CS 558: Computer Vision 11th Set of Notes

Instructor: Philippos Mordohai
Webpage: www.cs.stevens.edu/~mordohai
E-mail: Philippos.Mordohai@stevens.edu
Office: Lieb 215

Overview

- Object detection and recognition
  - Supervised Classification
  - Boosting and face detection
  - Pedestrian detection (HOG)
  - Part-based models

  - Based on slides by K. Grauman, D. Hoiem and S. Lazebnik

Why recognition?

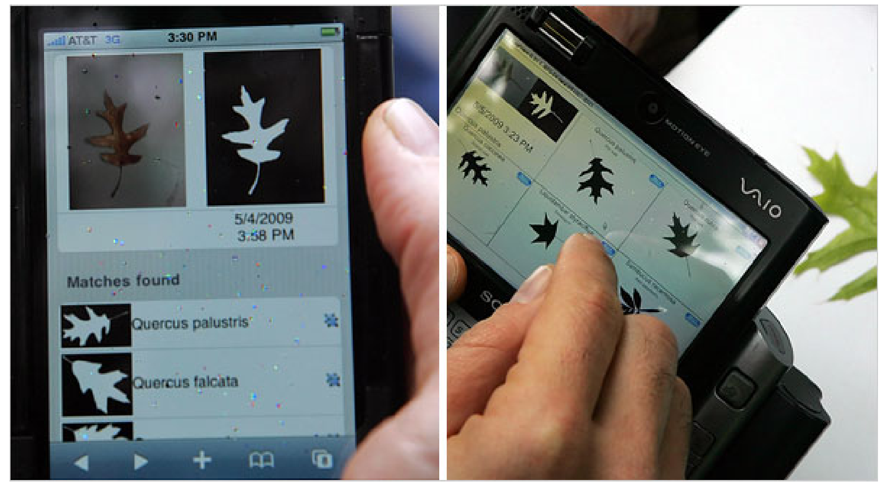
– Recognition a fundamental part of perception
    - e.g., robots, autonomous agents

– Organize and give access to visual content
    - Connect to information
    - Detect trends and themes

Posing visual queries

Yeh et al., MIT

Digital Field Guides Eliminate the Guesswork

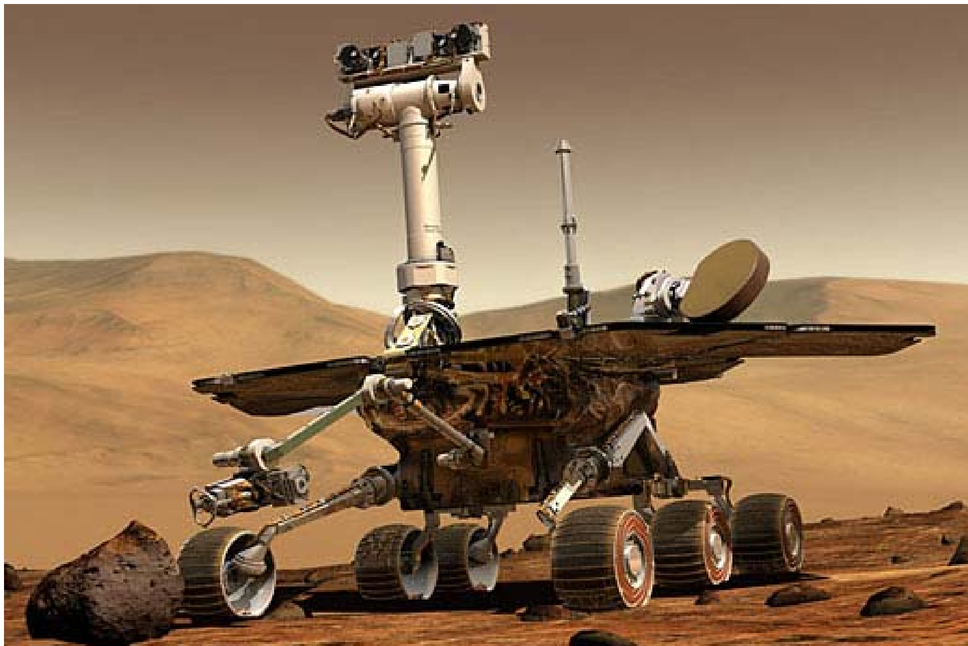


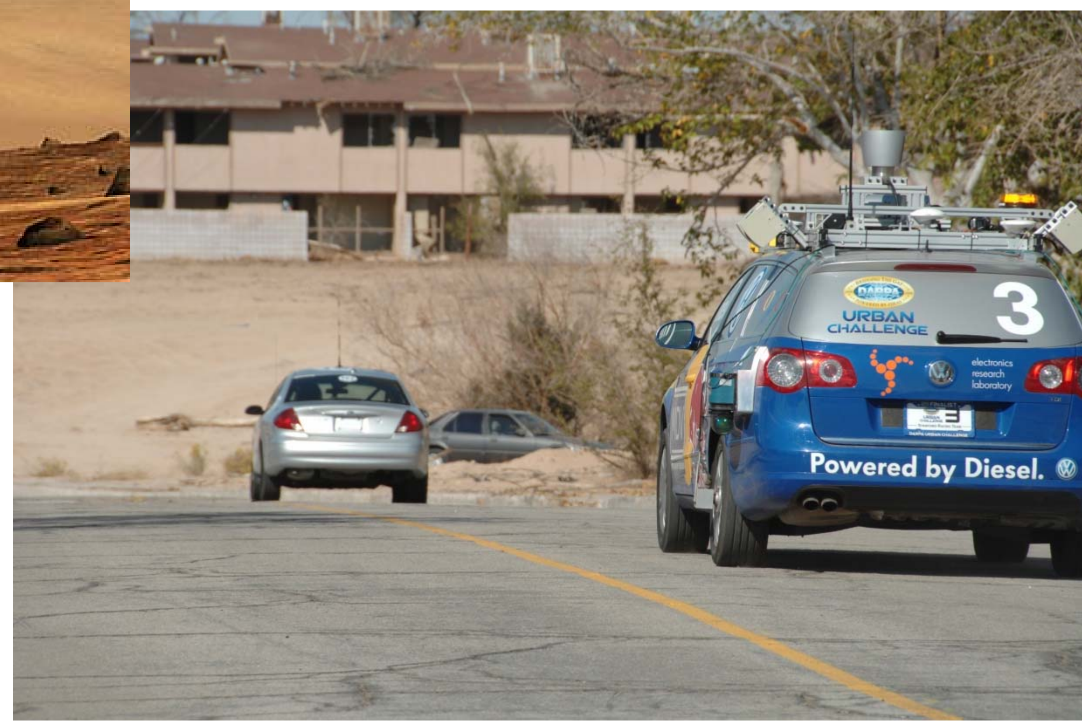
Belhumeur et al.

Outdated, but…..

Kooaba, Bay & Quack et al.

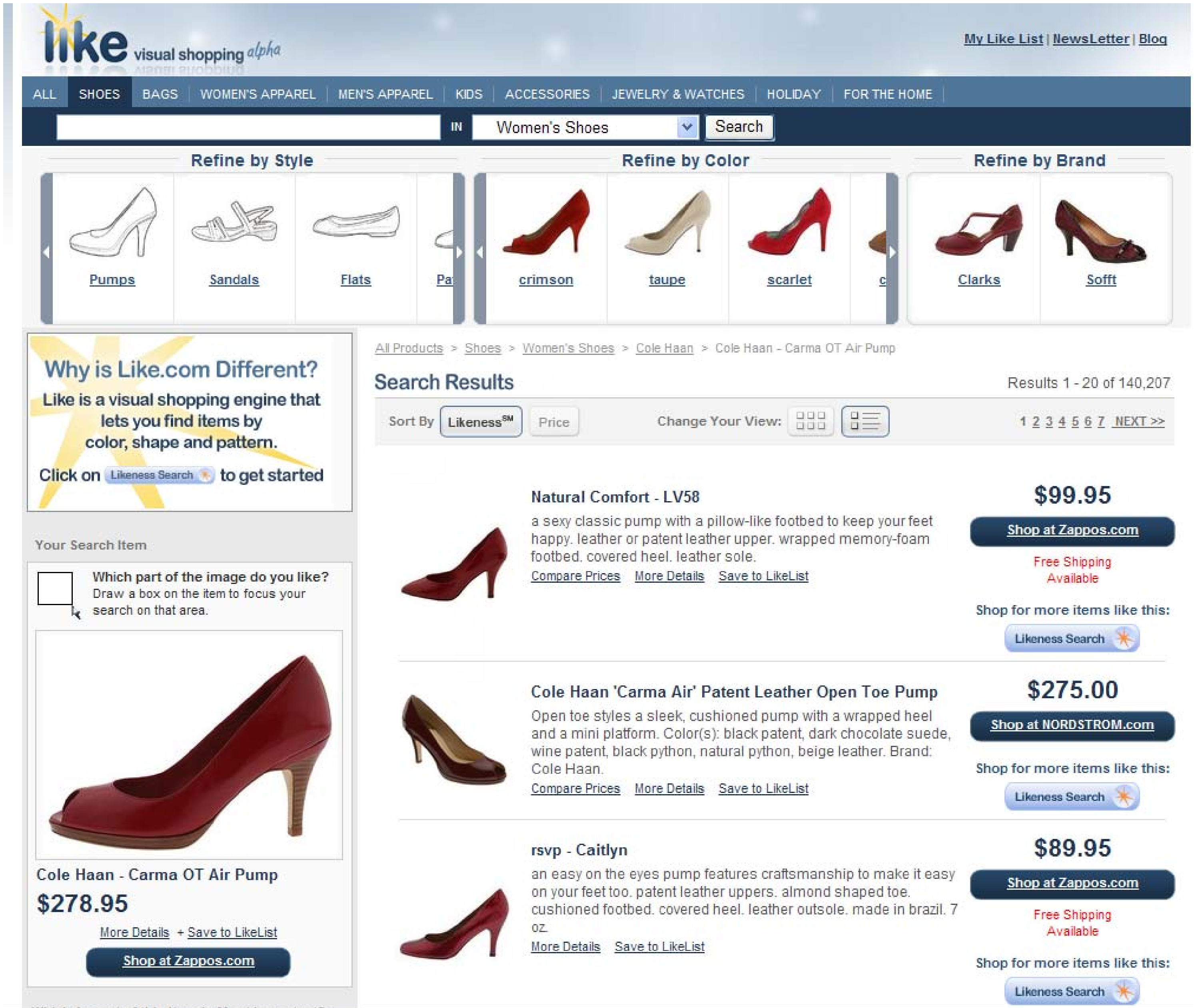
Autonomous agents able to detect objects





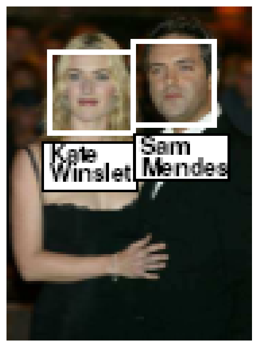
http://www.darpa.mil/grandchallenge/gallery.asp

Finding visually similar objects

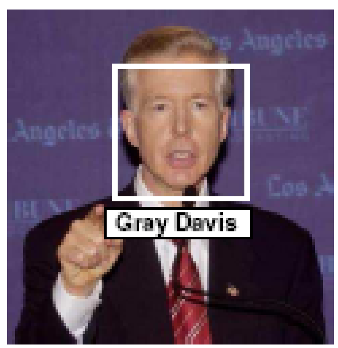


Auto-annotation

President George W. Bush makes a statement in the Rose Garden while Secretary of Defense Donald Rumsfeld looks on, July 23, 2003. Rumsfeld said the United States would release graphic photographs of the dead sons of Saddam Hussein to prove they were killed by American troops. Photo by Larry Downing/Reuters



British director Sam Mendes and his partner actress Kate Winslet arrive at the London premiere of 'The Road to Perdition', September 18, 2002. The films stars Tom Hanks as a Chicago hit man who has a separate family life and co-stars Paul Newman and Jude Law. REUTERS/Dan Chung



Incumbent California Gov. Gray Davis (news - web sites) leads Republican challenger Bill Simon by 10 percentage points – although 17 percent of voters are still undecided, according to a poll released October 22, 2002 by the Public Policy Institute of California. Davis is shown speaking to reporters after his debate with Simon in Los Angeles, on Oct. 7. (Jim Ruymen/Reuters)

T. Berg et al.

Kristen Grauman

Challenges: robustness

Illumination

Object pose

Clutter

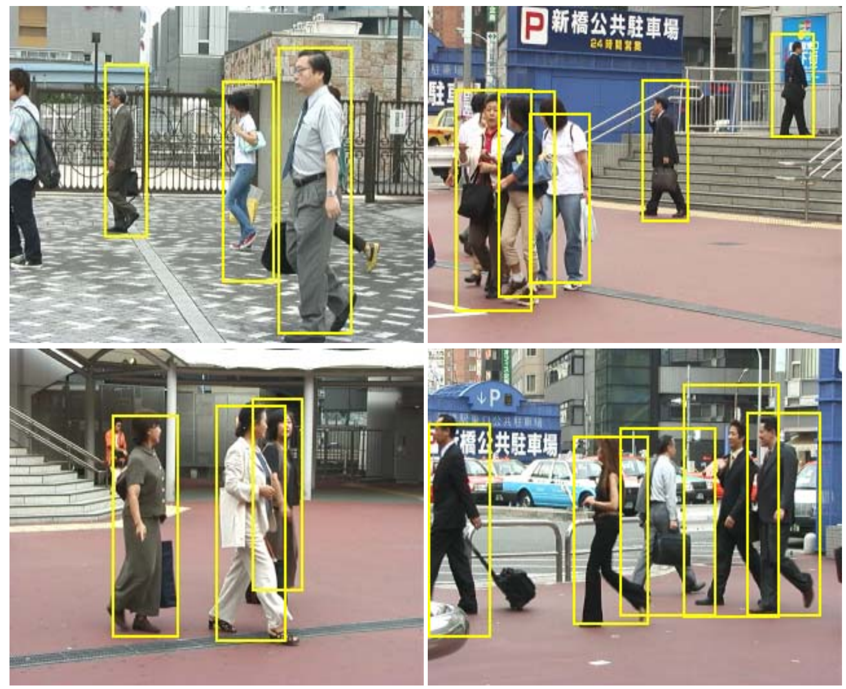
Occlusions

Intra-class appearance

Viewpoint

Kristen Grauman

Challenges: robustness

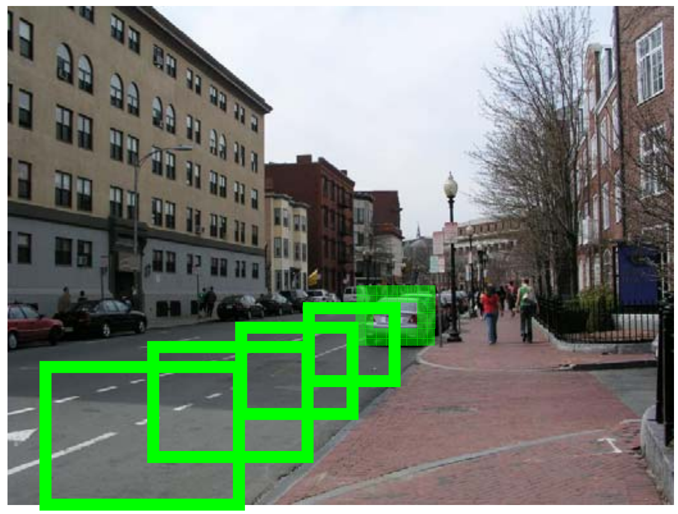


Realistic scenes are crowded, cluttered, have overlapping objects

Challenges: importance of context

slide credit: Fei-Fei, Fergus & Torralba

Challenges: importance of context

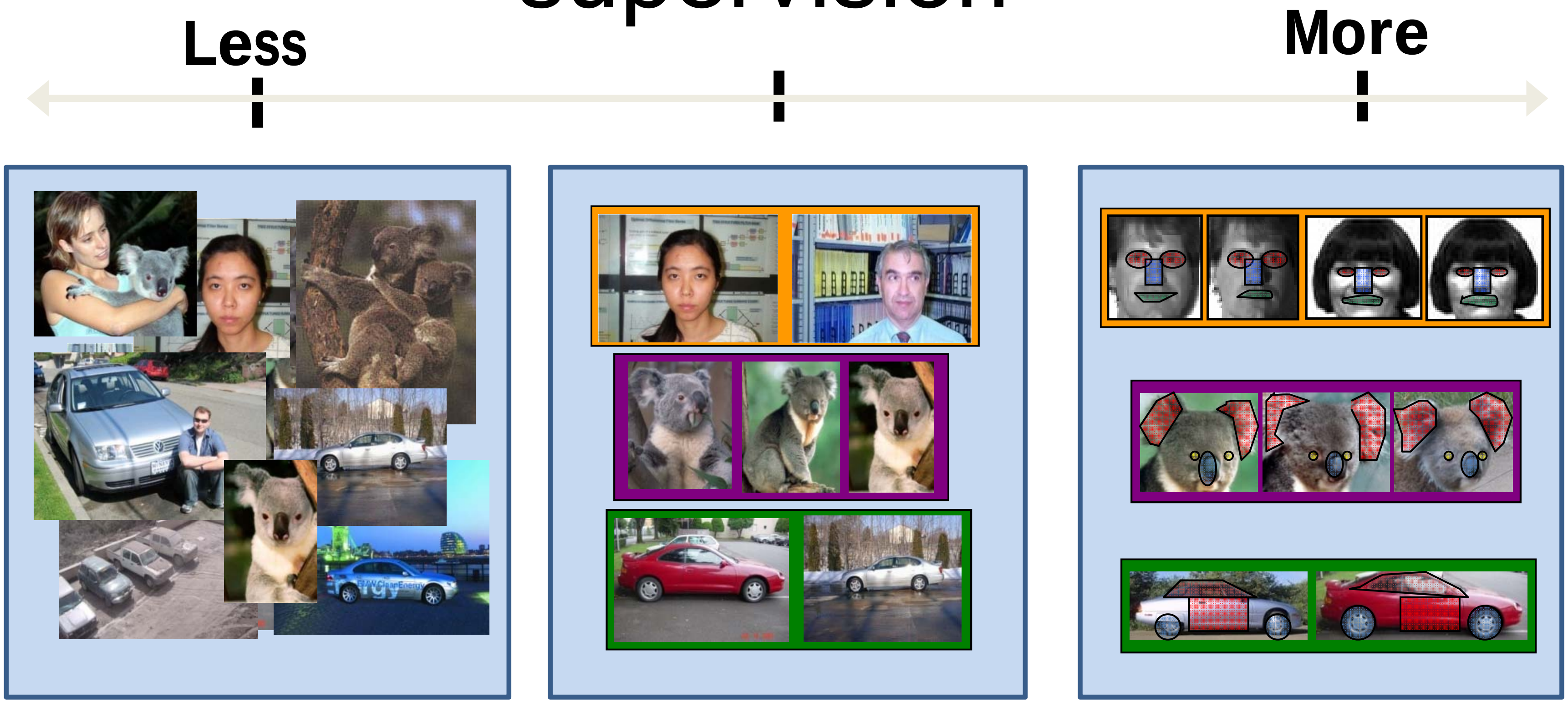


Challenges: complexity

- Thousands to millions of pixels in an image
- 3,000-30,000 human recognizable object categories
- 30+ degrees of freedom in the pose of articulated objects (humans)
- Billions of images indexed by Google Image Search
- About half of the cerebral cortex in primates is devoted to processing visual information [Felleman and van Essen 1991]

Kristen Grauman

Challenges: learning with minimal supervision



Unlabeled, multiple objects

Classes labeled, some clutter

Cropped to object, parts and classes labeled

Kristen Grauman

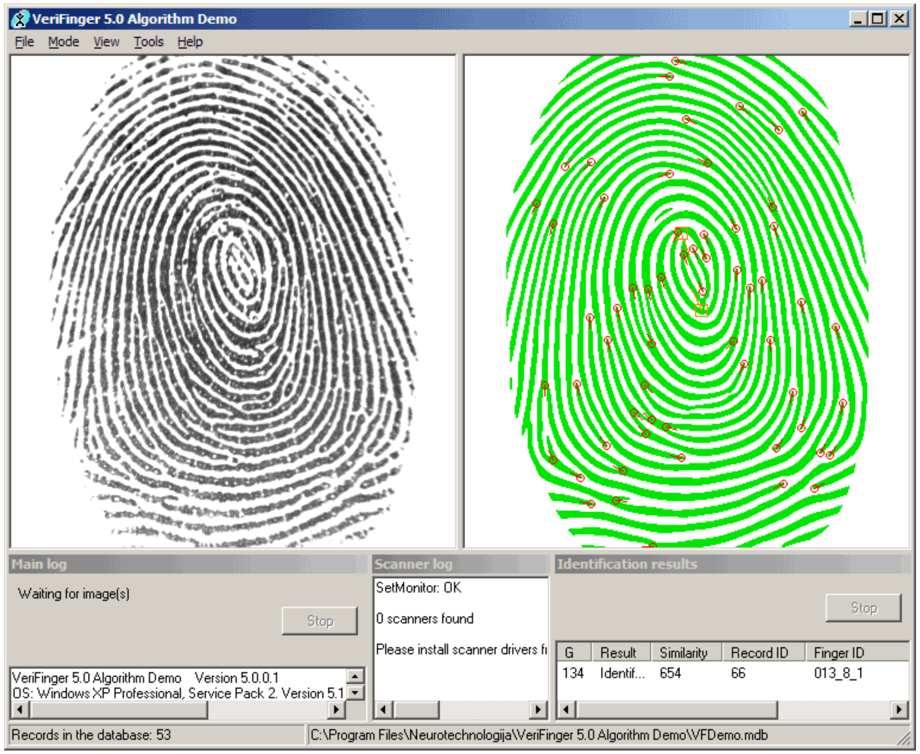
What works most reliably today

- Reading license plates, zip codes, checks

Source: Lana Lazebnik

What works most reliably today

- Reading license plates, zip codes, checks
- Fingerprint recognition



Source: Lana Lazebnik

What works most reliably today

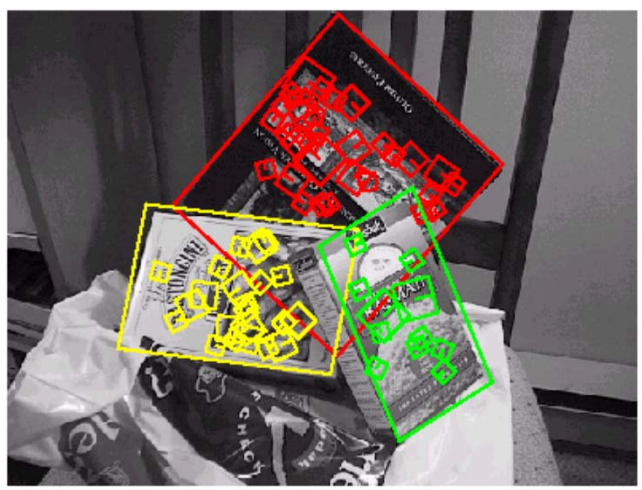
- Reading license plates, zip codes, checks
- Fingerprint recognition
- Face detection

[Face priority AE] When a bright part of the face is too bright

Source: Lana Lazebnik

What works most reliably today

- Reading license plates, zip codes, checks
- Fingerprint recognition
- Face detection
- Recognition of flat textured objects (CD covers, book covers, etc.)



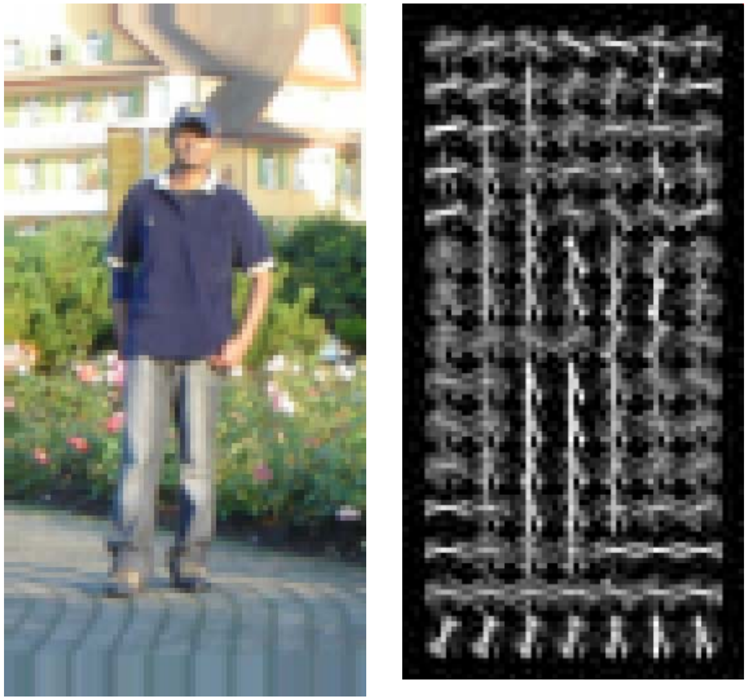
Source: Lana Lazebnik

Generic category recognition: basic framework

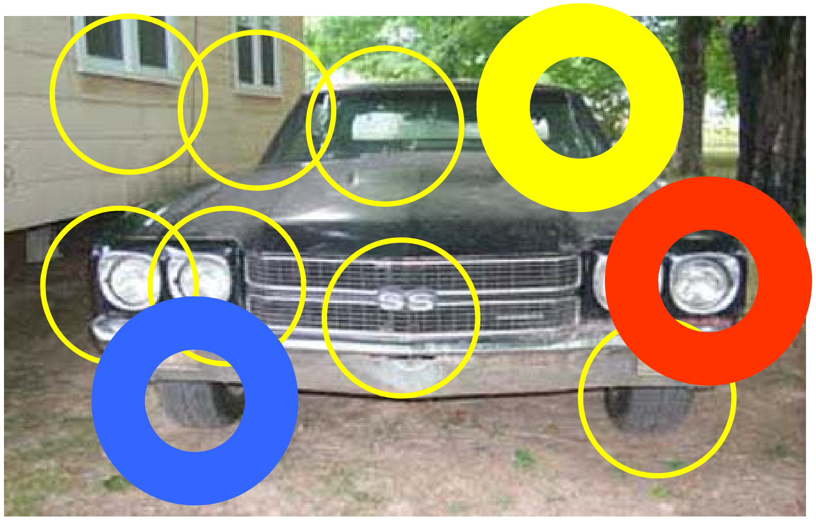
- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- Generate candidates in new image
- Score the candidates

Kristen Grauman

Generic category recognition: representation choice



Window-based



Part-based

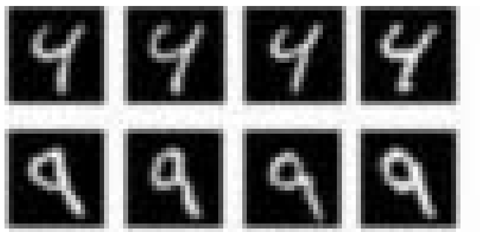
Kristen Grauman

Supervised classification

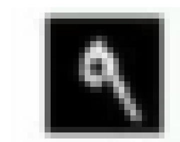
- Given a collection of *labeled* examples, come up with a function that will predict the labels of new examples.

“four”

“nine”



Training examples



?

Novel input

- How good is some function we come up with to do the classification?
- Depends on
  - Mistakes made
  - Cost associated with the mistakes

Kristen Grauman

Supervised classification

- Given a collection of *labeled* examples, come up with a function that will predict the labels of new examples.

- Consider the two-class (binary) decision problem
  - L(4→9): Loss of classifying a 4 as a 9
  - L(9→4): Loss of classifying a 9 as a 4

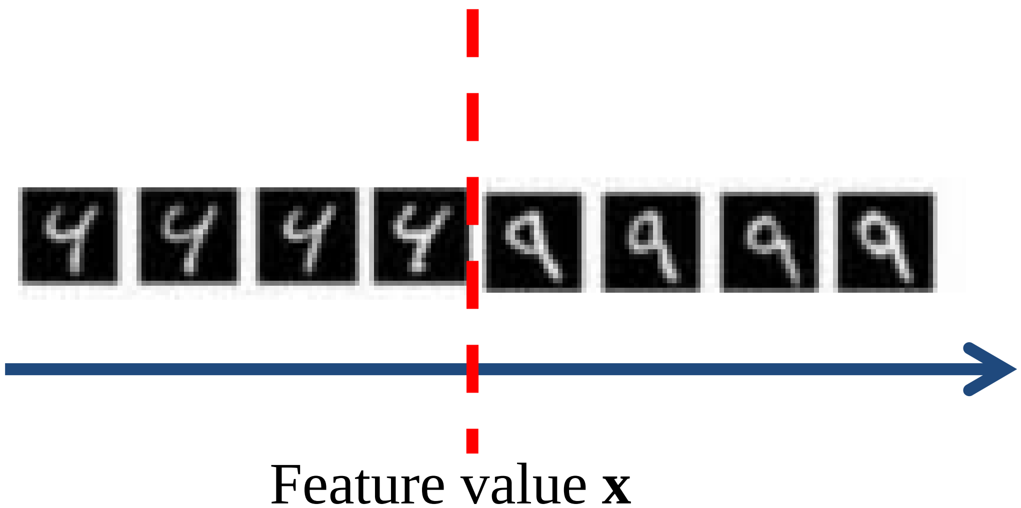
- **Risk** of a classifier *s* is expected loss:

$$R(s) = \Pr\left(4 \to 9 \mid \text{using } s\right) L\left(4 \to 9\right) + \Pr\left(9 \to 4 \mid \text{using } s\right) L\left(9 \to 4\right)$$

- We want to choose a classifier so as to minimize this total risk

Kristen Grauman

Supervised classification



Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

If we choose class "four" at boundary, expected loss is:

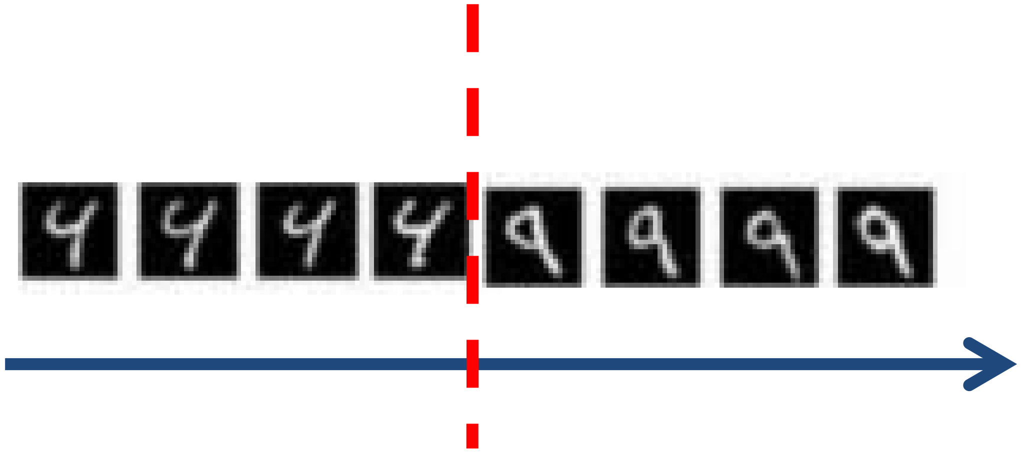
$$= P(\text{class is } 9 \mid \mathbf{x})\, L(9 \to 4) + P(\text{class is } 4 \mid \mathbf{x}) L(4 \to 4)$$

If we choose class "nine" at boundary, expected loss is:

$$= P(\text{class is } 4 \mid \mathbf{x})\, L(4 \to 9)$$

Kristen Grauman

Supervised classification



Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

So, best decision boundary is at point **x** where

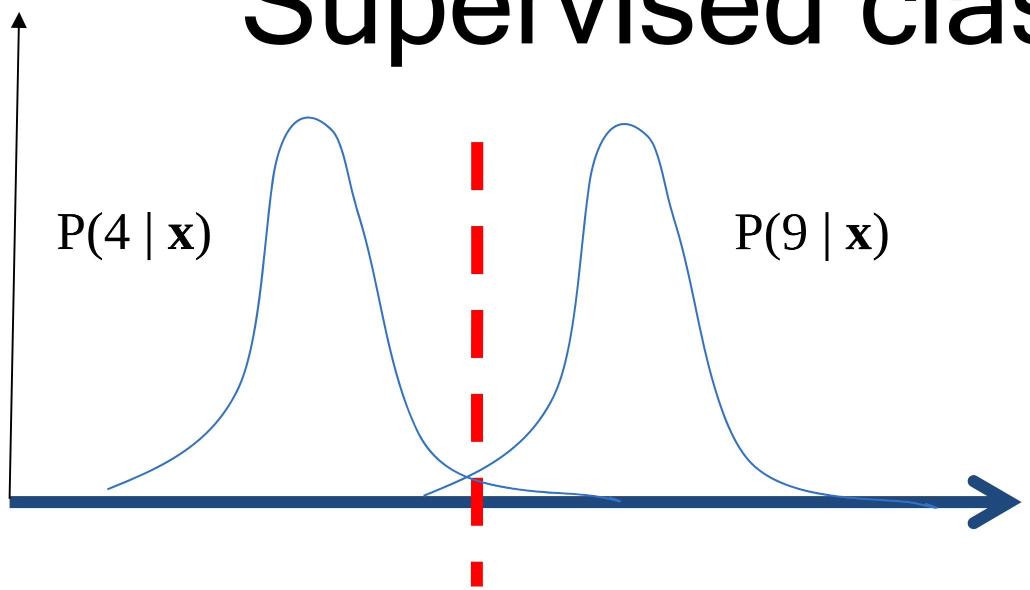
$$P(\text{class is } 9 \mid \mathbf{x})\, L(9 \rightarrow 4) = \mathrm{P}(\text{class is } 4 \mid \mathbf{x}) L(4 \rightarrow 9)$$

To classify a new point, choose class with lowest expected loss; i.e., choose "four" if

$$P(4 \mid \mathbf{x}) L(4 \rightarrow 9) > P(9 \mid \mathbf{x}) L(9 \rightarrow 4)$$

Kristen Grauman

Supervised classification



Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

So, best decision boundary is at point **x** where

$$P(\text{class is } 9 \mid \mathbf{x})\, L(9 \rightarrow 4) = \mathrm{P}(\text{class is } 4 \mid \mathbf{x}) L(4 \rightarrow 9)$$

To classify a new point, choose class with lowest expected loss; i.e., choose "four" if

$$\boxed{P(4 \mid \mathbf{x})} L(4 \rightarrow 9) > \boxed{P(9 \mid \mathbf{x})} L(9 \rightarrow 4)$$

How to evaluate these probabilities?

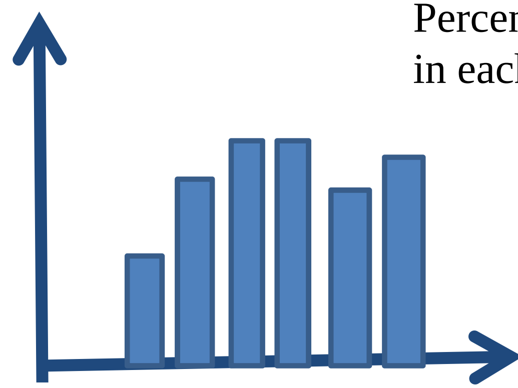
Kristen Grauman

Example: learning skin colors

- We can represent a class-conditional density using a histogram (a “non-parametric” distribution)

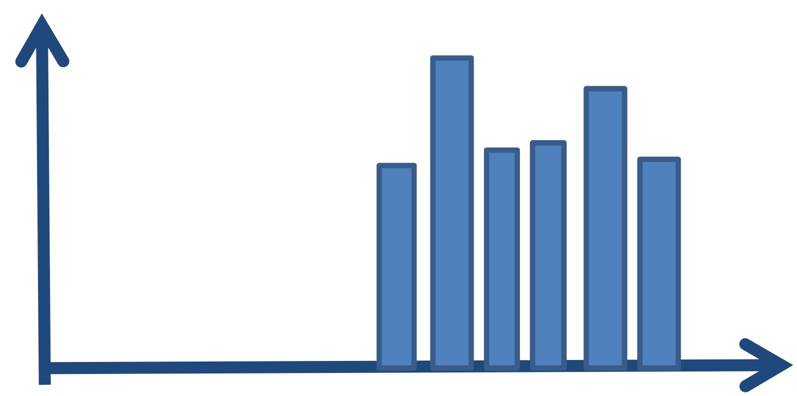
Percentage of skin pixels in each bin

P(x|skin)



Feature x = Hue

P(x|not skin)

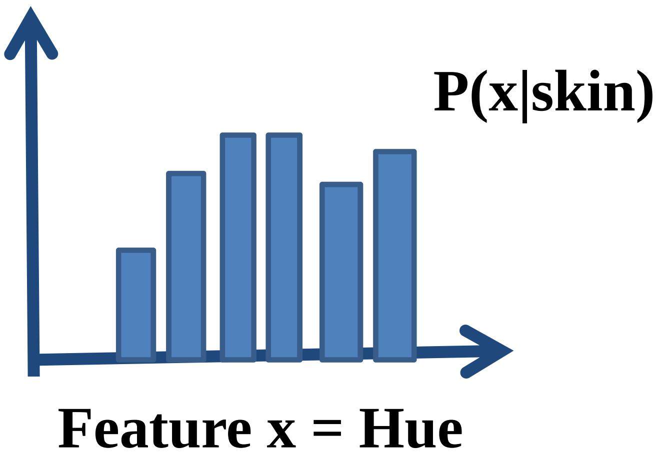


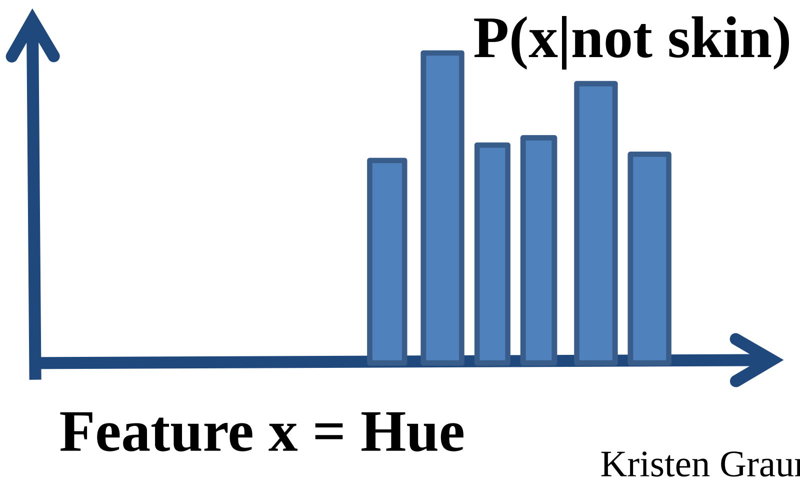
Feature x = Hue

Kristen Grauman

Example: learning skin colors

- We can represent a class-conditional density using a histogram (a “non-parametric” distribution)





Now we get a new image, and want to label each pixel as skin or non-skin.

What’s the probability we care about to do skin detection?

Kristen Grauman

Bayes rule

$$\underbrace{P(skin \mid x)}_{\text{posterior}} = \frac{\underbrace{P(x \mid skin)}_{\text{likelihood}}\underbrace{P(skin)}_{\text{prior}}}{P(x)}$$

$$P(skin \mid x) \,\alpha\, P(x \mid skin)P(skin)$$

Where does the prior come from?

Why use a prior?

Example: classifying skin pixels

Now for every pixel in a new image, we can estimate probability that it is generated by skin.

Brighter pixels → higher probability of being skin

Classify pixels based on these probabilities

- if $p(\mathrm{skin}|\boldsymbol{x}) > \theta$, classify as skin
- if $p(\mathrm{skin}|\boldsymbol{x}) < \theta$, classify as not skin

Example: classifying skin pixels

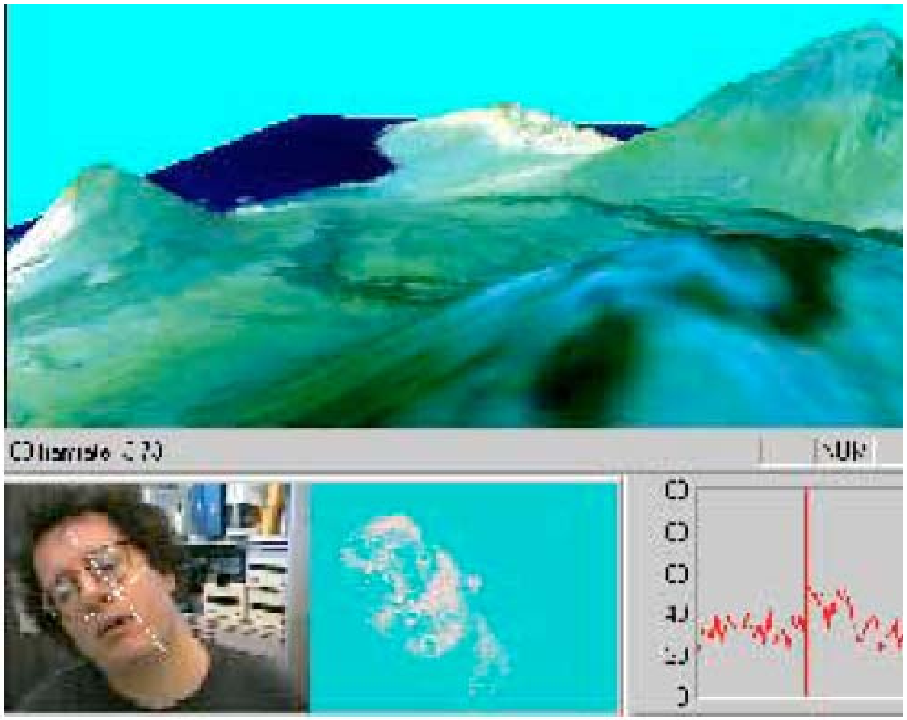


Figure 13: CAMSHIFT-based face tracker used to over a 3D graphic's model of Hawaii

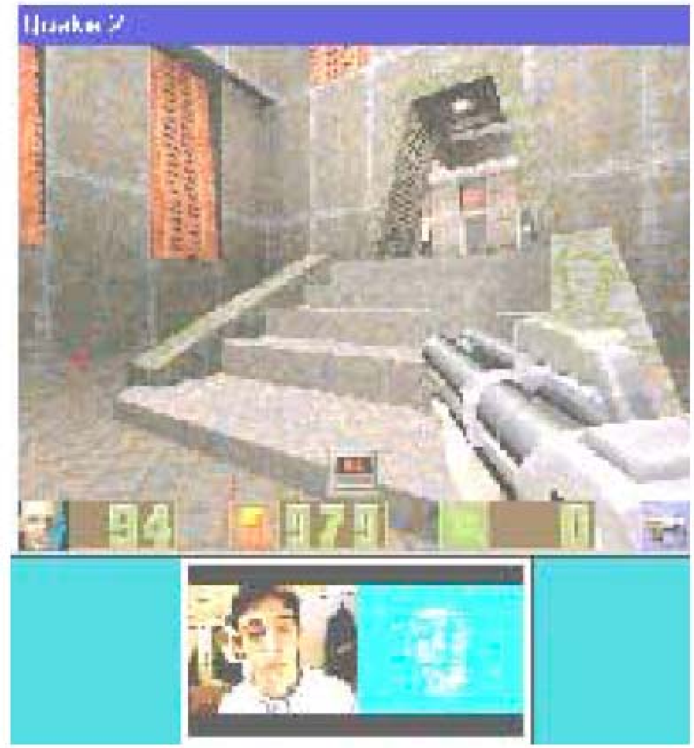


Figure 12: CAMSHIFT-based face tracker used to play Quake 2 hands free by inserting control variables into the mouse queue

Using skin color-based face detection and pose estimation as a video-based interface

Gary Bradski, 1998

Kristen Grauman

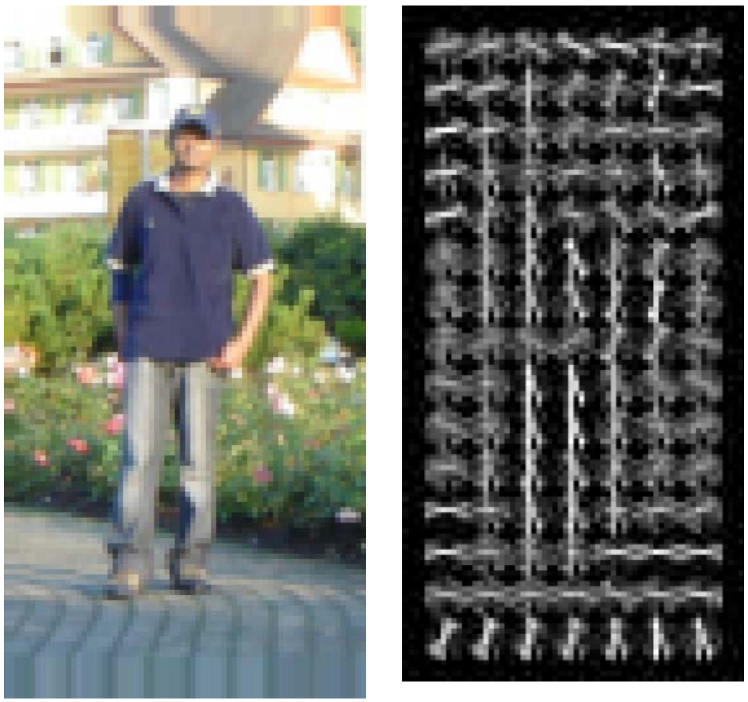
Supervised classification

- Want to minimize the expected misclassification
- Two general strategies
  - Use the training data to build representative probability model; separately model class-conditional densities and priors (*generative*)
  - Directly construct a good decision boundary, model the posterior (*discriminative*)

Generic category recognition: basic framework

- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- Generate candidates in new image
- Score the candidates

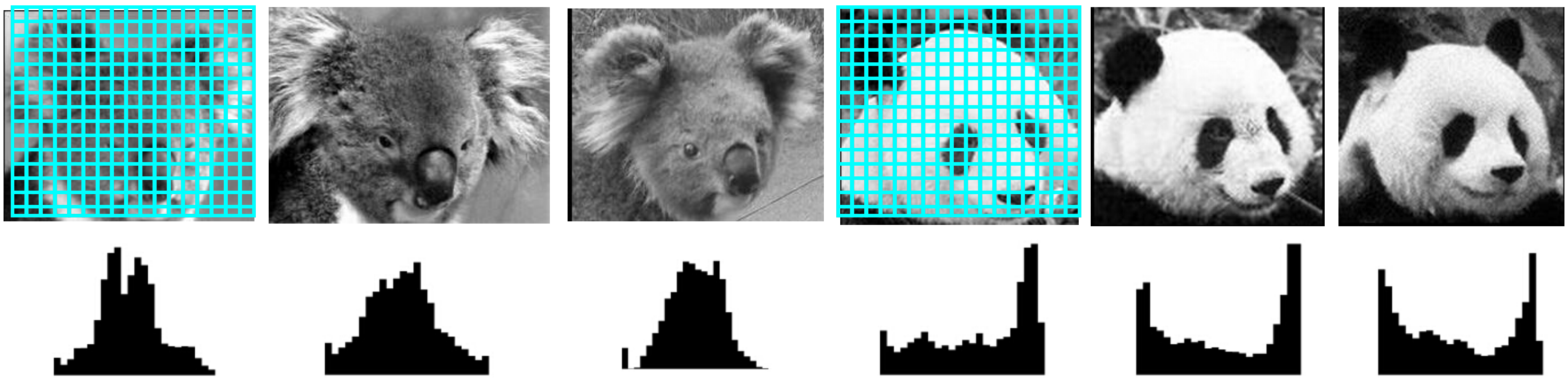
Generic category recognition: representation choice



Window-based

Part-based

Window-based models
Building an object model



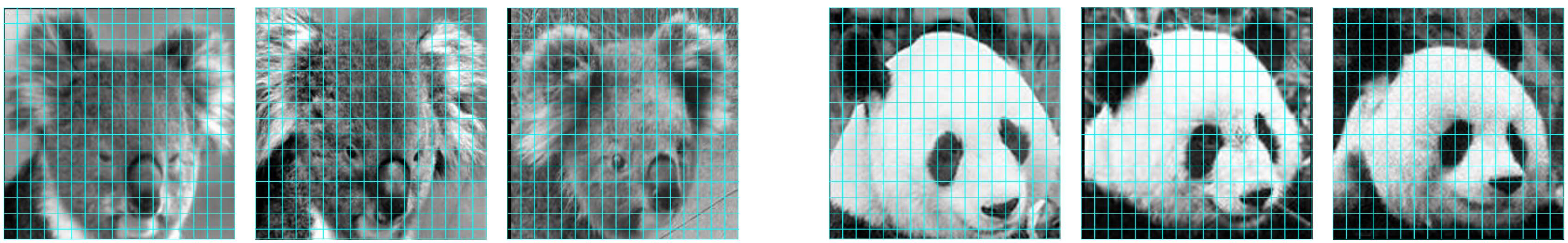
Simple holistic descriptions of image content

- **grayscale / color histogram**
- **vector of pixel intensities**

Kristen Grauman

Window-based models
Building an object model

- **Pixel-based representations sensitive to small shifts**

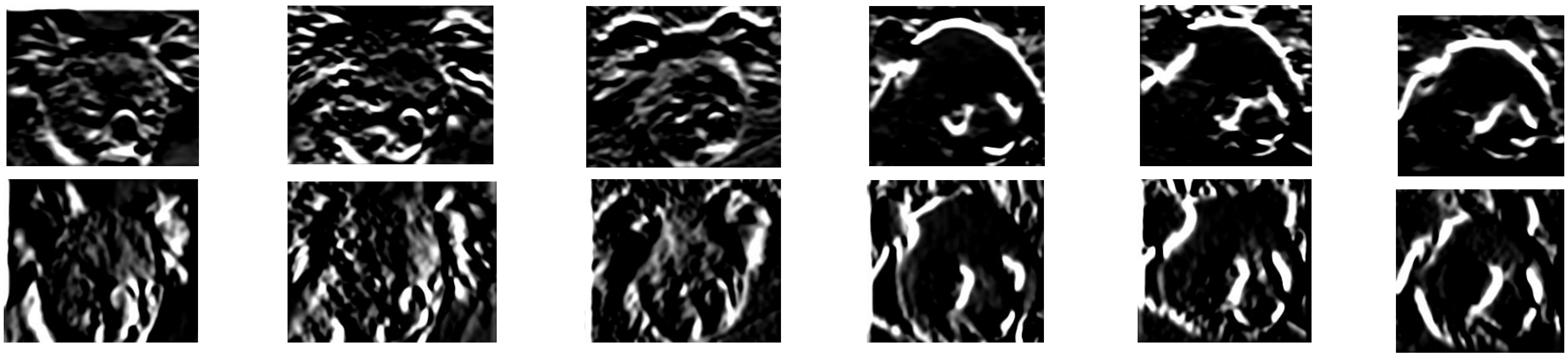


- **Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation**

Kristen Grauman

Window-based models
Building an object model

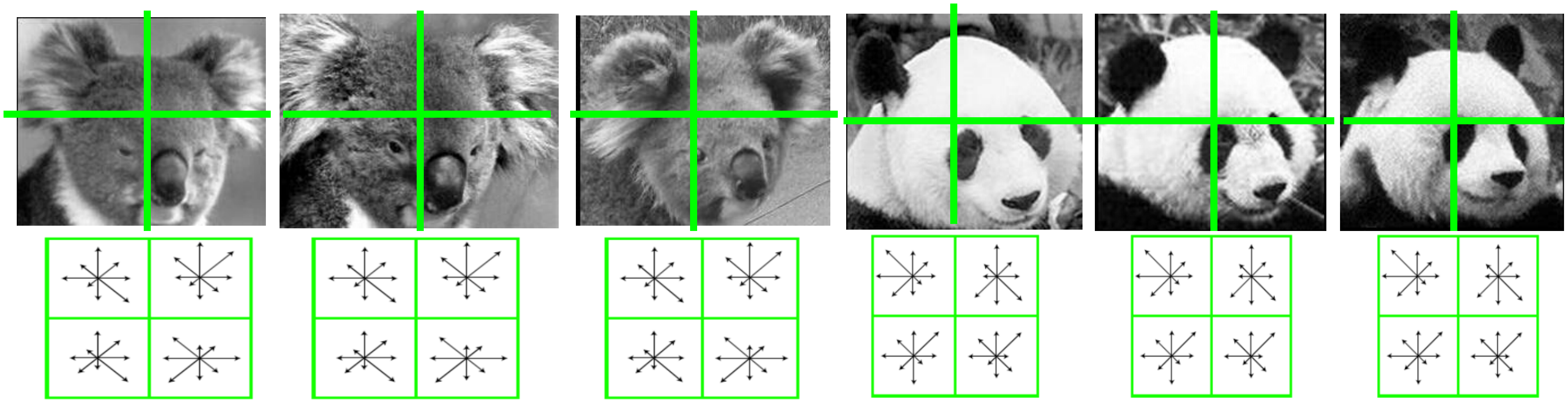
- **Consider edges, contours, and (oriented) intensity gradients**



Kristen Grauman

Window-based models
Building an object model

- Consider edges, contours, and (oriented) intensity gradients



- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination

Kristen Grauman

Window-based models
Building an object model

Given the representation, train a binary classifier

Car/non-car Classifier

Yes, car.

Car/non-car Classifier

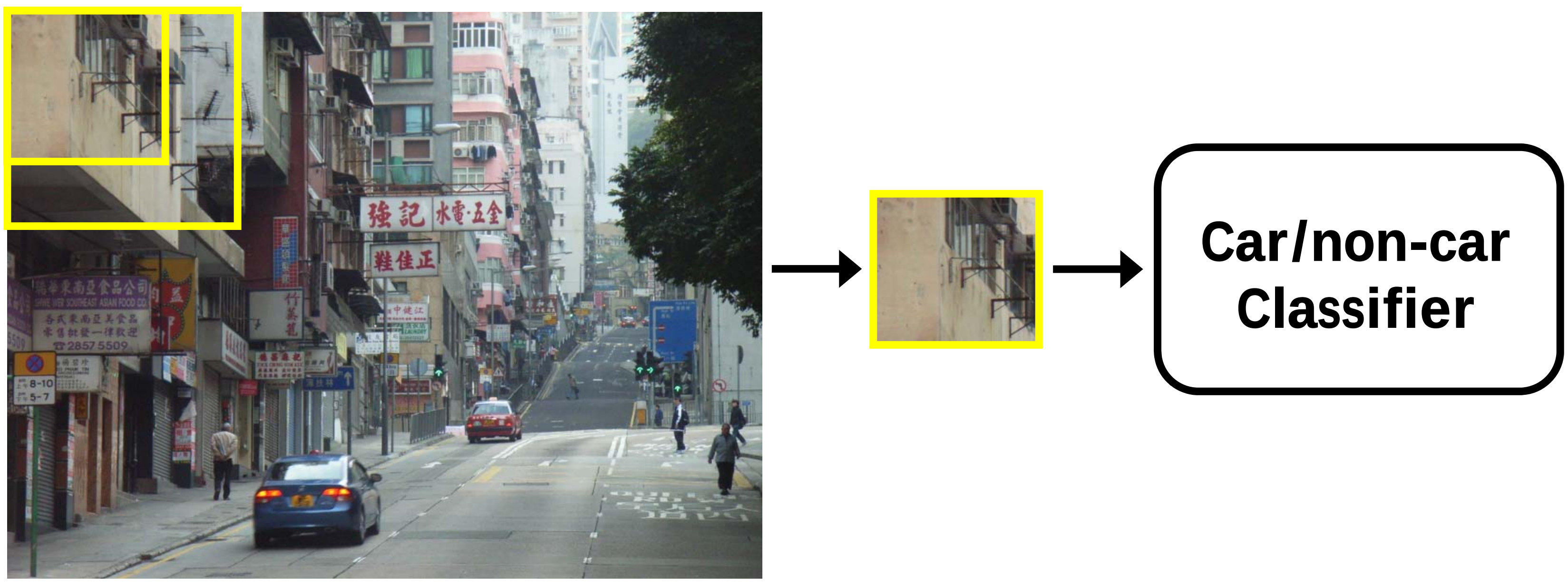
No, not a car.

Kristen Grauman

Generic category recognition: basic framework

- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- **Generate candidates in new image**
- **Score the candidates**

Window-based models
Generating and scoring candidates

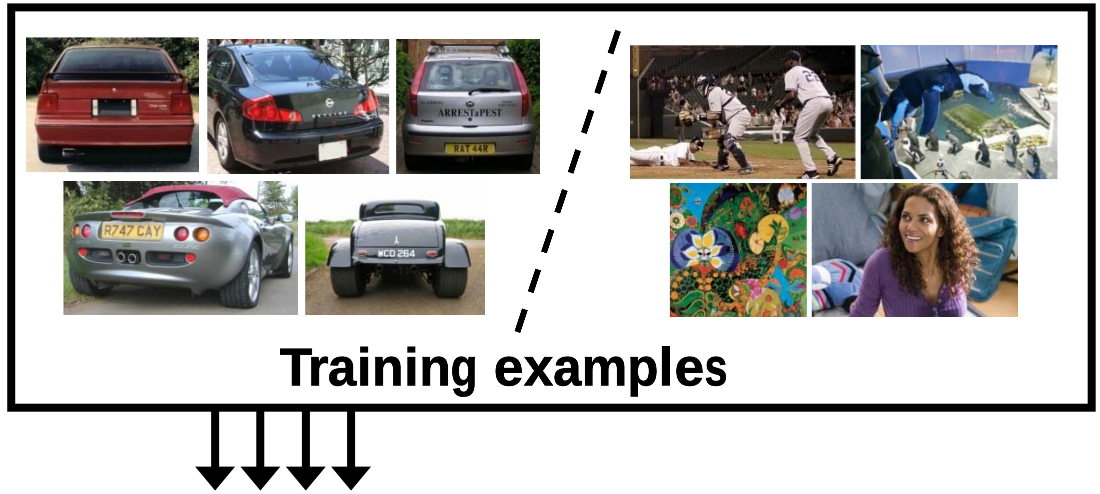


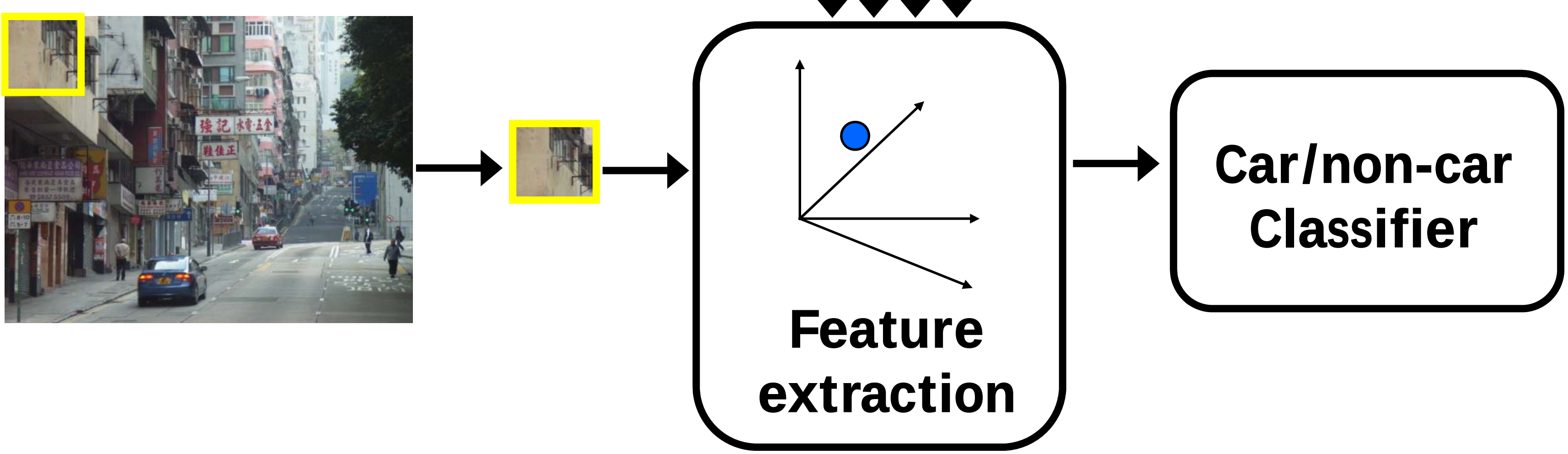
Kristen Grauman

Window-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

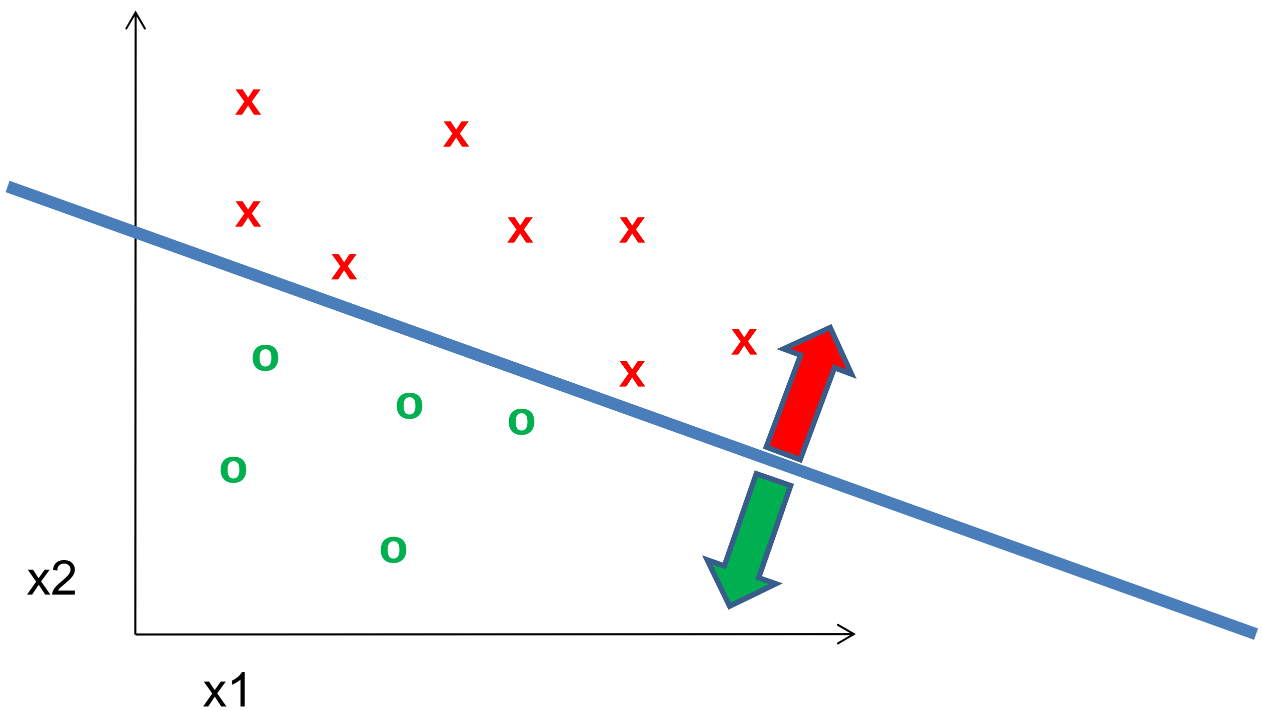
Given new image:
1. Slide window
2. Score by classifier





Classifier

A classifier maps from the feature space to a label



Derek Hoiem

Different types of classification

- Exemplar-based: transfer category labels from examples with most similar features
  - What similarity function? What parameters?
- Linear classifier: confidence in positive label is a weighted sum of features
  - What are the weights?
- Non-linear classifier: predictions based on more complex function of features
  - What form does the classifier take? Parameters?
- Generative classifier: assign to the label that best explains the features (makes features most likely)
  - What is the probability function and its parameters?

Note: You can always fully design the classifier by hand, but usually this is too difficult. Typical solution: learn from training examples.

Derek Hoiem

One way to think about it…

- Training labels dictate that two examples are the same or different, in some sense
- Features and distance measures define visual similarity
- Goal of training is to learn feature weights or distance measures so that visual similarity predicts label similarity
- *We want the simplest function that is confidently correct*

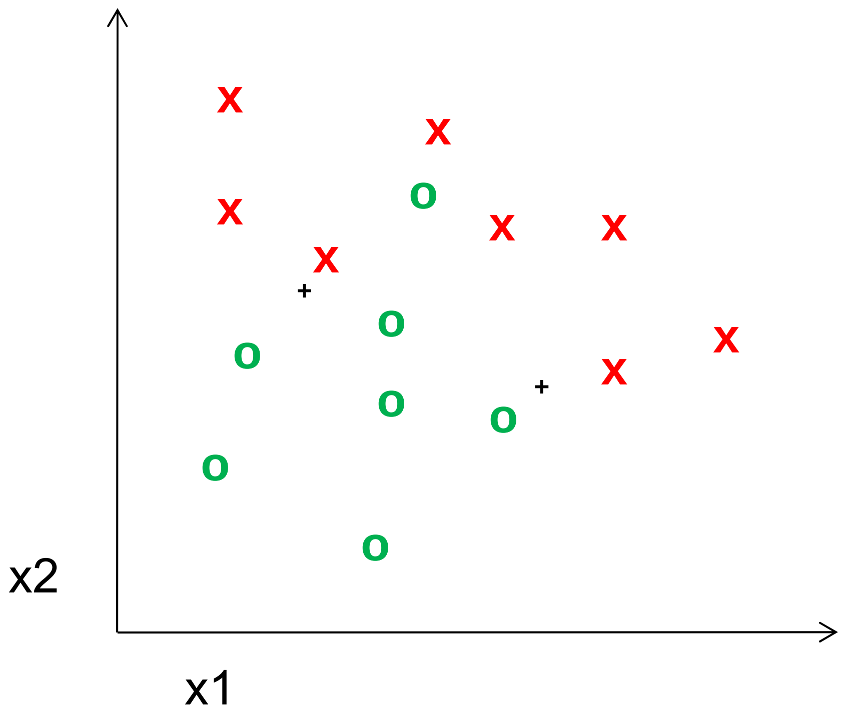
Derek Hoiem

Exemplar-based Models

- Transfer the label(s) of the most similar training examples

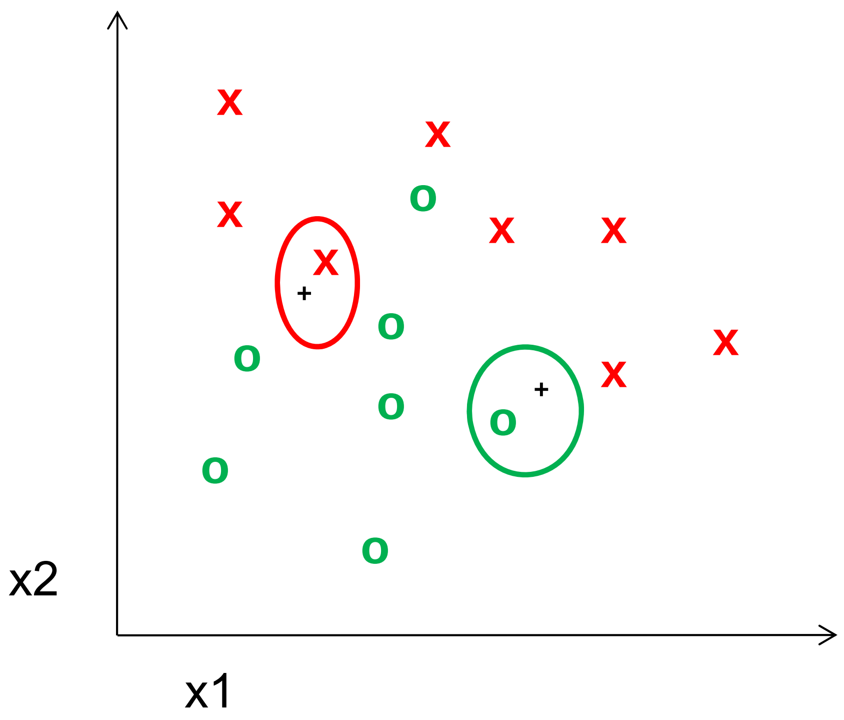
Derek Hoiem

K-nearest neighbor classifier



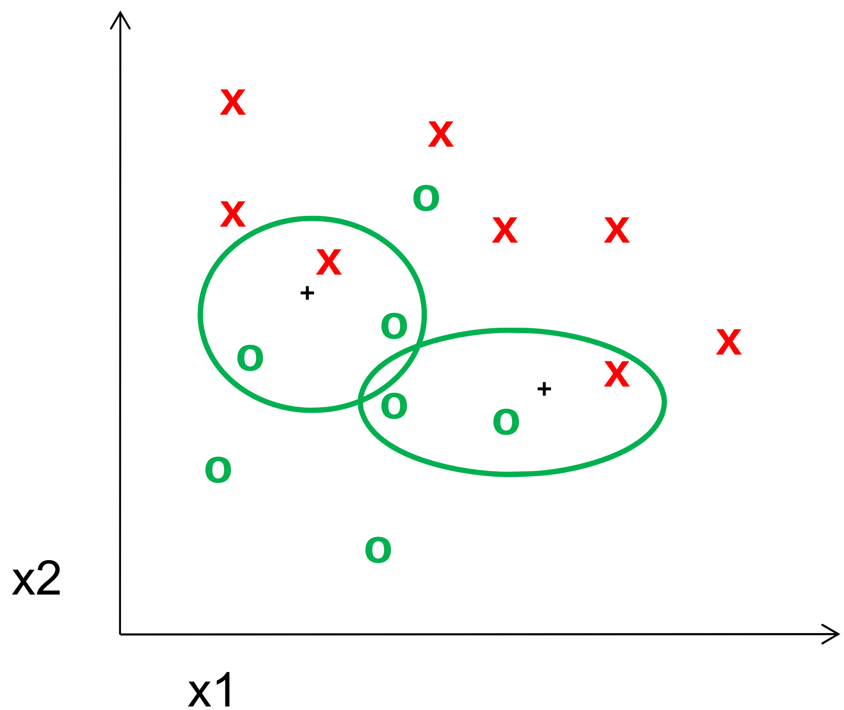
Derek Hoiem

1-nearest neighbor



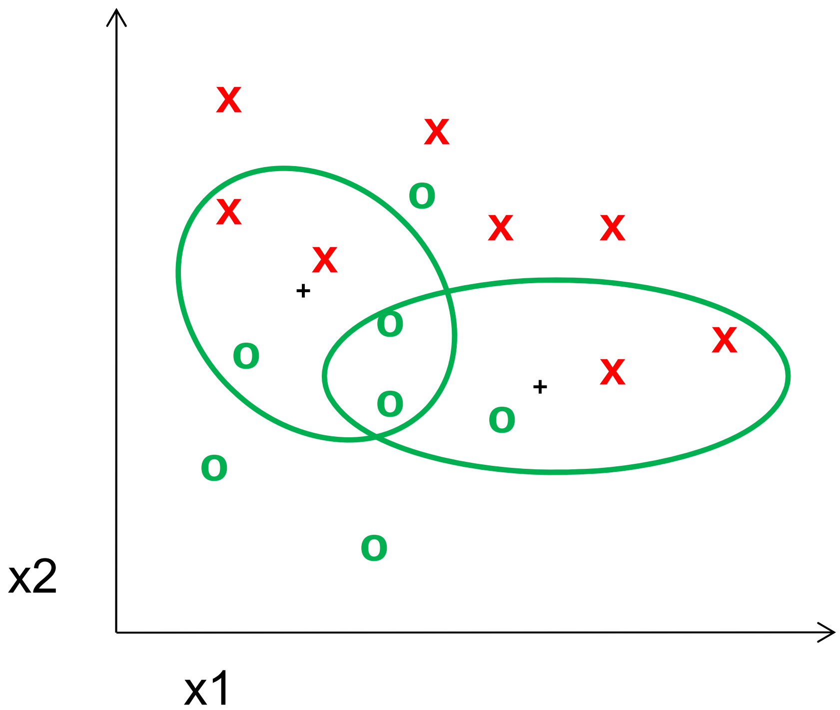
Derek Hoiem

3-nearest neighbor



Derek Hoiem

5-nearest neighbor



Derek Hoiem

Using K-NN

- Simple, a good classifier to try first

- No training time (unless you want to learn a distance function)

- With infinite examples, 1-NN provably has error that is at most twice Bayes optimal error

Derek Hoiem

Discriminative classifier construction

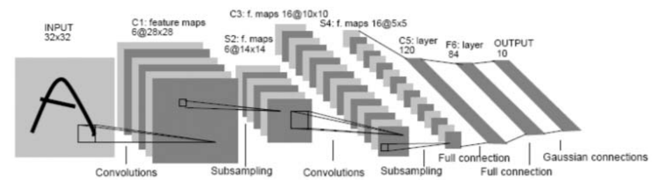
Nearest neighbor



10^6 examples

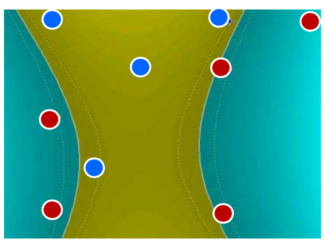
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

Neural networks



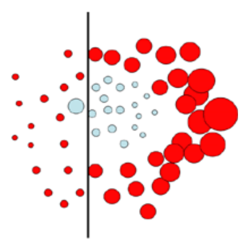
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Support Vector Machines



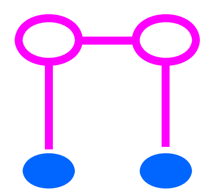
Guyon, Vapnik
Heisele, Serre, Poggio, 2001,…

Boosting



Viola, Jones 2001, Torralba et al. 2004, Opelt et al. 2006,…

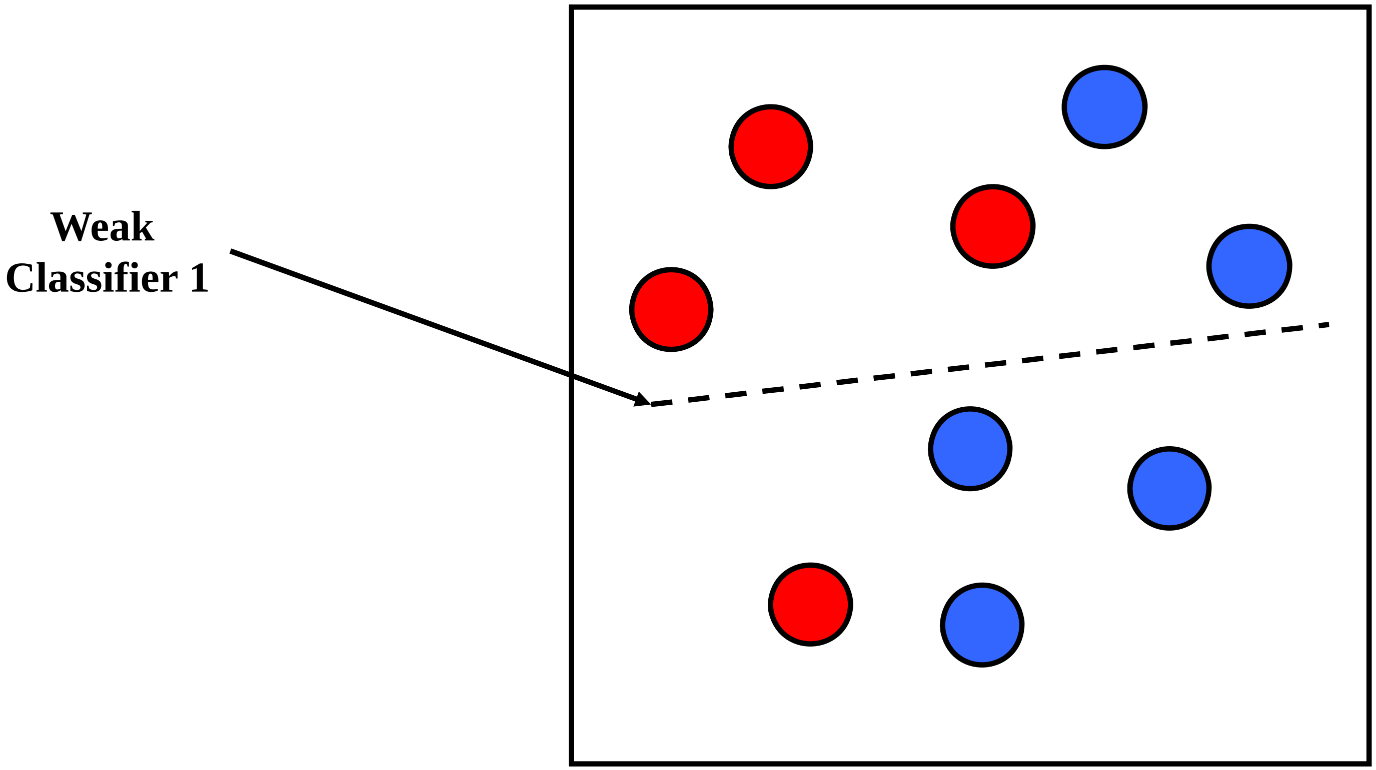
Conditional Random Fields



McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003
…

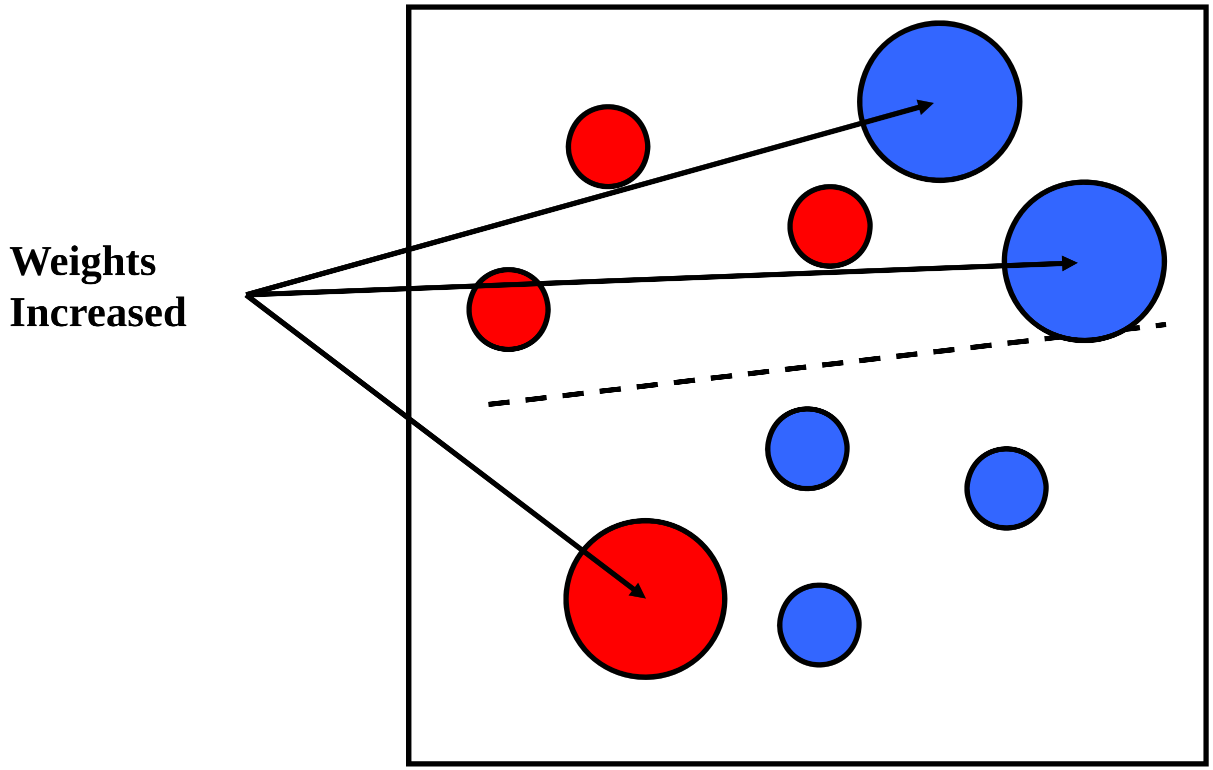
Slide adapted from Antonio Torralba

Boosting intuition

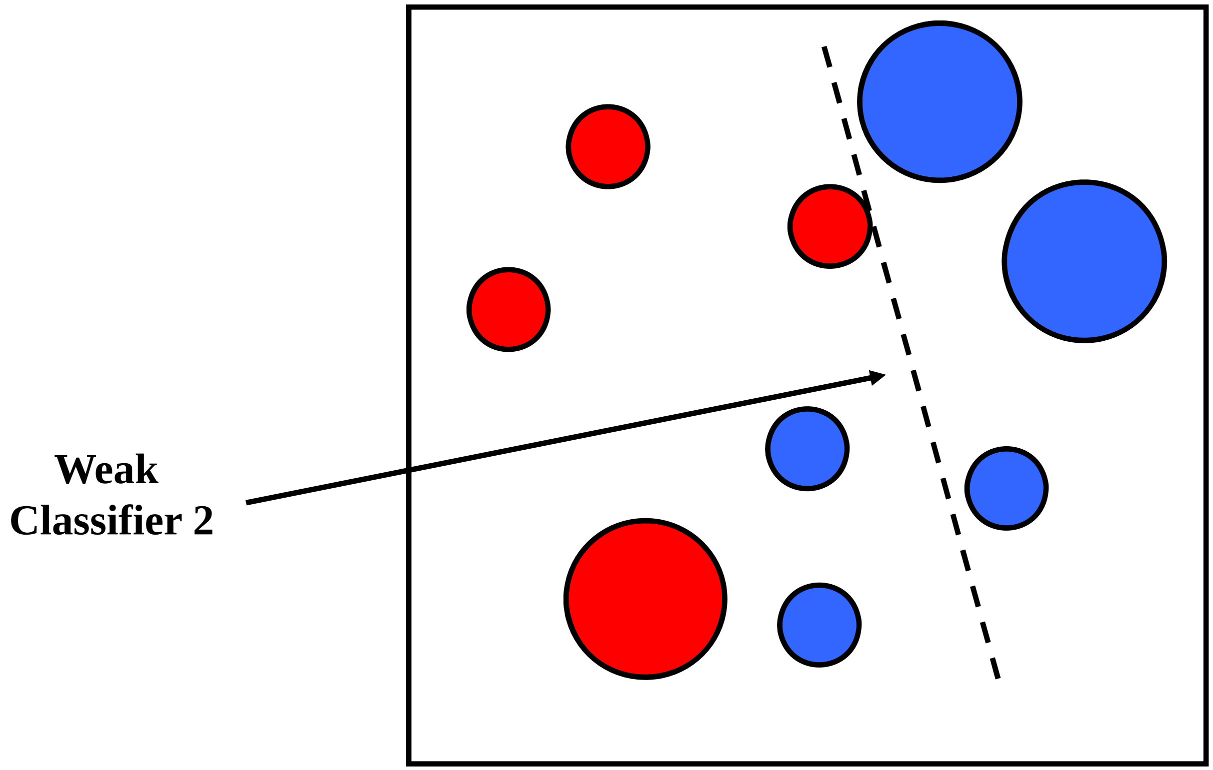


Slide credit: Paul Viola

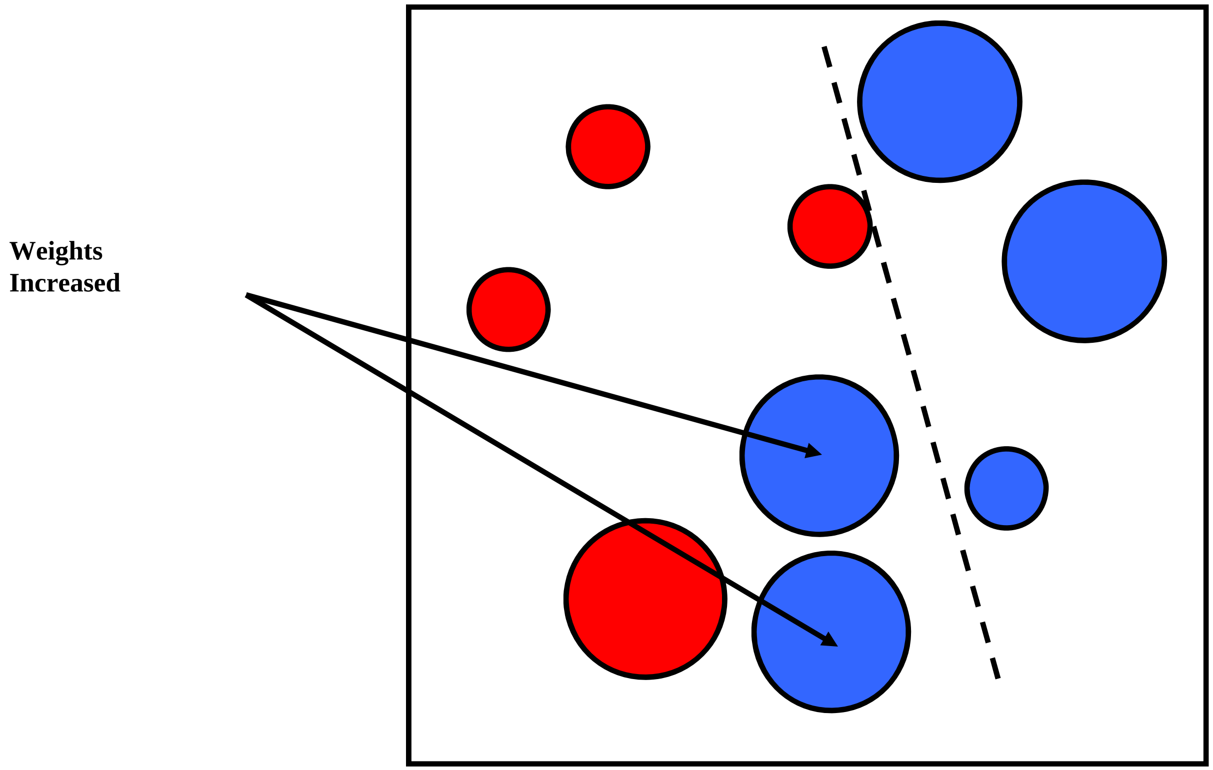
Boosting illustration



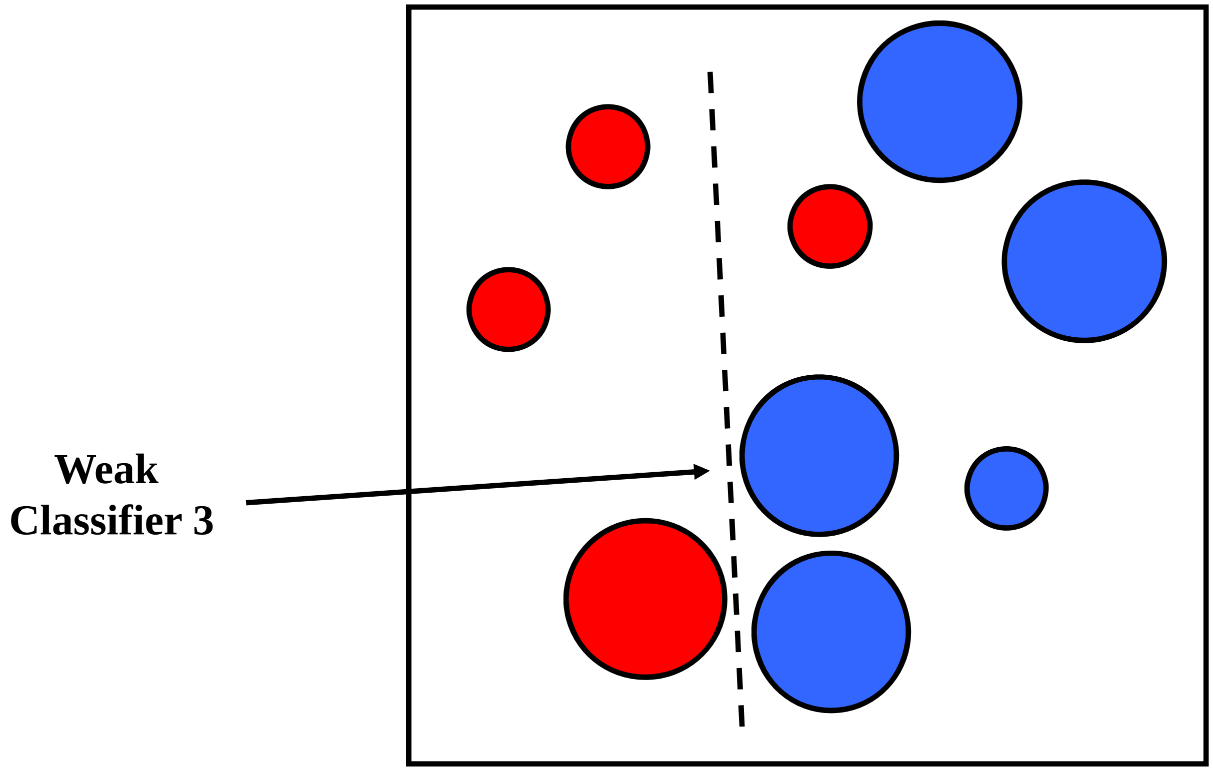
Boosting illustration



Boosting illustration

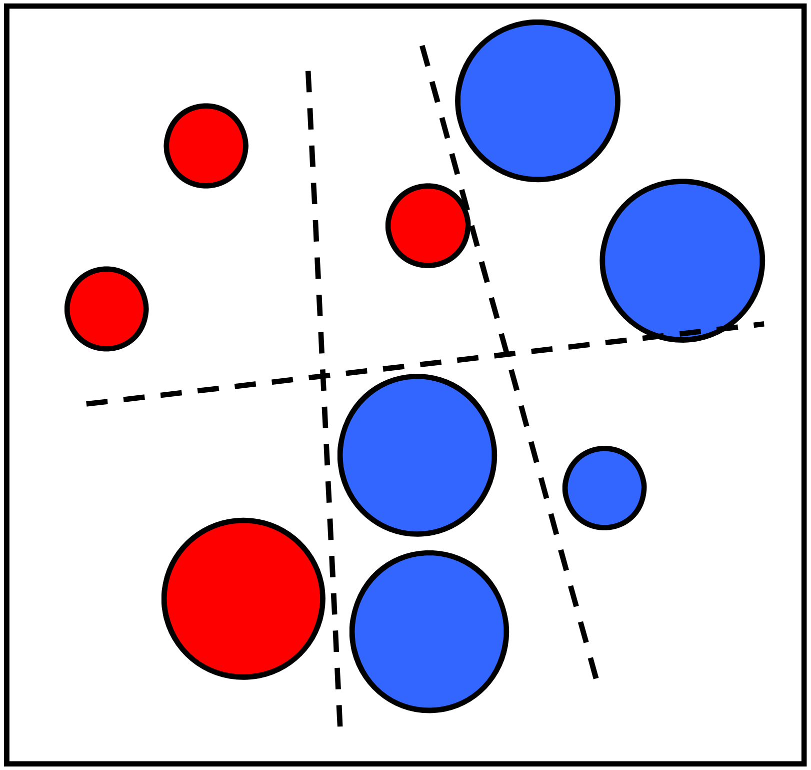


Boosting illustration



Boosting illustration

Final classifier is
a combination of weak
classifiers



Boosting: training

- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner that achieves the lowest *weighted* training error
  - Raise 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)

Slide credit: Lana Lazebnik

Challenges of face detection

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

The Viola/Jones Face Detector

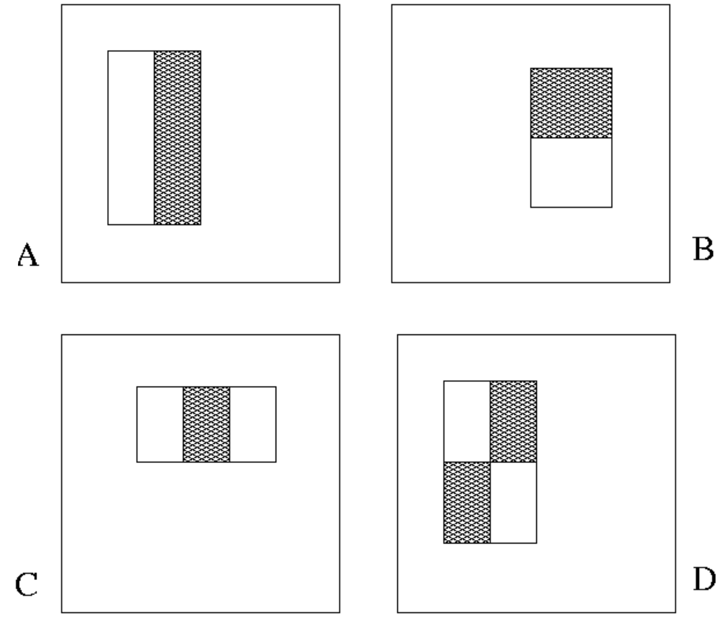
- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - *Integral images* for fast feature evaluation
  - *Boosting* for feature selection
  - *Attentional cascade* for fast rejection of non-face windows

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

"Rectangle filters"

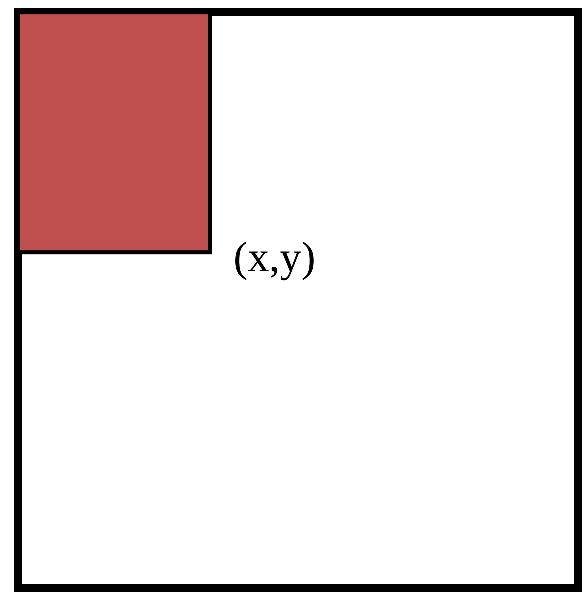


Value =

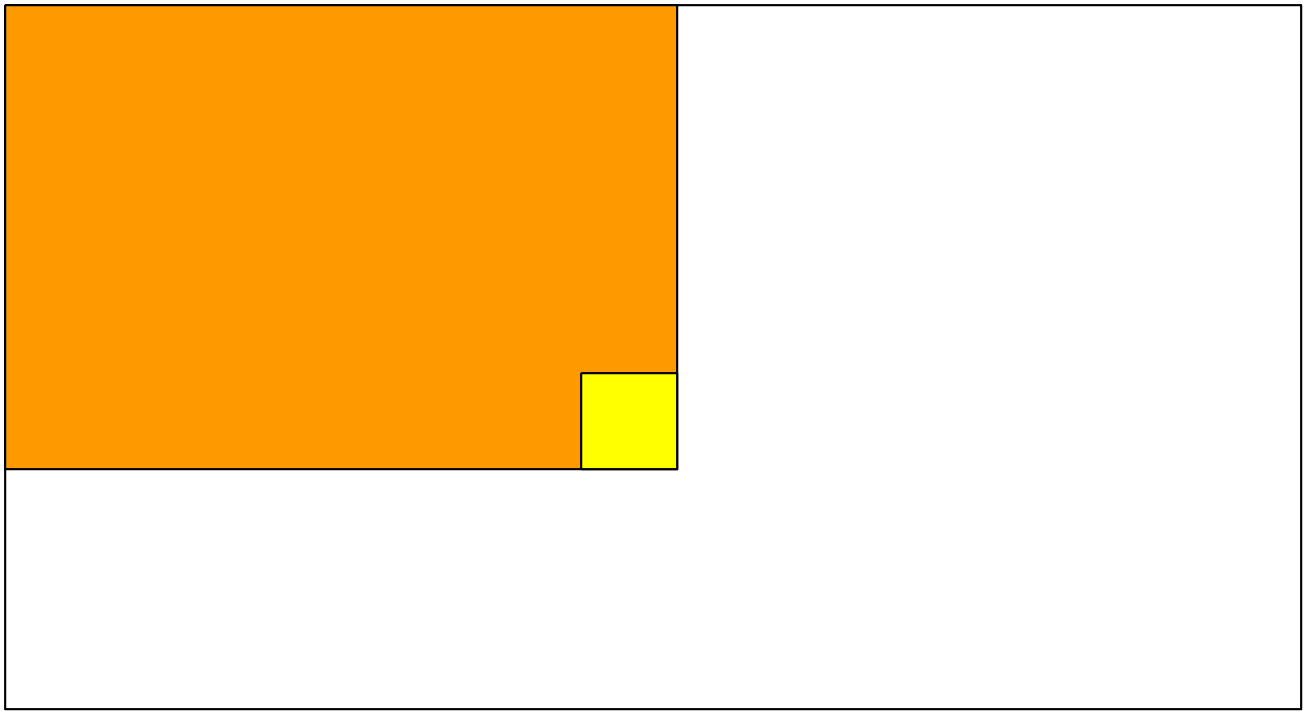
$\sum$ *(pixels in white area) –*
$\sum$ *(pixels in black area)*

Fast computation with integral images

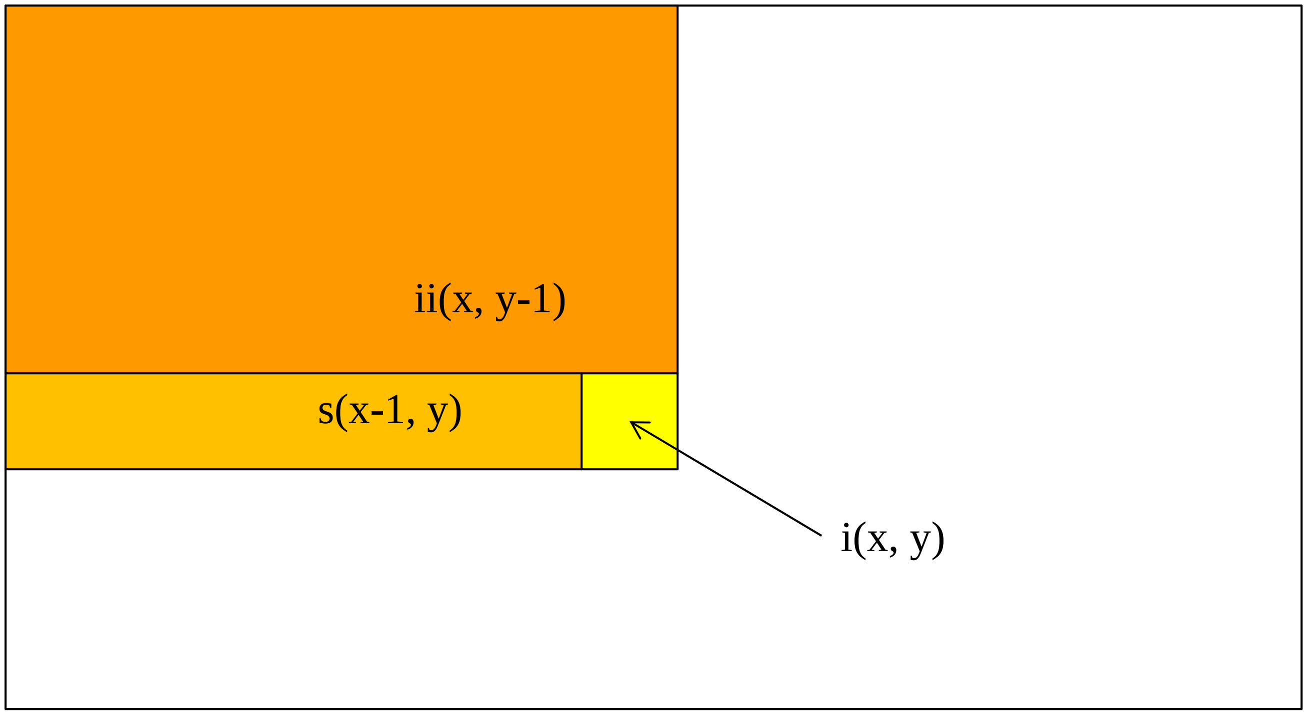
- The *integral image* computes a value at each pixel (x,y) that is the sum of the pixel values above and to the left of (x,y), inclusive
- This can quickly be computed in one pass through the image



Computing the integral image



Computing the integral image

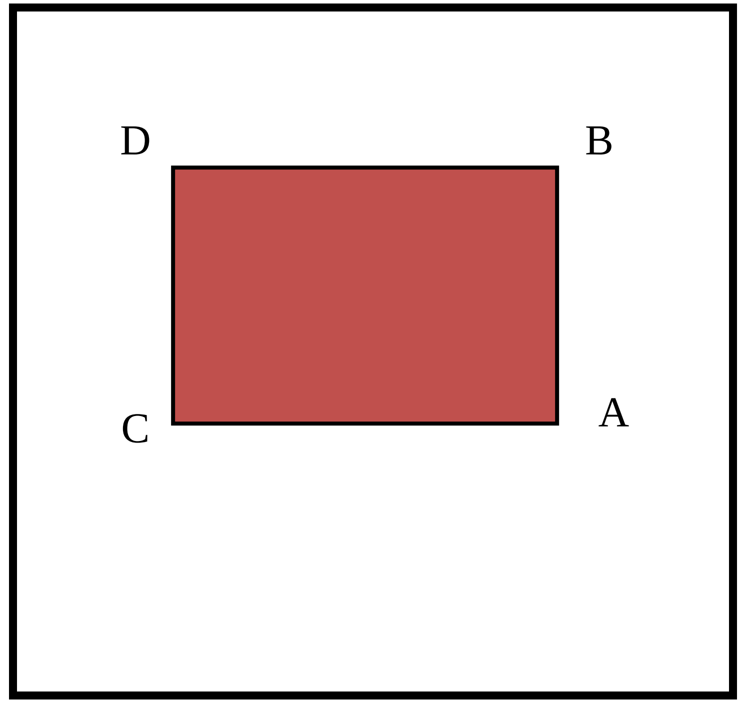


- Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$
- Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$

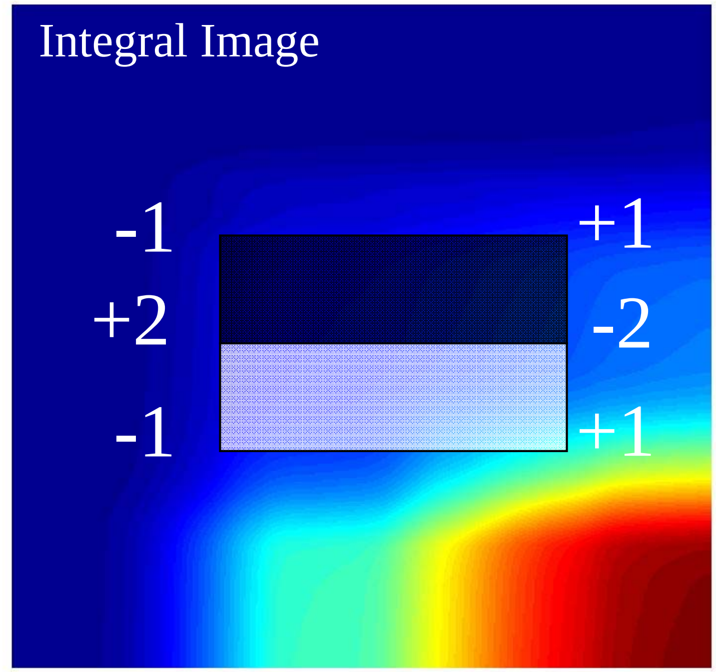
Computing sum within a rectangle

- 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!

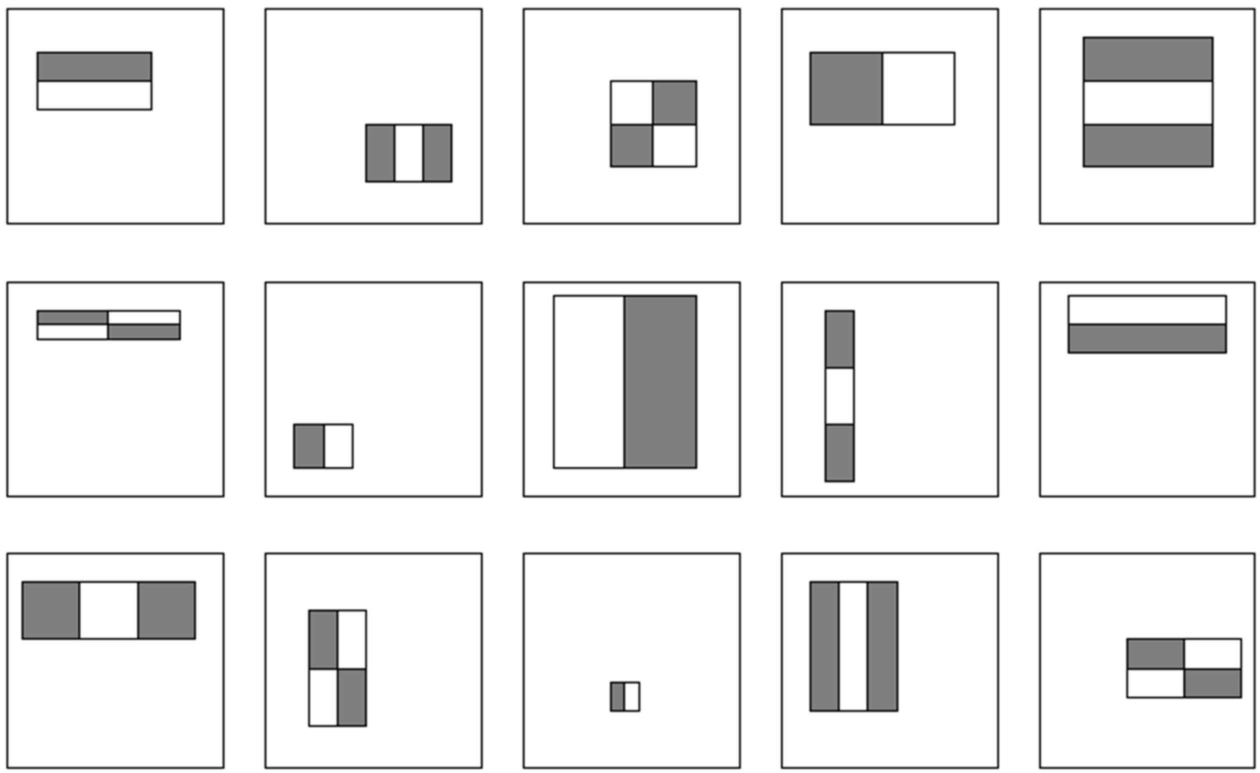


Computing a rectangle feature



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* combines *weak learners* into a more accurate *ensemble classifier*
- Weak learners based on rectangle filters:

value of rectangle feature

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

window

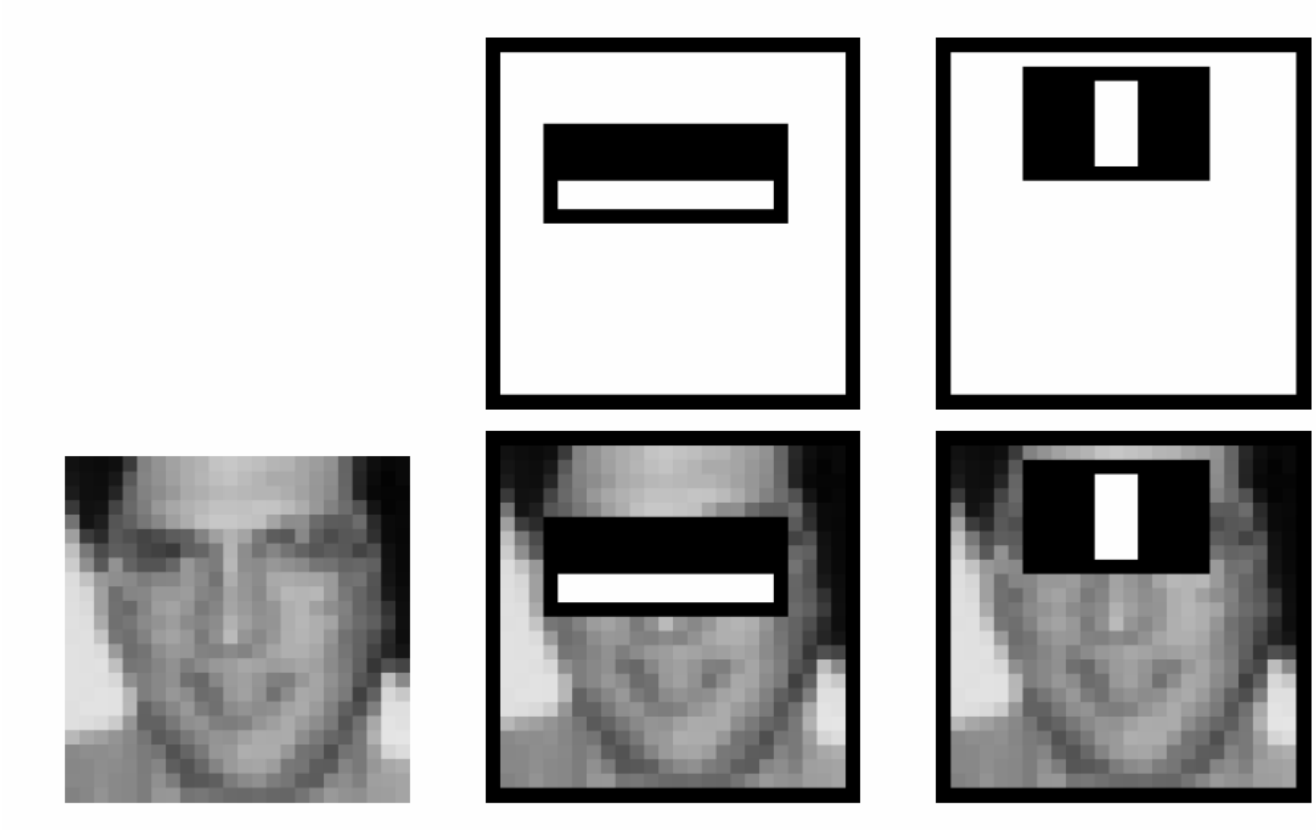
threshold

- 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

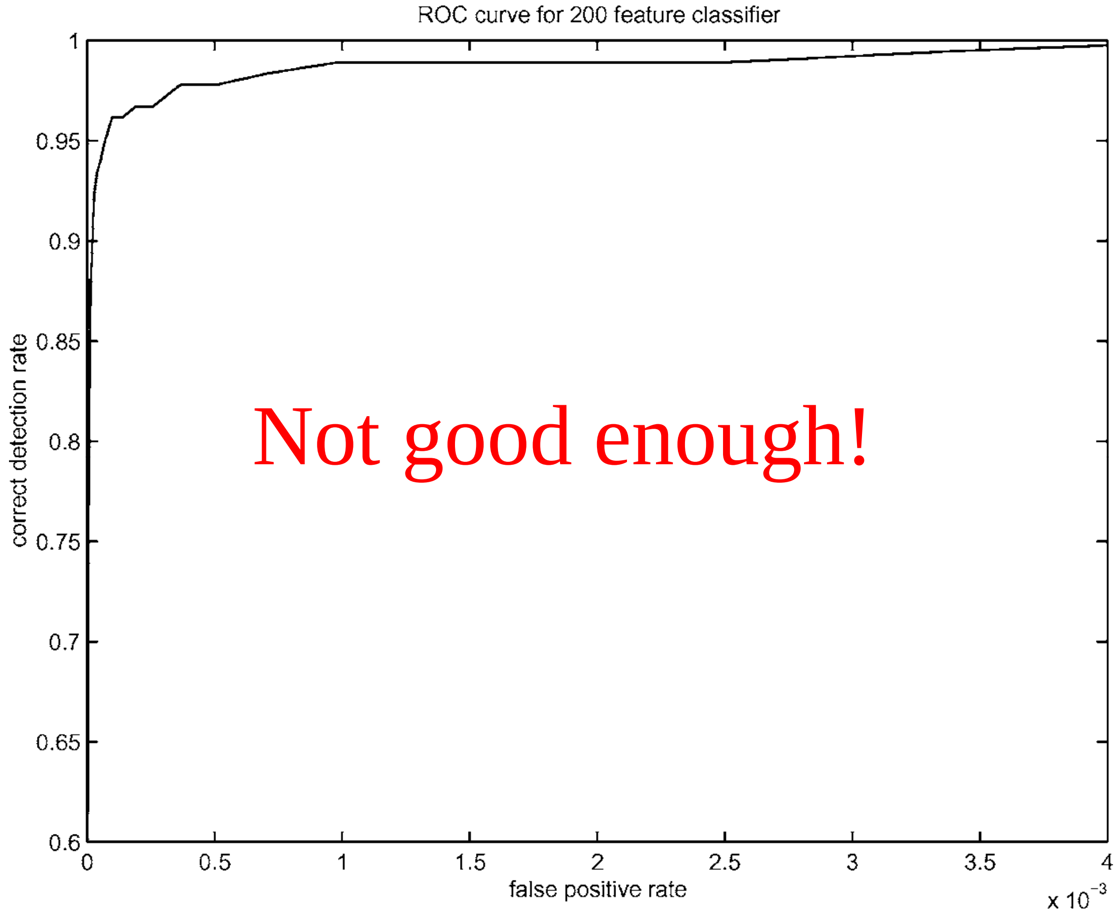
Boosting for face detection

- First two features selected by boosting:
- 

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

Boosting for face detection

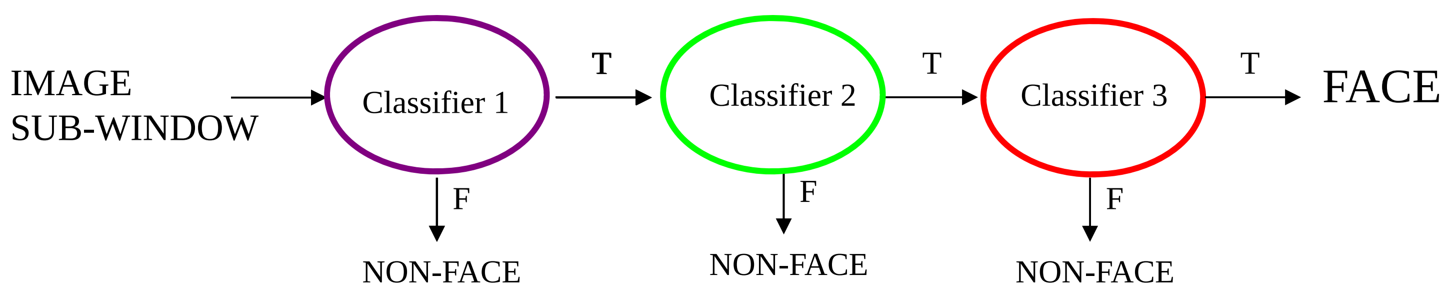
- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084



Receiver operating characteristic (ROC) curve

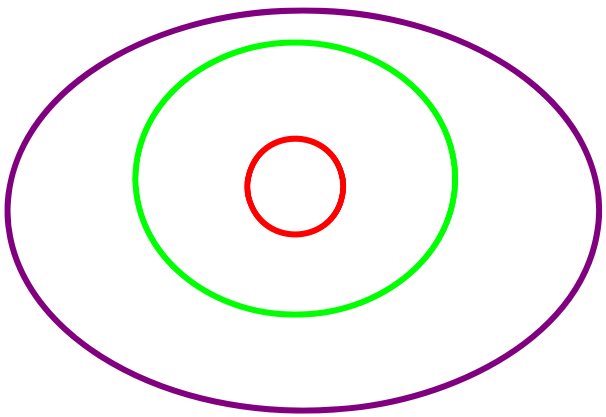
Attentional cascade

- 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

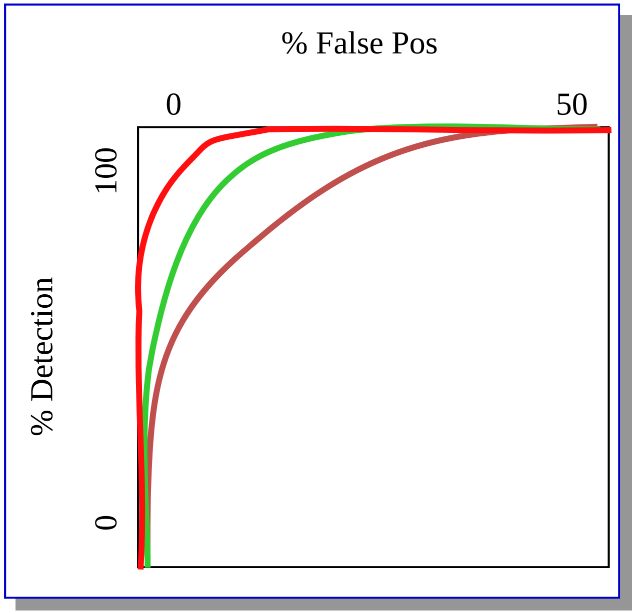


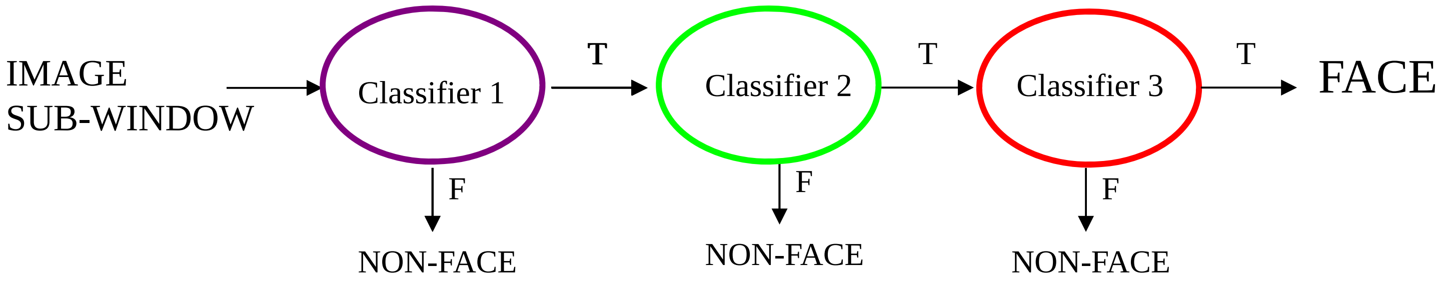
Attentional cascade

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



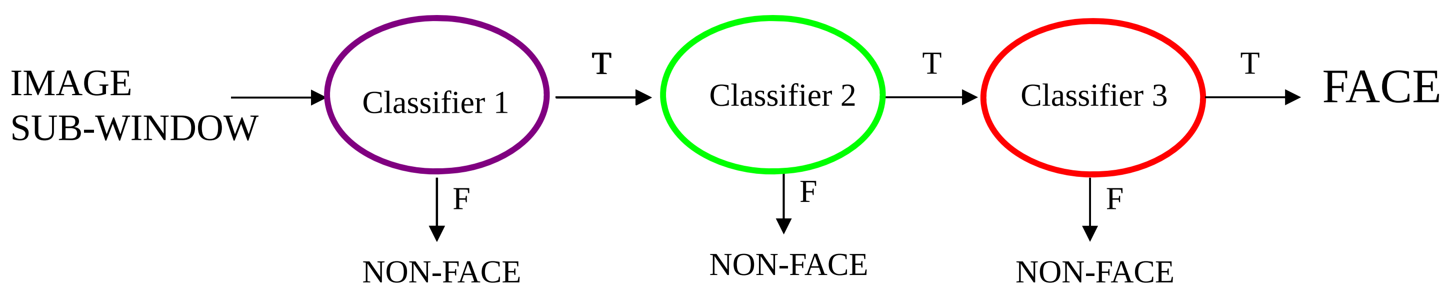
Receiver operating characteristic





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

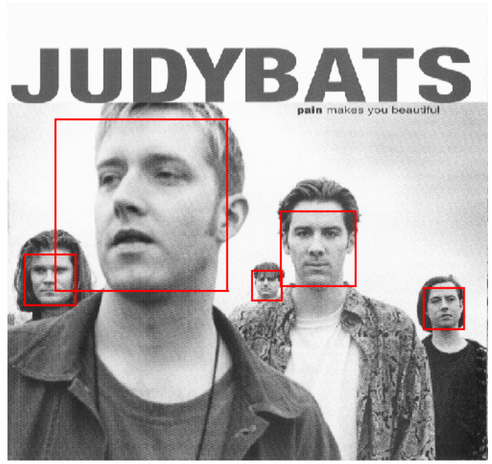
The implemented system

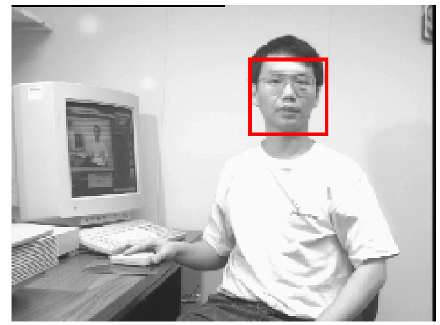
- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
    - 9500 non-face images
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose

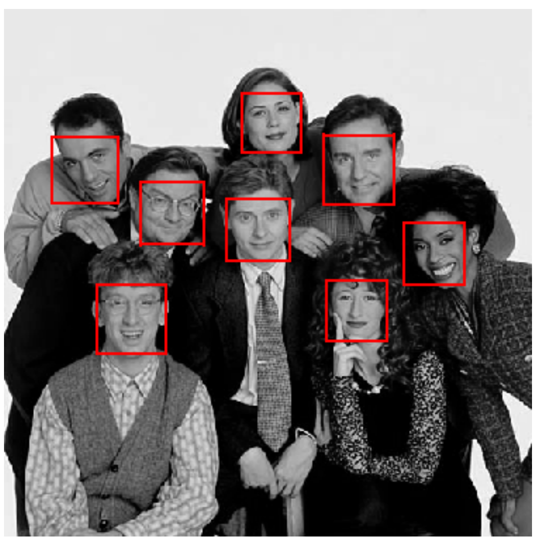
System performance

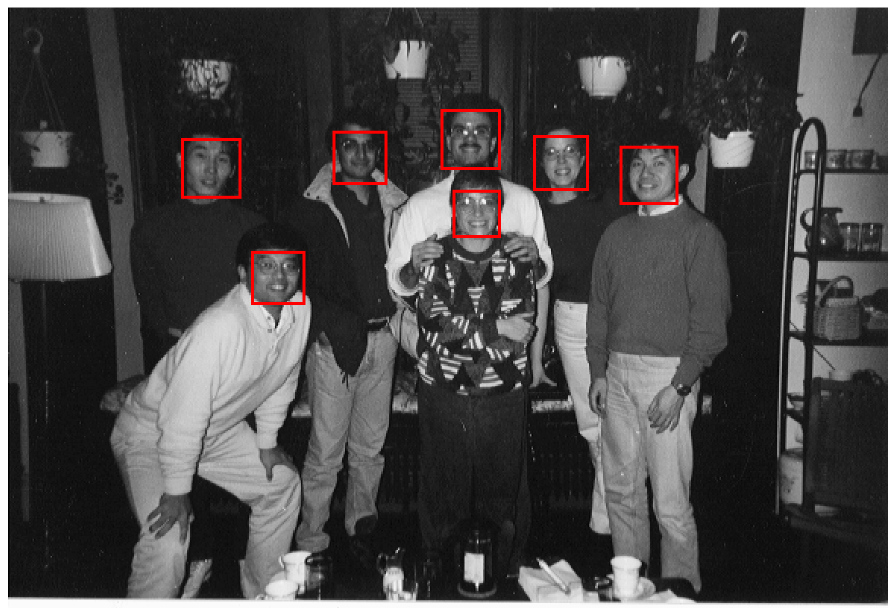
- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous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

Output of Face Detector on Test Images



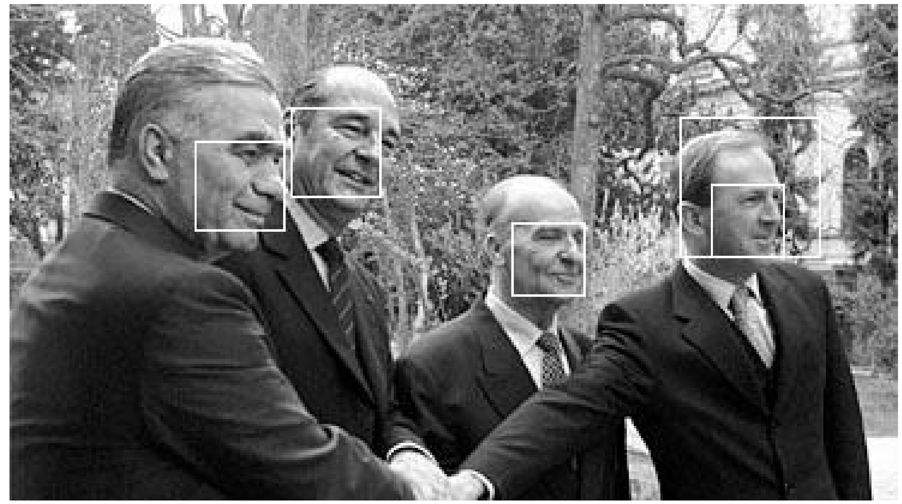






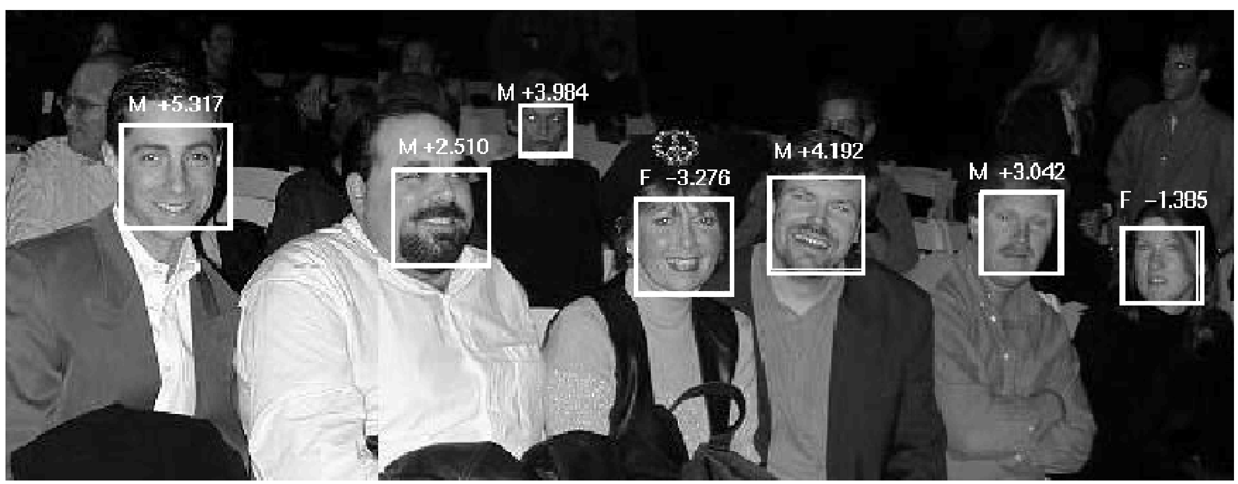
Related detection tasks

Facial Feature Localization

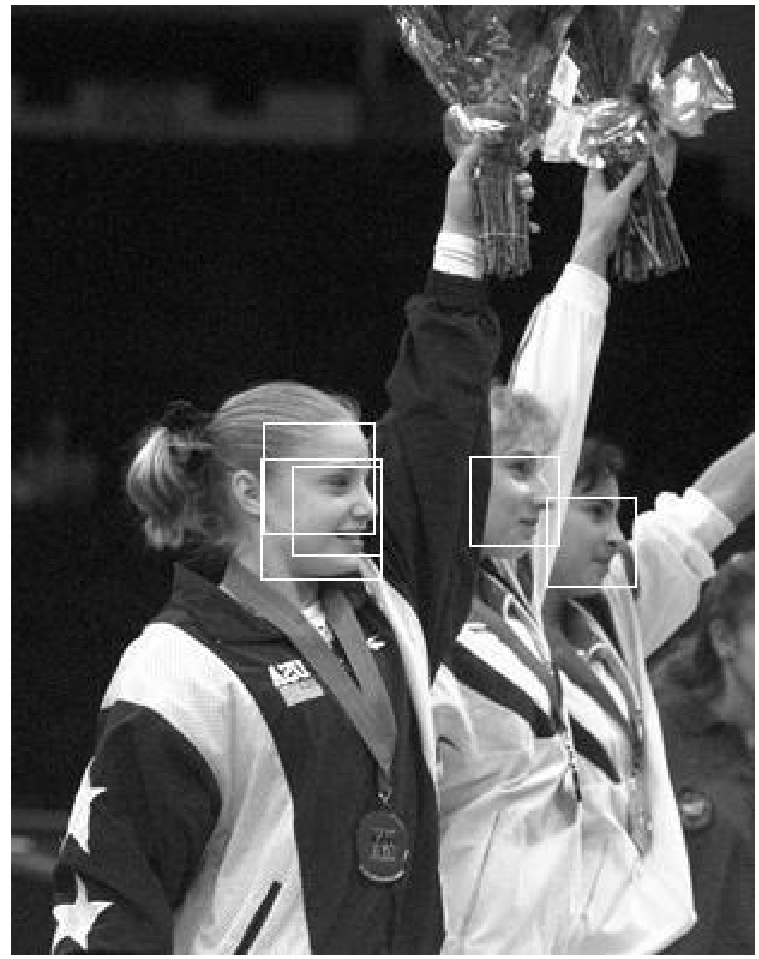


Profile Detection

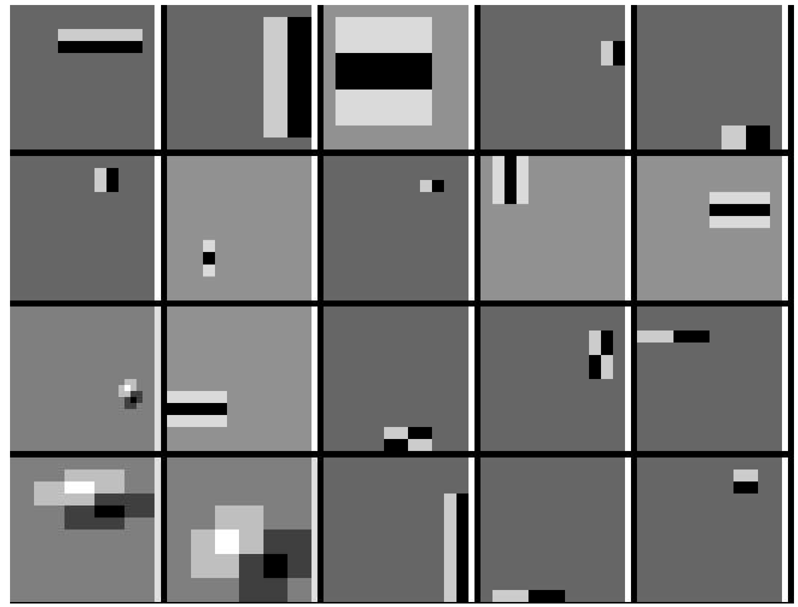
Male vs.
female



Profile Detection



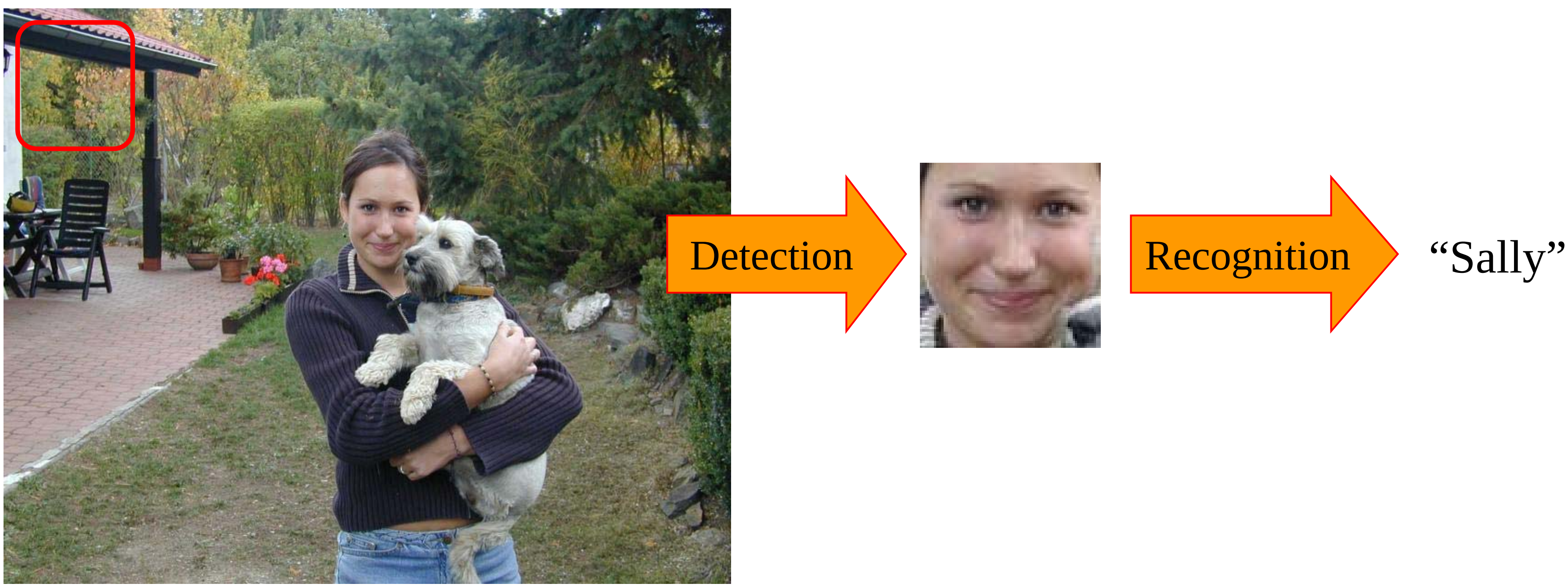
Profile Features



Summary: Viola/Jones detector

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

Face detection and recognition

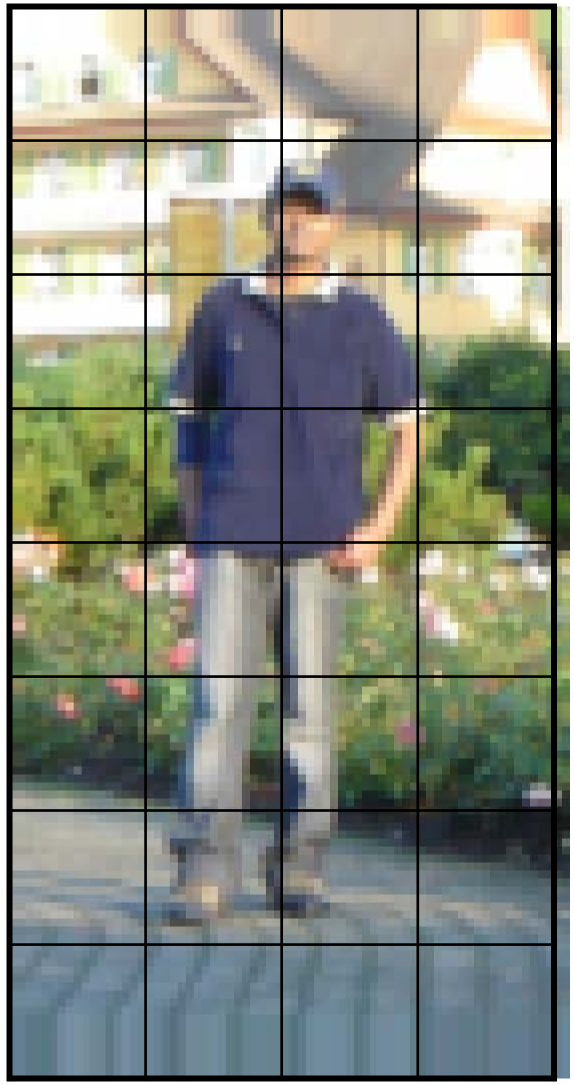


Dalal-Triggs pedestrian detector

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

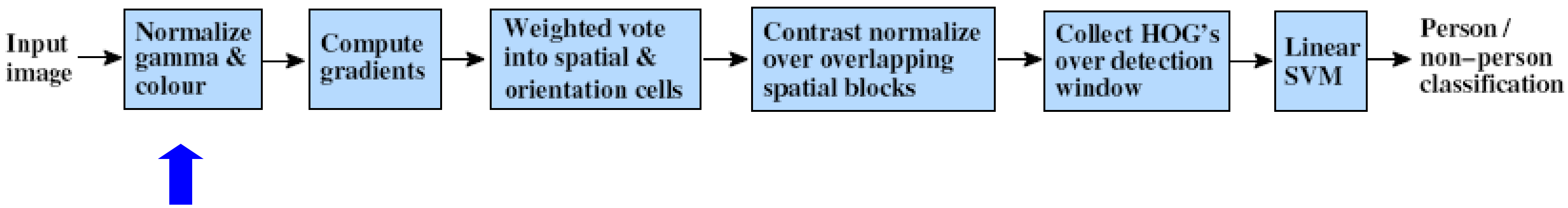
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Input image → Normalize gamma & colour → Compute gradients → Weighted vote into spatial & orientation cells → Contrast normalize over overlapping spatial blocks → Collect HOG's over detection window → Linear SVM → Person / non–person classification



Slides by Pete Barnum

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

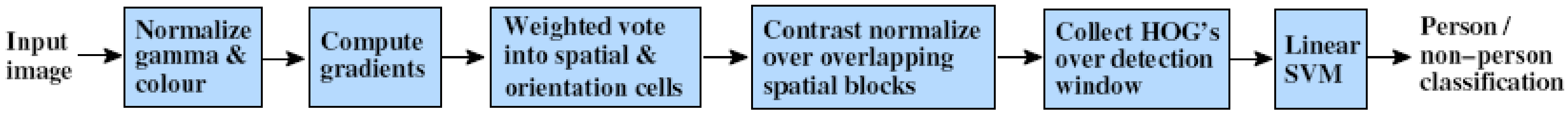


- Tested with
  - RGB
  - LAB
  - Grayscale

Slightly better performance vs. grayscale

- Gamma Normalization and Compression
  - Square root
  - Log

Very slightly better performance vs. no adjustment



Works best

-1	0	1

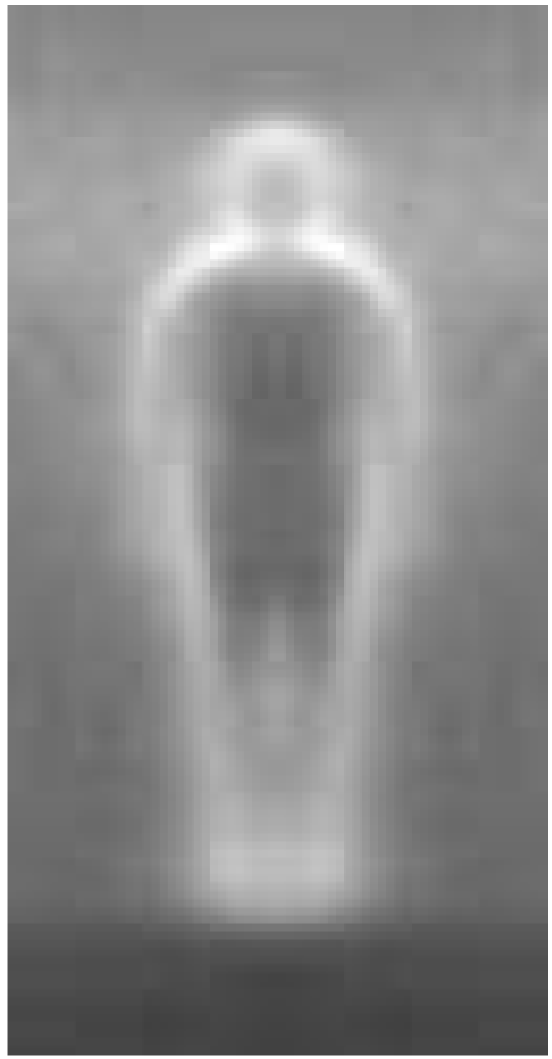
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



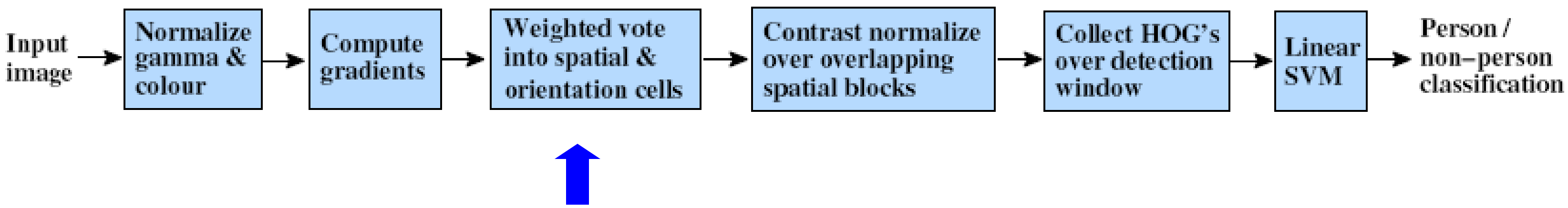
0	1
-1	0

diagonal

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

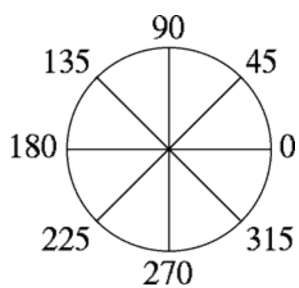
Sobel

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

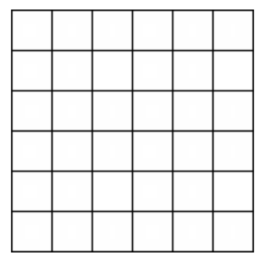


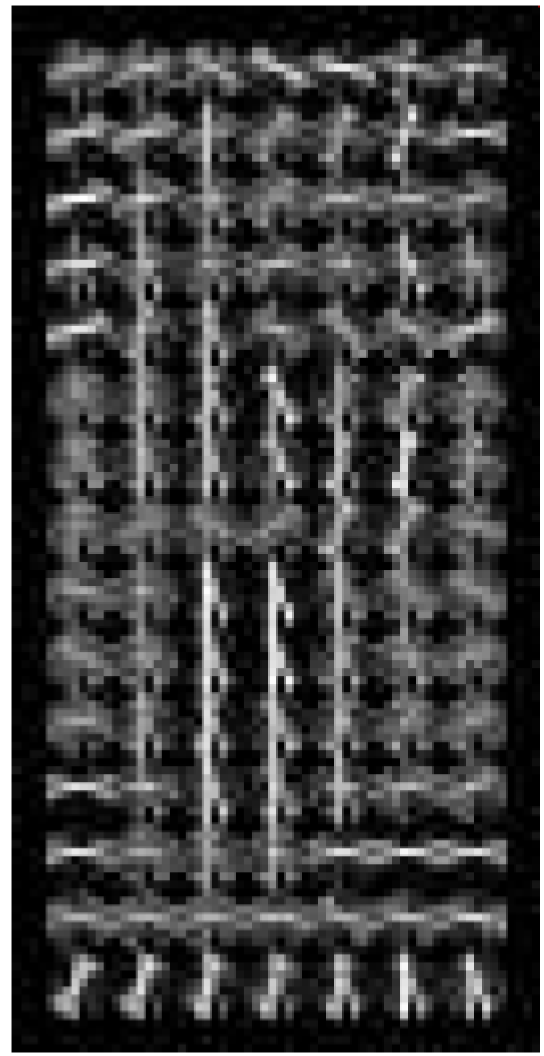
- Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles 0-180)



Histograms in 8x8 pixel cells

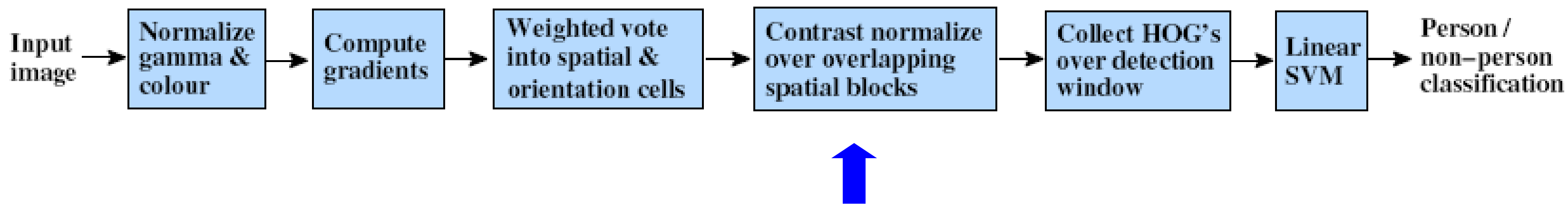


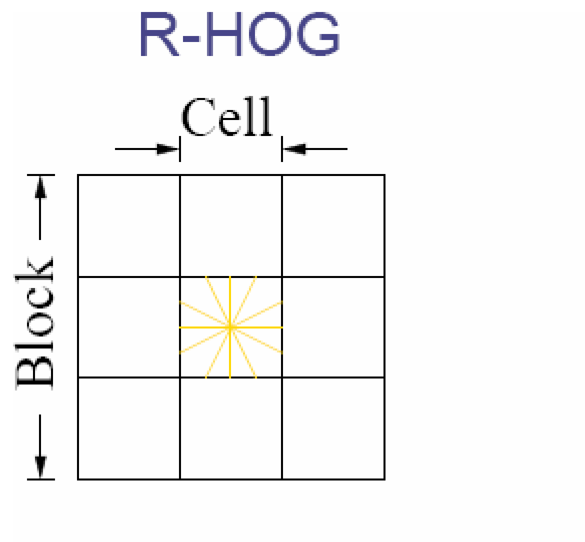


  – Votes weighted by magnitude
  – Bilinear interpolation between cells

Slides by Pete Barnum

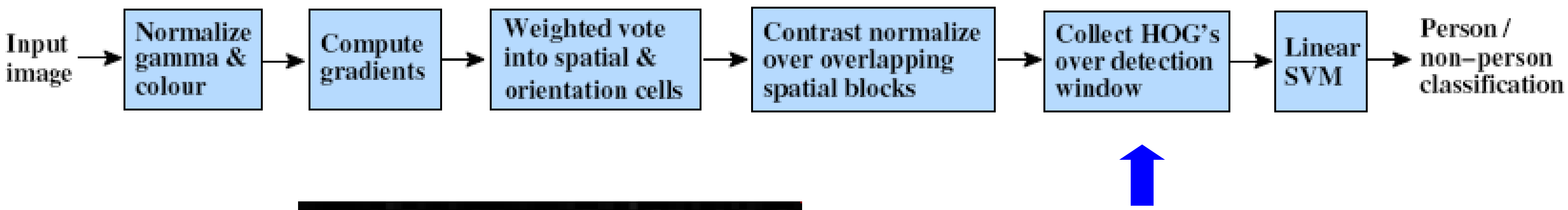
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



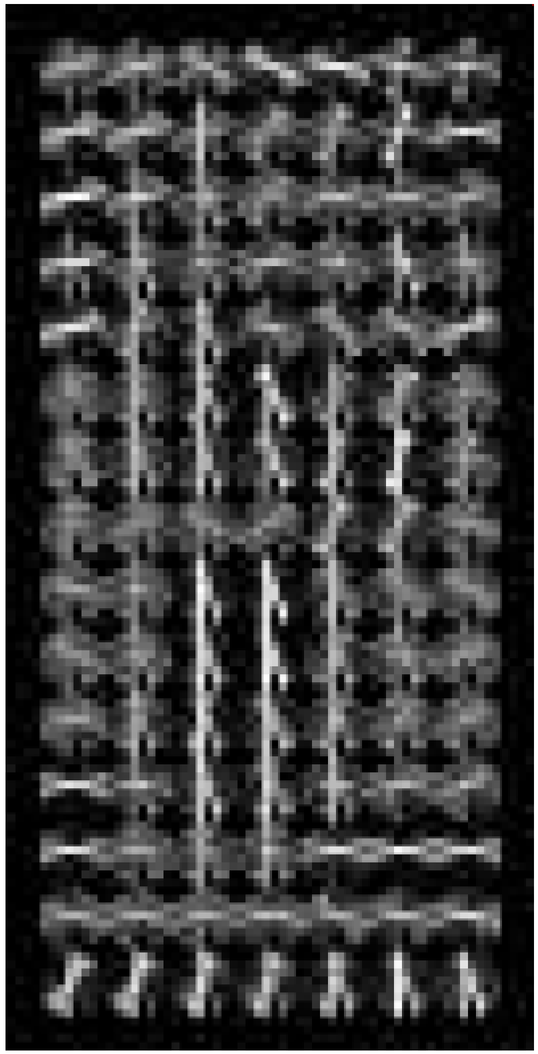


Normalize with respect to surrounding cells in overlapping blocks with different cell and block sizes

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



X=



orientations

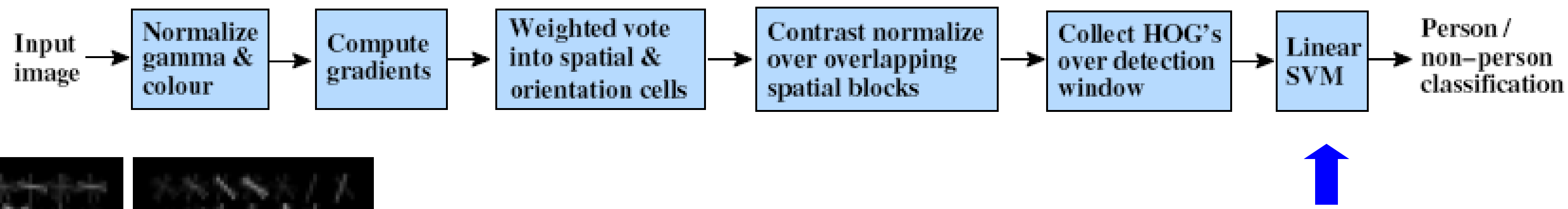
features = 15 x 7 x 9 x 4 = 3780

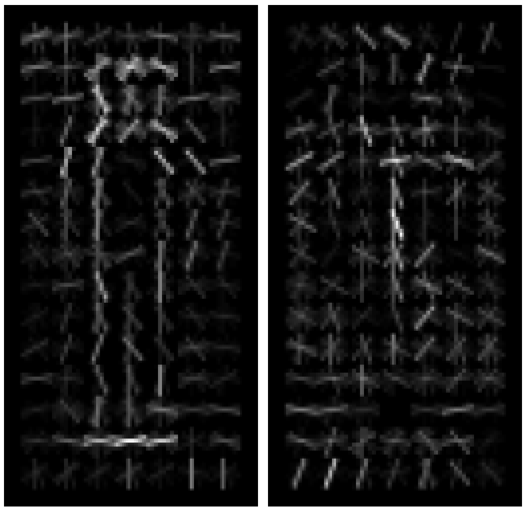
cells

normalizations by neighboring cells

Slides by Pete Barnum

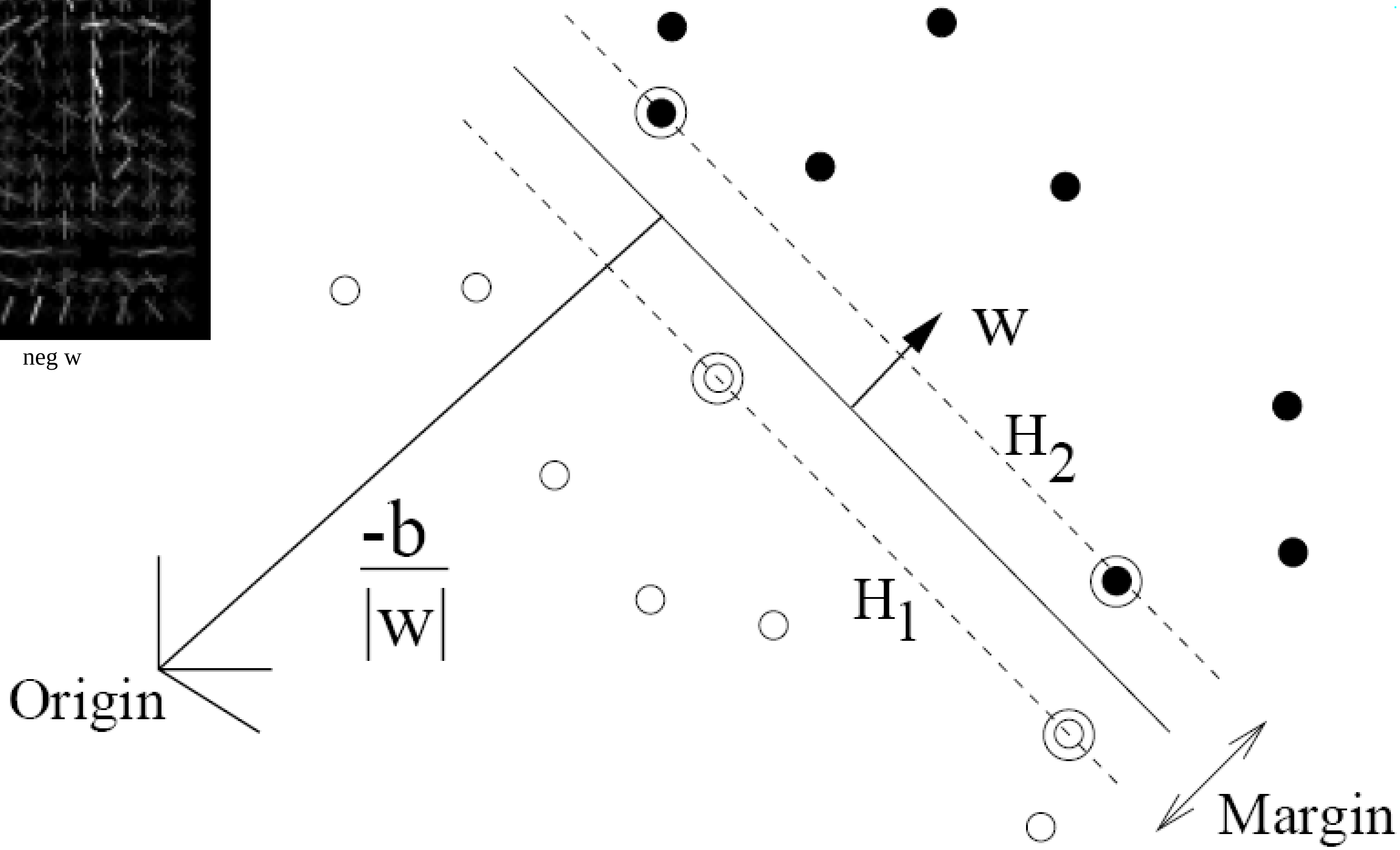
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





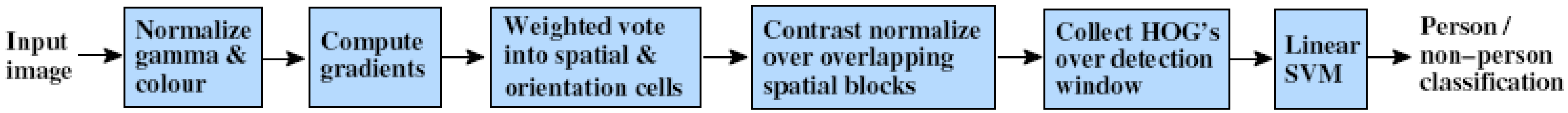
pos w

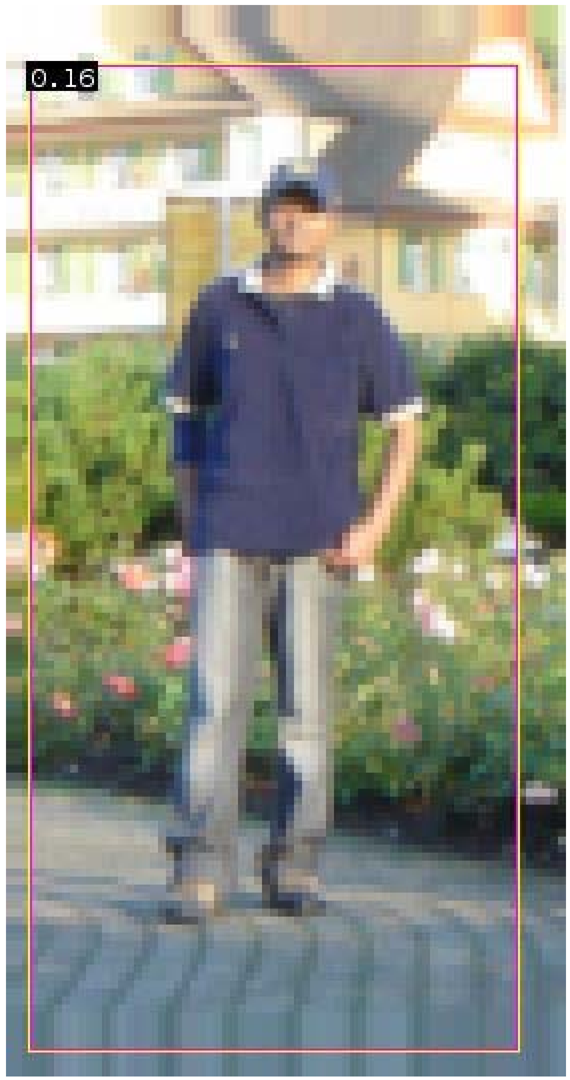
neg w



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

Pedestrian detection with HOG

- Train a pedestrian template using a linear support vector machine

positive training examples

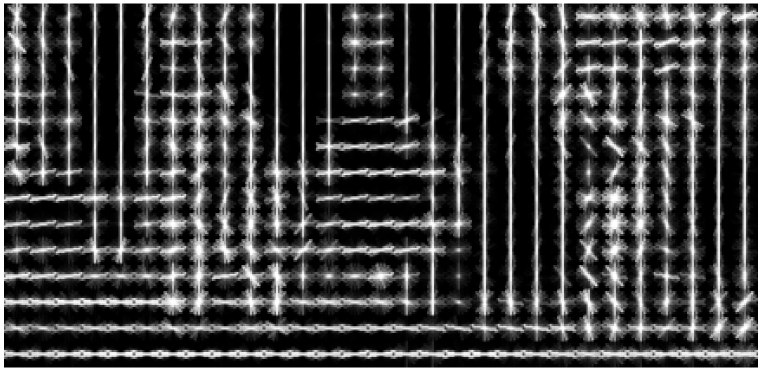
negative training examples

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

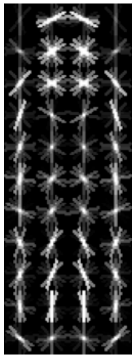
Pedestrian detection with HOG

- Train a pedestrian template using a linear support vector machine
- At test time, convolve feature map with template
- Find local maxima of response – apply non-max suppression
- For multi-scale detection, repeat over multiple levels of a HOG *pyramid*

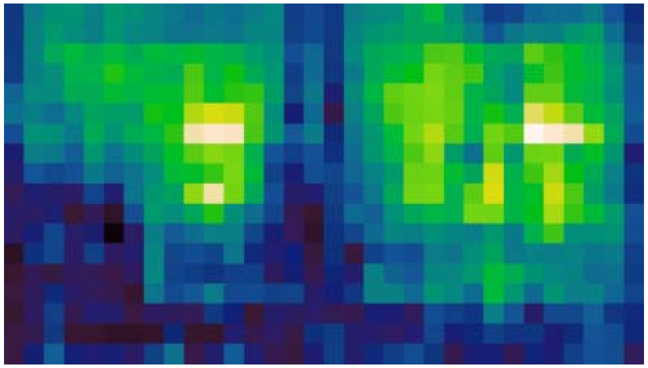
HOG feature map



Template



Detector response map



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

Detection examples

Part-based Models

Part-based Models

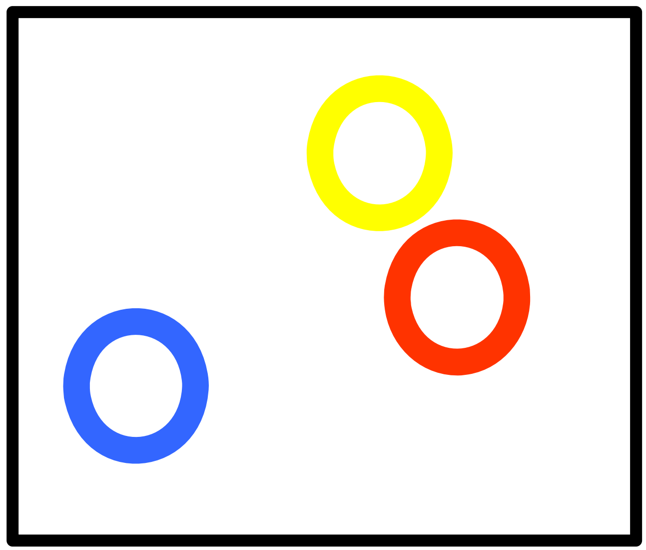
Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration

Slide credit: Rob Fergus

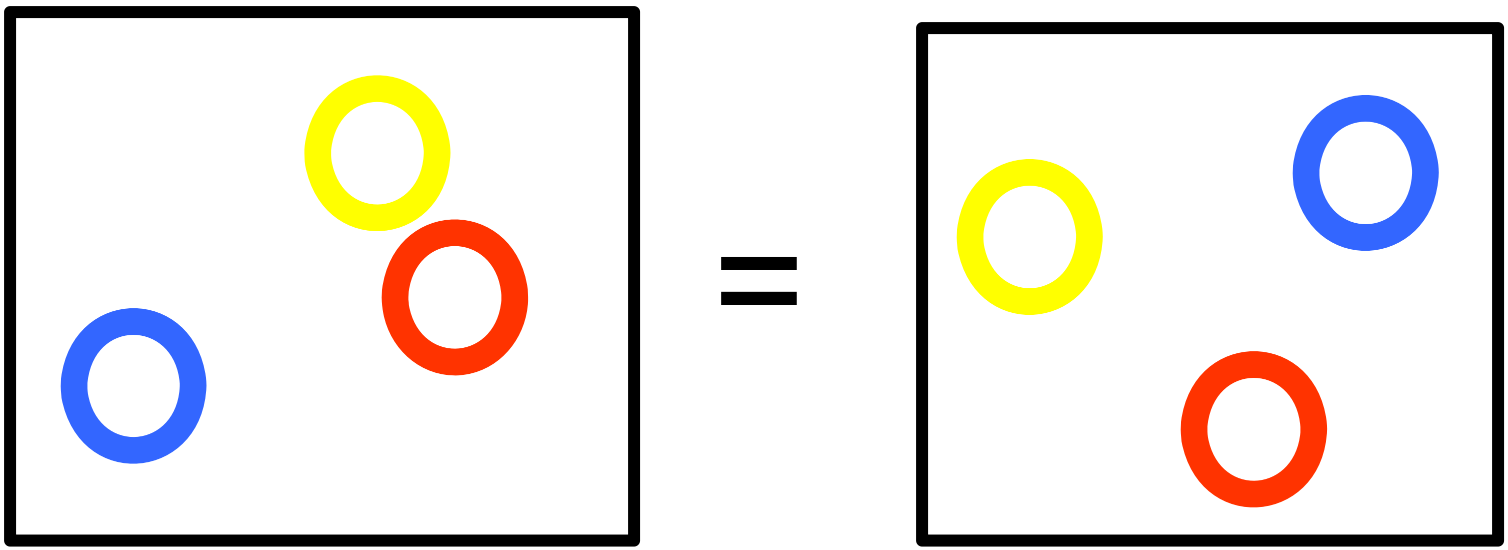
How to model spatial relations?

- One extreme: fixed template



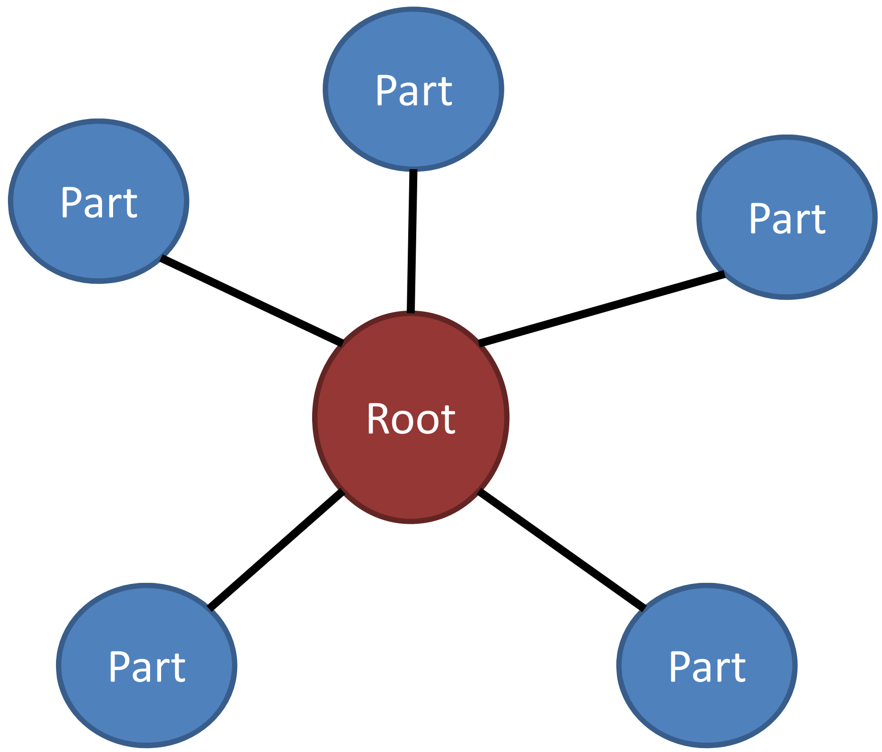
How to model spatial relations?

- Another extreme: bag of words



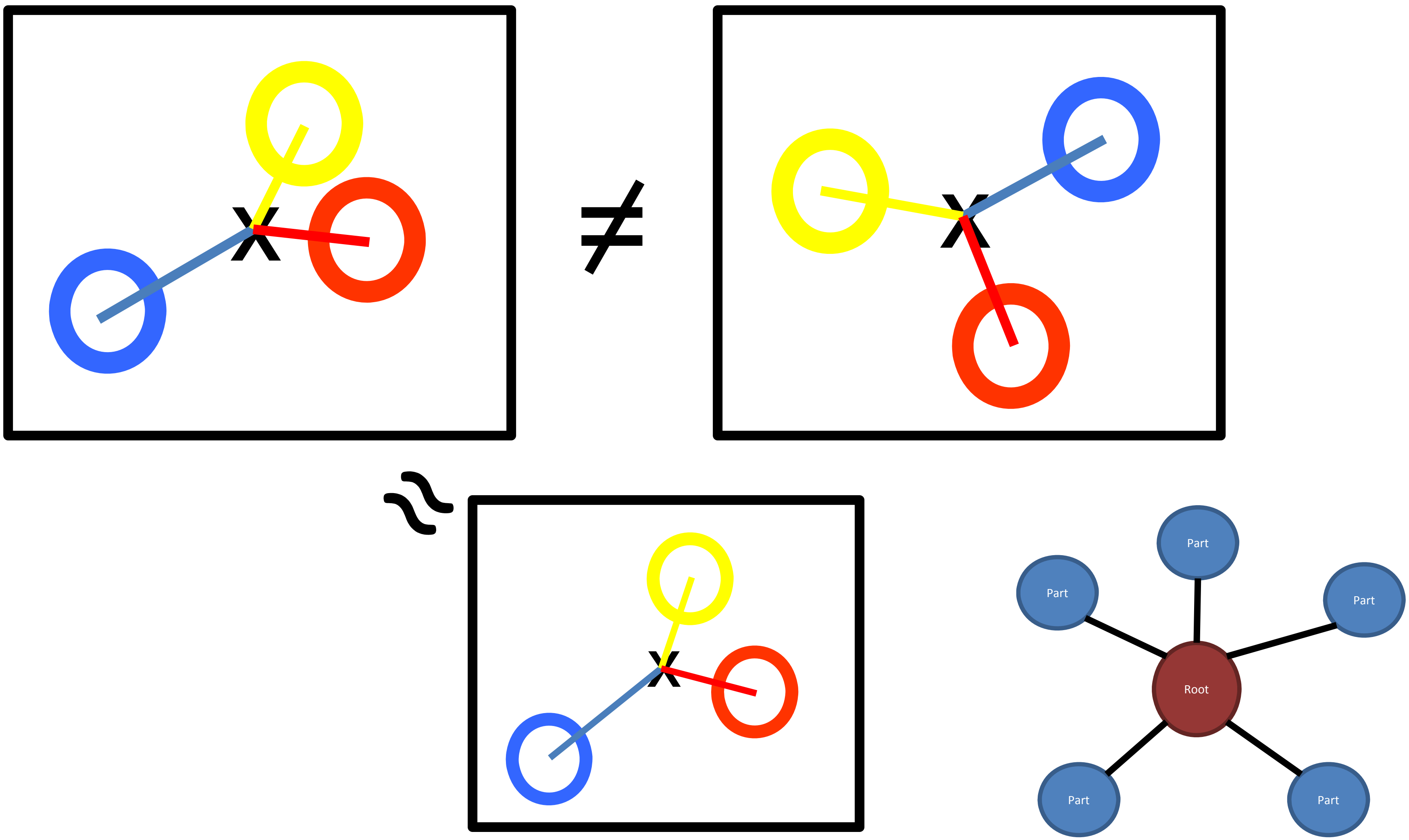
How to model spatial relations?

- Star-shaped model



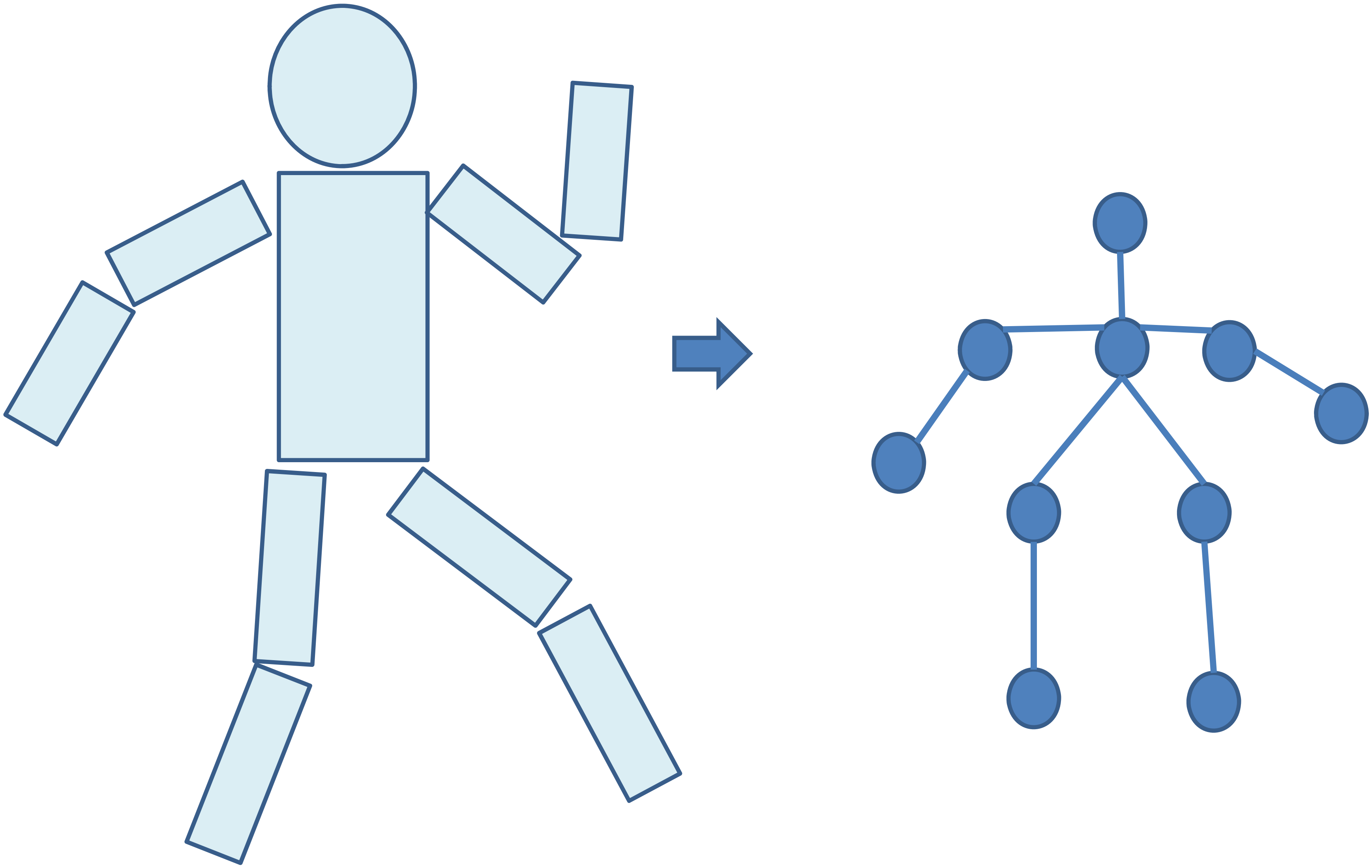
How to model spatial relations?

- Star-shaped model



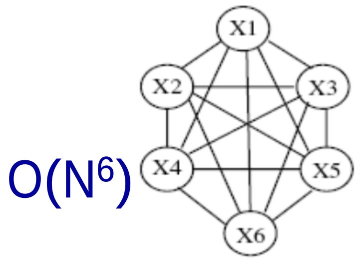
How to model spatial relations?

- Tree-shaped model



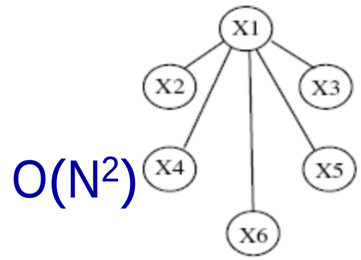
How to model spatial relations?

- Many others...



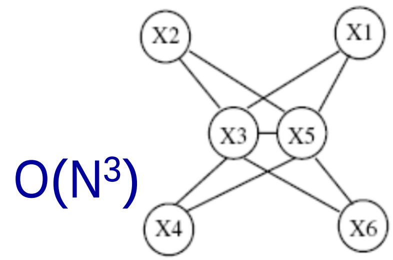
a) Constellation

Fergus et al. ’03
Fei-Fei et al. ‘03



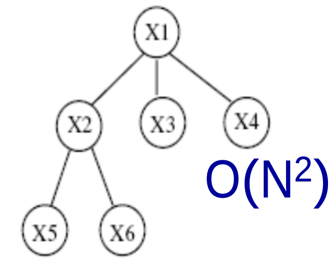
b) Star shape

Leibe et al. ’04, ‘08
Crandall et al. ‘05
Fergus et al. ’05



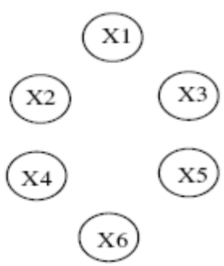
c) k-fan ($k = 2$)

Crandall et al. ‘05



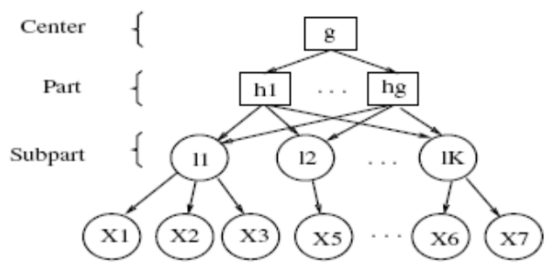
d) Tree

Felzenszwalb &
Huttenlocher ‘05



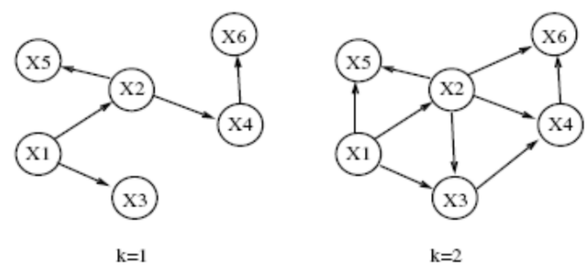
e) Bag of features

Csurka ’04
Vasconcelos ‘00



f) Hierarchy

Bouchard & Triggs ‘05

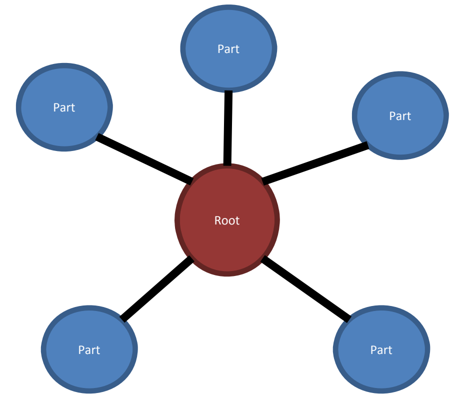


g) Sparse flexible model

Carneiro & Lowe ‘06

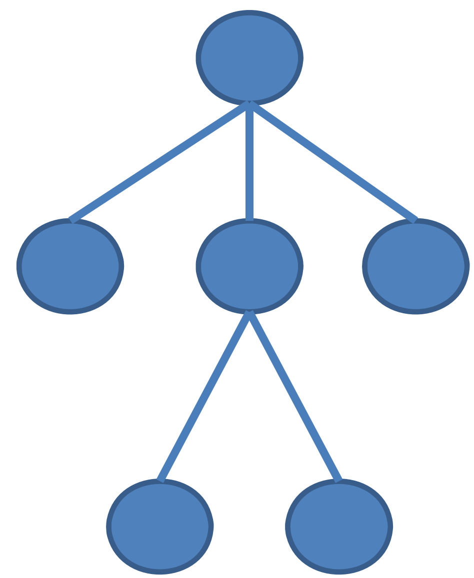
from [Carneiro & Lowe, ECCV’06]

Star and Tree-shaped Models



1. Star-shaped model
   – Example: Deformable Parts Model
     • Felzenswalb et al. 2010

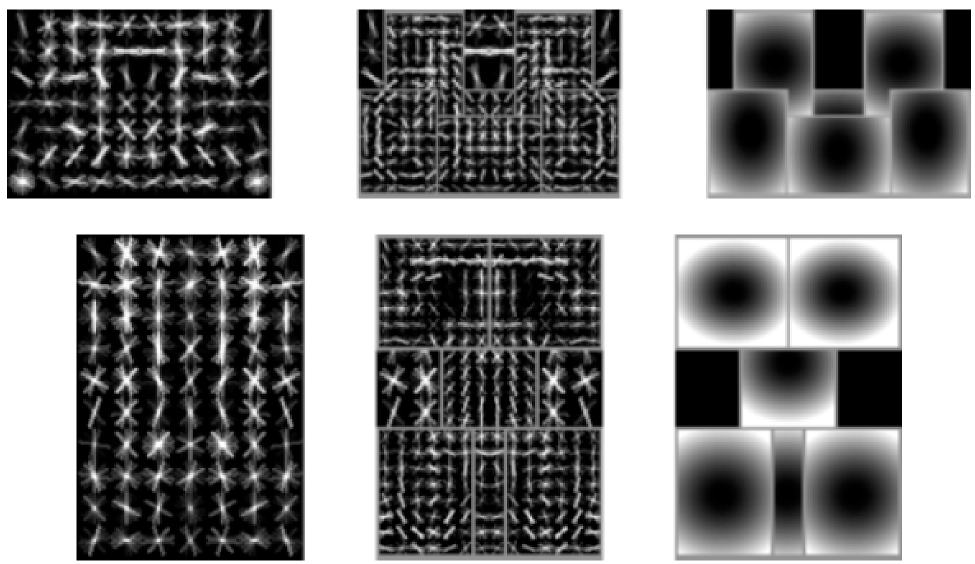
2. Tree-shaped model
   – Example: Pictorial structures
     • Felzenszwalb Huttenlocher 2005



Deformable Part Model (DPM)

Detections

Template Visualization



root filters
coarse resolution

part filters
finer resolution

deformation
models

Felzenszwalb et al. 2008, 2010

Review: Dalal-Triggs detector

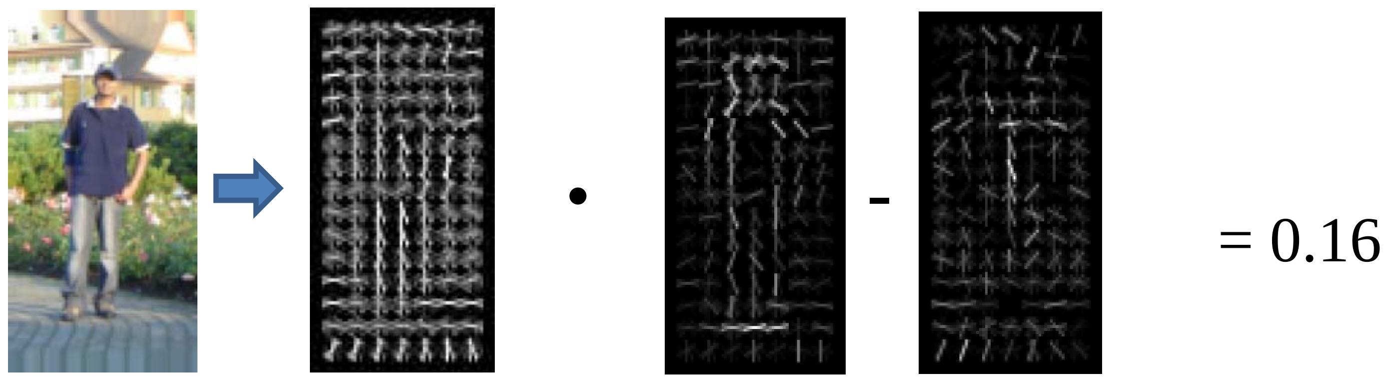
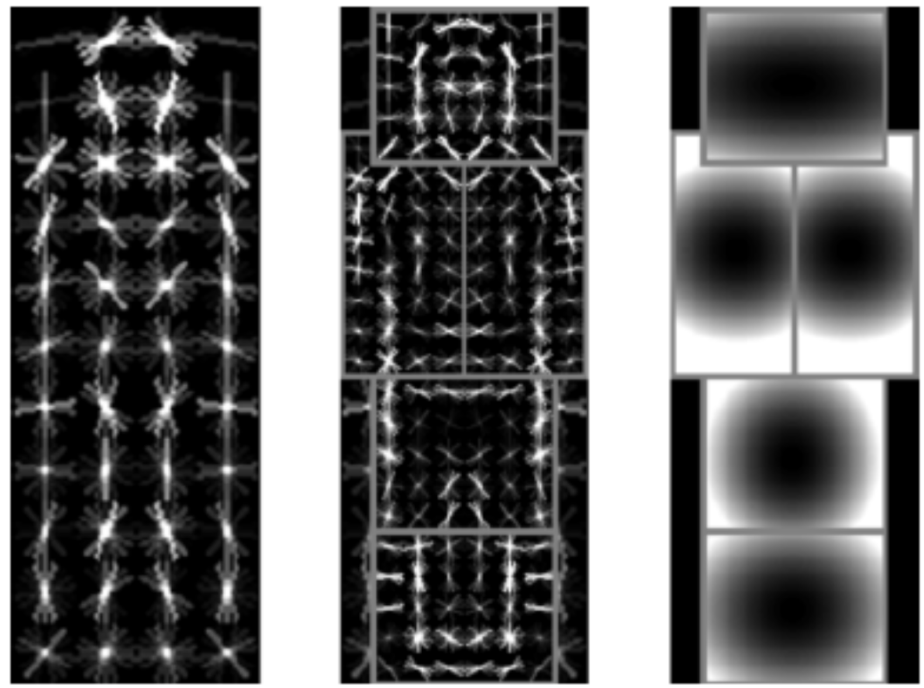


Image Window HOG SVM weights (pos/neg) score

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

Deformable parts model

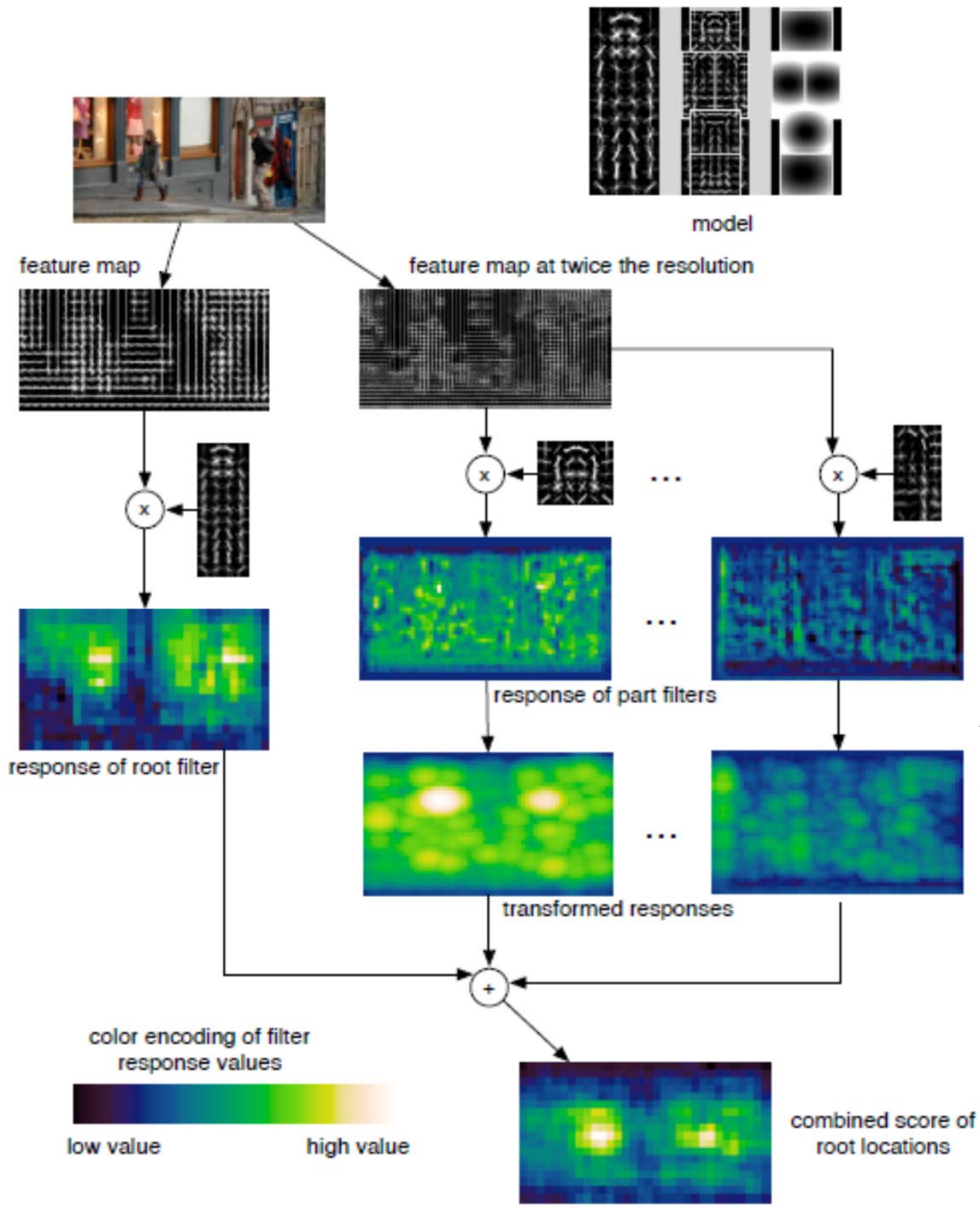
- Root filter models coarse whole-object appearance
- Part filters model finer-scale appearance of smaller patches
- For each root window, part positions that maximize appearance score minus spatial cost are found
- Total score is sum of scores of each filter and spatial costs



Root filter Part filters Spatial costs

DPM: computing object score

Scores from root detector

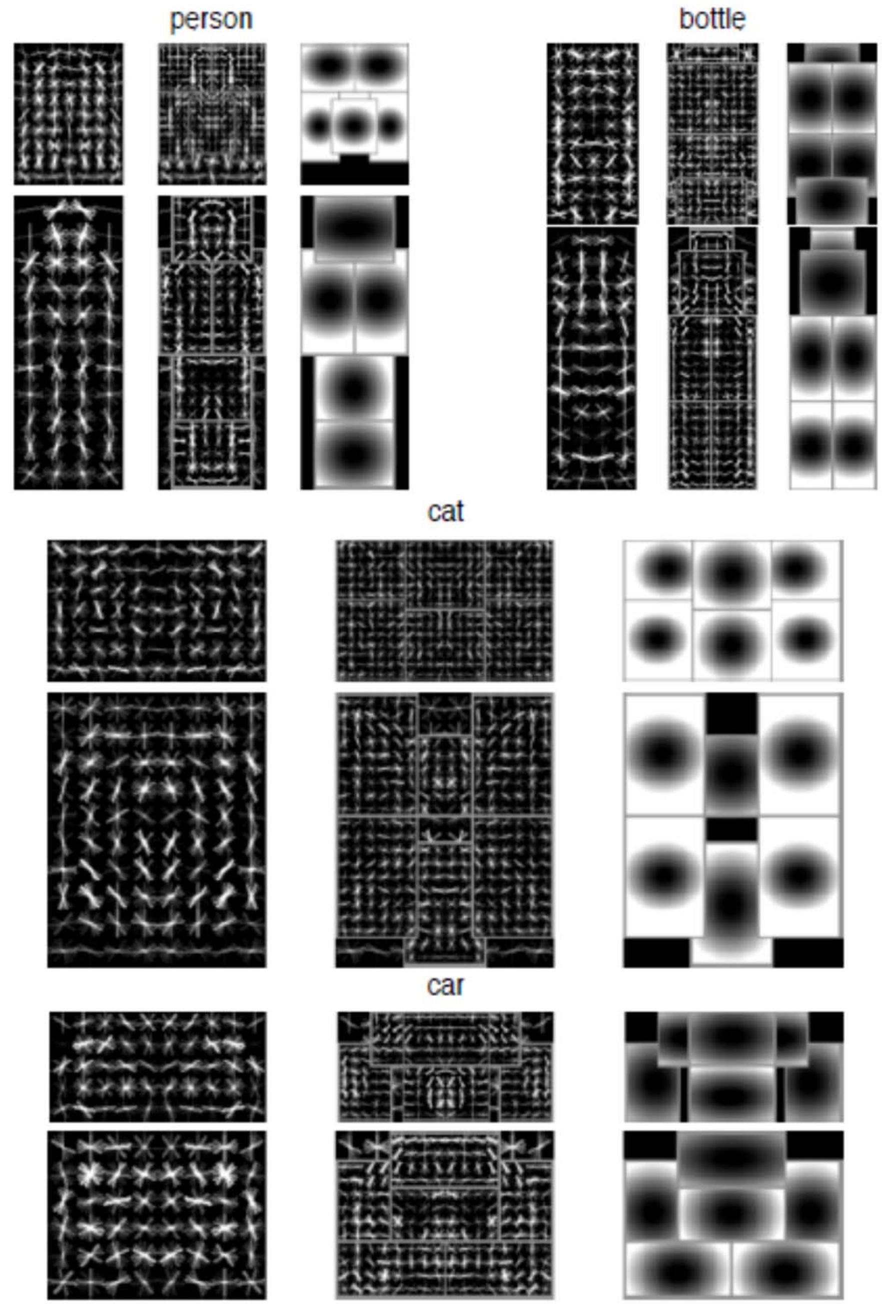


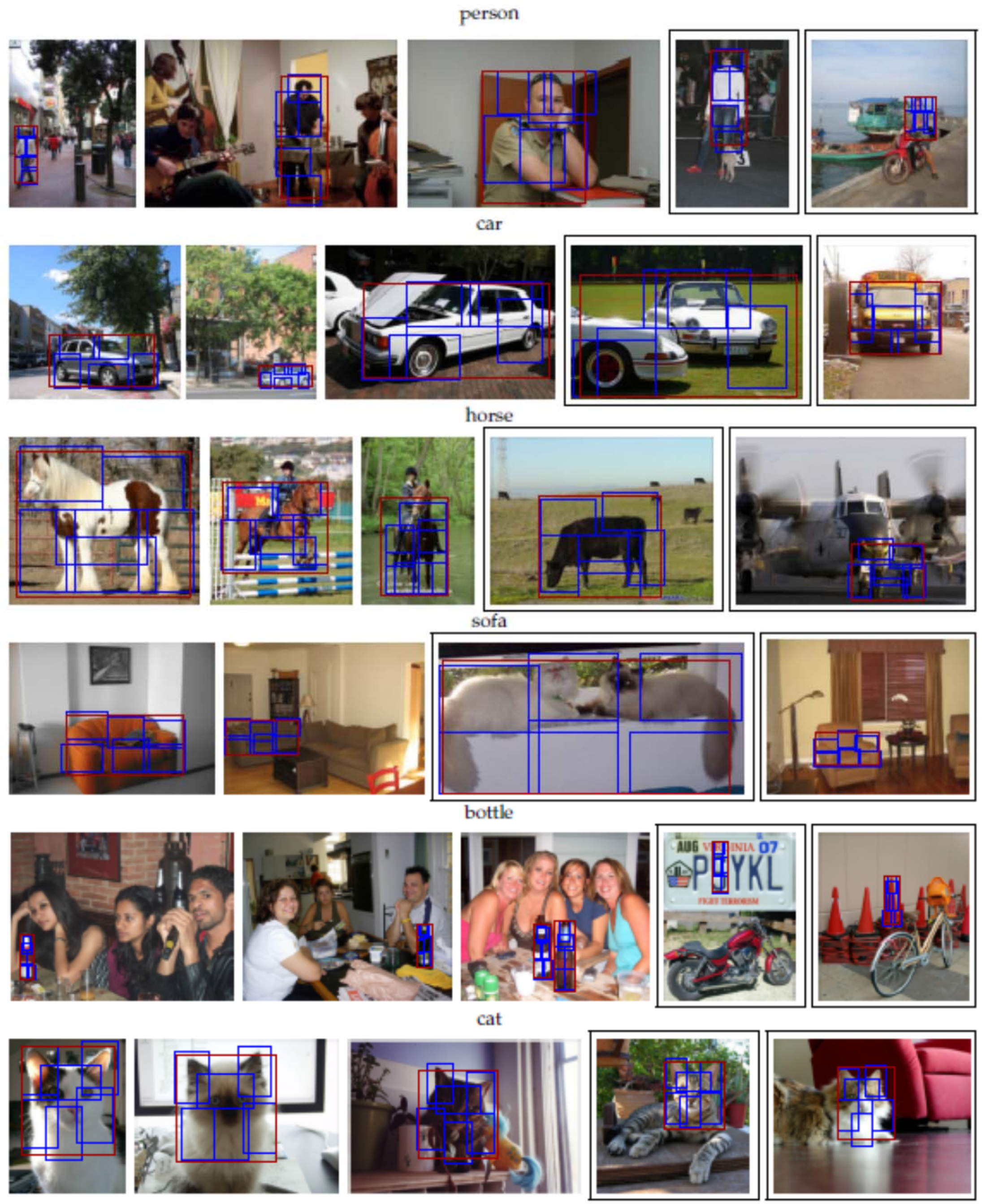
Scores from individual part detectors

With generalized distance transform, compute the maximum part score corresponding to each root position

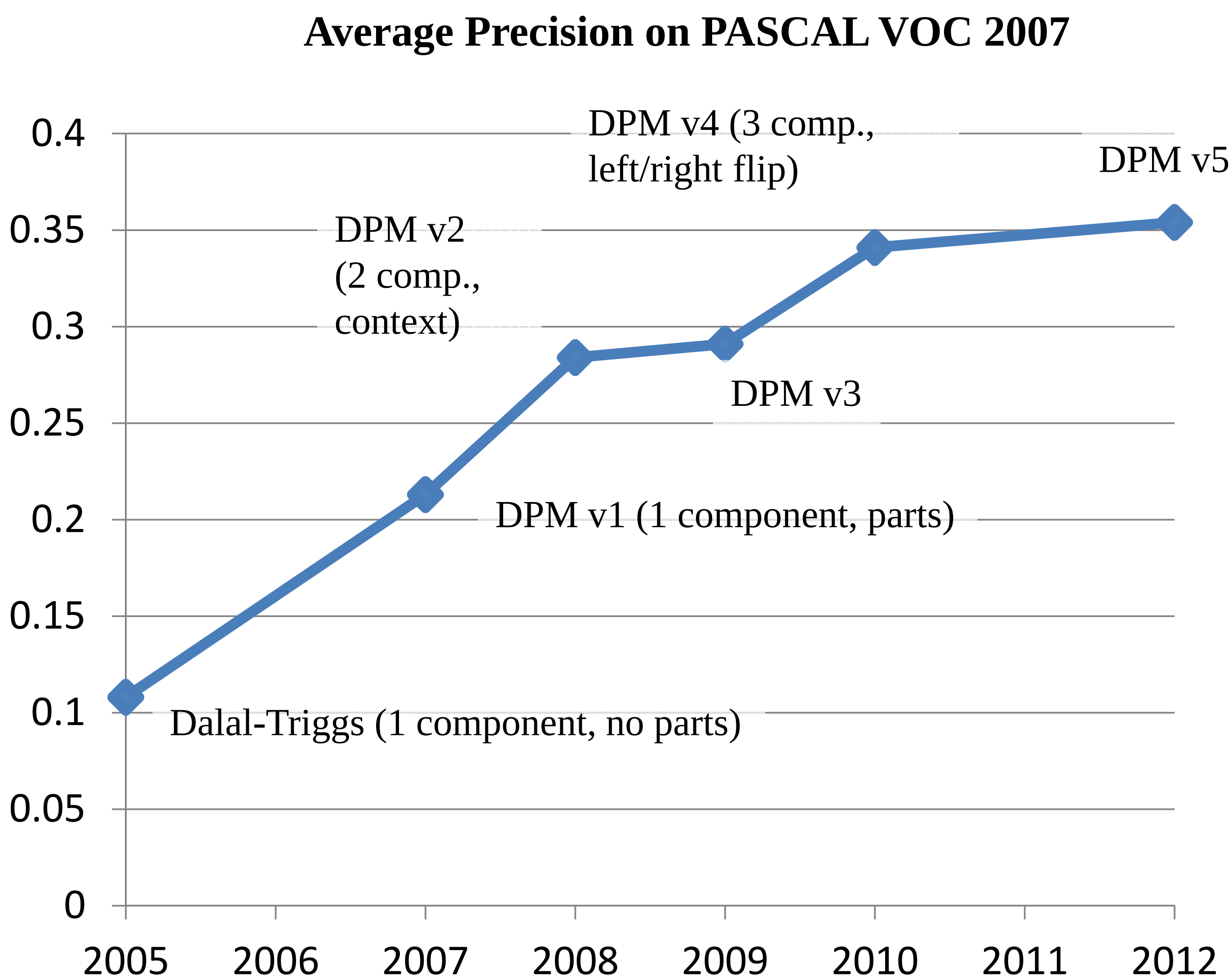
DPM: mixture model

- Each positive example is modeled by one of M detectors
- In testing, all detectors are applied with non-max suppression

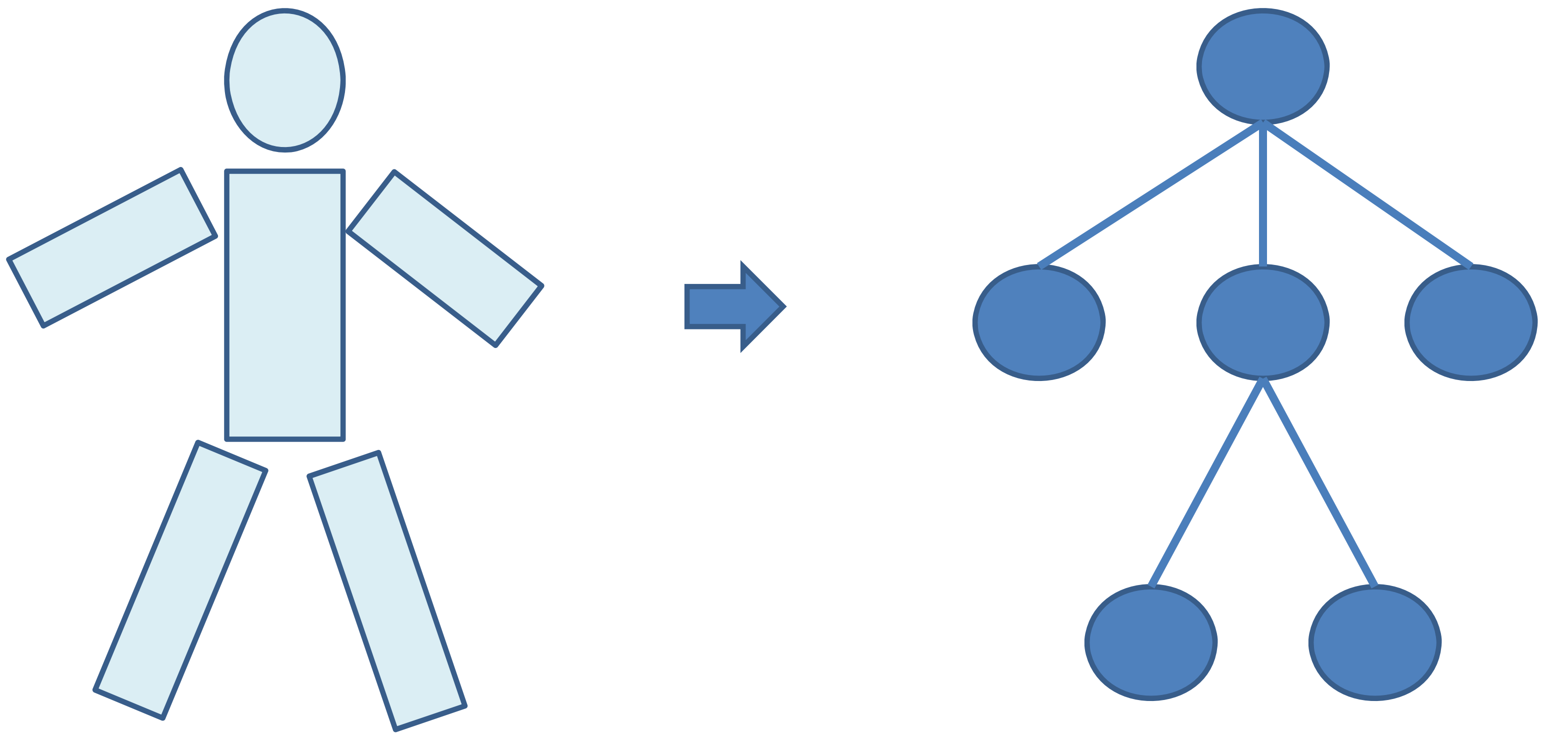



person
car
horse
sofa
bottle
cat

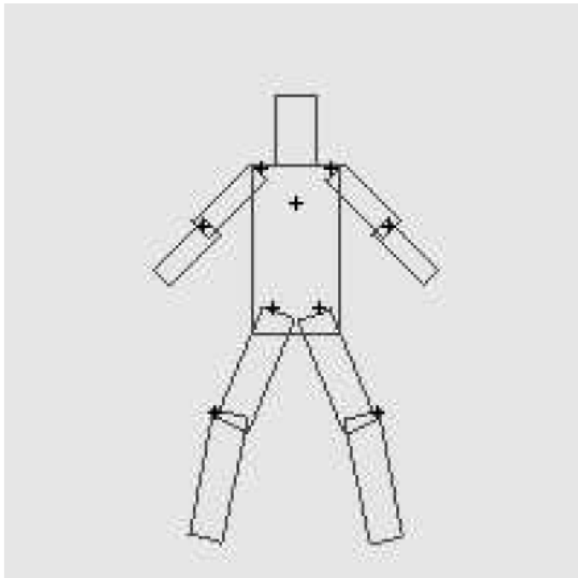
Improvement over time for HOG-based detectors

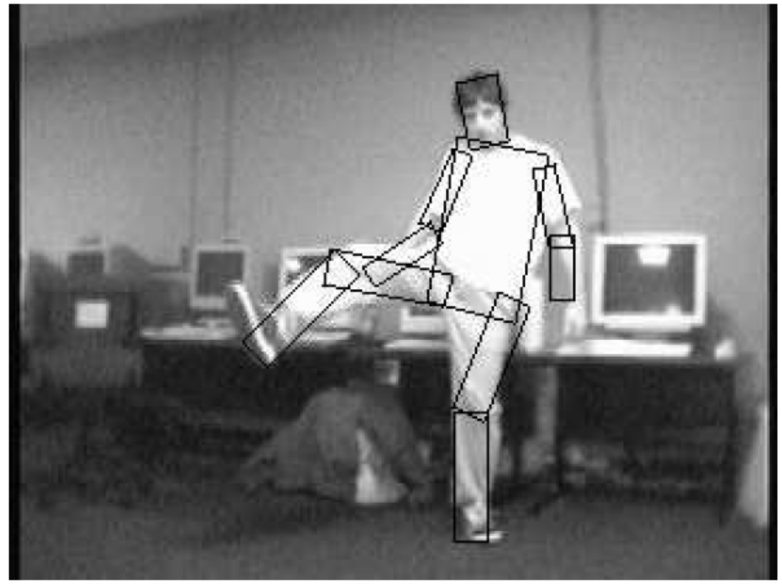


Tree-shaped model



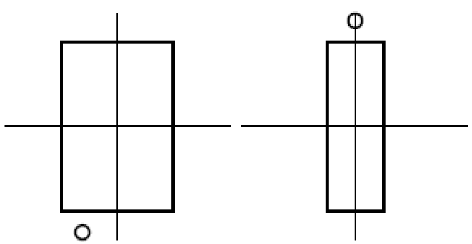
Pictorial Structures



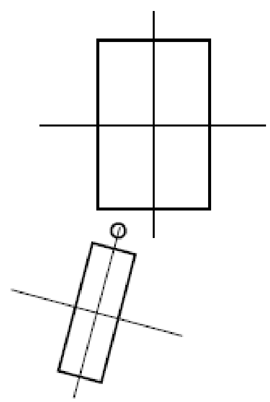


Part = oriented rectangle

Spatial model = relative size/orientation



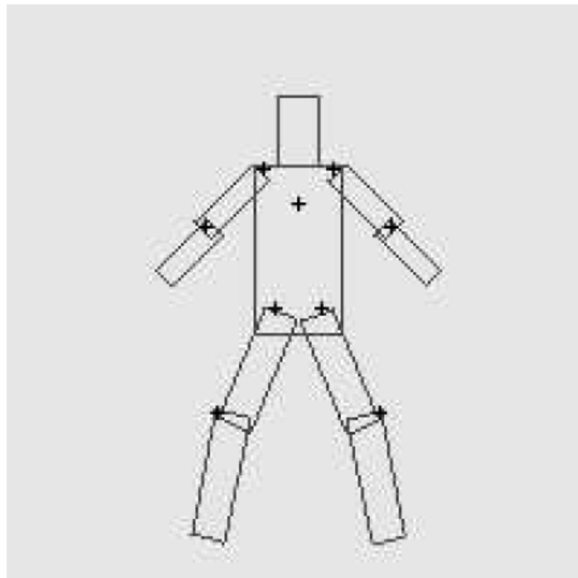
a



b

Felzenszwalb and Huttenlocher 2005

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

Modeling the Appearance

- Any appearance model could be used
  - HOG Templates, etc.
  - Here: rectangles fit to background subtracted binary map
- Can train appearance models independently (easy, not as good) or jointly (more complicated but better)

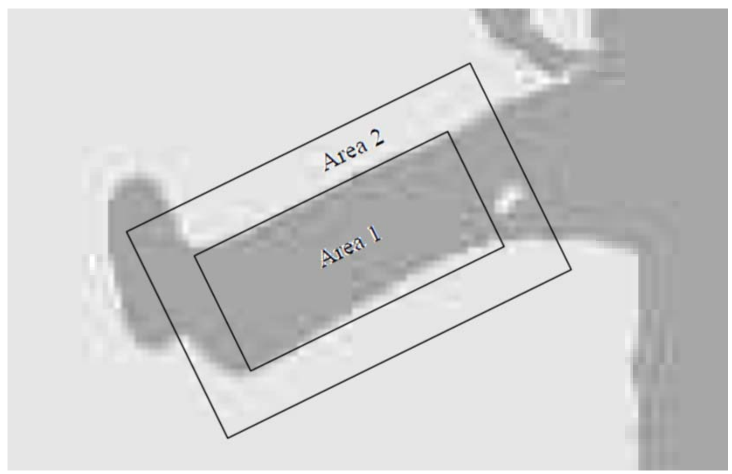
$$P(L|I,\theta) \propto \left(\prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i,v_j)\in E} p(l_i, l_j|c_{ij}) \right)$$

Appearance likelihood

Geometry likelihood

Part representation

- Background subtraction

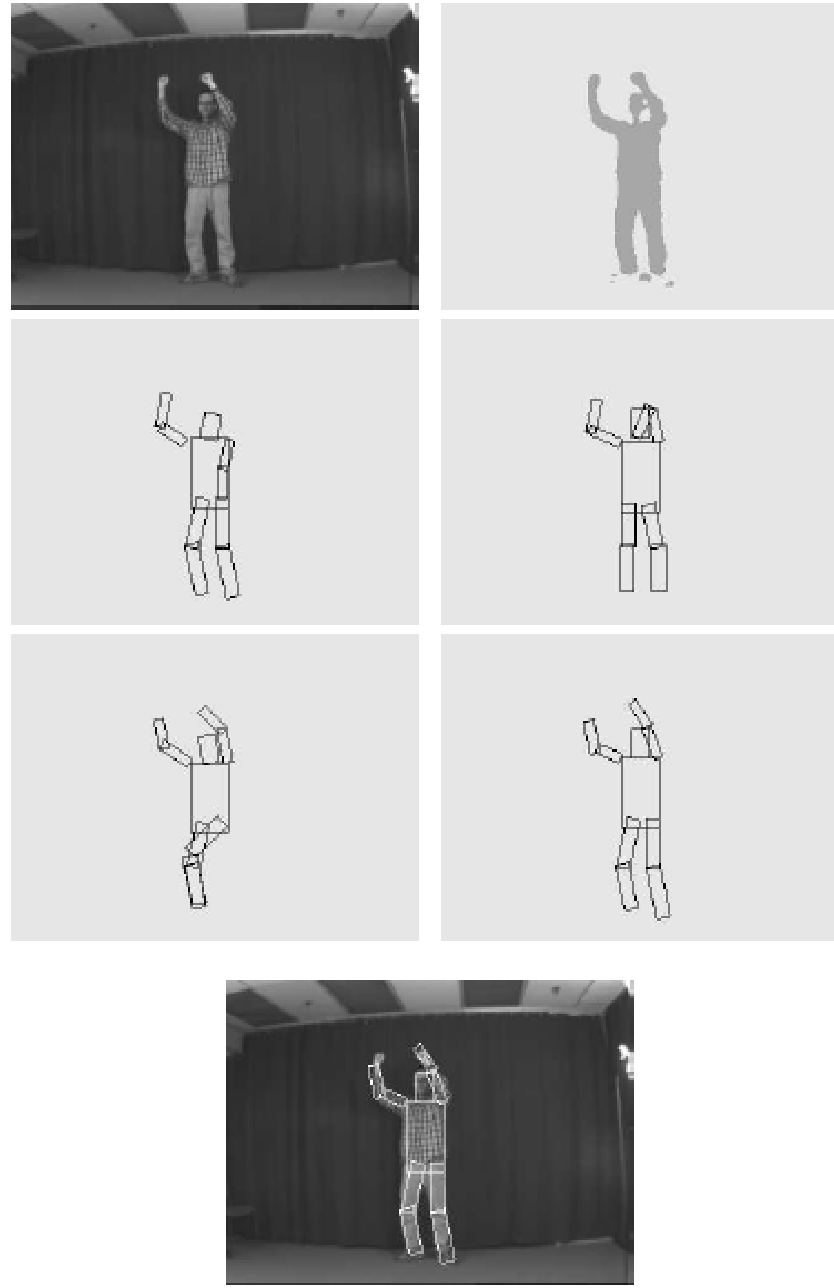


Pictorial structures model

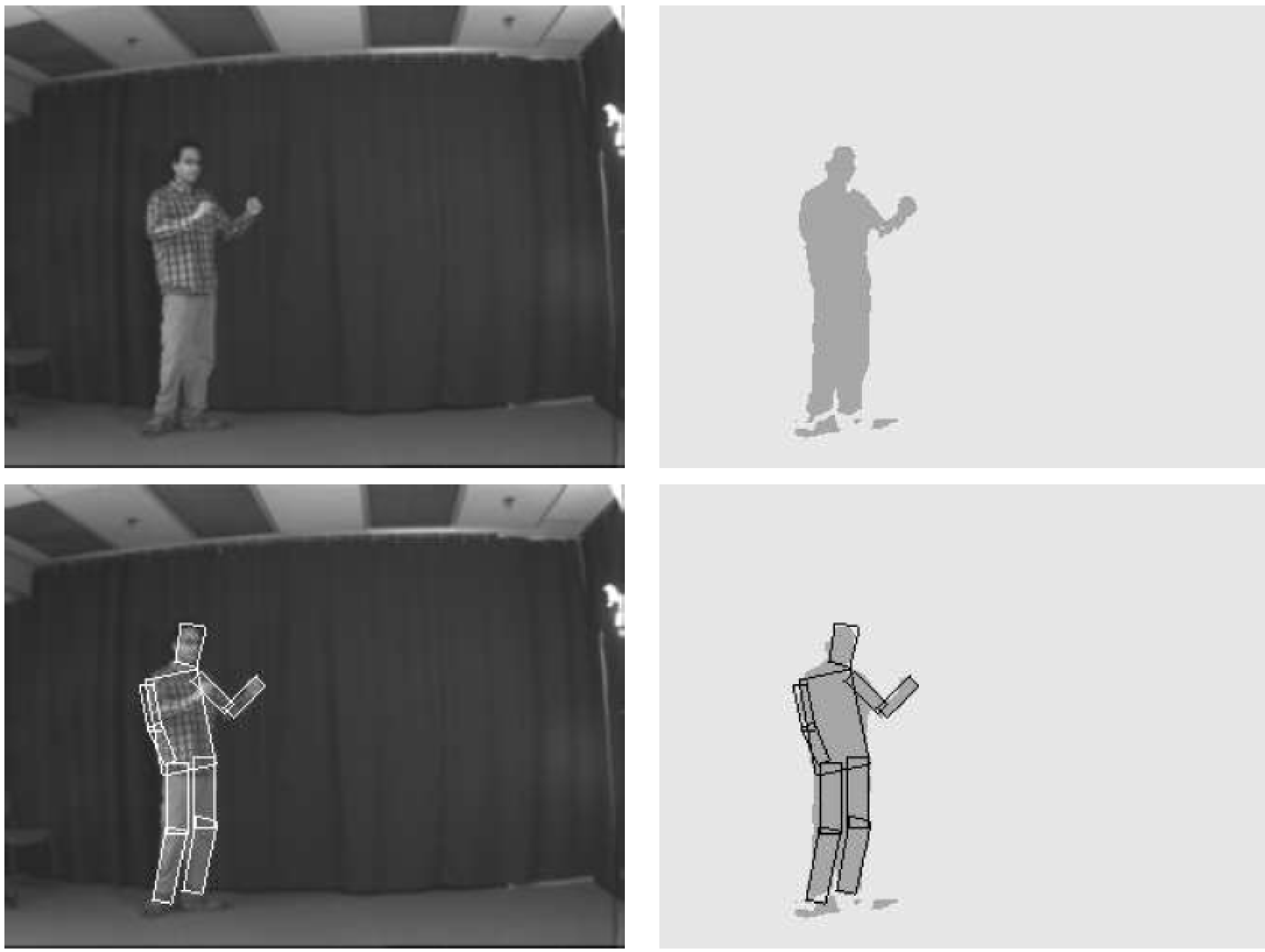
To create multiple likely candidates

- Sample root node, then each node given parent, until all parts are sampled

Sample poses from likelihood and choose best match



Results for person matching



Results for person matching