Face, Human Detection

Announcement

- Lab Sessions will happen at this and next week for Lab Assignment 2. Please attend the lab sessions and ask questions.

- Final exam timetable is to be confirmed. The initially suggested time by the exam office is on 9:00 am Canberra time, June 3rd , which might be too early for students overseas. We have requested changes.

Outline

- Review
- Viola-Jones Face detection- continue
- HOG features
- Bag of words for object detection

Review: Eigenfaces (PCA on face images)

1. Compute covariance matrix of face images

2. Compute the principal components ("eigenfaces")
   - K eigenvectors with top K largest eigenvalues

3. Represent all face images in the dataset as linear combinations of eigenfaces
   - Perform nearest neighbor on these coefficients

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Review: Eigen Face, Representation and reconstruction

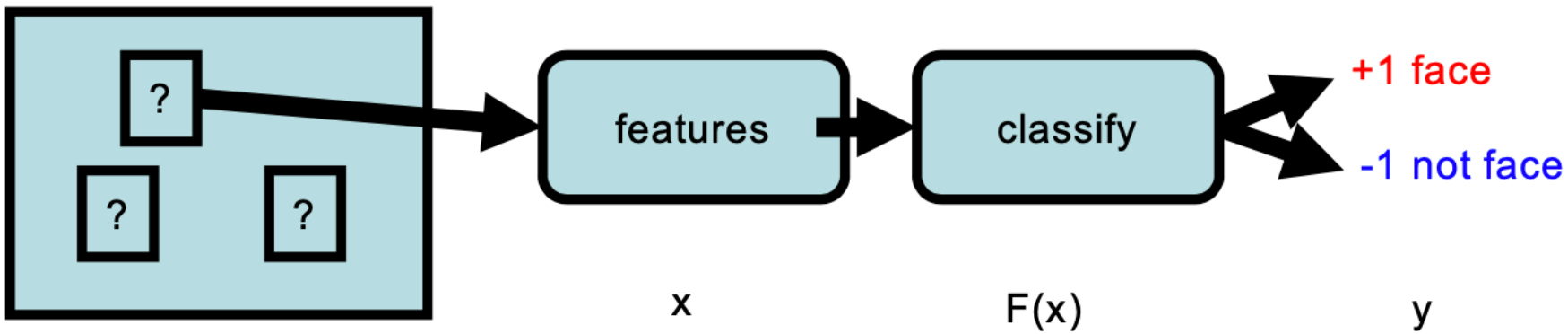
- Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$
$$= \quad w_1, \ldots, w_k$$

- Reconstruction:

$\hat{x} = \mu + w_1u_1+w_2u_2+w_3u_3+w_4u_4+ \ldots$

Face detection -- basic scheme



- We slide a window over the image
- Extract features for each window
- Classify each window into face/non-face

Viola-Jones face detection

Paul A. Viola and Michael J. Jones

Intl. J. Computer Vision

57(2), 137–154, 2004

Some slides *(slides adapted from Bill Freeman, MIT 6.869, April 2005)*

45

Characteristics of the algorithm

– Feature set (…is huge about 16M features)
– Efficient feature selection using AdaBoost
– New image representation: Integral Image
– Cascaded Classifier for rapid detection

➢ Fastest known face detector for gray scale images (circa 2001, Marr Prize Winner).

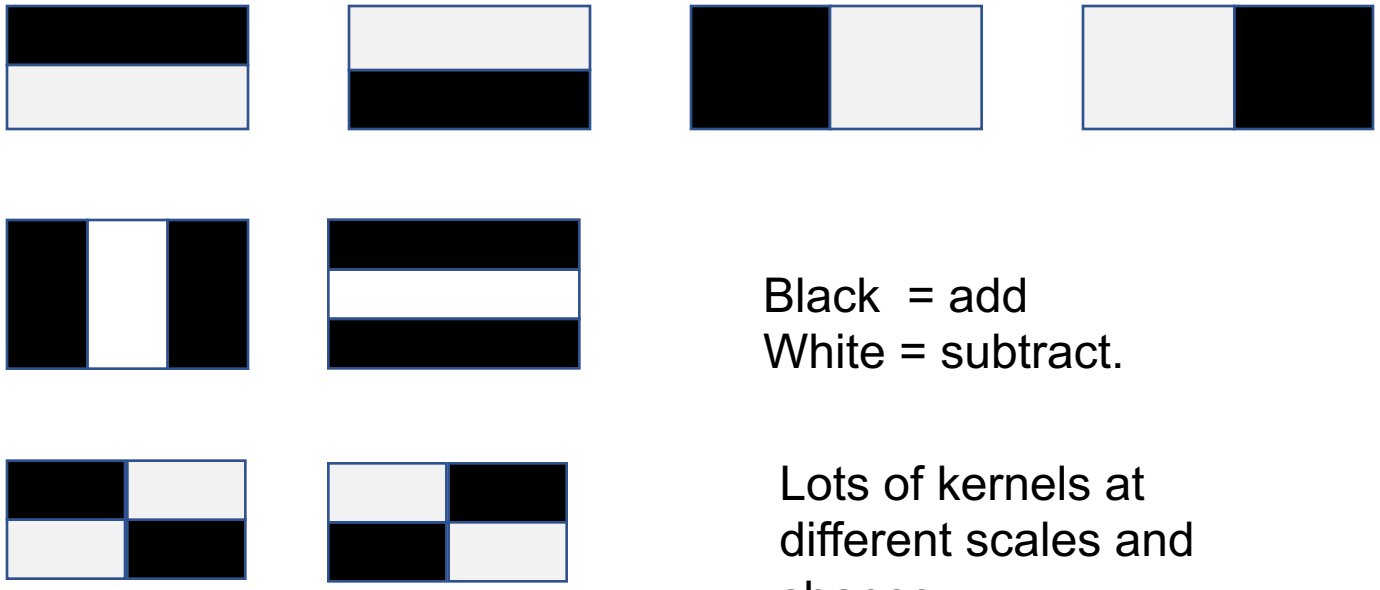
48

Viola-Jones face detection

- 3 key ideas
  - ➢ Use of Haar features
  - ➢ Integral Image
  - ➢ Cascade classifier

- For more information, read the original paper.
- Also, lots of videos on the web.
  - ➢ https://www.youtube.com/watch?v=uEJ71VlUmMQ&vl=en-GB (Google: Viola-Jones Computerphile)

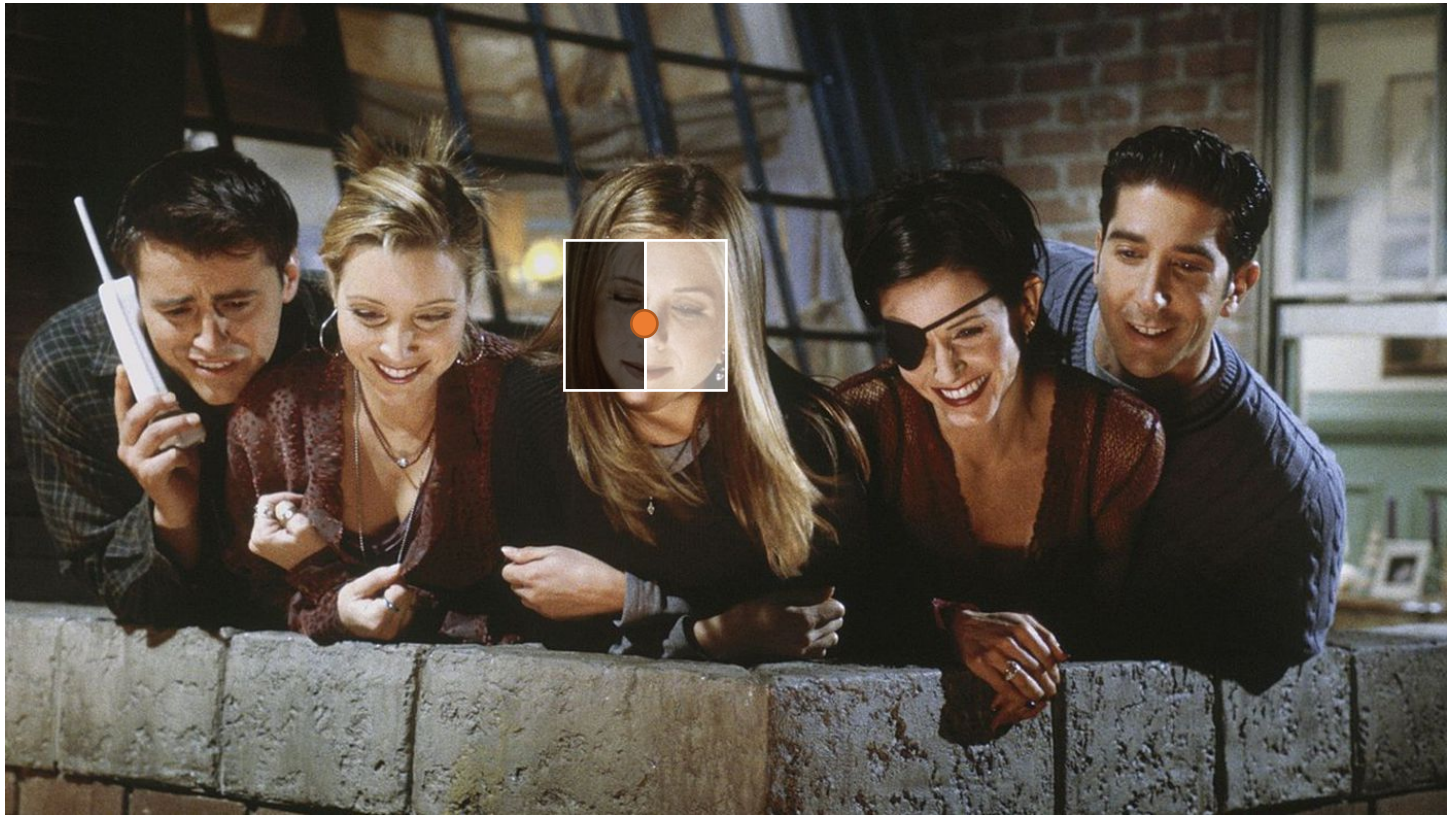
Haar features



Black = add
White = subtract.

Lots of kernels at different scales and shapes.

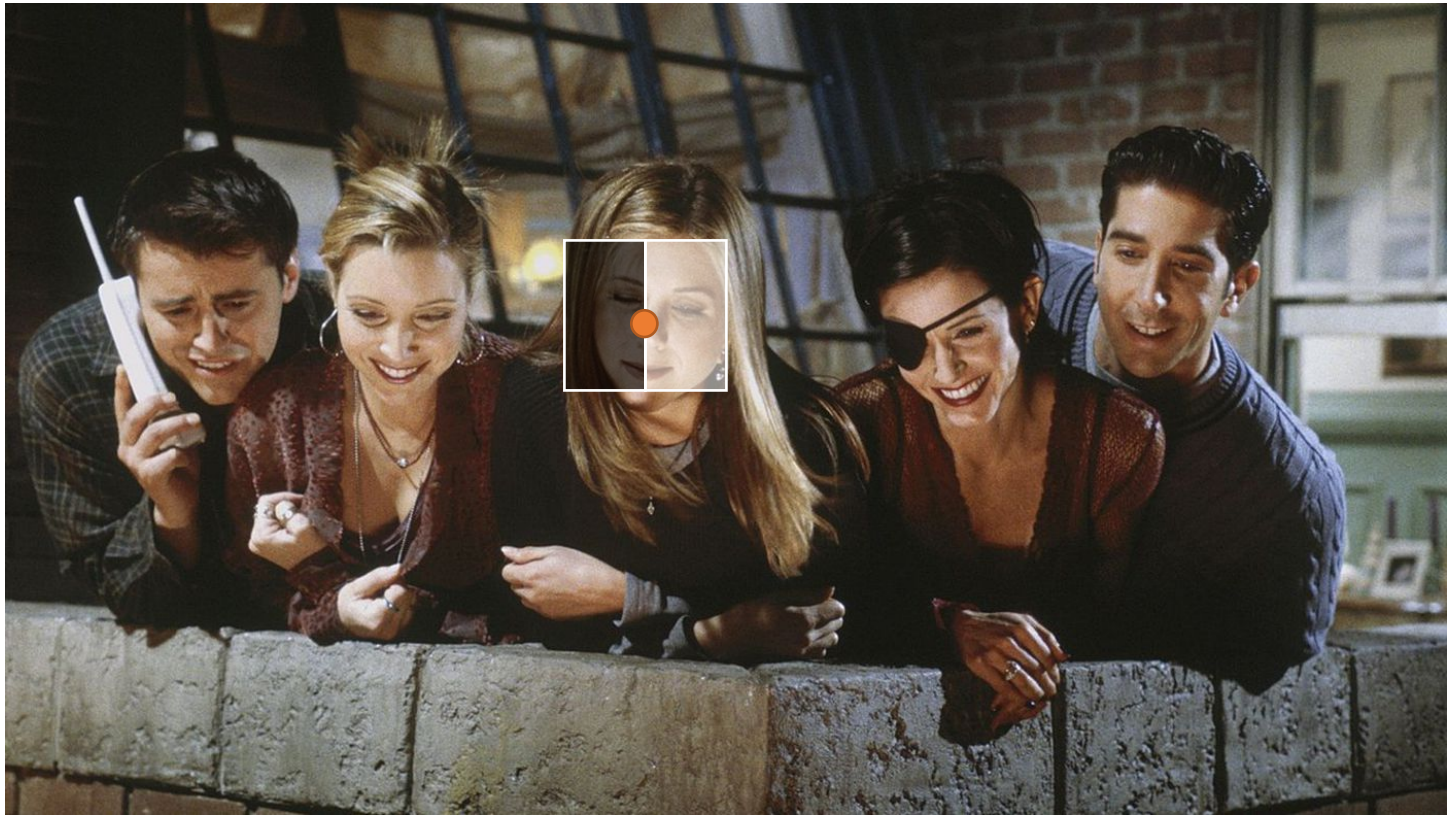
Haar Features

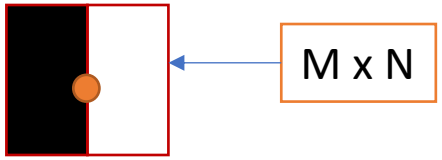


$$V_A[i,j] = \sum(\text{pixel intensities in black area}) - \sum(\text{pixel intensities in white area})$$

Slides from Shree Nayar

Haar Features





Computation Cost
= (NxM-1) additions per pixel, per filter, per scale.

How could we compute the features Efficiently?

$$V_A[i,j] = \sum(\text{pixel intensities in black area}) - \sum(\text{pixel intensities in white area})$$

Slides from Shree Nayar

Integral Image

Integral Image

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image

- A table that holds the sum of all pixel values to the left and top of a given pixel, inclusive.

Slides from Shree Nayar

Integral Image

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image

- A table that holds the sum of all pixel values To the left and top of a given pixel, inclusive.

Slides from Shree Nayar

Integral Image

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image

- A table that holds the sum of all pixel values To the left and top of a given pixel, inclusive.

Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image

Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

P

Integral Image II

Sum = II_P …
=7371

Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Q

P

Integral Image II

Sum = II_P – II_Q….
=7371 - 3240

Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I

162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Q

P

S

Integral Image II

Sum = II_P – II_Q-II_S…
=7371 – 3240 – 3159…

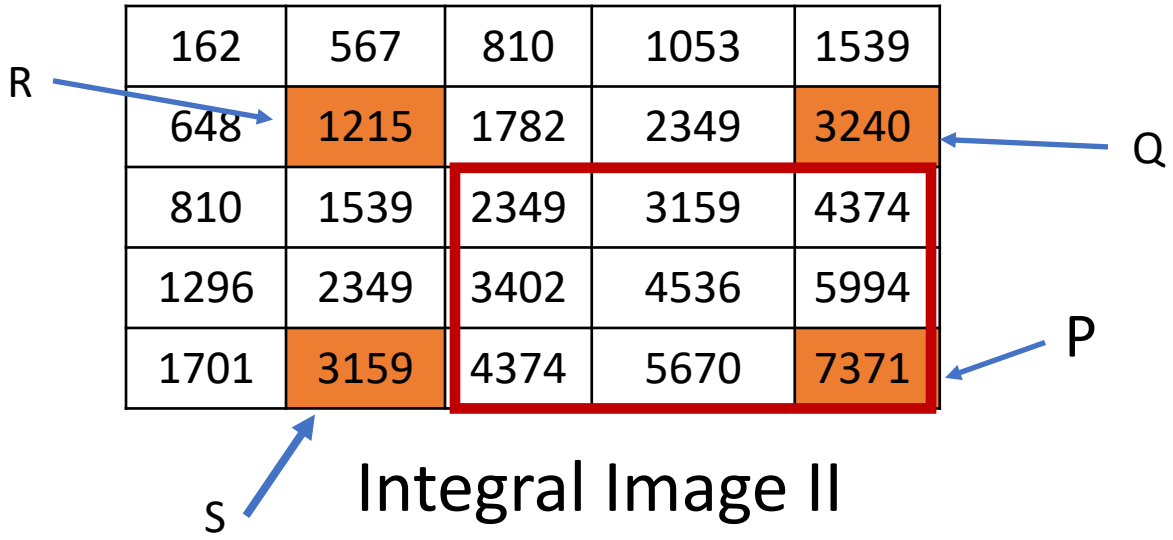
Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I



162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image II

Sum = II_P – II_Q-II_S + II_R
=7371 – 3240 – 3159 + 1215
= 2187

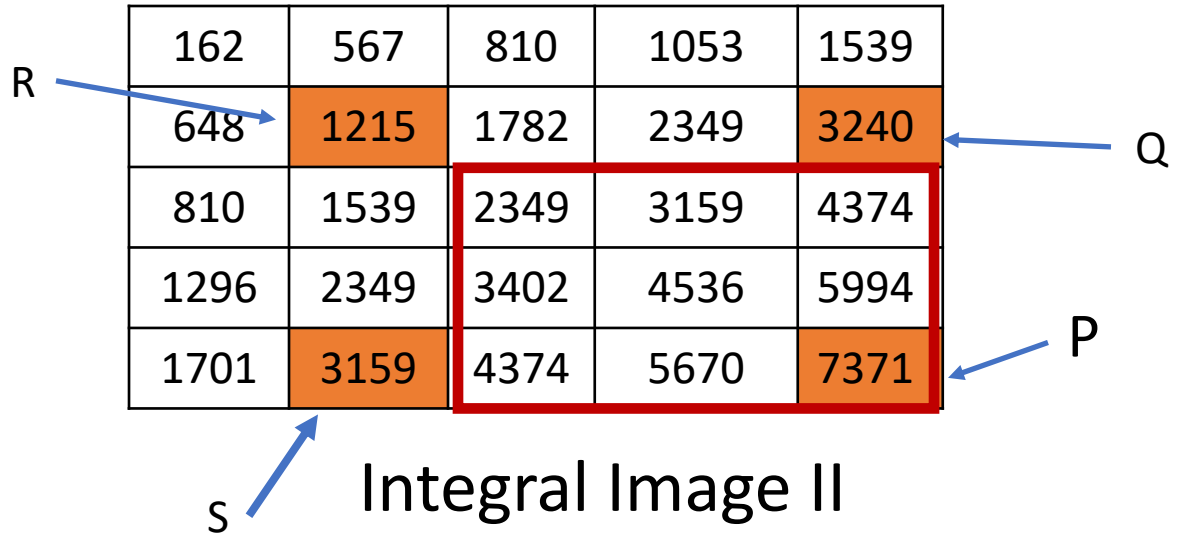
Slides from Shree Nayar

Integral Image

- Fast computation of arbitrary rectangles using integral image.

162	405	243	243	486
486	162	324	324	405
162	162	243	243	324
486	324	243	324	243
405	405	162	162	243

Image I



162	567	810	1053	1539
648	1215	1782	2349	3240
810	1539	2349	3159	4374
1296	2349	3402	4536	5994
1701	3159	4374	5670	7371

Integral Image II

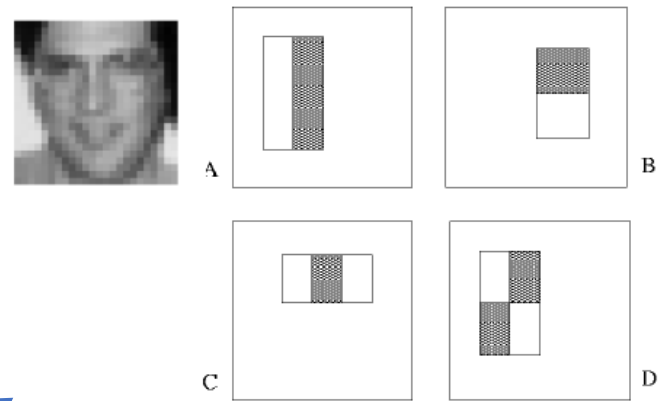
Sum = II_P – II_Q-II_S + II_R
=7371 – 3240 – 3159 + 1215
= 2187

Computational cost: only 3 additions which is Invariant to the size of the rectangle.

Slides from Shree Nayar

Image features

- “Rectangle filters”
- Differences between sums of pixels in adjacent rectangles



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

Threshold determined by training

50

Cascade classifier

Slides adapted from Prof. Richard Hartley

Constructing the classifier

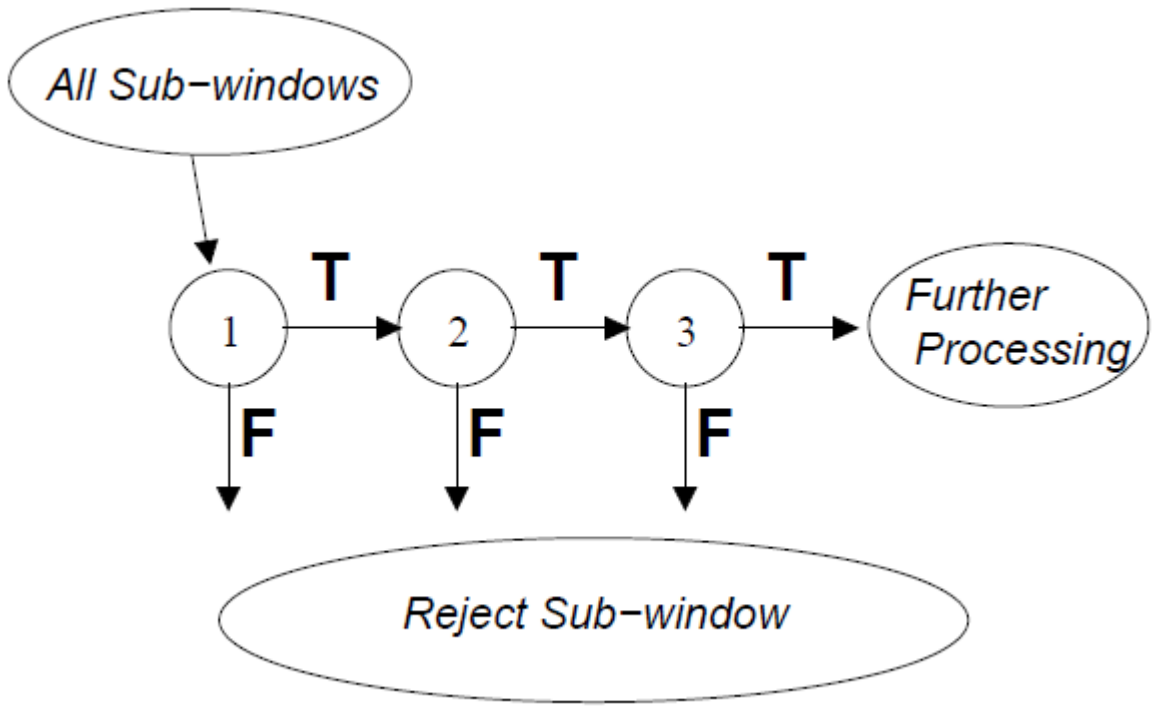
- Perceptron yields a sufficiently powerful weak classifier

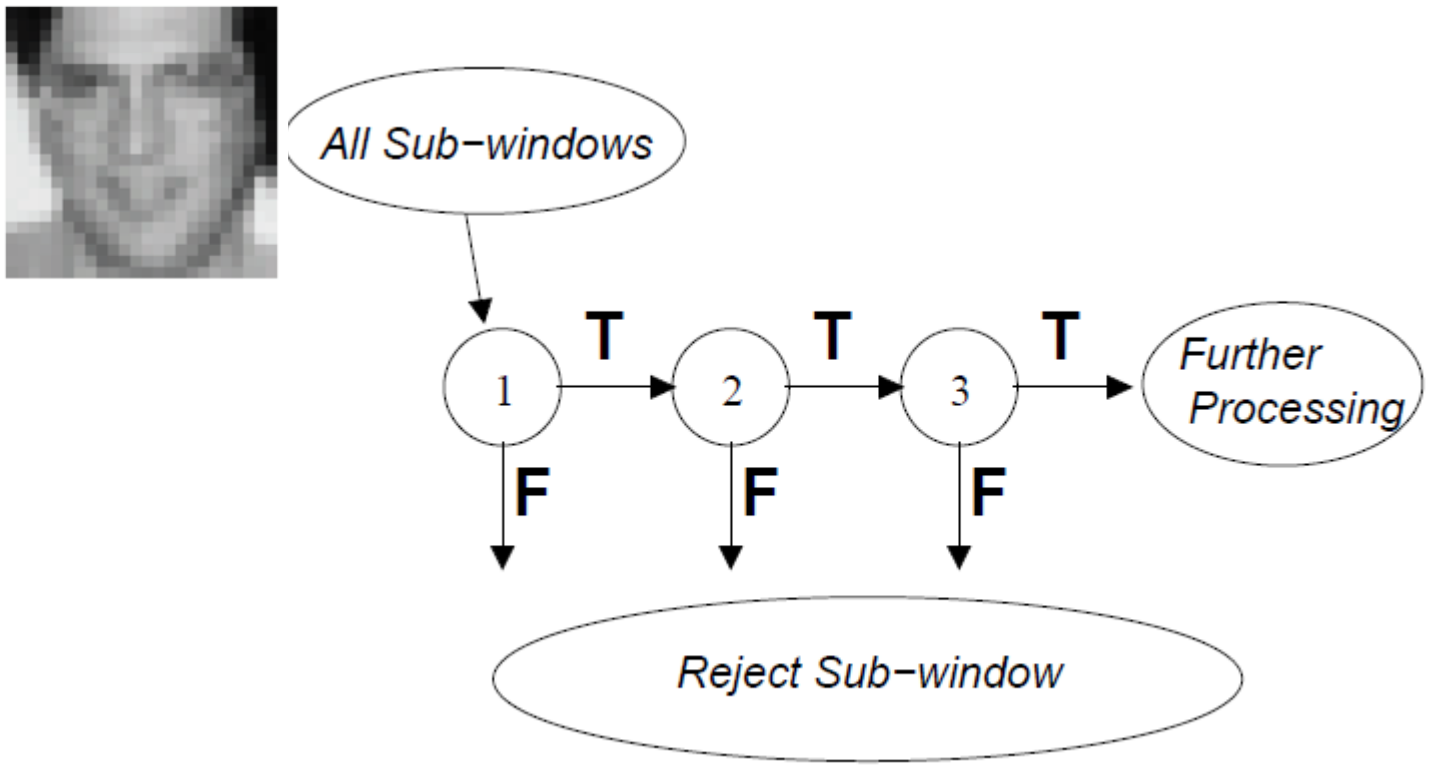
$$C(x) = \theta\left(\sum_i \alpha_i h_i(x) + b\right)$$

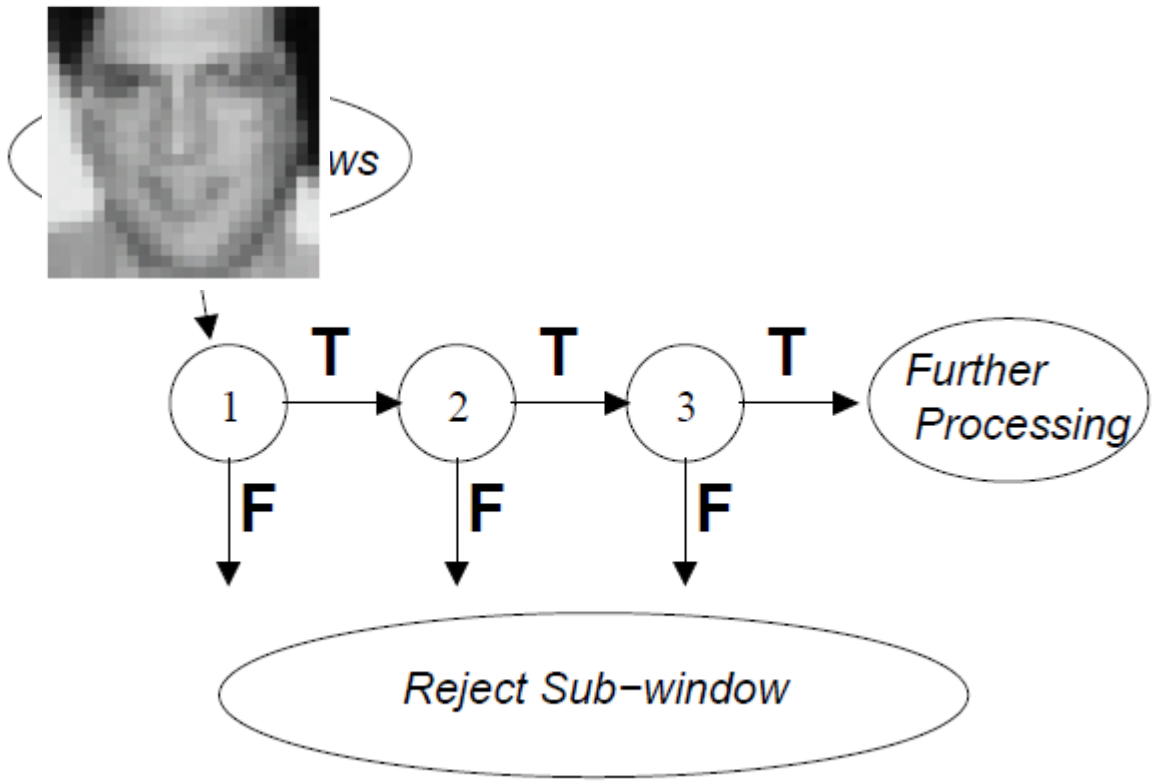
- Use AdaBoost to efficiently choose best features among the feature library.
- add a new $h_i(x)$ at each round
- each $h_i(x_k)$ is a “decision stump”

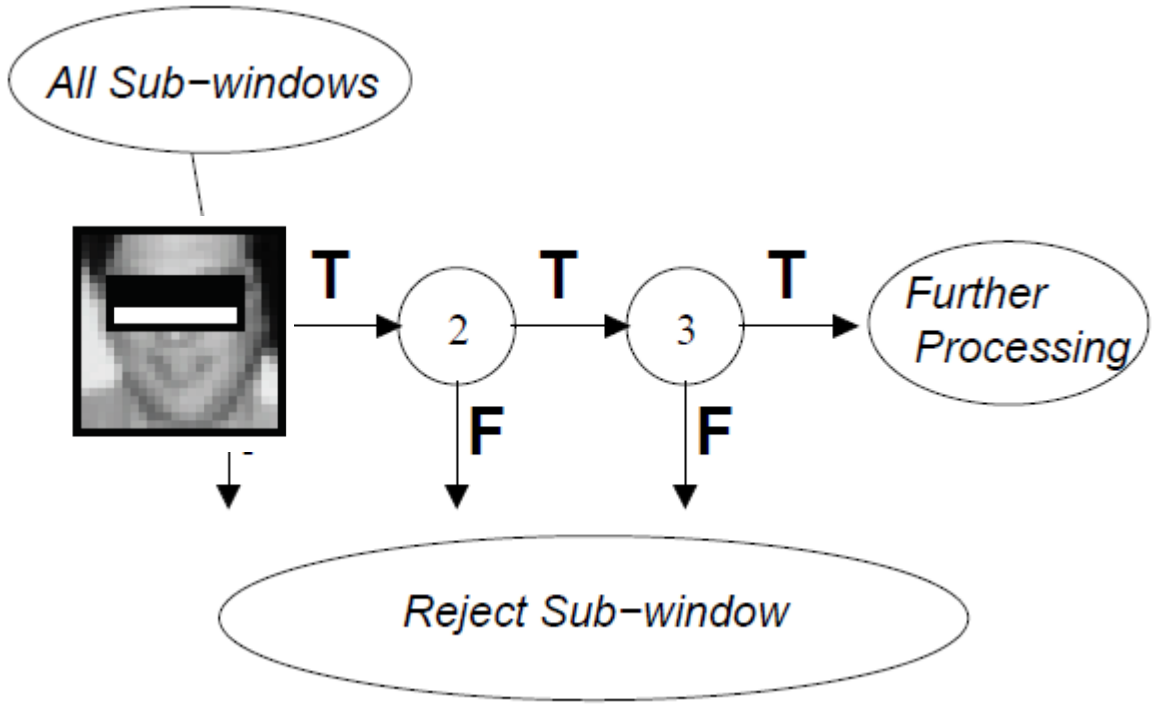
Constructing the classifier

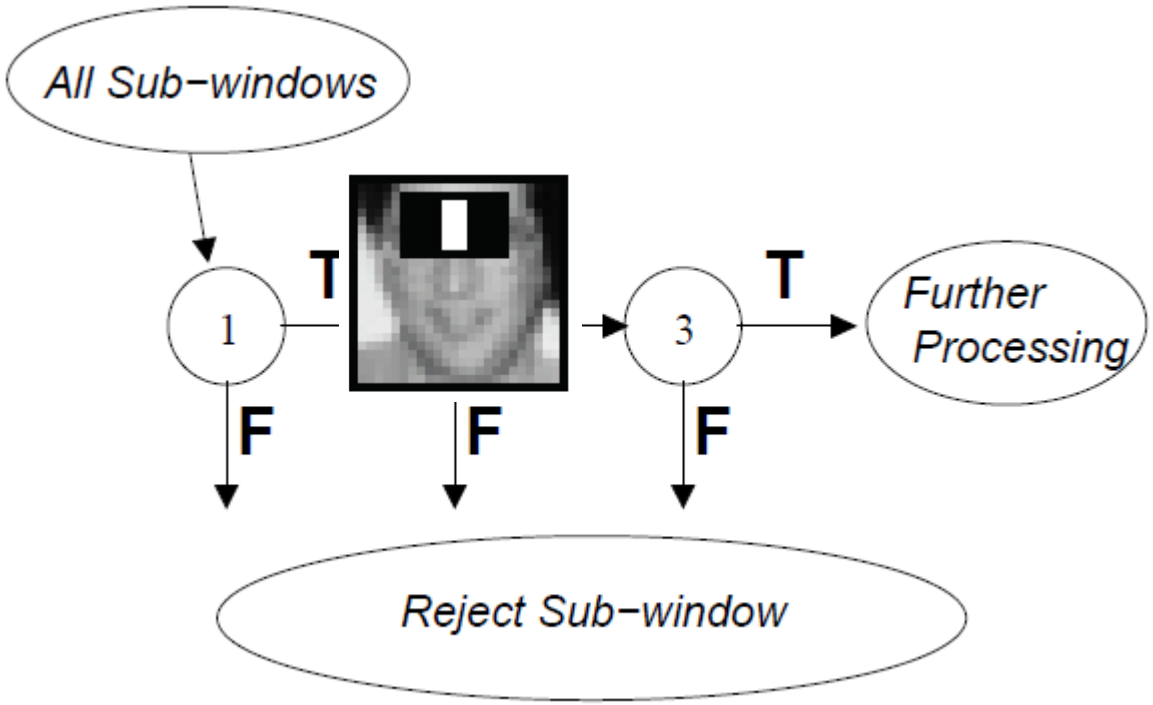
- For each round of boosting:
- Evaluate each rectangle filter on each example
- Sort examples by filter values
- Select best threshold for each filter (min error)
- Use sorting to quickly scan for optimal threshold
- Select best filter/threshold combination
- Weight is a simple function of error rate
- Reweight examples

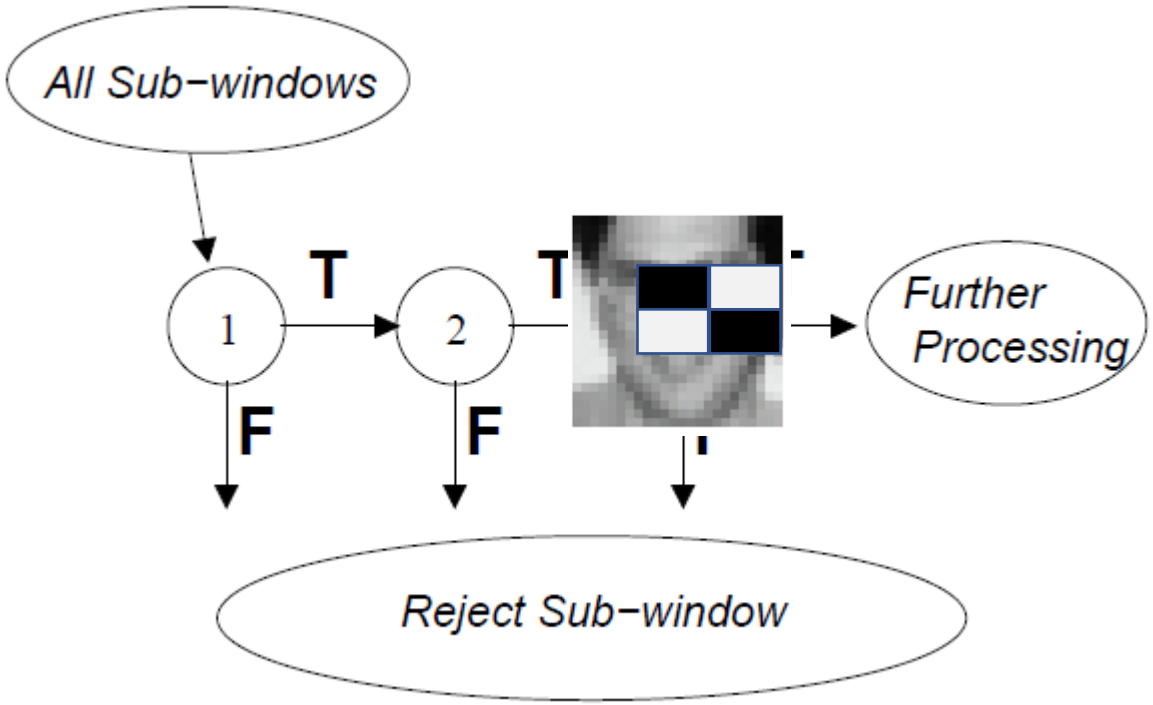

All Sub−windows
1
2
3
T
T
T
F
F
F
Further
Processing
Reject Sub−window

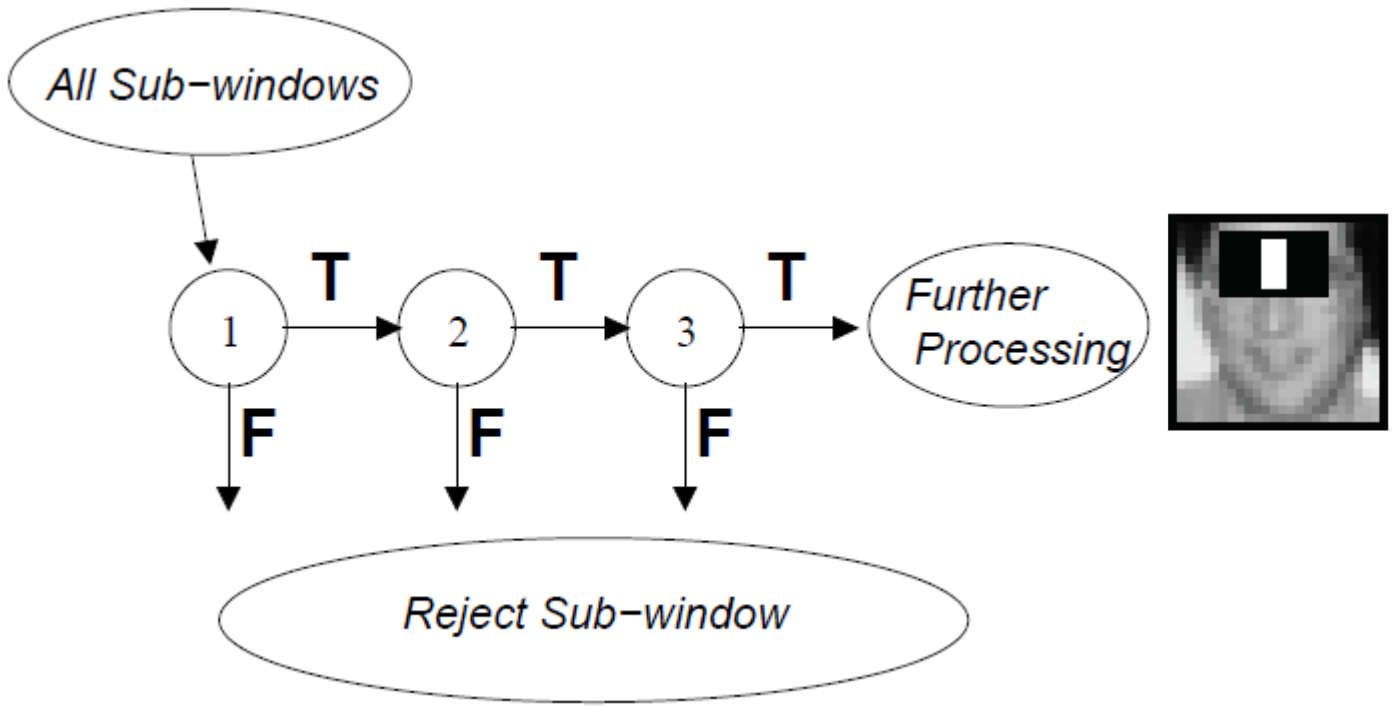

All Sub−windows
1
2
3
T
T
T
F
F
F
Further
Processing
Reject Sub−window

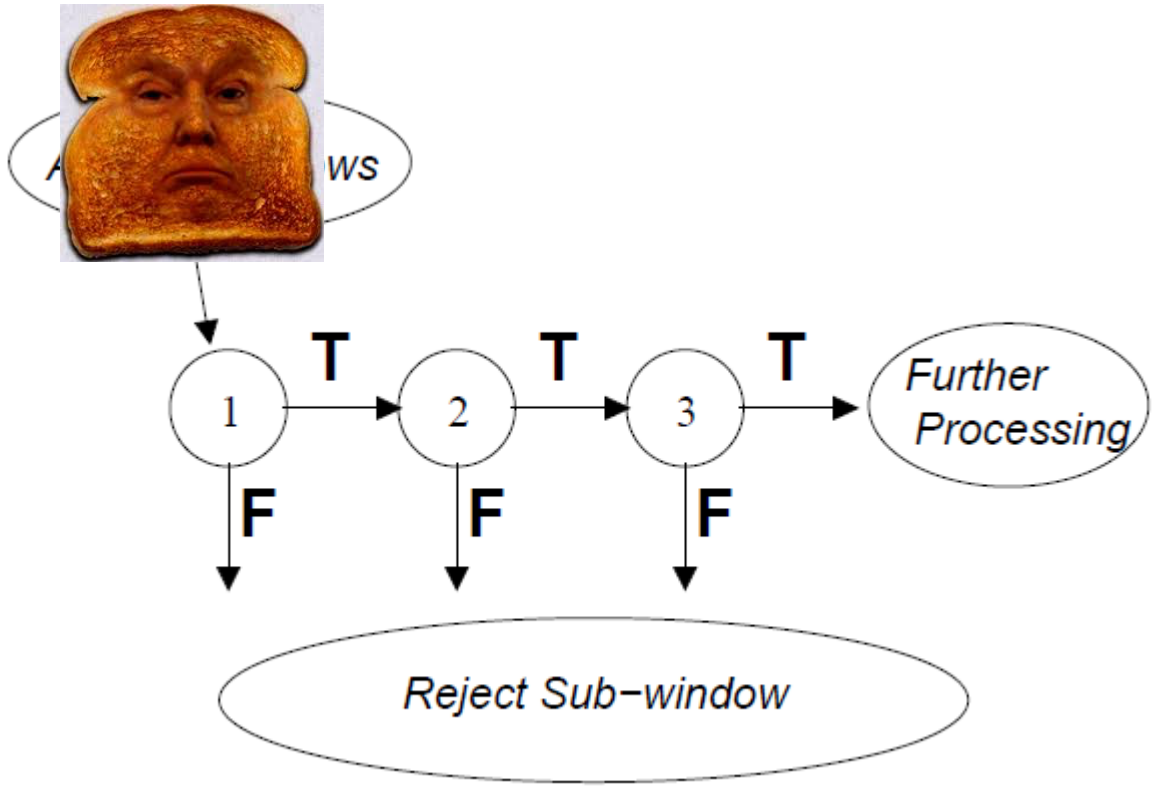

ws
1
2
3
T
T
T
F
F
F
Further
Processing
Reject Sub−window

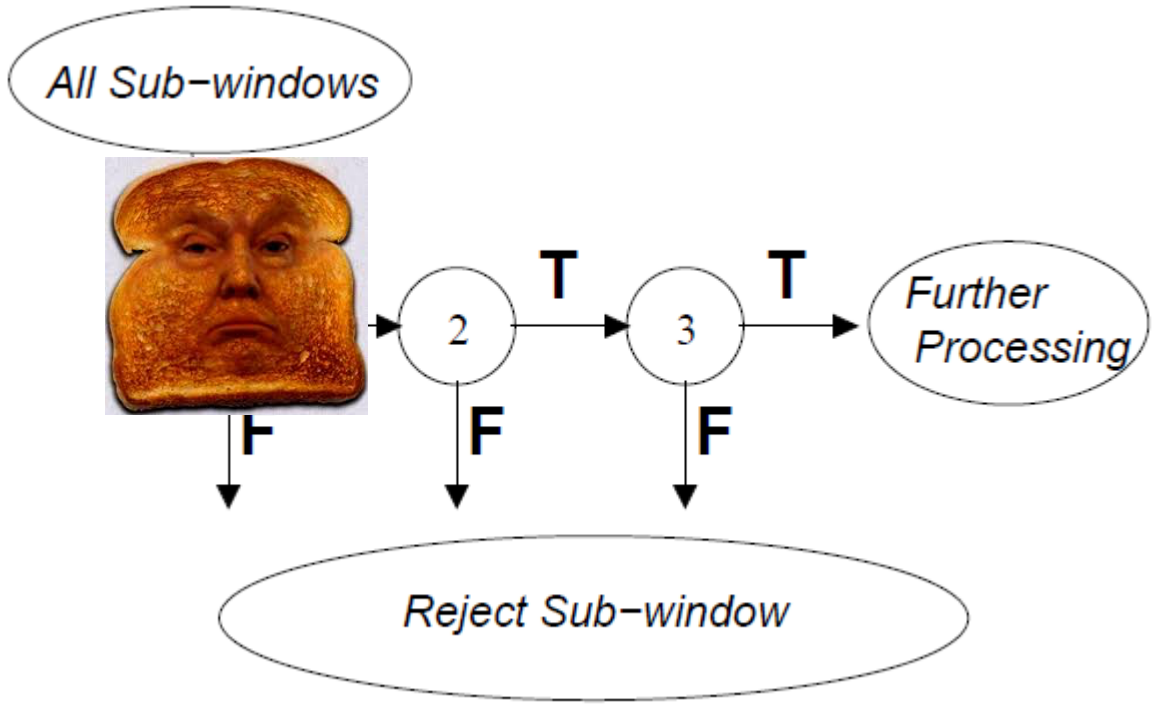

All Sub-windows
T
T
T
2
3
Further Processing
F
F
Reject Sub-window

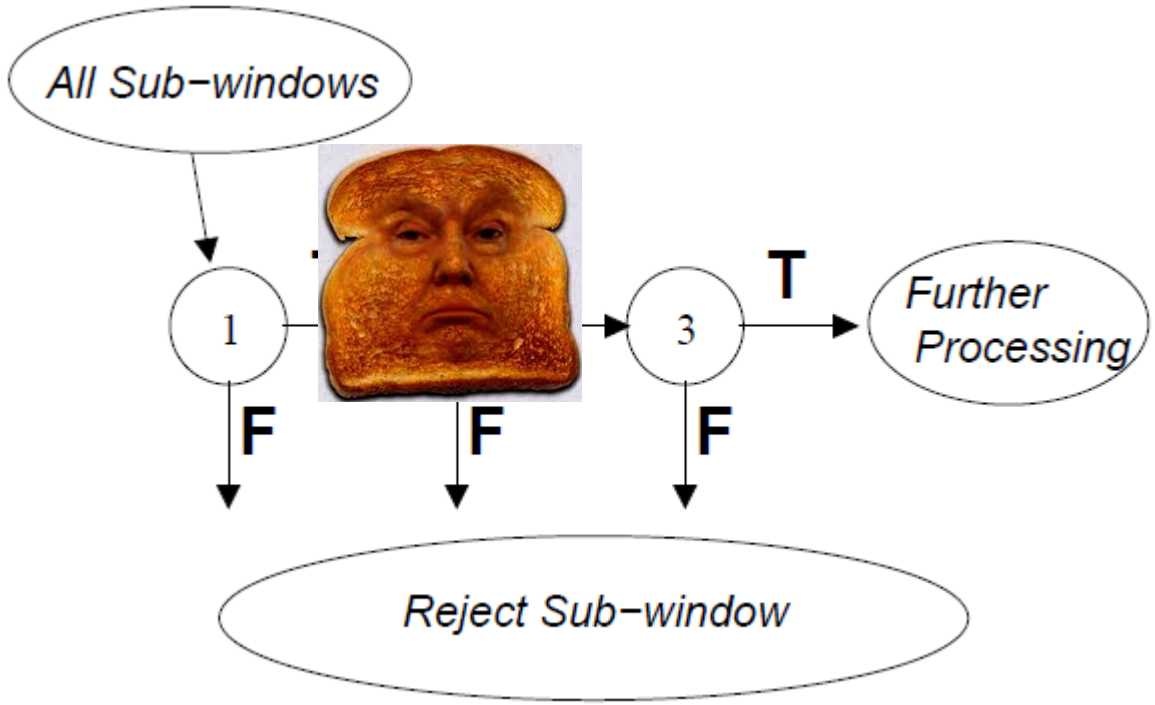

All Sub-windows
1
T
3
T
Further Processing
F
F
F
Reject Sub-window

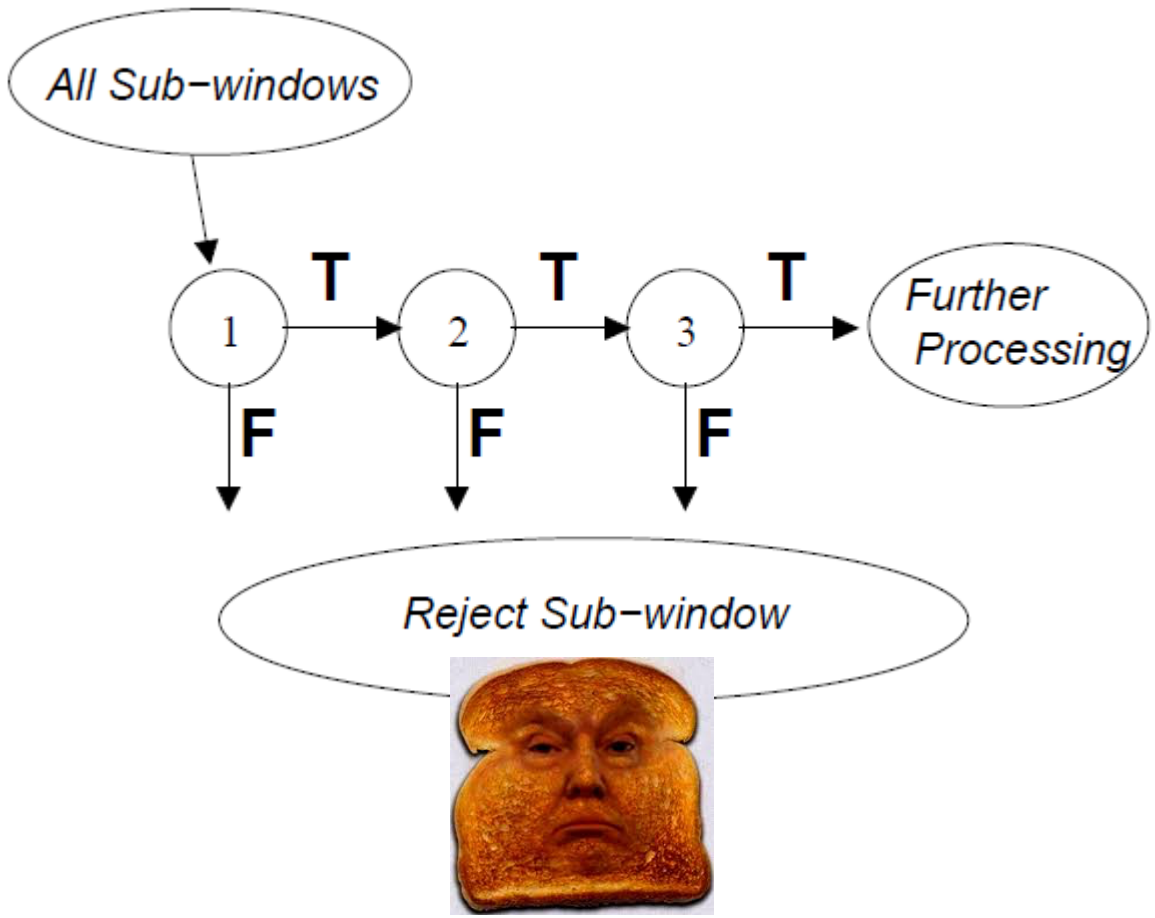

All Sub-windows
1
T
2
T
F
F
Further
Processing
Reject Sub-window


All Sub-windows
1
2
3
T
T
T
F
F
F
Further Processing
Reject Sub-window

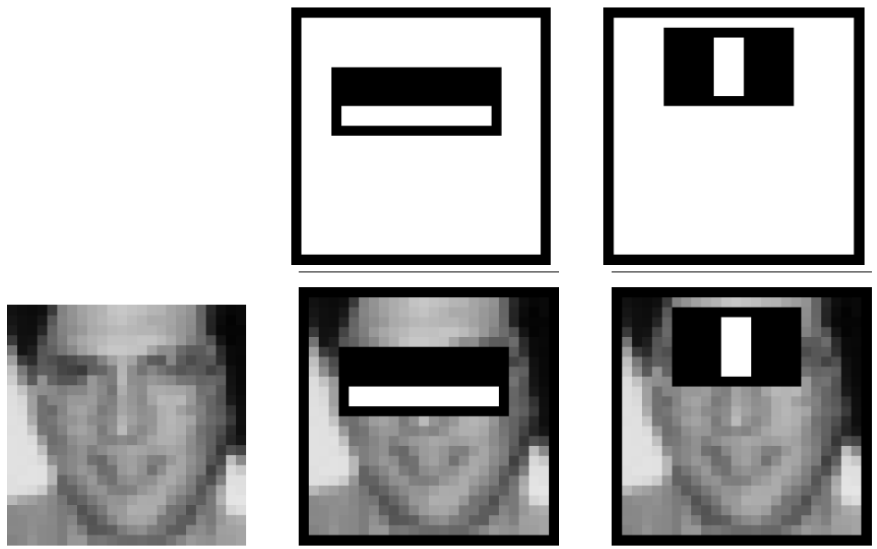

ws
T
T
T
1
2
3
Further
Processing
F
F
F
Reject Sub−window


All Sub−windows
T
T
2
3
Further
Processing
F
F
F
Reject Sub−window


All Sub−windows
1
3
T
Further
Processing
F
F
F
Reject Sub−window


All Sub−windows
1
2
3
T
T
T
F
F
F
Further Processing
Reject Sub−window

Viola-Jones Face Detector: Results



First two features selected

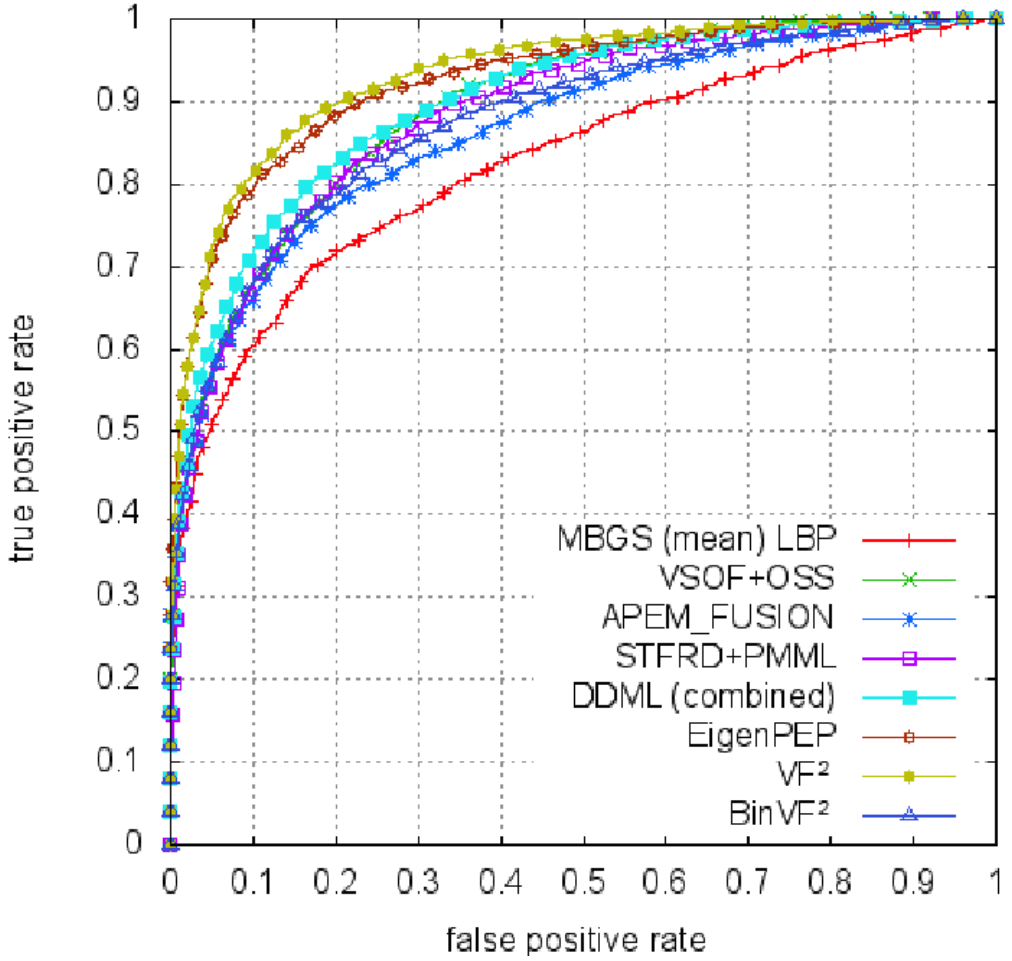
Possible outcomes of a classifier

	is face	is NOT face
says face	TP	FP
Says NOT face	FN	TN

TP : true positive
FP : false positive
TN : true negative
FN : false negative

$$\mathrm{TP} + \mathrm{FN} = 1.0$$

$$\mathrm{FP} + \mathrm{TN} = 1.0$$



ROC curve for face recognition.

A **receiver operating characteristic curve**, or **ROC curve**, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. Wiki

Assuming tests $i = 1, \ldots, n$ are independent:

$$\text{Probability of true face passing test } i = \text{TP}_i$$

$$\text{Probability of true face passing all tests} = \text{TP}_{\text{all}} = \prod_i \text{TP}_i$$

$$\text{Probility of false face passing test } i = \text{FP}_i$$

$$\text{Probability of false face passing all tests} = \text{FP}_{\text{all}} = \prod_i \text{FP}_i$$

Assuming tests $i = 1, \ldots, n$ are independent:

$$\text{Probability of true face passing test } i = \text{TP}_i$$

$$\text{Probability of true face passing all tests} = \text{TP}_{\text{all}} = \prod_i \text{TP}_i$$

$$\text{Probility of false face passing test } i = \text{FP}_i$$

$$\text{Probability of false face passing all tests} = \text{FP}_{\text{all}} = \prod_i \text{FP}_i$$

We want TP_{all} to be large, and FP_{all} small.
For example, suppose that

$$\text{TP}_i = 0.999, \quad \text{FN}_i = 0.001$$
$$\text{FP}_i = 0.95, \quad \text{TN}_i = 0.05$$

and suppose there are 100 tests. Then

$$\text{TP}_{\text{all}} = 0.999^{100} = 0.904$$
$$\text{FP}_{\text{all}} = 0.95^{100} = 0.006$$

Test is relatively cheap on negative examples.

Expected number of tests applied to a non-face is

$$E[\text{number tests}] = \frac{1}{\text{TN}_i}$$

which is 20 if $\text{FP}_i = 0.95$

Proof:

$$
\begin{aligned}
E[\text{number tests}] &= 1 \times \mathrm{TN} + 2 \times \mathrm{FP} \times \mathrm{TN} \\
&+ 3 \times \mathrm{FP}^2 \times \mathrm{TN} + 4 \times \mathrm{FP}^3 \times \mathrm{TN} + \cdots \\
&= \sum_{i=1}^{\infty} i \times \mathrm{FP}^{i-1} \times \mathrm{TN} \\
&= aa \qquad (1)
\end{aligned}
$$

We multiply FP on both sides of Eq. 1, we can get

$$
\frac{aa}{TN} \times \mathrm{FP} = \sum_{i=1}^{\infty} i \times \mathrm{FP}^{i} \qquad (2)
$$

$$
\begin{aligned}
\frac{aa}{TN}(1 - \mathrm{FP}) &= 1 + \mathrm{FP} + \mathrm{FP}^2 + \mathrm{FP}^3 + \cdots \qquad (3) \\
aa &= \frac{1}{1 - \mathrm{FP}} = \frac{1}{\mathrm{TN}} \qquad (4)
\end{aligned}
$$

Given $\mathrm{TN} = 0.05$, then $E[\text{number tests}] = 20$.

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

Table 1: The boosting algorithm for learning a query online. T hypotheses are constructed each using a single feature. The final hypothesis is a weighted linear combination of the T hypotheses where the weights are inversely proportional to the training errors.

56

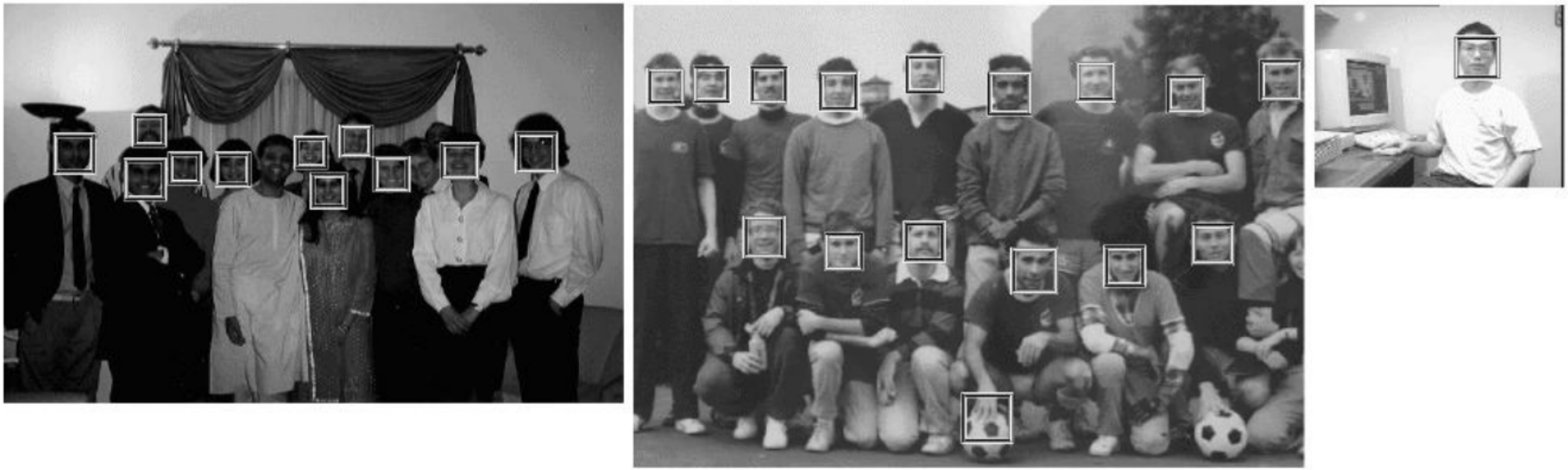
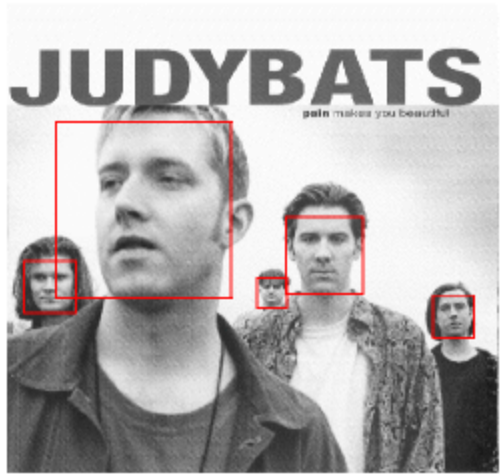
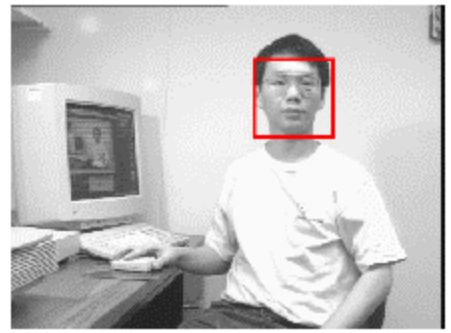
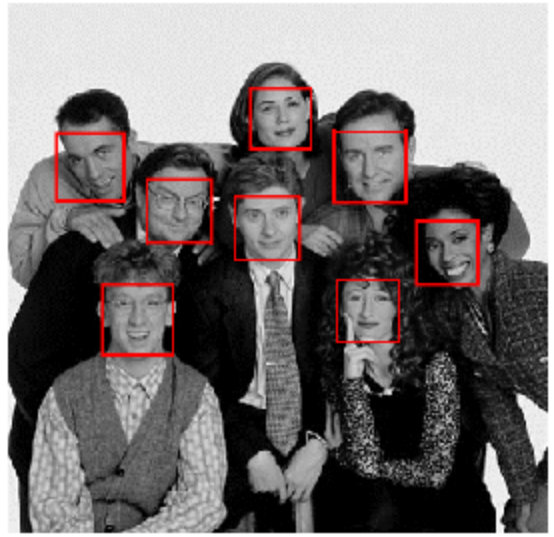


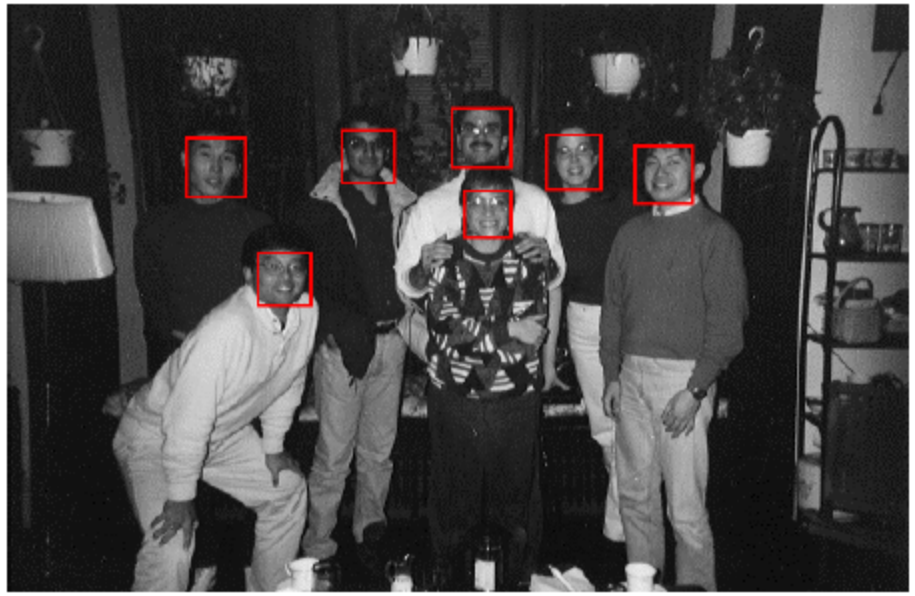
Figure 7: Output of our face detector on a number of test images from the MIT+CMU test set.

Viola-Jones Face Detector: Results



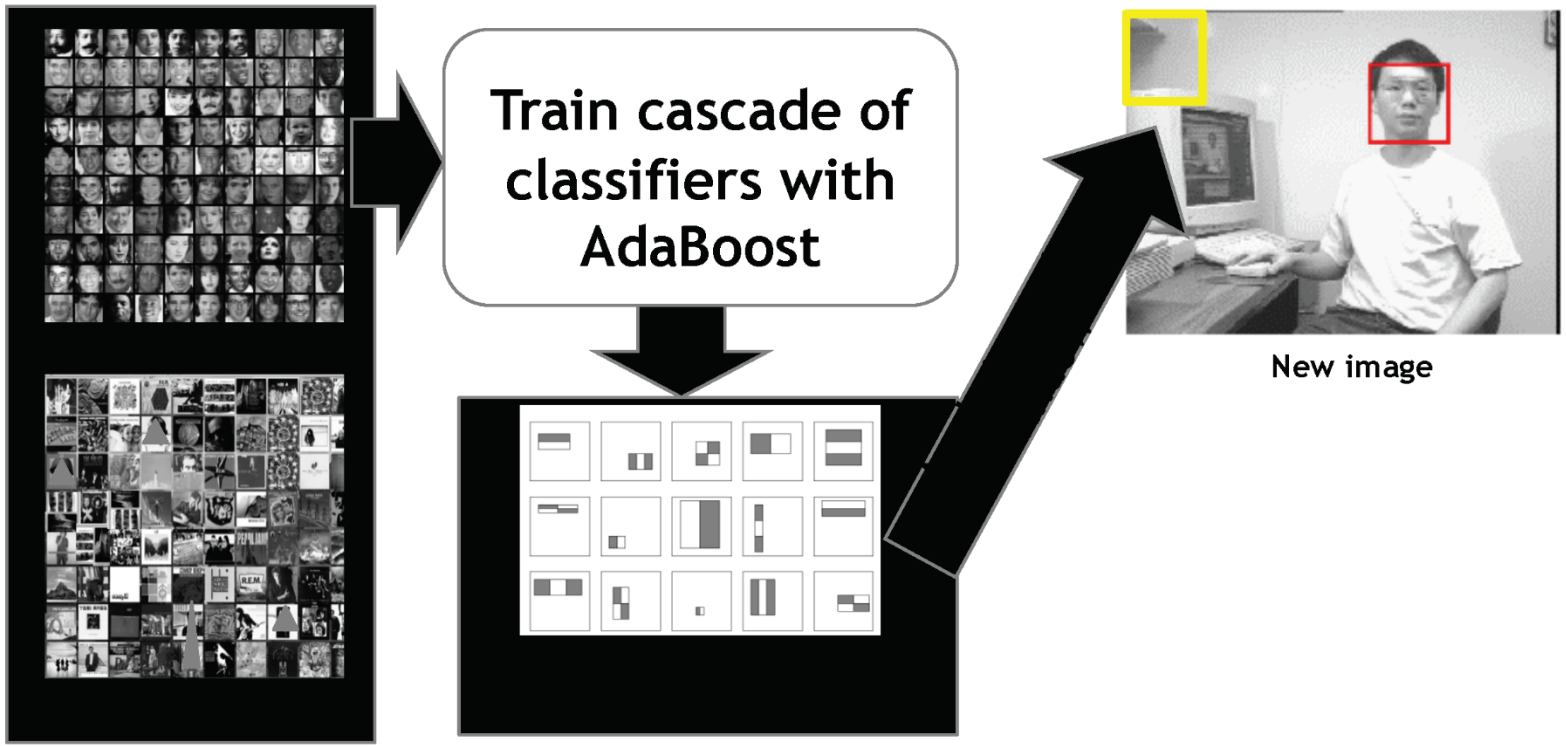






60

Viola-Jones Face Detector: Summary



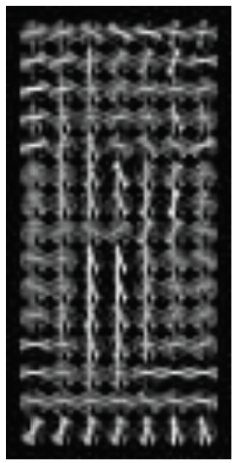
- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://www.intel.com/technology/computing/opencv/]

Summary (Viola-Jones)

- Fastest known real-time face detector for gray images (in 2001)
- Three contributions with broad applicability:
  - ❖Cascaded classifier yields rapid classification
  - ❖AdaBoost as an extremely efficient feature selector
  - ❖Rectangle Features + Integral Image can be used for rapid image analysis

Linear SVM and HOG for pedestrian detection

Person detection with HoG's & linear SVM's



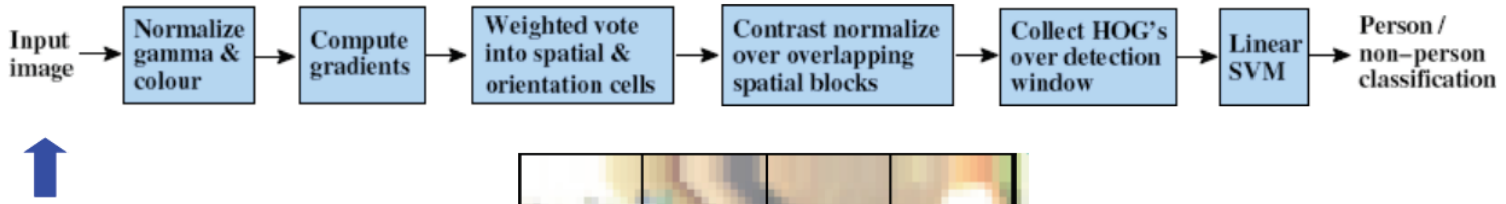
- Histogram of oriented gradients (HoG): Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

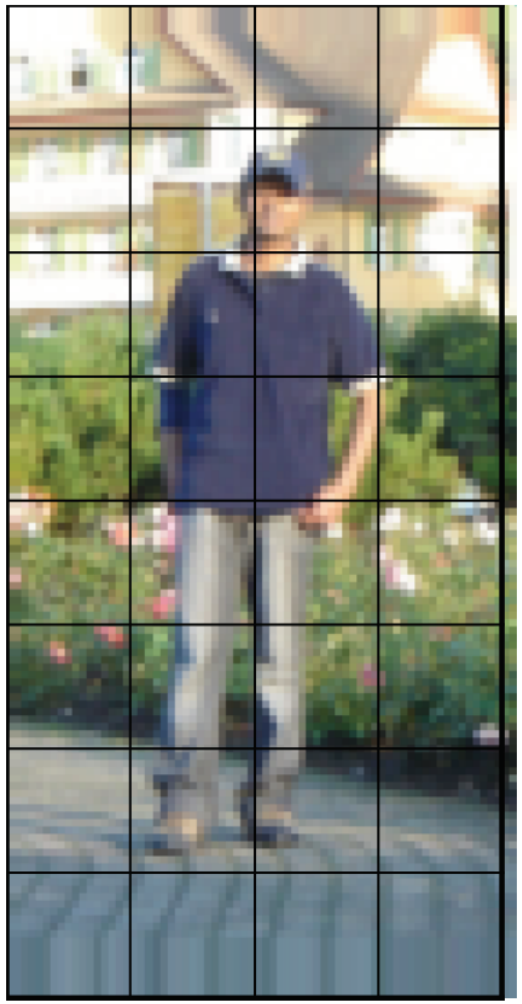
Dalal & Triggs, CVPR 2005

Person detection with HoGs & linear SVMs

- 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/

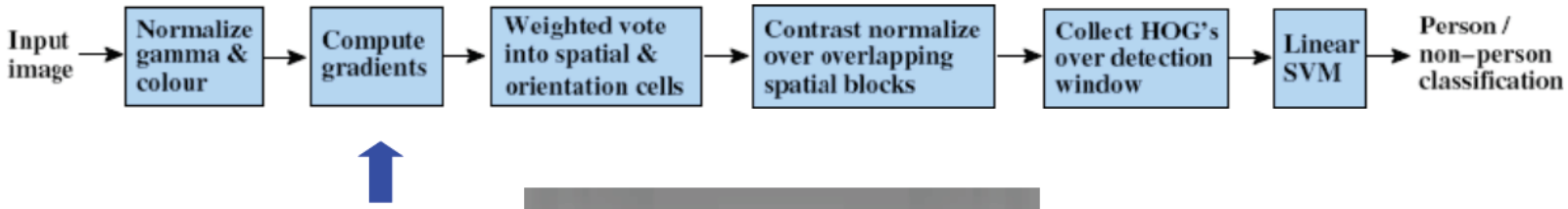
65

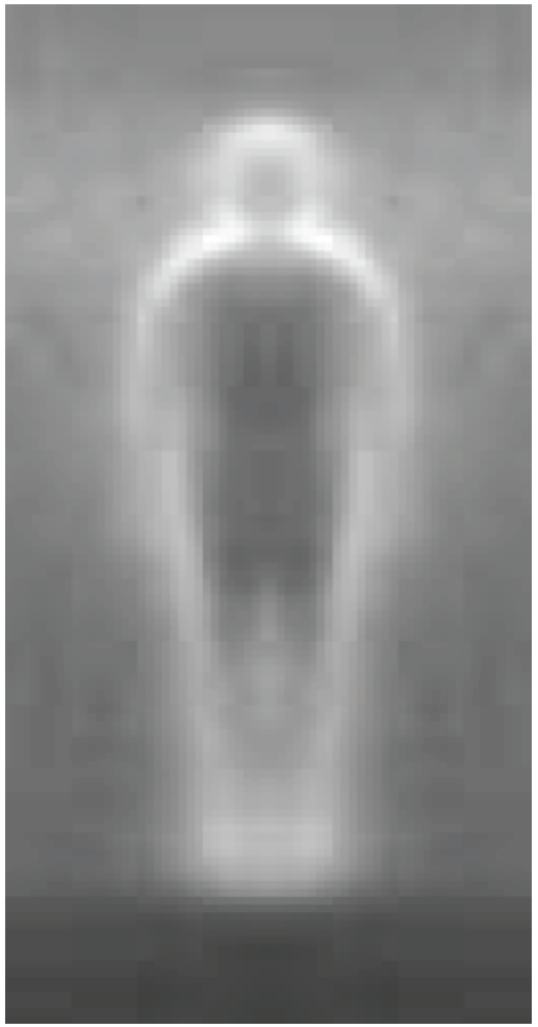




Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05 66





-1	0	1

centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected

0	1
-1	0

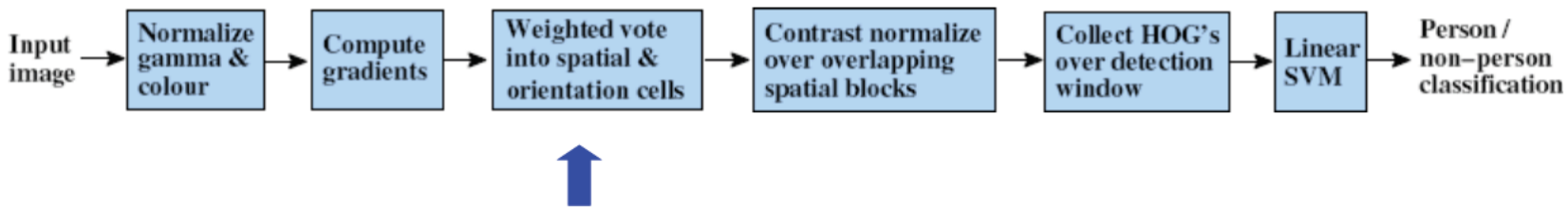
diagonal

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

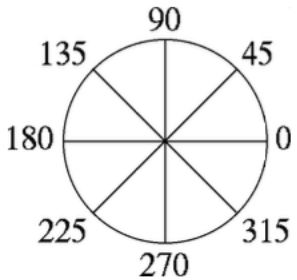
Sobel

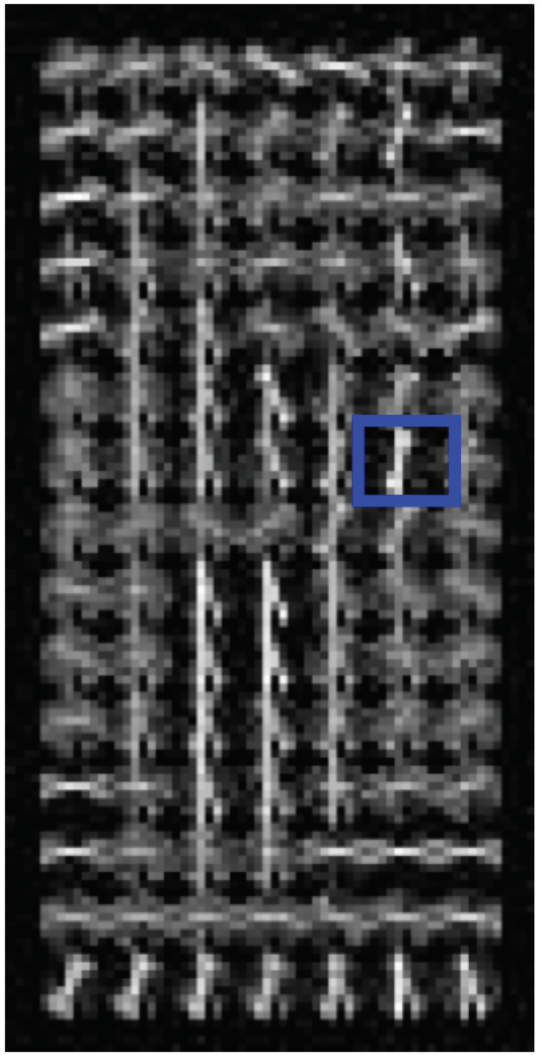
Slides by Pete Barnum

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



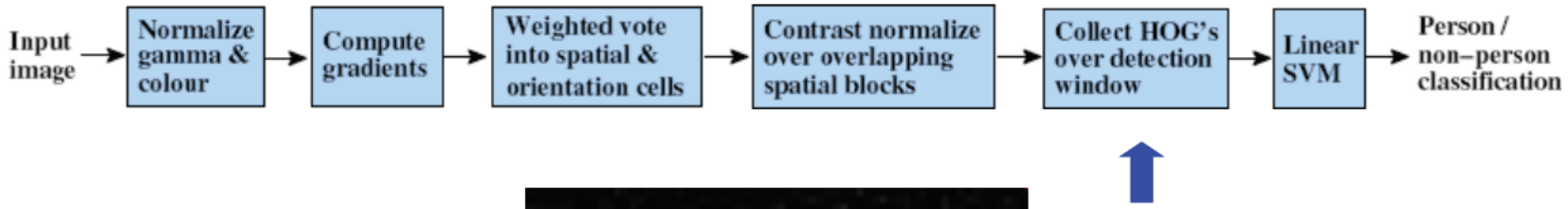
- Histogram of gradient orientations
  - -Orientation





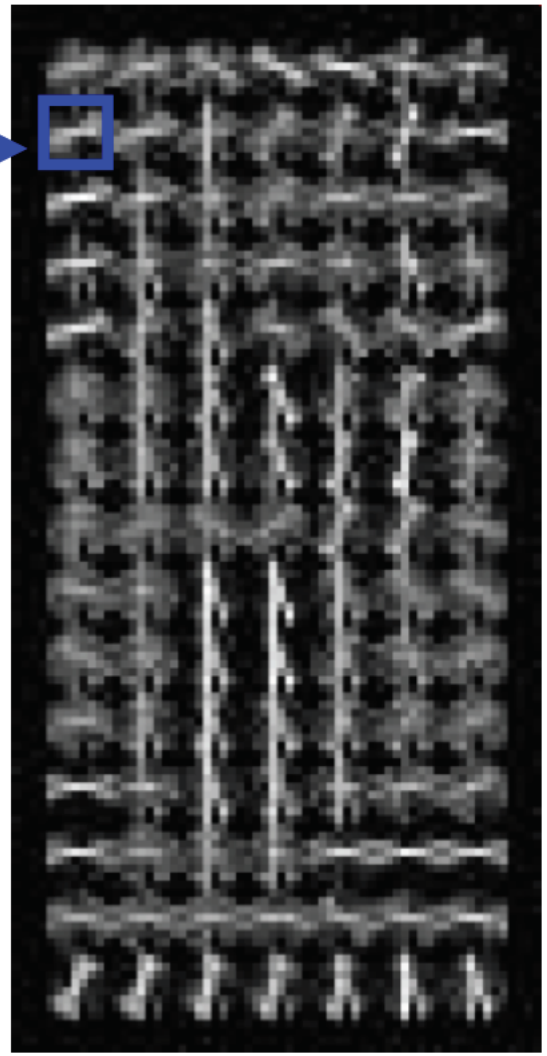
Slides by Pete Barnum

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



8 orientations

X=

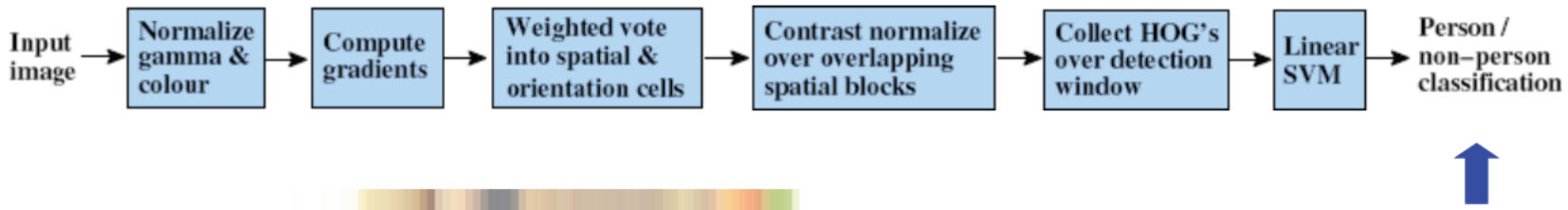


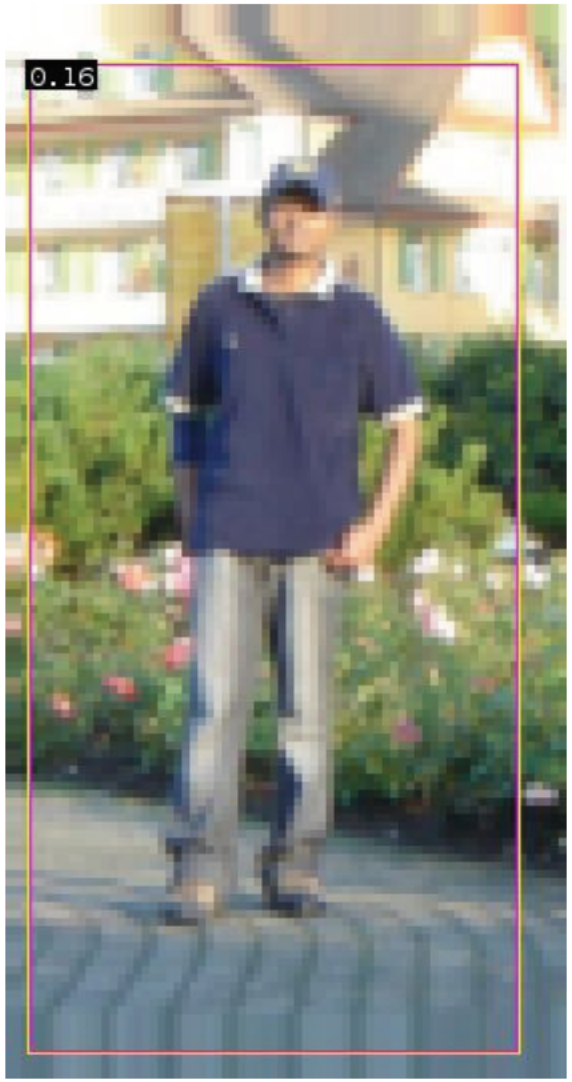
$\in R^{840}$

15x7 cells

Slides by Pete Barnum

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





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

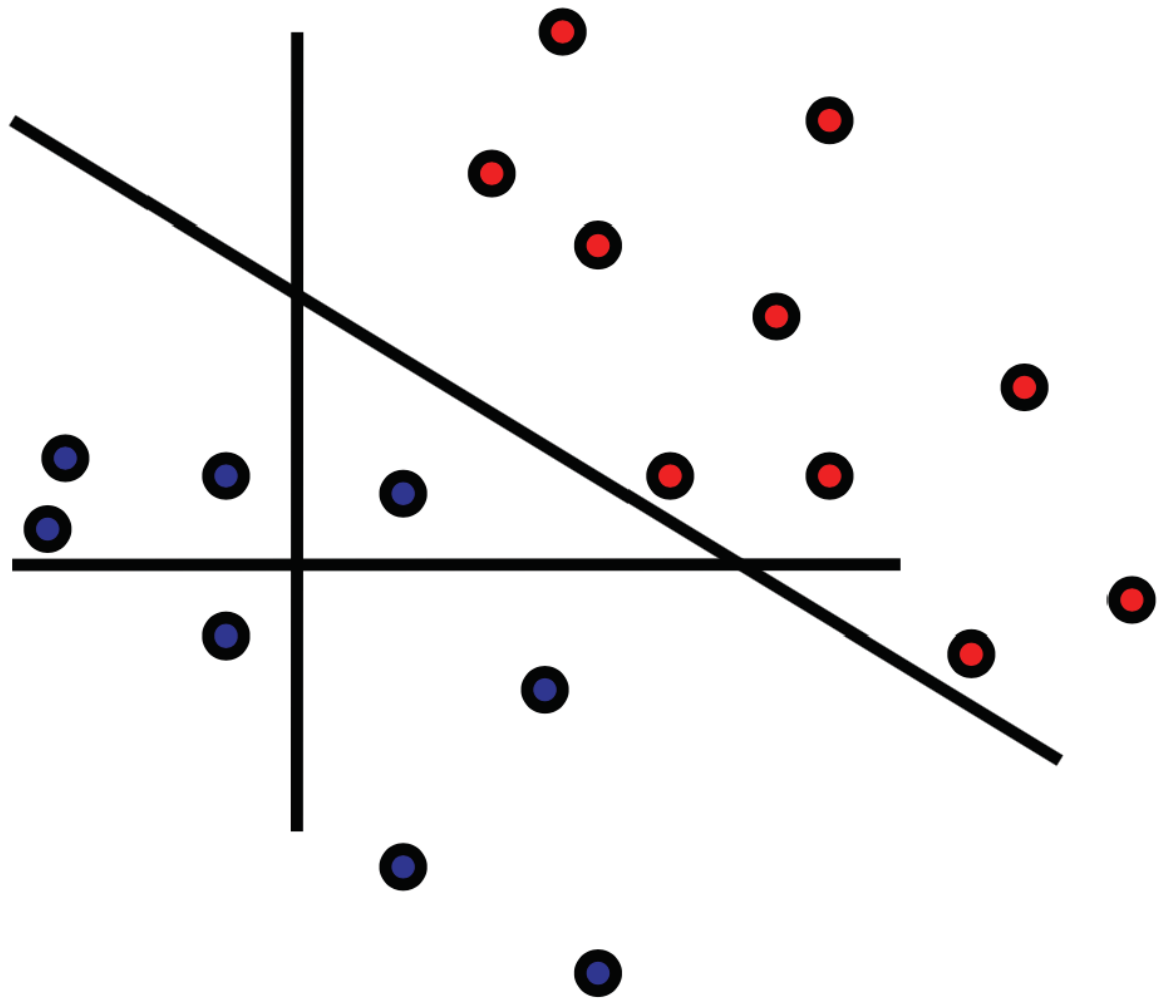
$$sign(0.16) = 1$$

$$=>$$ pedestrian

Slides by Pete Barnum

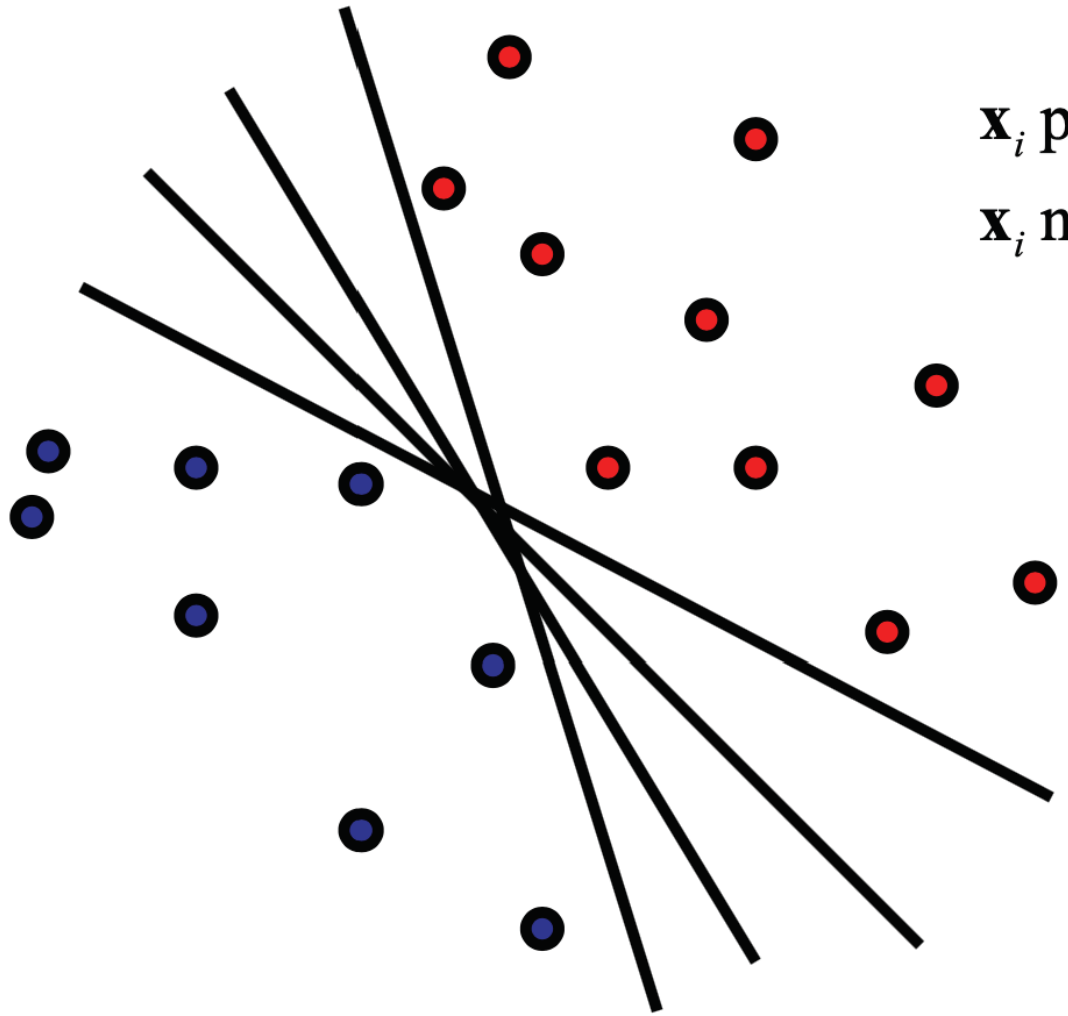
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Linear classifiers



Linear classifiers

- Find linear function to separate positive and negative examples

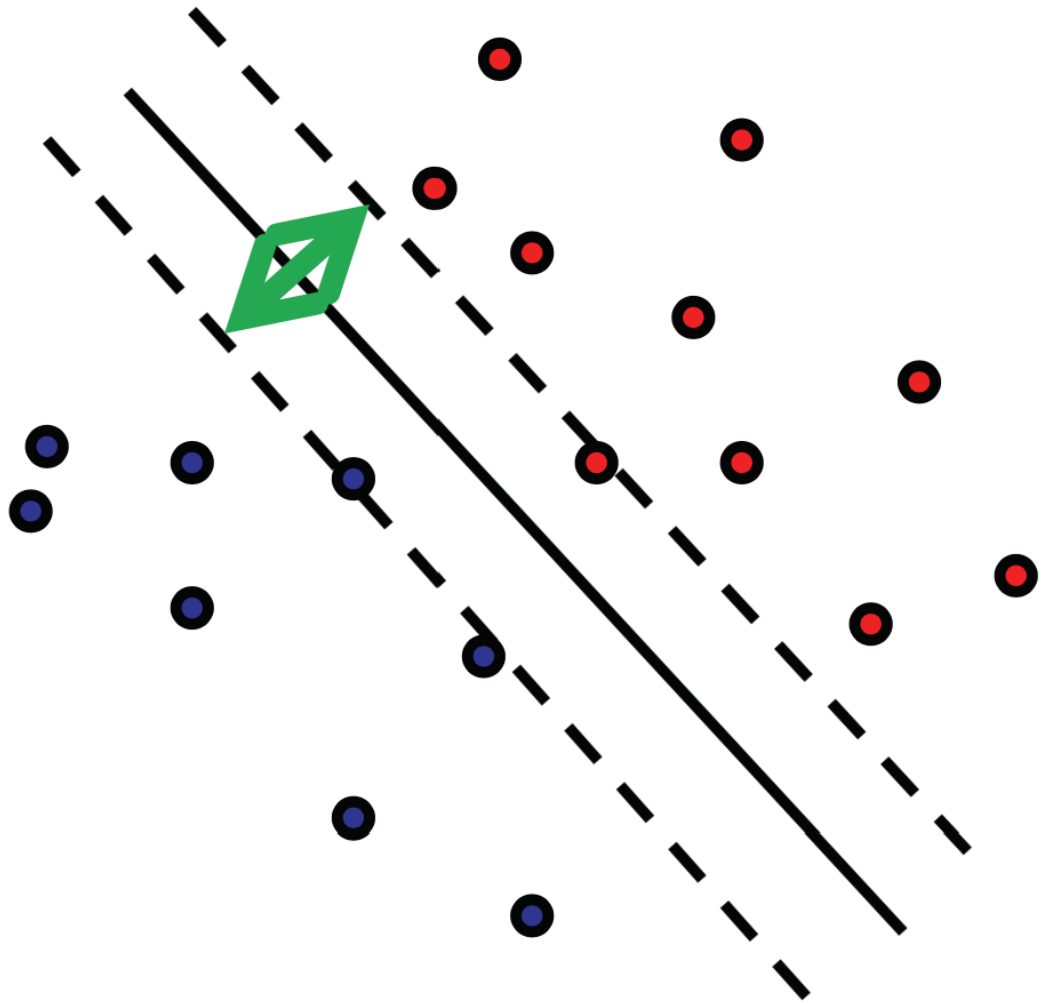


$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Which line is best?

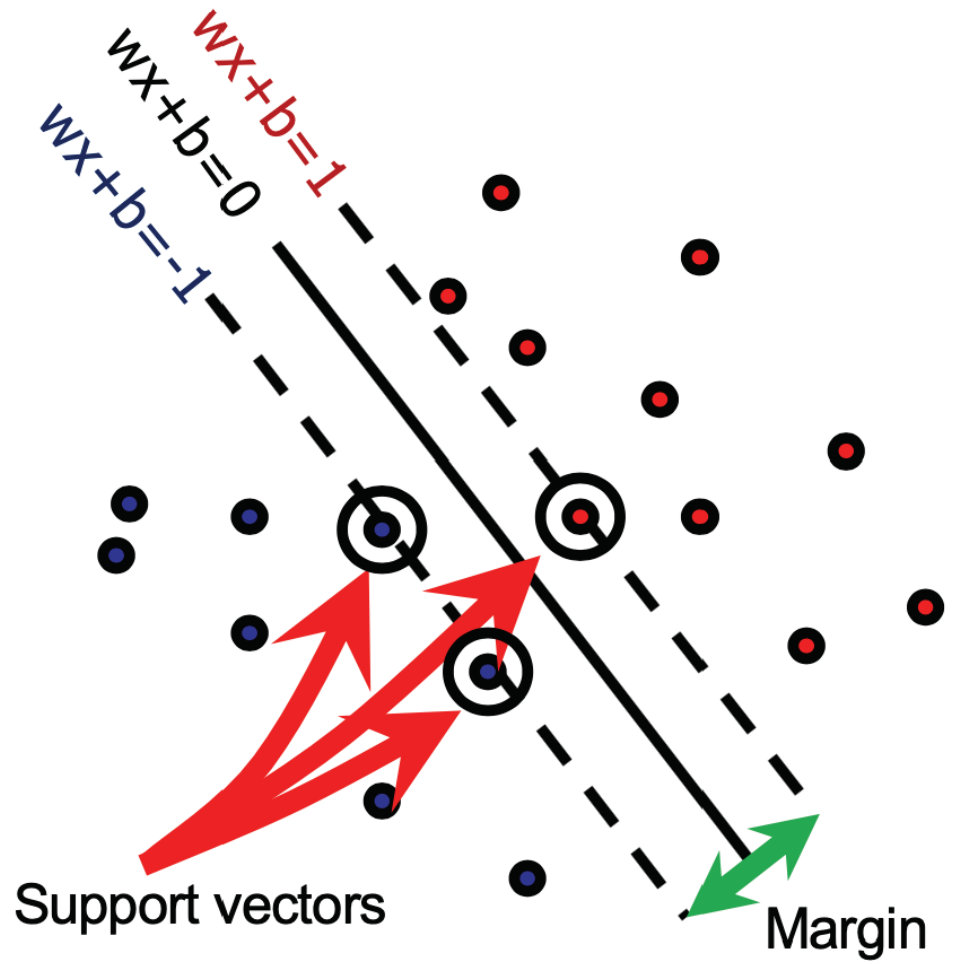
Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

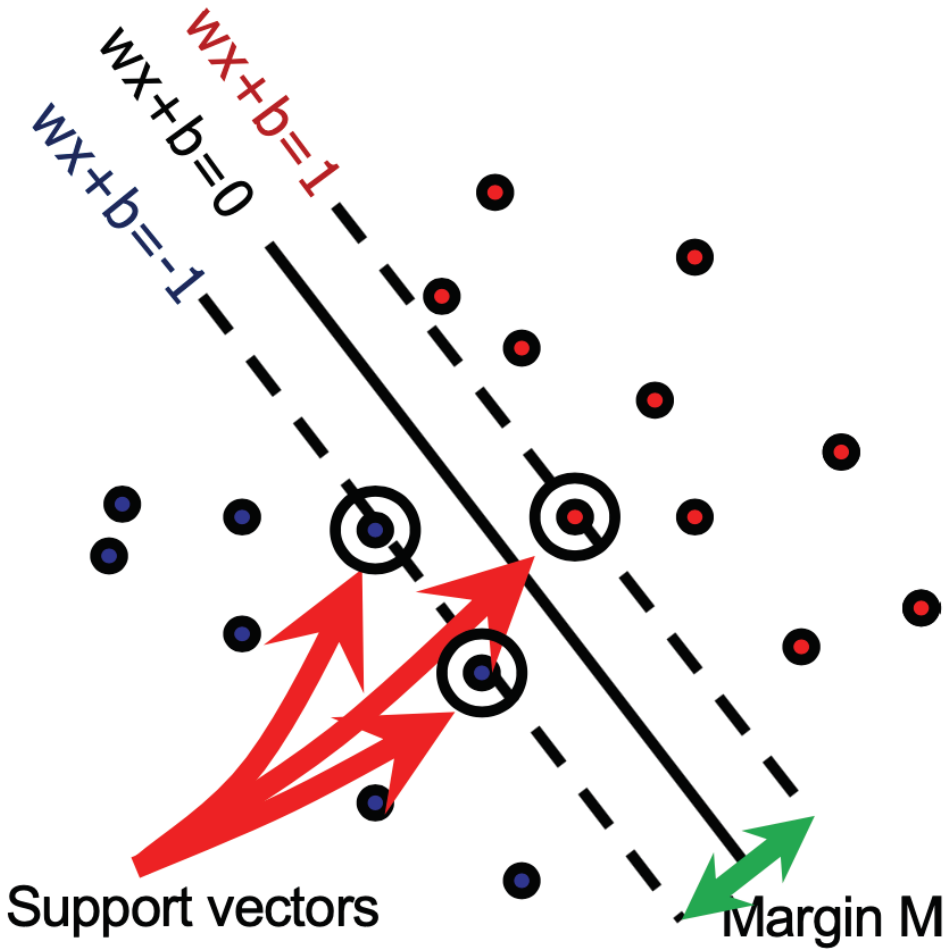
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

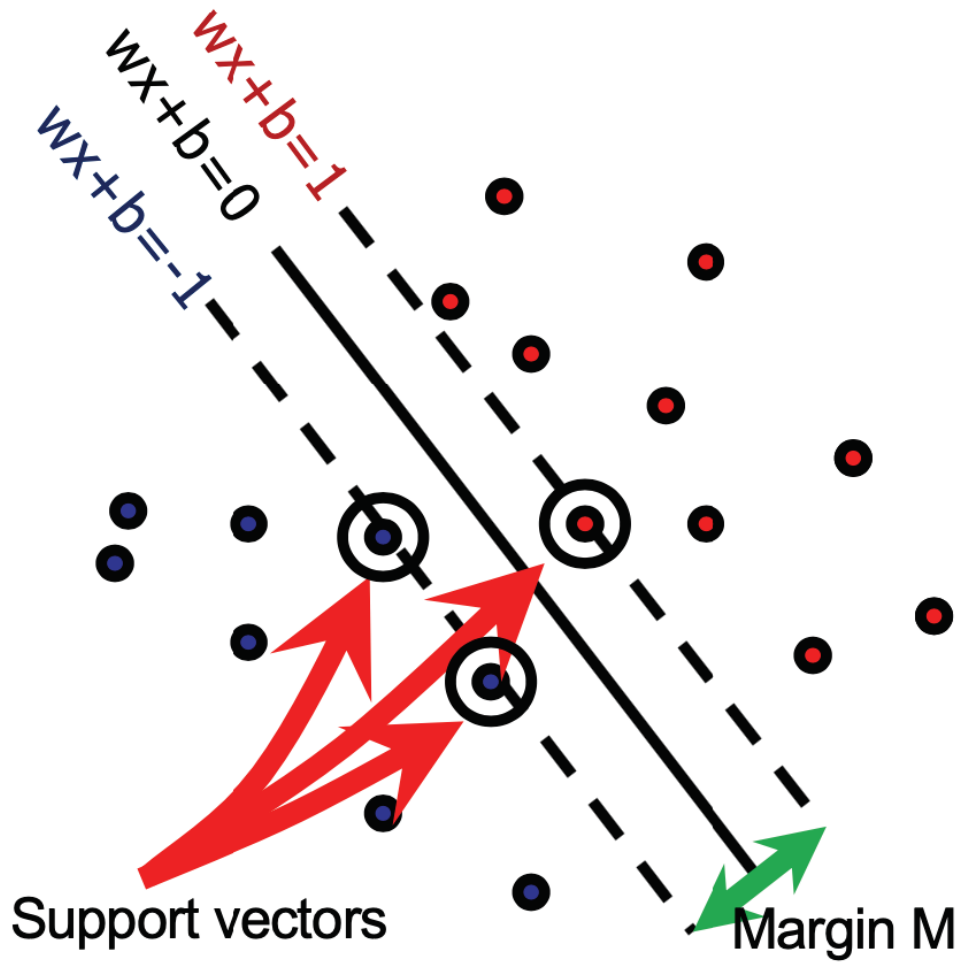
Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

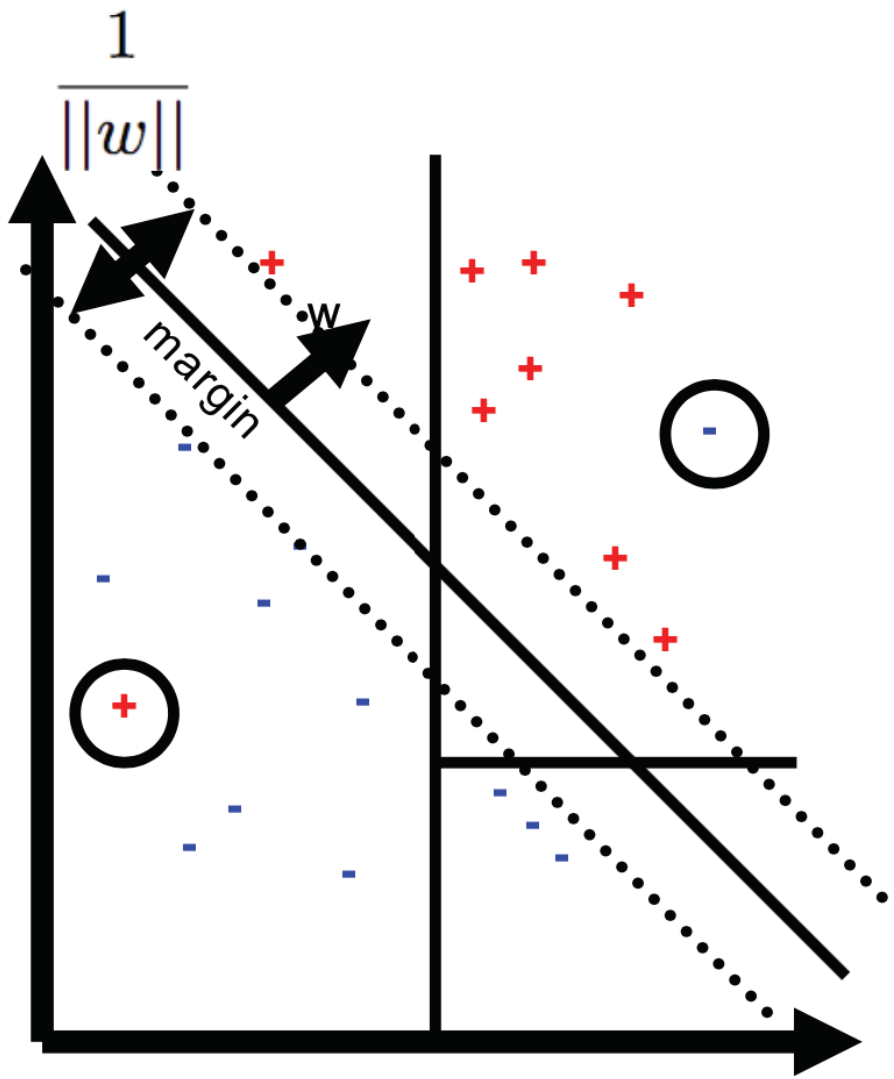
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

Support vector machines



- Simple decision
- Good classification
- Good generalization

$$\min_{w,\xi} \frac{||w||^2}{2} + C\sum_j \xi_j$$

$$y_j w^T x_j \geq 1 - \xi_j$$

$$\xi_j \geq 0$$

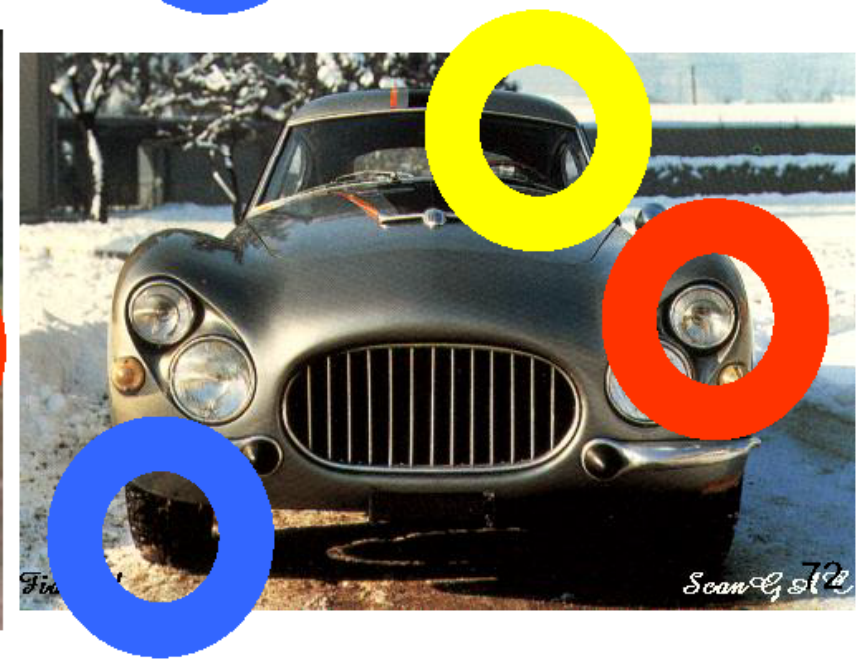
How do I solve the SVM problem?

- It's a convex optimization problem
  - Can solve in Matlab (don't)
- Download from the web
  - SMO: Sequential Minimal Optimization
  - SVM-Light http://svmlight.joachims.org/
  - LibSVM http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - LibLinear http://www.csie.ntu.edu.tw/~cjlin/liblinear/
  - SVM-Perf http://svmlight.joachims.org/
  - Pegasos http://ttic.uchicago.edu/~shai/

Bag of Words model for object detection

Adapted from slides by Rob Fergus

Different appearance, similar parts



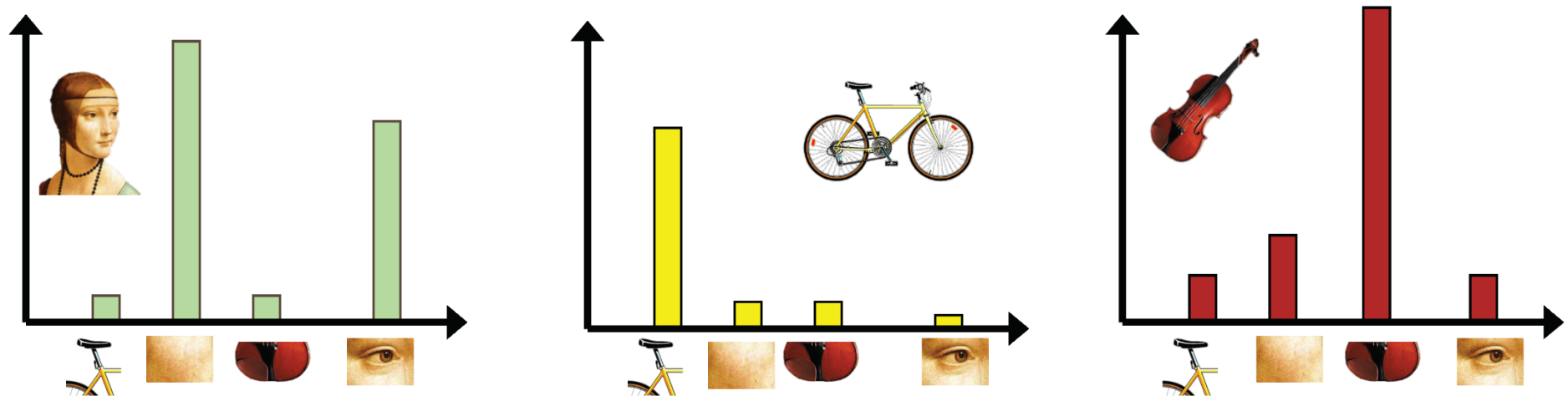
Object → **Bag of ‘words’**

73

Bag of Words

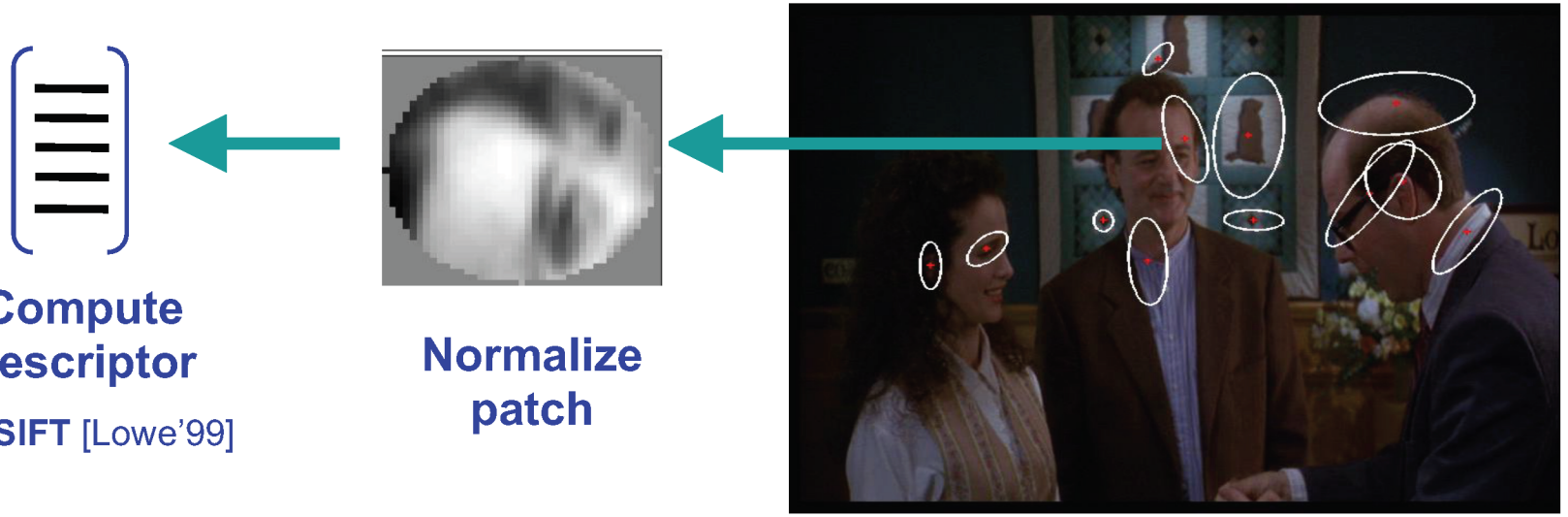
- Independent features

- Histogram representation



74

1.Feature detection and representation



Compute descriptor

e.g. **SIFT** [Lowe'99]

Normalize patch

Detect patches

[Mikojaczyk and Schmid '02]

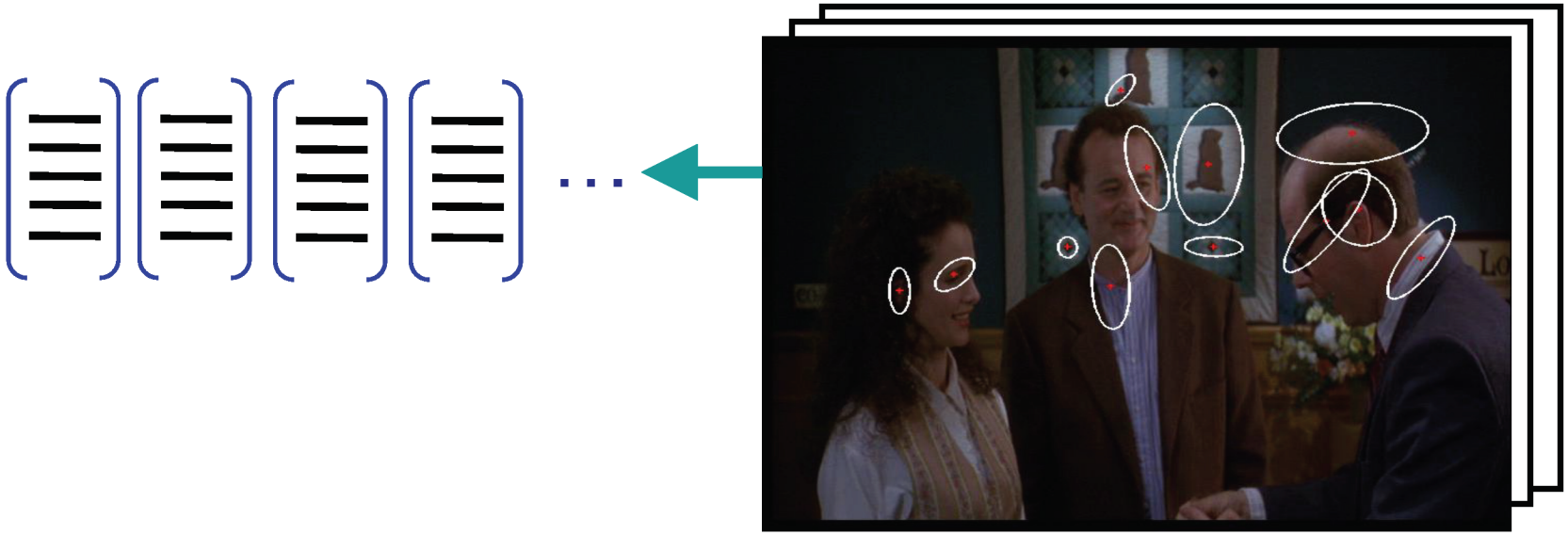
[Mata, Chum, Urban & Pajdla, '02]

[Sivic & Zisserman, '03]

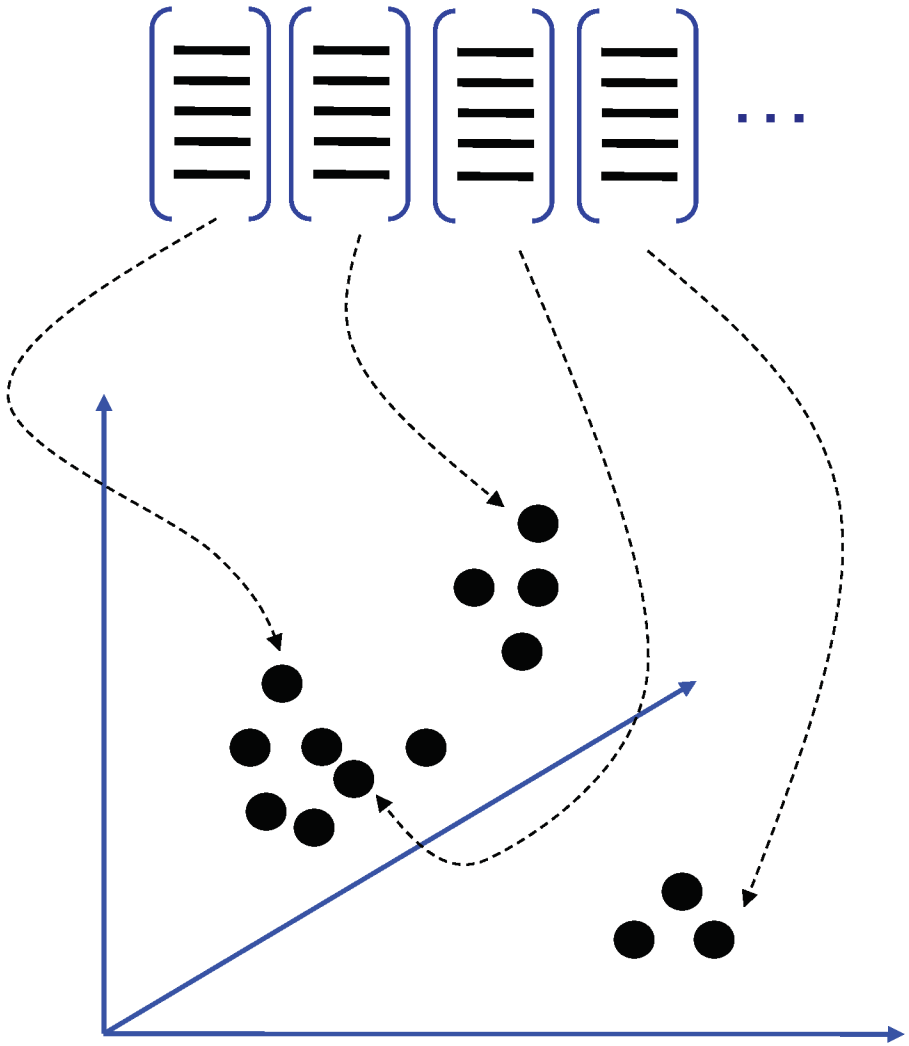
Local interest operator
or
Regular grid

Slide credit: Josef Sivic

1.Feature detection and representation

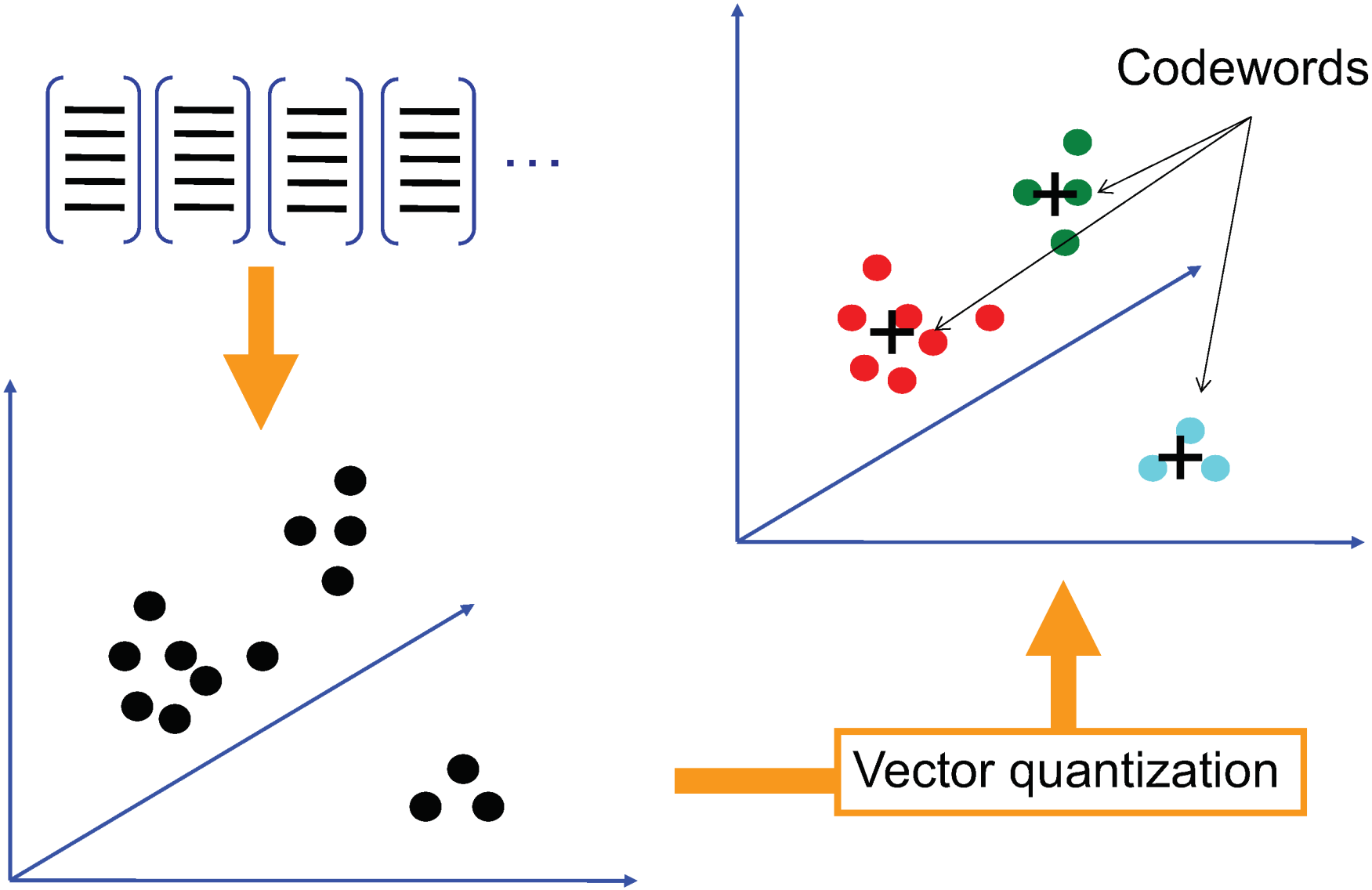


2. Codewords dictionary formation



77

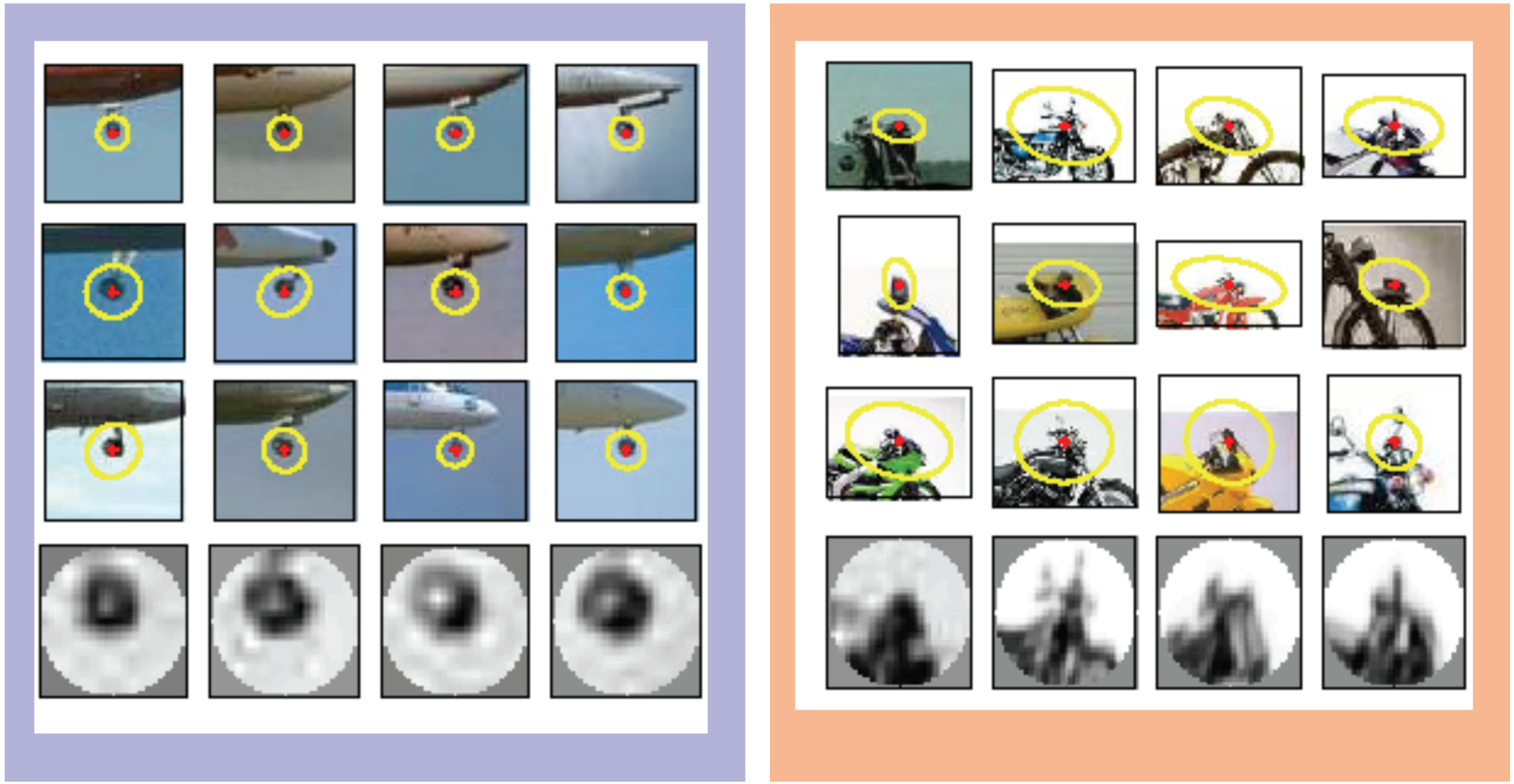
2. Codewords dictionary formation



78

Slide credit: Josef Sivic

Image patch examples of codewords

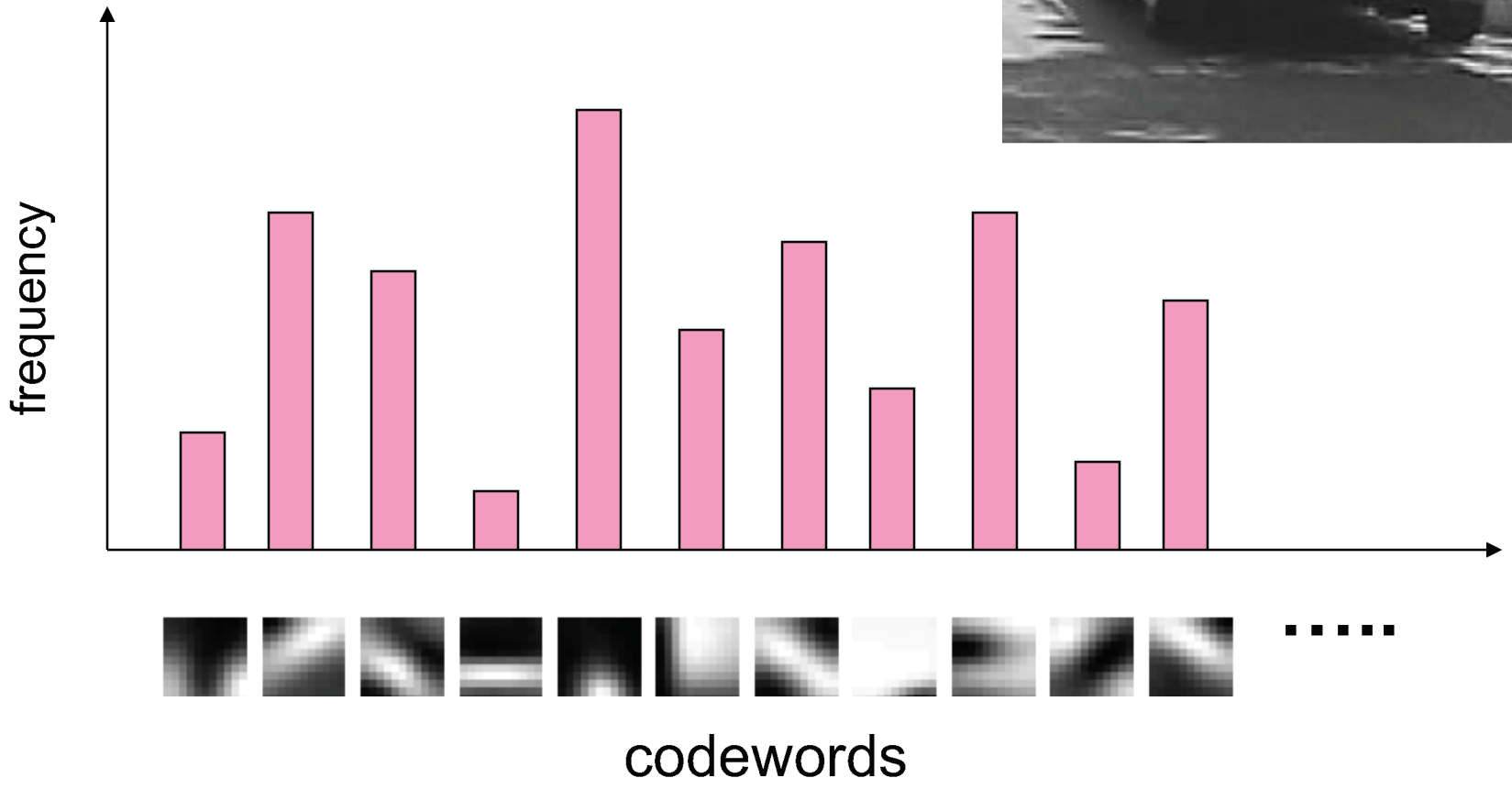


79

Sivic et al. 2005

Image representation

Histogram of features
assigned to each cluster



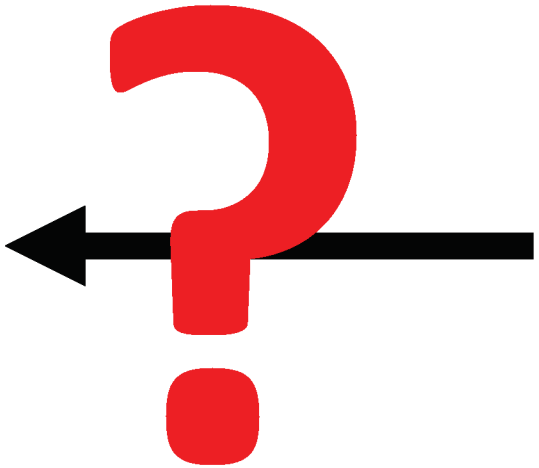
80

Uses of BoW representation

- Treat as feature vector for standard classifier
  - e.g k-nearest neighbors, support vector machine

- Cluster BoW vectors over image collection
  - Discover visual themes

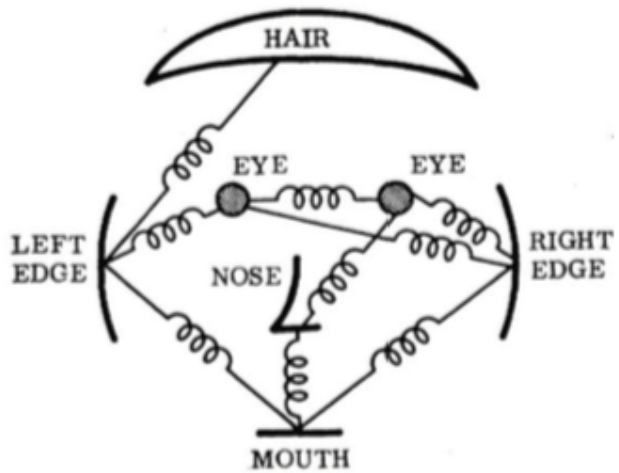
What about spatial info?



82

Deformable Part-based Models

- Felzenszwalb et al. PAMI'10
- Winner of the PASCAL detection challenge (2008,2009)
- Objects are decomposed into parts and spatial relations among parts



Fischler and Elschlager '73

Deformable Part-based Models

- Score of hypothesis

$$\text{score}(p_0, \ldots, p_n) = \underbrace{\sum_{i=0}^{n} F_i \cdot \phi(H, p_i)}_{\text{“data term”}} - \underbrace{\sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)}_{\text{“spatial prior”}}$$

filters

displacements

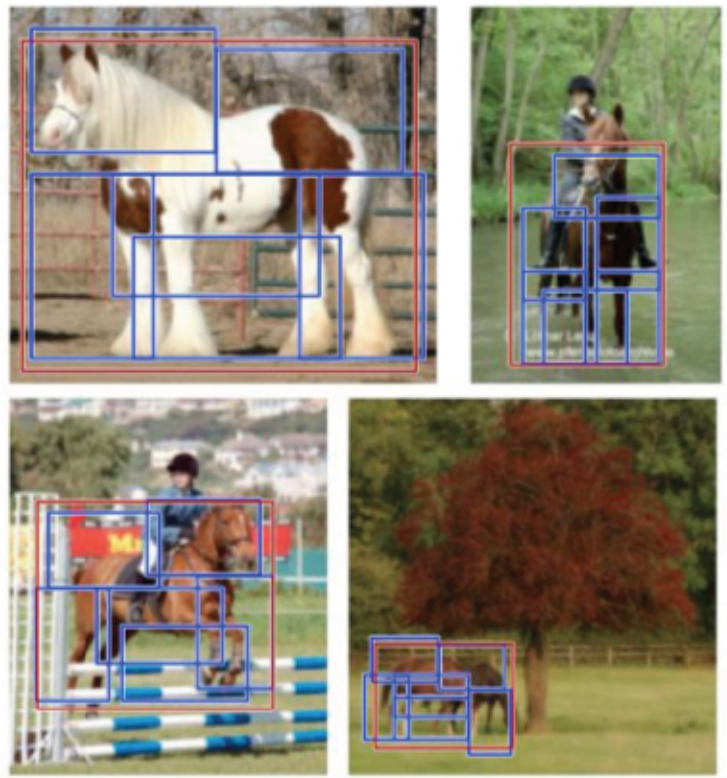
deformation parameters

Score is sum of filter scores minus deformation costs

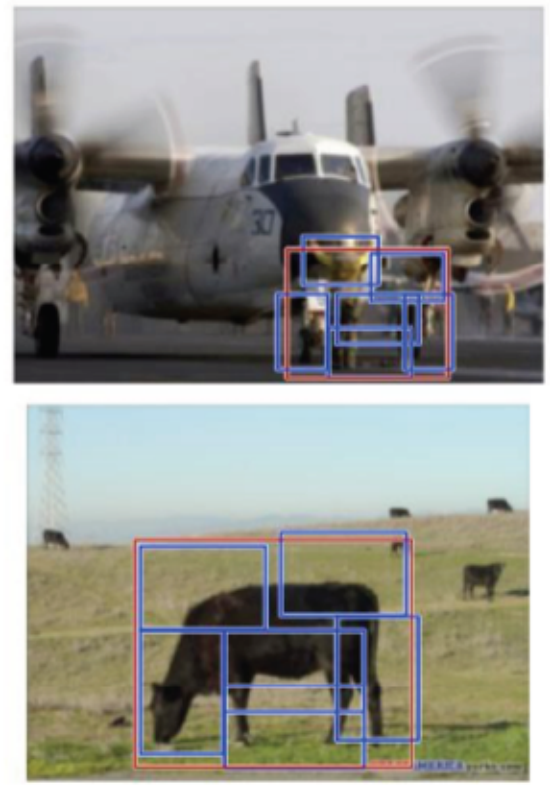
Deformable Part-based Models: Results

- Horse detections

high scoring true positives

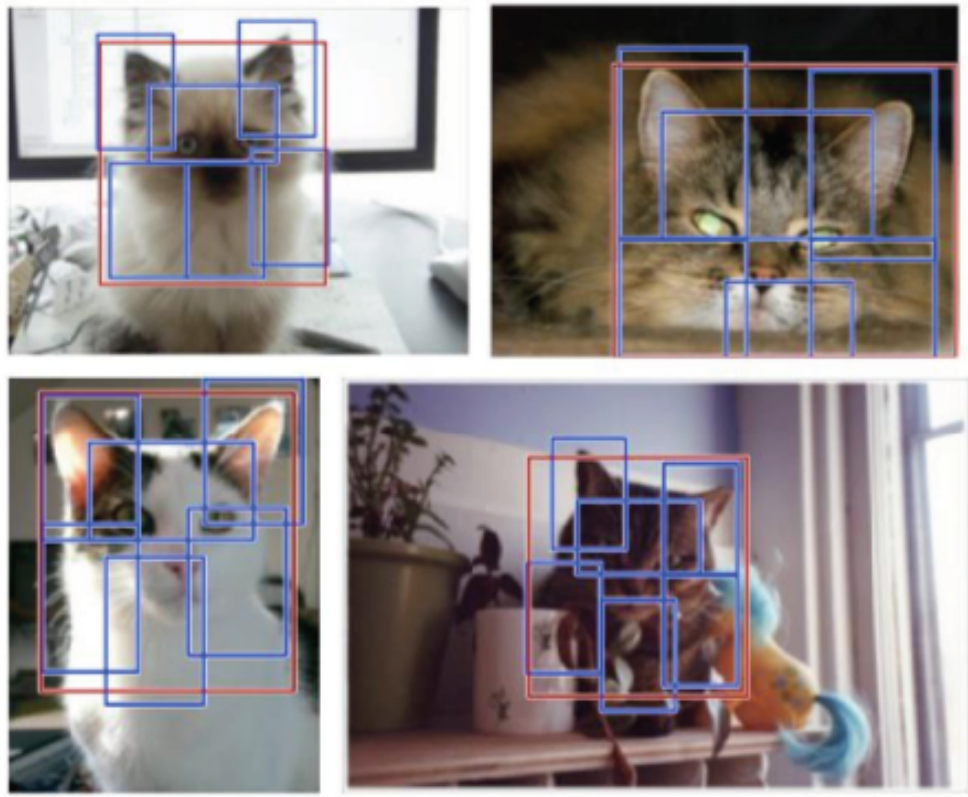


high scoring false positives

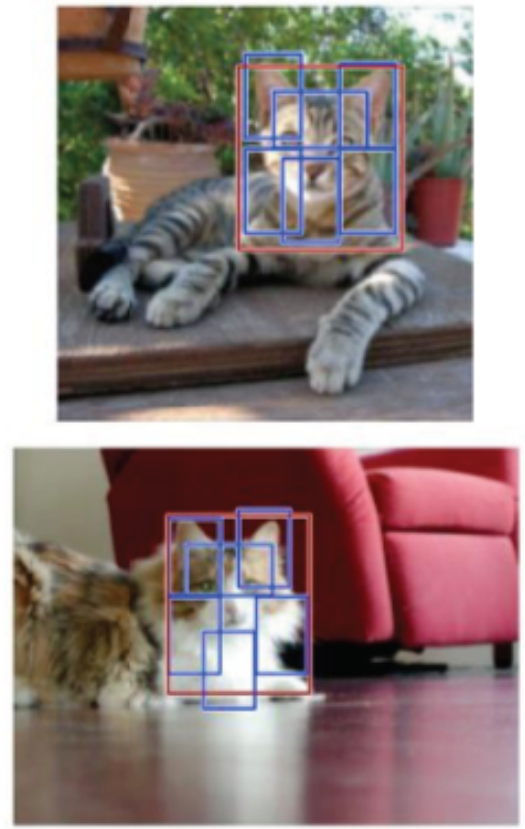


- Cat detections

high scoring true positives



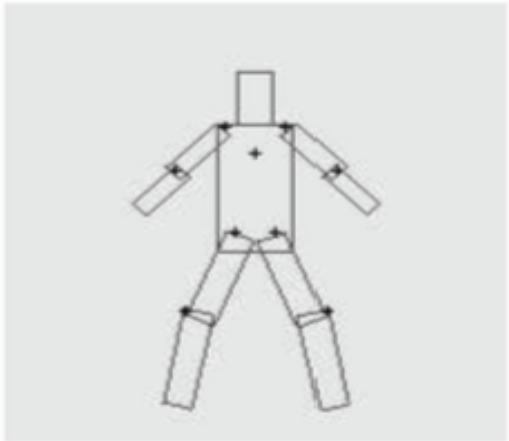
high scoring false positives
(not enough overlap)

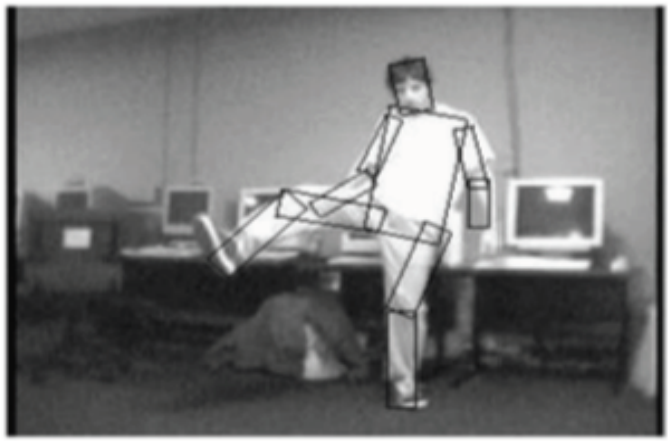


86

Part-based representation

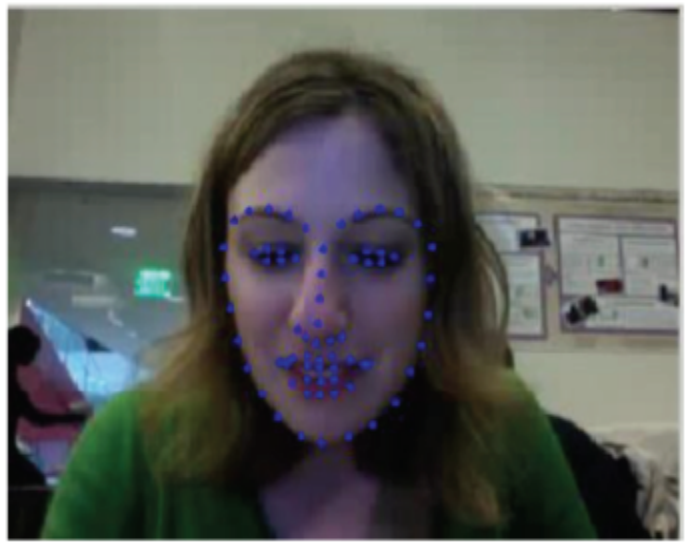
- Tree model ➔ Articulated objects





Part-based models: Pose estimation

- Pose estimation in video (Ramanan et al, 2007; Yang et al, 2015)

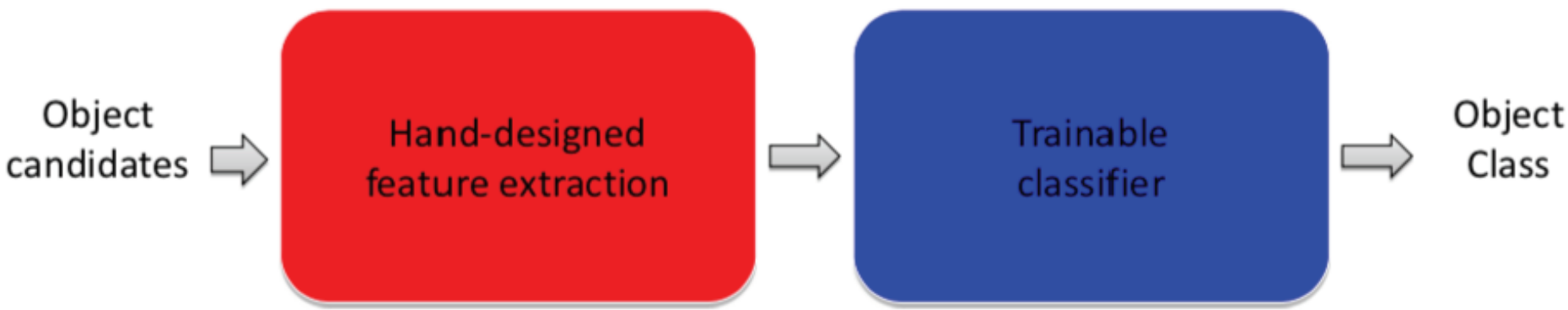


Face capturing

Dancing

DPMs: Limitations

- Manually designed feature (HOG)
- Pre-defined object-part structure
- Shallow structure: limited representation capacity

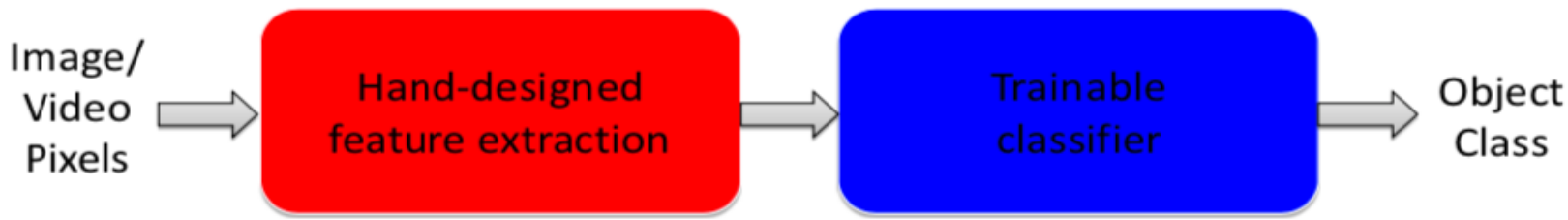


89

Move into “deep learning” regime

Next: “deep” architectures

Traditional recognition: “Shallow” architecture



Deep learning: “Deep” architecture

Readings

- M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991
- M. Turk and A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neuroscience, 1991
- P. Viola and M. Jones, Robust Real-time Object Detection, IJCV 2001
- N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
- P. Felzenszwalb, R. Girshick, D. McAllester and D. Ramanan, Object Detection with Discriminatively Trained Part Based Models, TPAMI 2010