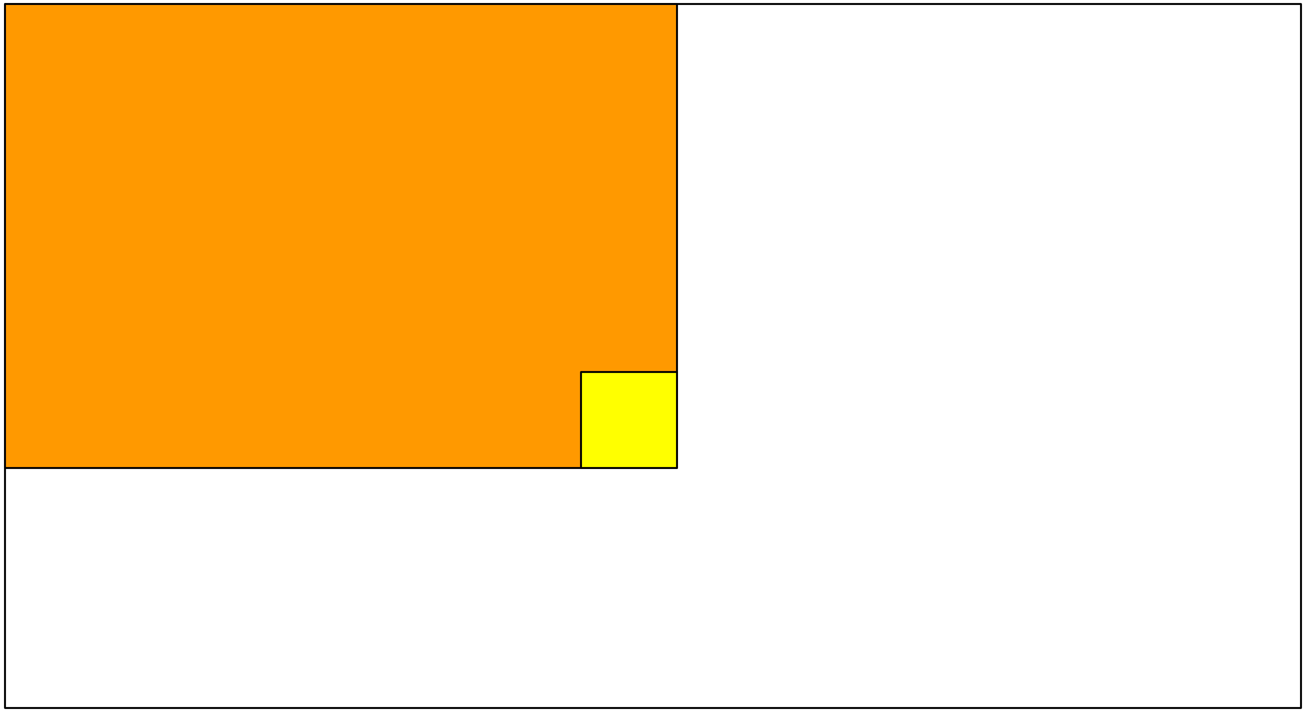
Fast computation with integral images

- The ***integral image*** computes a value at each pixel (x, y) that is the sum of the pixel values **above** and to **the left** of (x, y), inclusive
- This can quickly be computed in one pass through the image

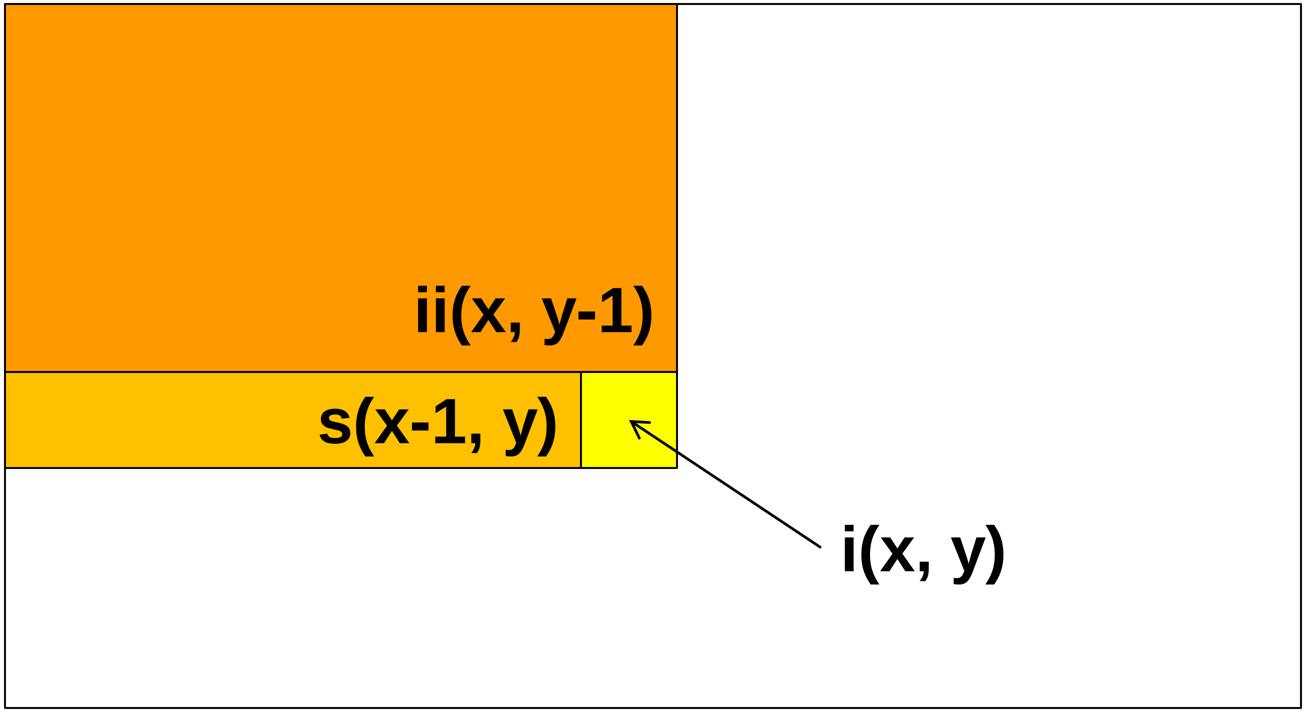


$$ii(x,y) = \sum_{x' \le x,\, y' \le y} i(x', y'),$$

Computing the integral image



Computing the integral image



Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$

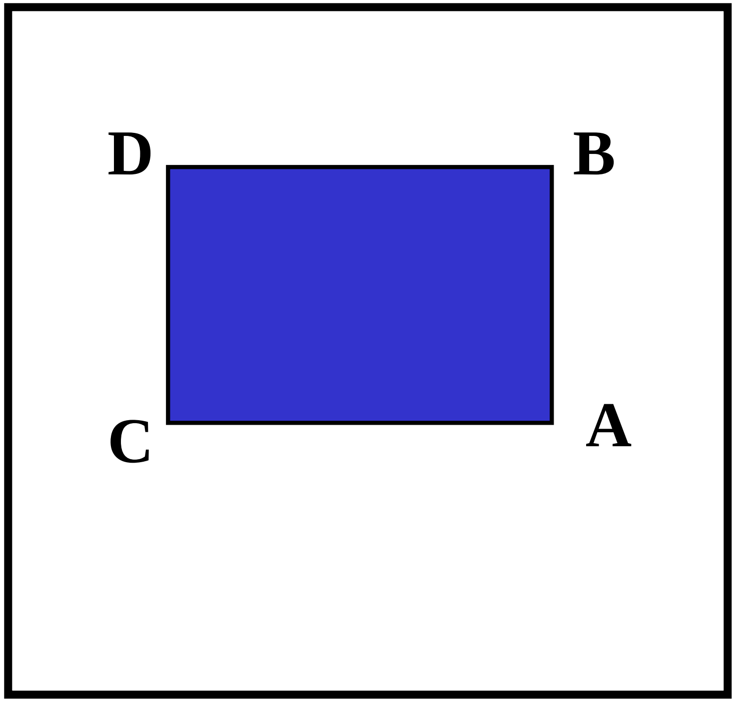
Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$

MATLAB: `ii = cumsum(cumsum(double(i)), 2);`

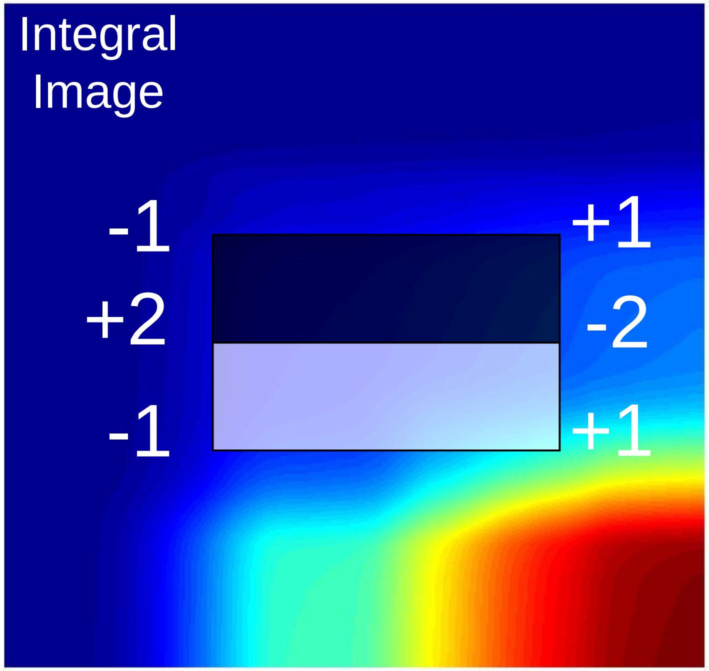
Computing sum within a rectangle

- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:

  sum = A – B – C + D
- Only 3 additions are required for any size of rectangle!

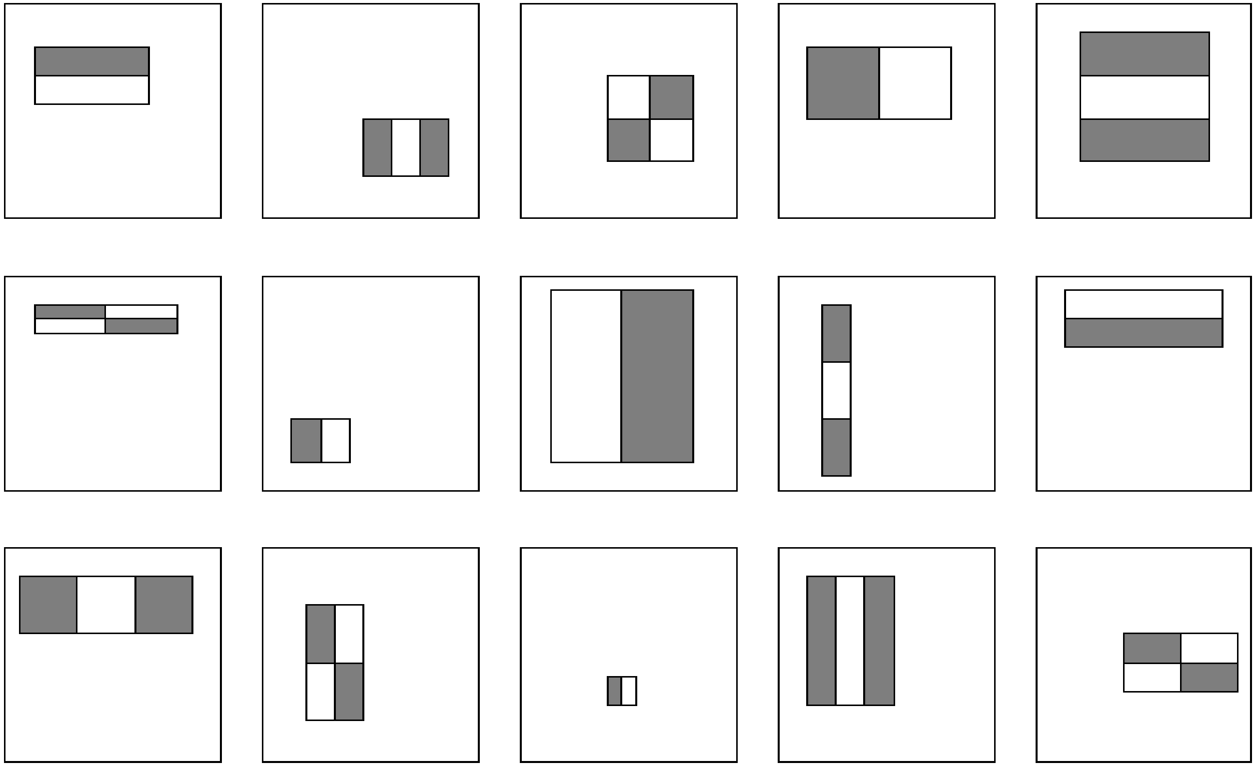


Computing a rectangle feature



Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!



Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!
- At test time, it is impractical to evaluate the entire feature set
- Can we create a good classifier using just a small subset of all possible features?
- How to select such a subset?

Boosting

- *Boosting* is a learning scheme that combines *weak learners* into a more accurate *ensemble classifier*
- Weak learners based on rectangle filters:

value of rectangle feature

$$h_t(x) = \begin{cases} 1 & \text{if } \; p_t f_t(x) > p_t \theta_t \\ 0 & \text{otherwise} \end{cases}$$

window

parity

threshold

- Ensemble classification function:

$$C(x) = \begin{cases} 1 & \text{if } \; \sum_{t=1}^{T} \alpha_t h_t(x) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

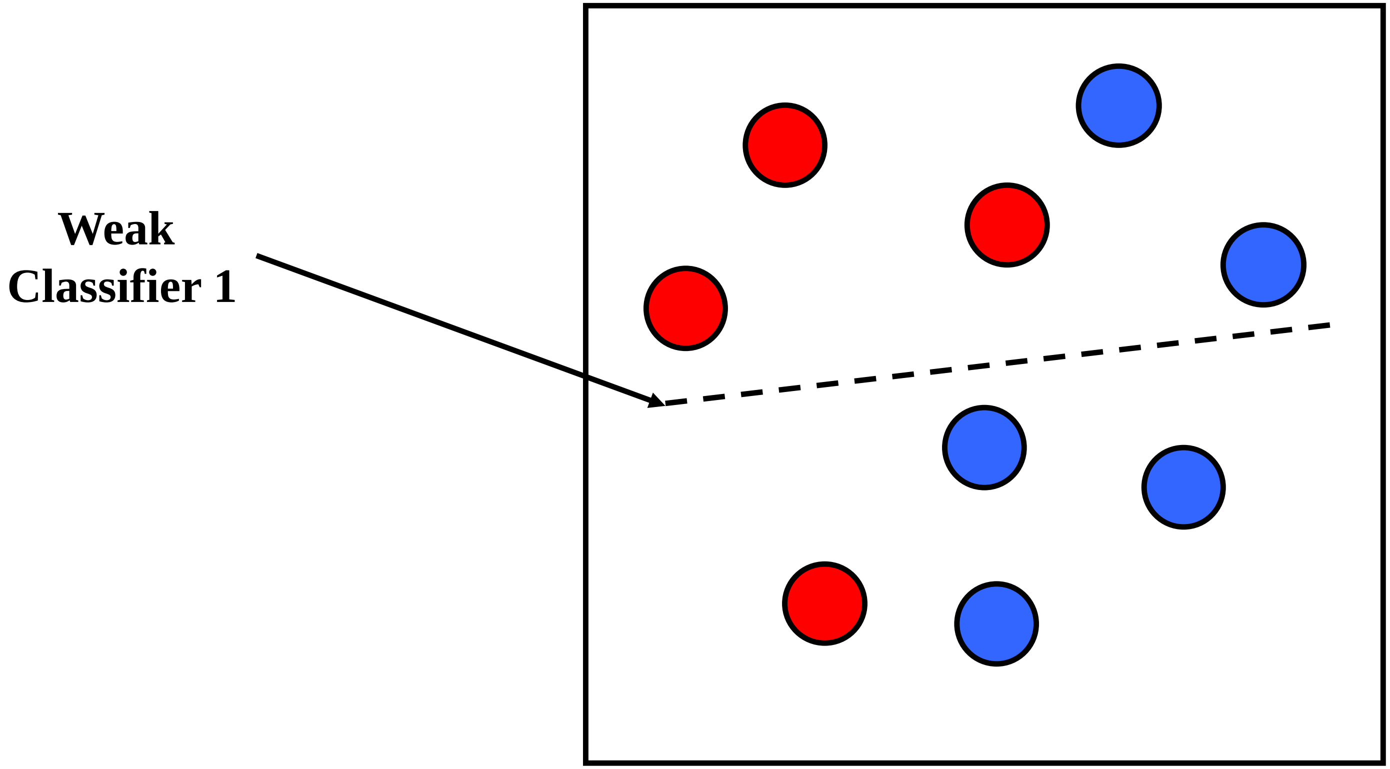
learned weights

Training procedure

- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner that achieves the lowest *weighted* training error
  - Raise the weights of training examples misclassified by current weak learner
- Compute final classifier as linear combination of all weak learners (weight of each learner is directly proportional to its accuracy)
  - Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme (e.g., AdaBoost)

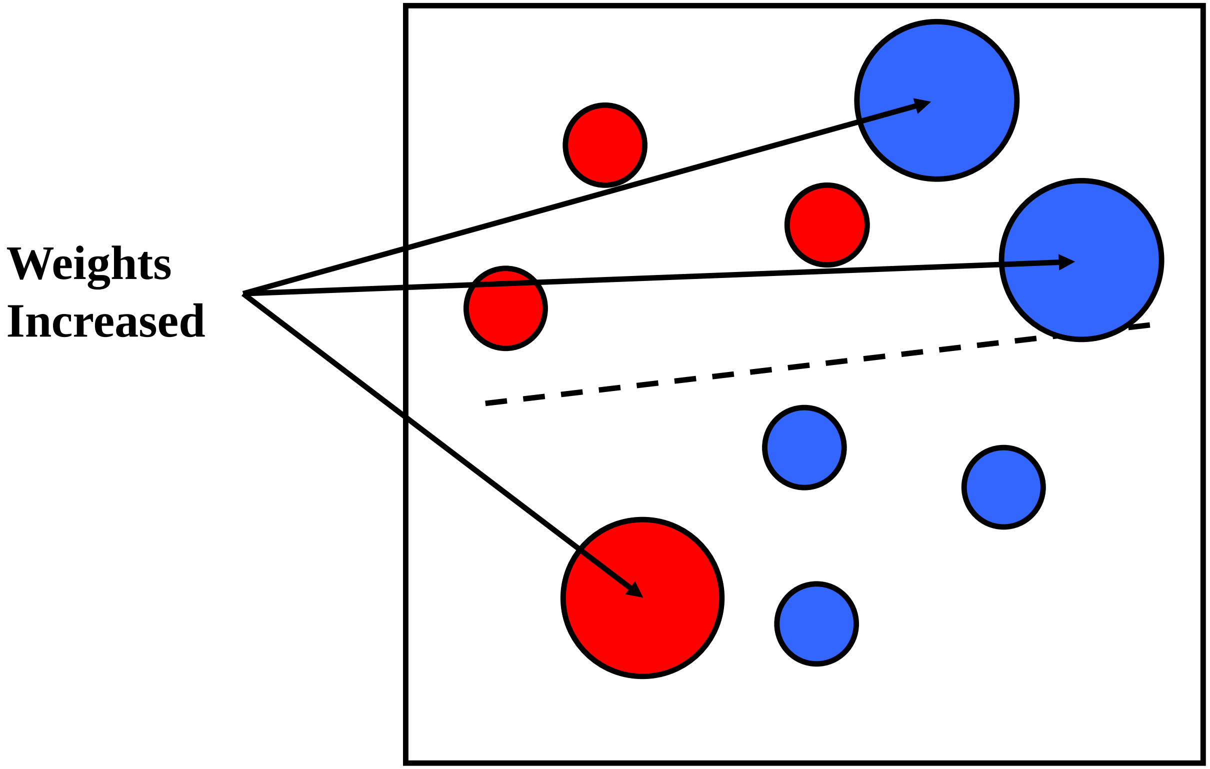
Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Boosting intuition

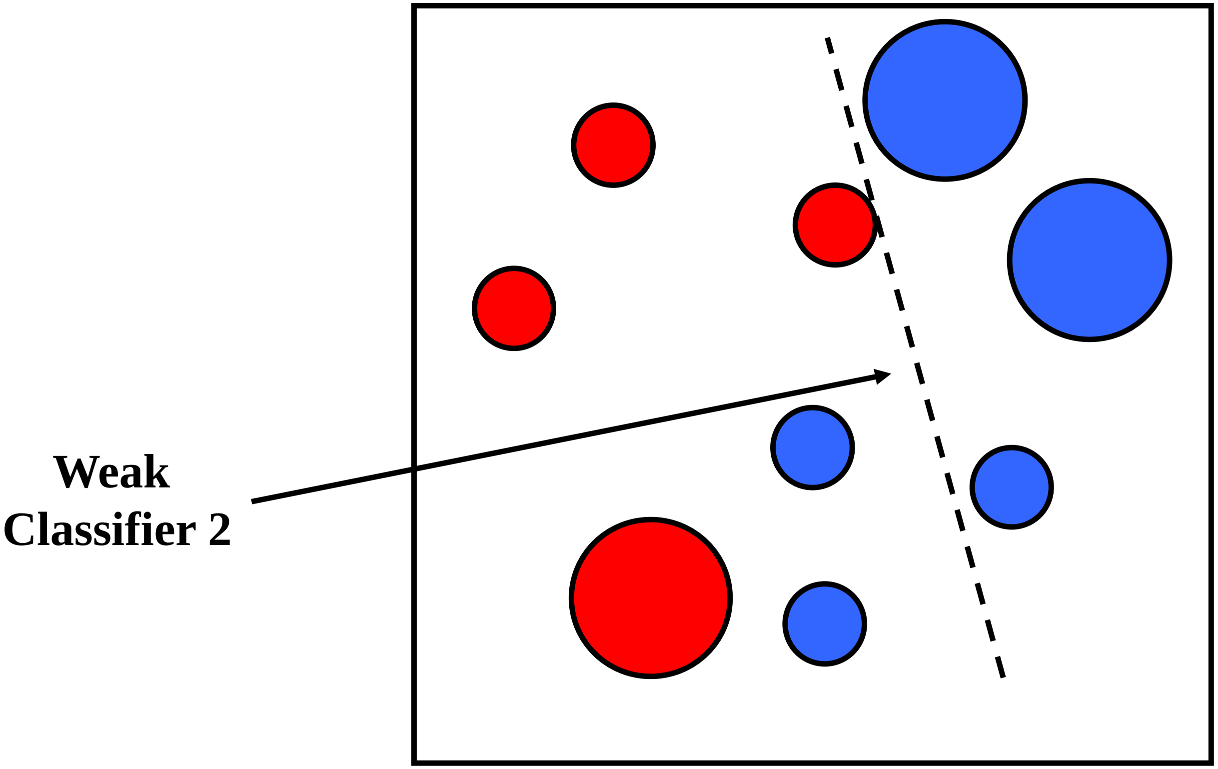


Slide credit: Paul Viola

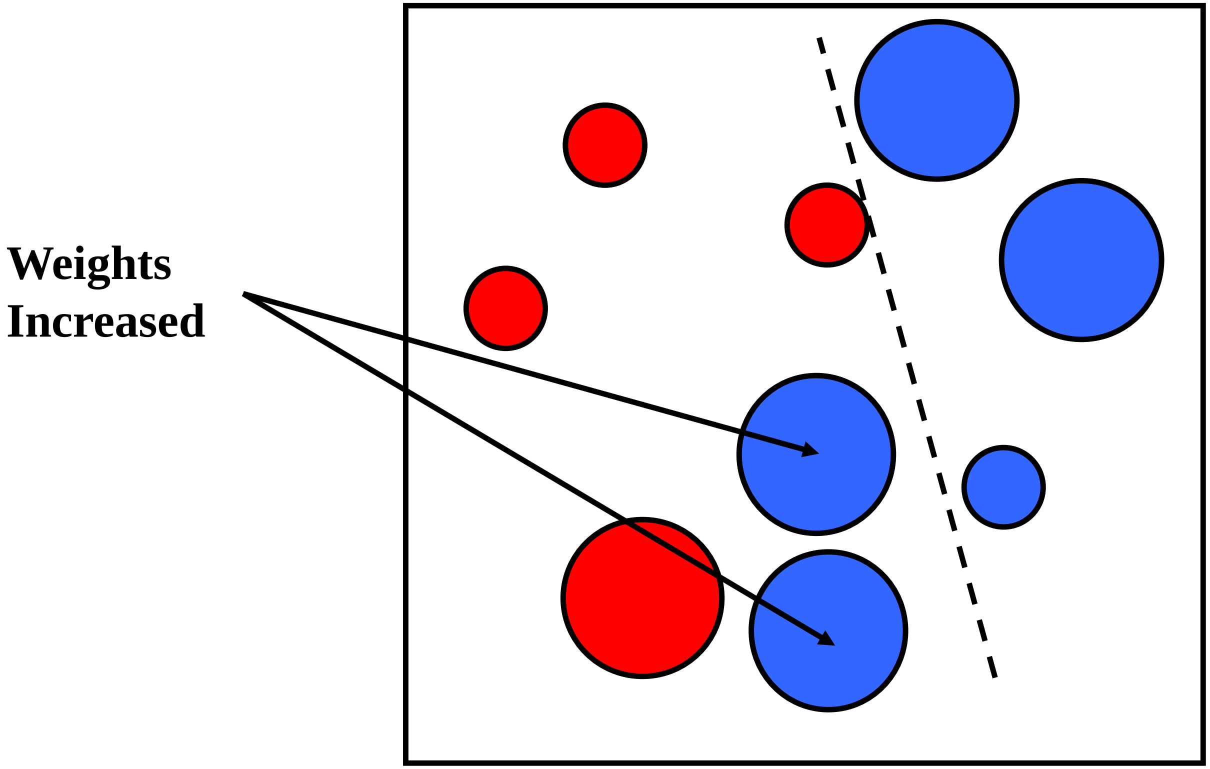
Boosting illustration



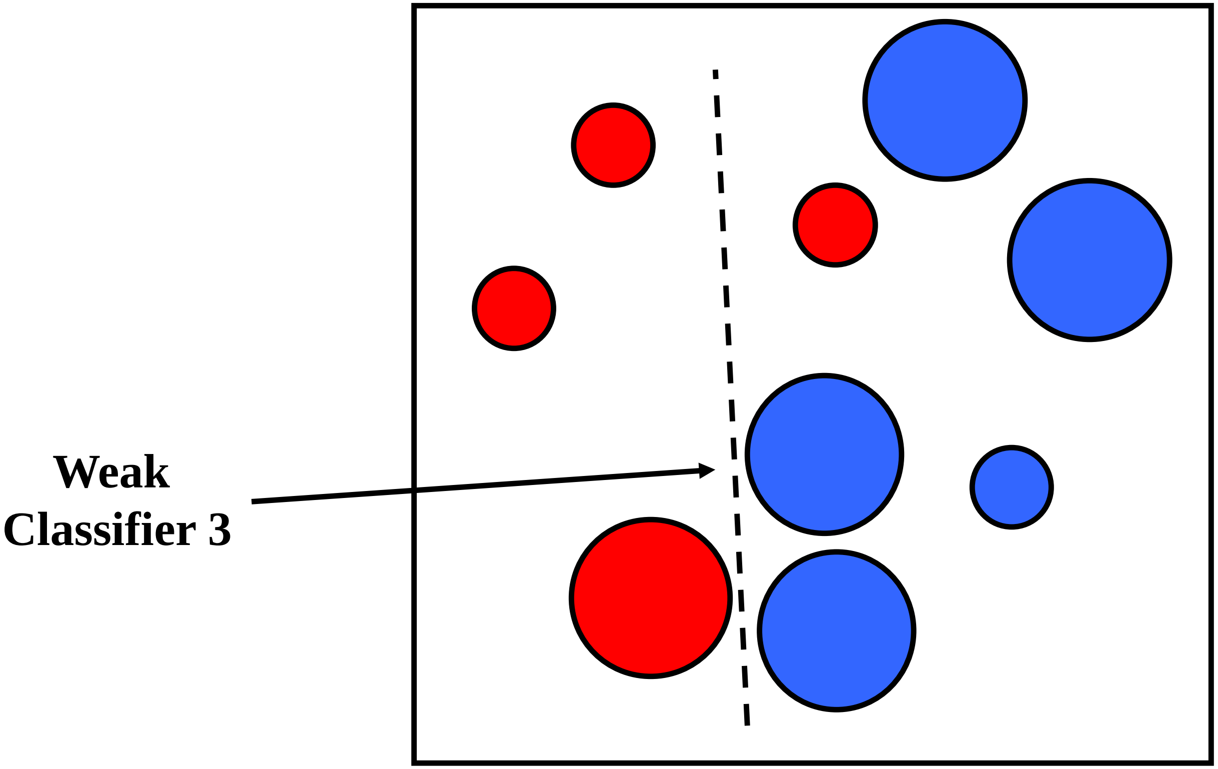
Boosting illustration



Boosting illustration

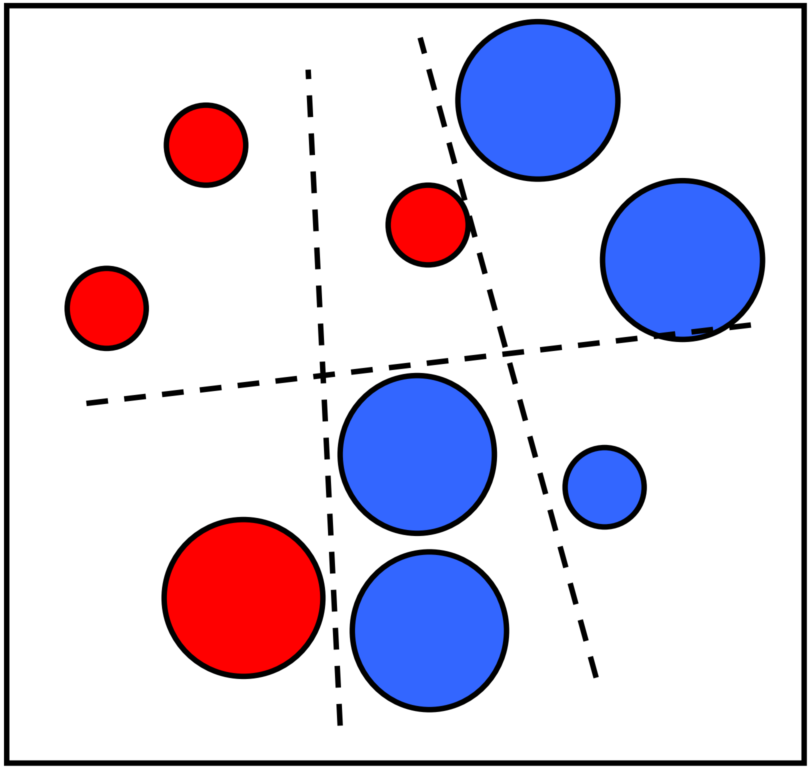


Boosting illustration



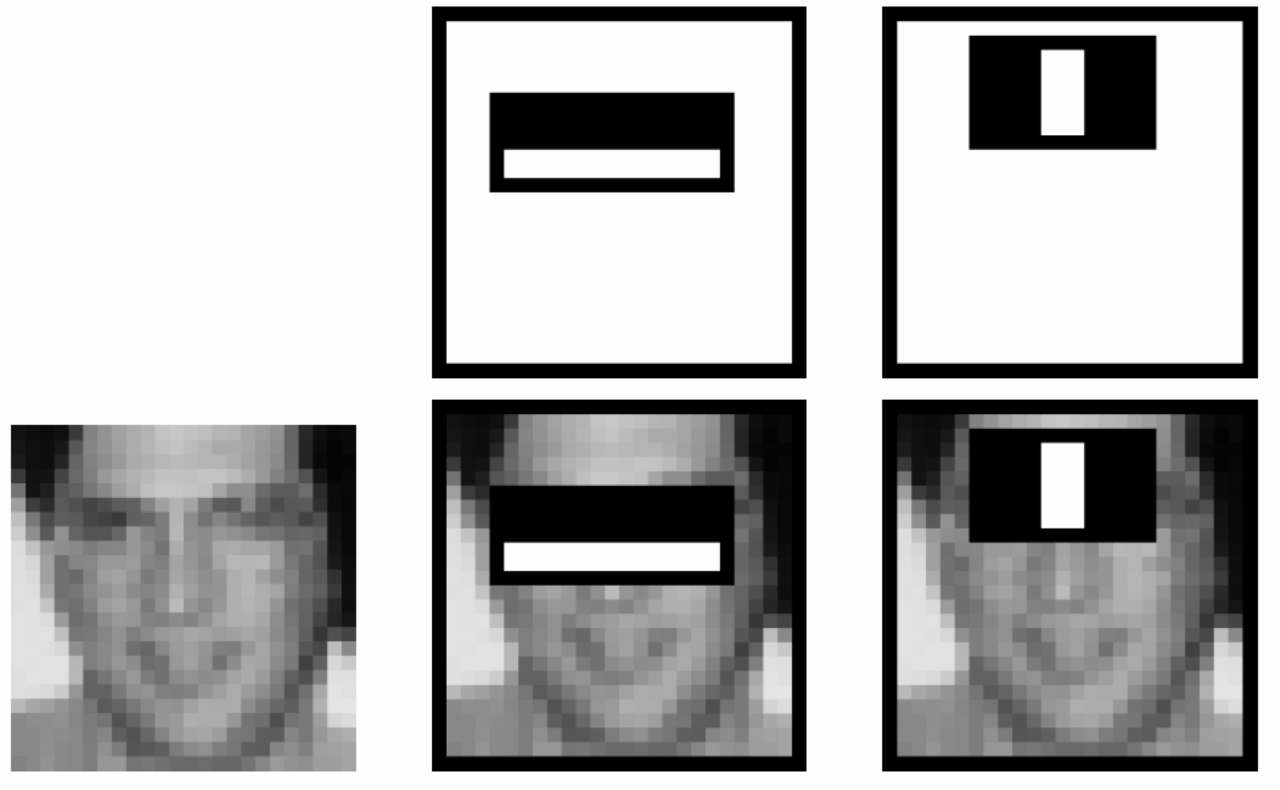
Boosting illustration

Final classifier is a combination of weak classifiers



Boosting for face detection

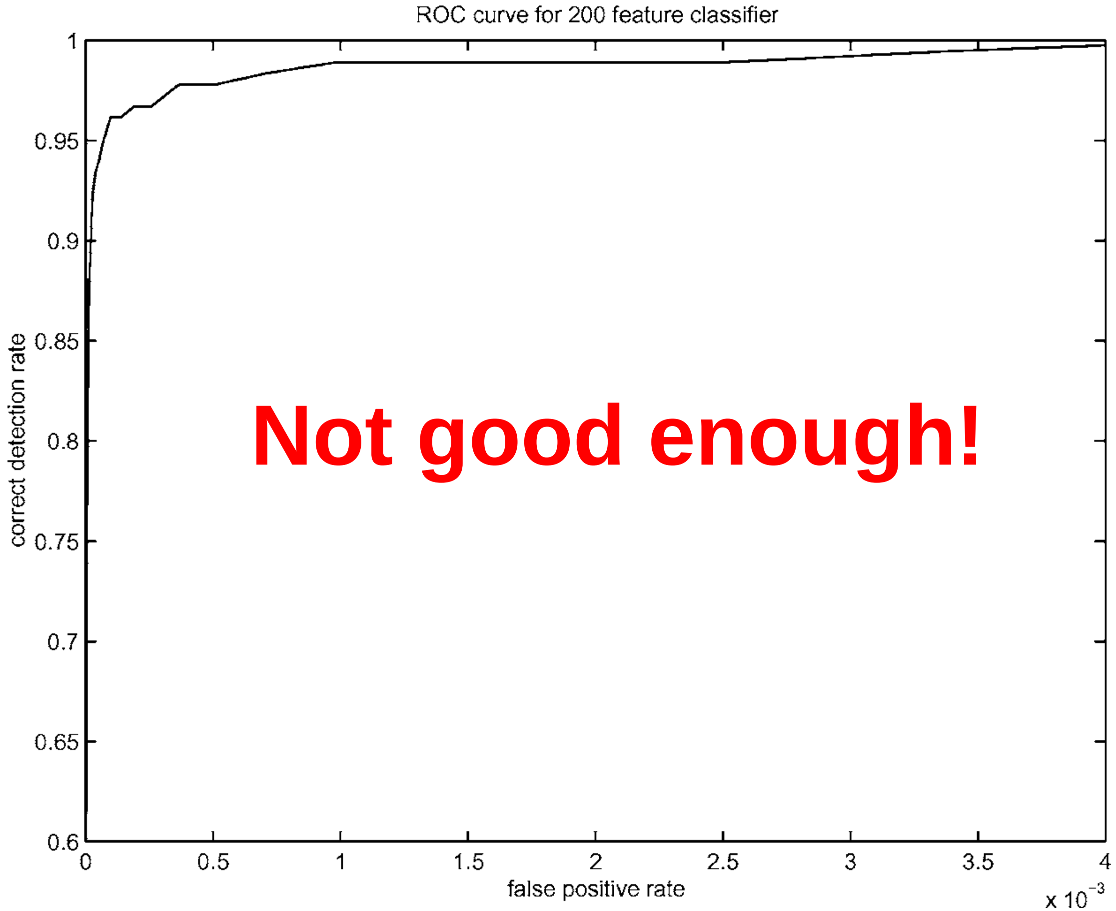
- First two features selected by boosting:



This feature combination can yield 100% recall and 50% false positive rate

Boosting for face detection

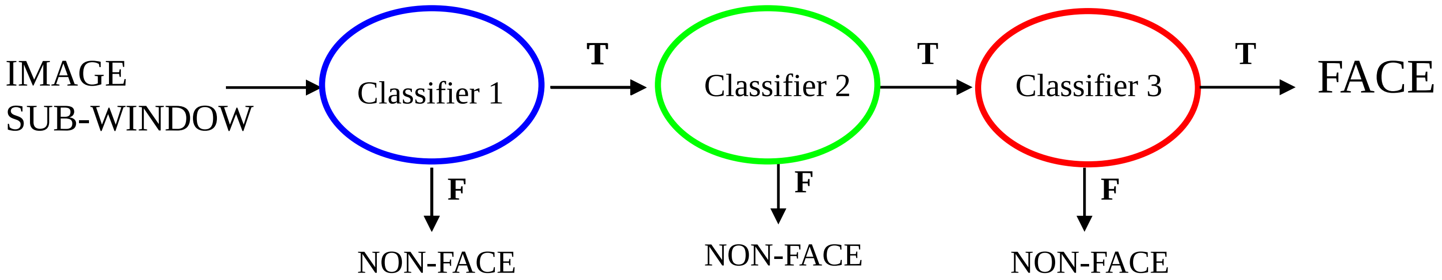
- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084



Receiver operating characteristic (ROC) curve

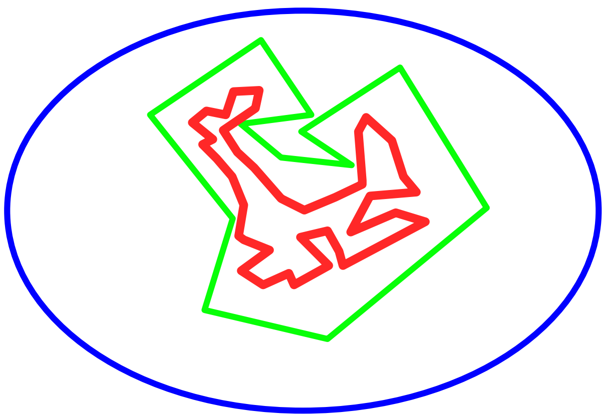
Attentional cascade

- We start with simple classifiers which reject many of the negative sub-windows while detecting almost all positive sub-windows
- Positive response from the first classifier triggers the evaluation of a second (more complex) classifier, and so on
- A negative outcome at any point leads to the immediate rejection of the sub-window

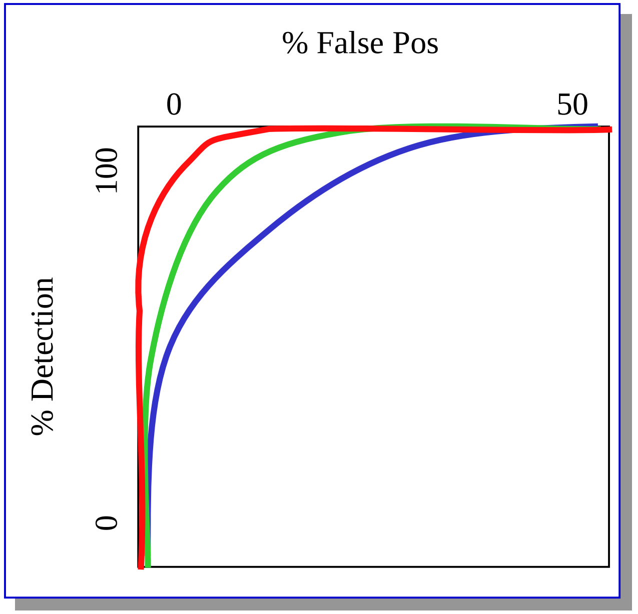


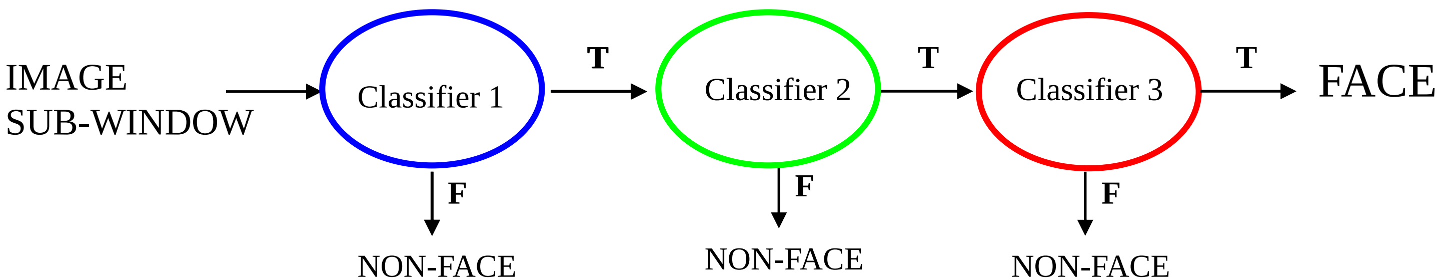
Attentional cascade

- Chain classifiers that are progressively more complex and have lower false positive rates:



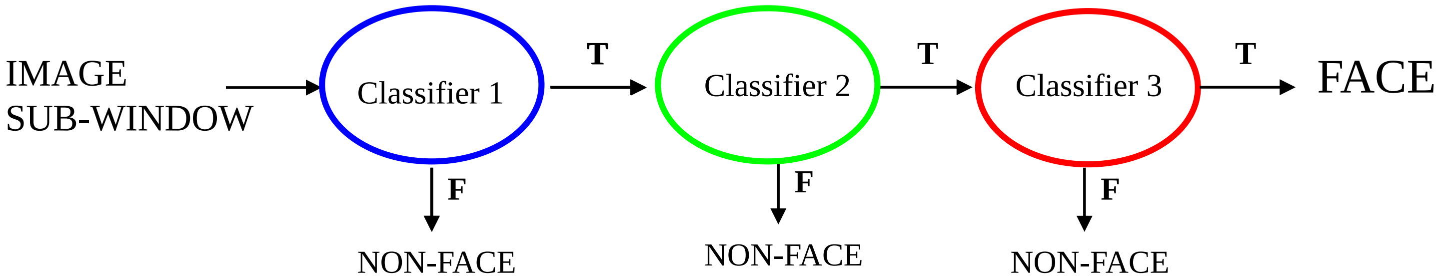
Receiver operating characteristic





Attentional cascade

- The detection rate and the false positive rate of the cascade are found by multiplying the respective rates of the individual stages
- A detection rate of 0.9 and a false positive rate on the order of 10^{-6} can be achieved by a 10-stage cascade if each stage has a detection rate of 0.99 ($0.99^{10} \approx 0.9$) and a false positive rate of about 0.30 ($0.3^{10} \approx 6\times10^{-6}$)



Training the cascade

- Set target detection and false positive rates for each stage
- Keep adding features to the current stage until its target rates have been met
  - Need to lower AdaBoost threshold to maximize detection (as opposed to minimizing total classification error)
  - Test on a *validation set*
- If the overall false positive rate is not low enough, then add another stage
- Use false positives from current stage as the negative training examples for the next stage

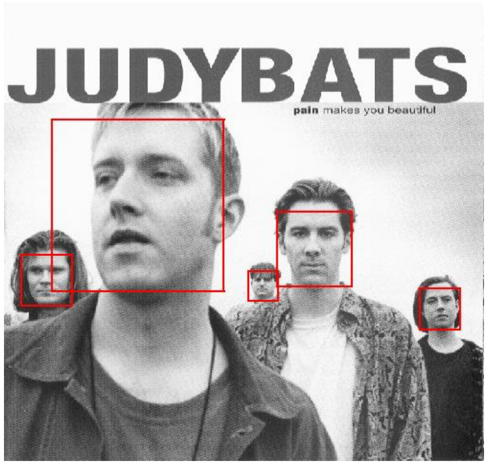
The implemented system

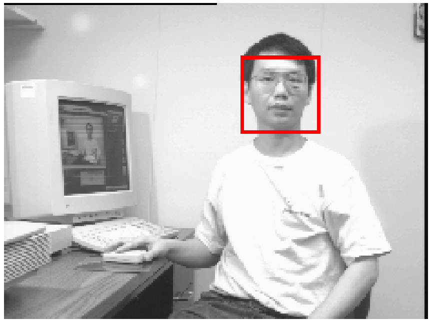
- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
    - 9500 non-face images
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose

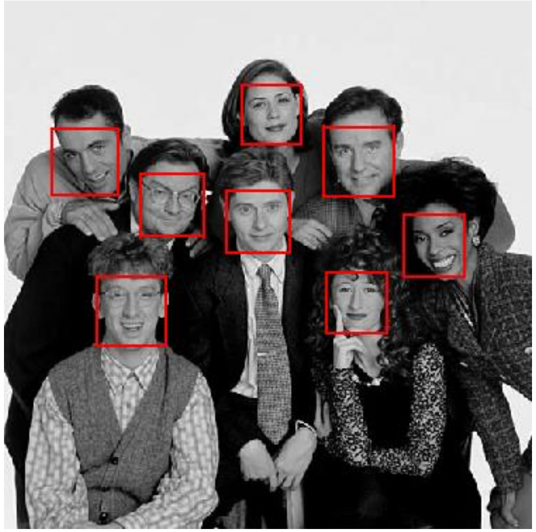
System performance

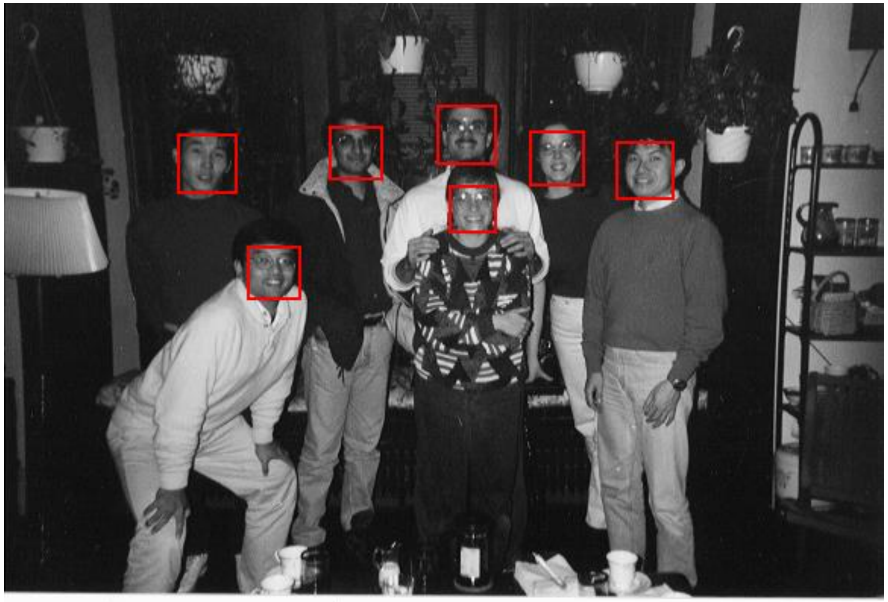
- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous detector of comparable accuracy (Rowley et al., 1998)

Output of Face Detector on Test Images



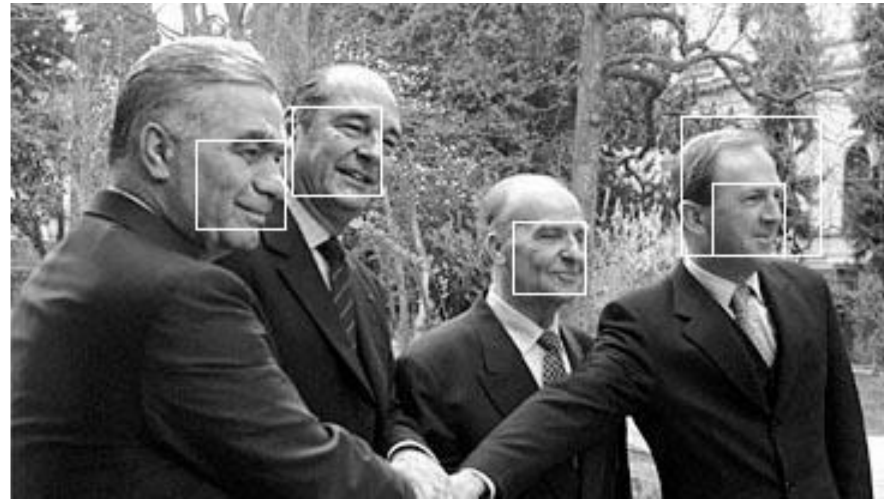






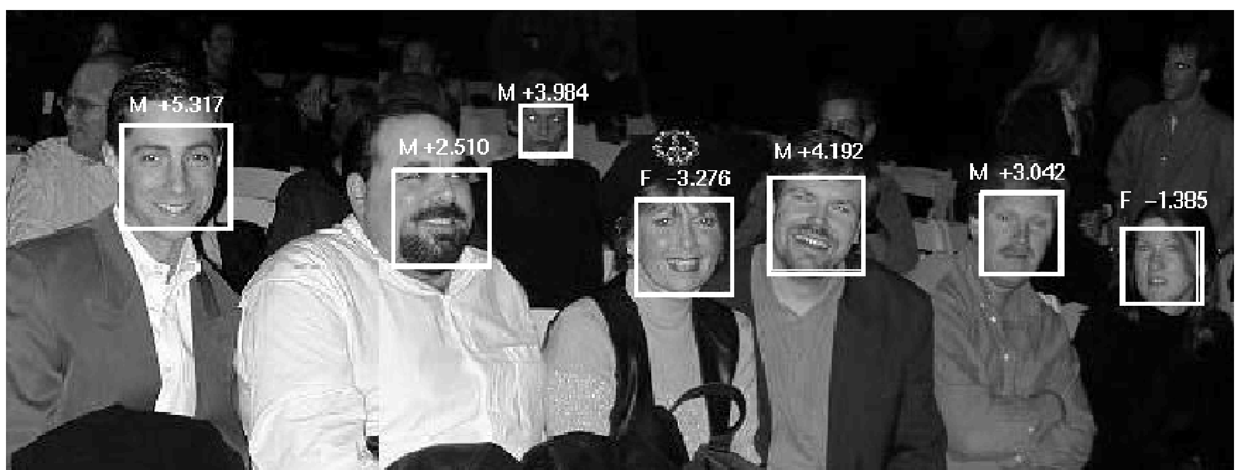
Other detection tasks

Facial Feature Localization

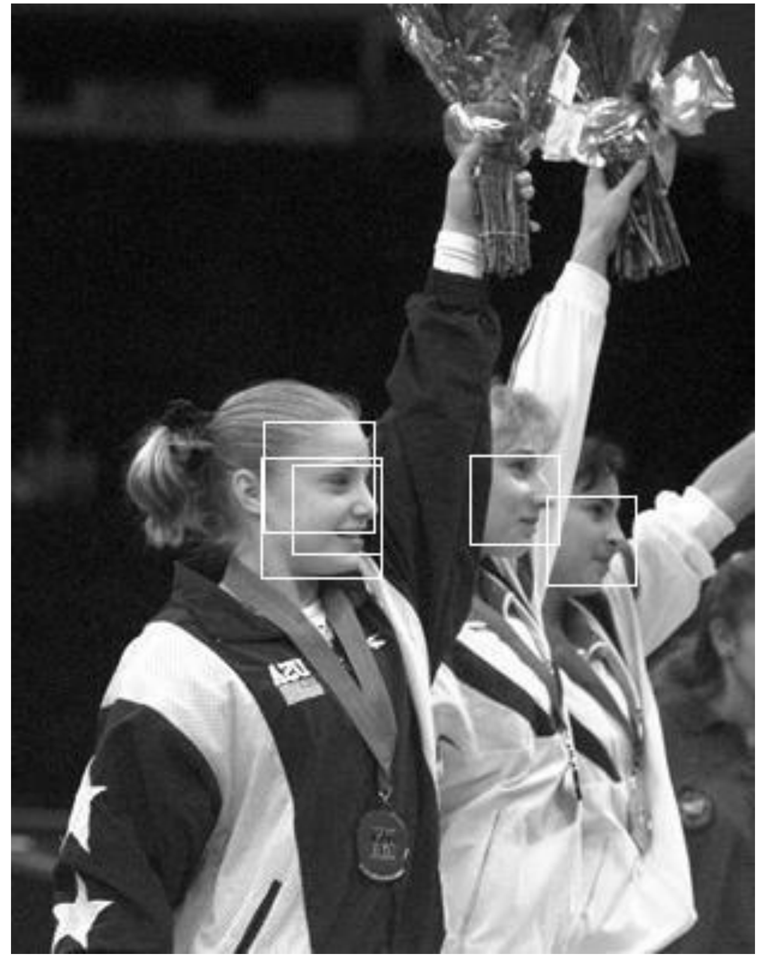


Profile Detection

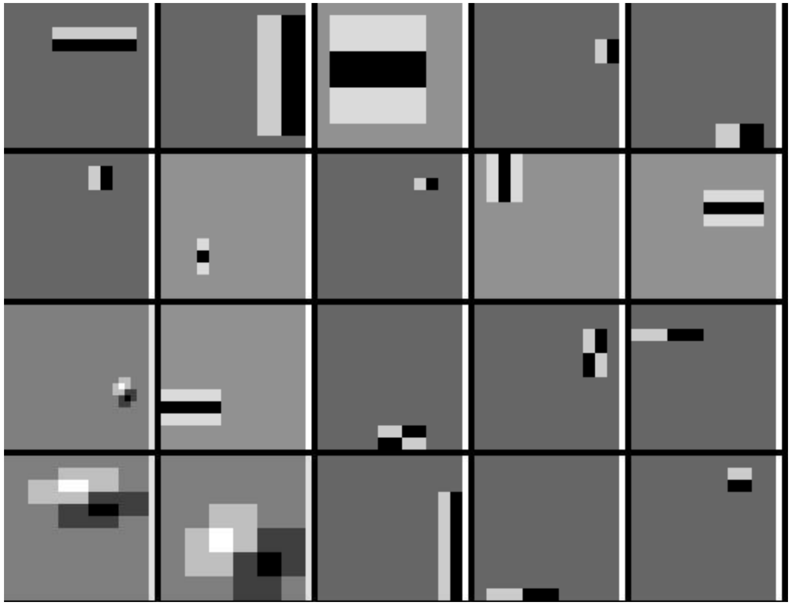
Male vs. female



Profile Detection



Profile Features



Summary: **Viola/Jones detector**

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows