Face detection

Bill Freeman, MIT

6.869 April 5, 2005

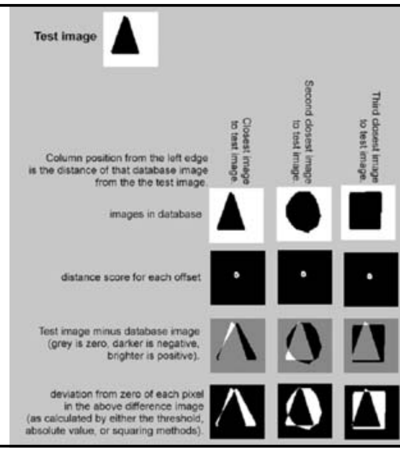
Today (April 5, 2005)

- Face detection
  - Subspace-based
  - Distribution-based
  - Neural-network based
  - Boosting based

Some slides courtesy of: Baback Moghaddam, Trevor Darrell, Paul Viola

Photos of class

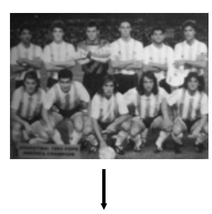
- What makes detection easy or hard?
- What makes recognition easy or hard?



E5 class, and recognition machine

Face Detection

- Goal: Identify and locate human faces in an image (usually gray scale) regardless of their position, scale, in plane rotation, orientation, pose and illumination
- The first step for any automatic face recognition system
- A very difficult problem!
- First aim to detect upright frontal faces with certain ability to detect faces with different pose, scale, and illumination
- One step towards Automatic Target Recognition or generic object recognition



Where are the faces, if any?

Why Face Detection is Difficult?

- **Pose**: Variation due to the relative camera-face pose (frontal, 45 degree, profile, upside down), and some facial features such as an eye or the nose may become partially or wholly occluded.
- **Presence or absence of structural components**: Facial features such as beards, mustaches, and glasses may or may not be present, and there is a great deal of variability amongst these components including shape, color, and size.
- **Facial expression**: The appearance of faces are directly affected by a person's facial expression.
- **Occlusion**: Faces may be partially occluded by other objects. In an image with a group of people, some faces may partially occlude other faces.
- **Image orientation**: Face images directly vary for different rotations about the camera's optical axis.
- **Imaging conditions**: When the image is formed, factors such as lighting (spectra, source distribution and intensity) and camera characteristics (sensor response, lenses) affect the appearance of a face.

Face detectors

- Subspace-based
- Distribution-based
- Neural network-based
- Boosting-based

Subspace Methods

- PCA ("Eigenfaces", Turk and Pentland)
- PCA (Bayesian, Moghaddam and Pentland)
- LDA/FLD ("Fisherfaces", Belhumeur & Kreigman)
- ICA

Principal Component Analysis

Joliffe (1986)

- data modeling & visualization tool
- discrete (partial) Karhunen-Loeve expansion
- dimensionality reduction tool $R^N \rightarrow R^M$
- makes no assumption about *p(x)*
- if *p(x)* is Gaussian, then $p(x) = \prod_i N(y_i; 0, \lambda_i)$

Eigenfaces (PCA)

Kirby & Sirovich (1990), Turk & Pentland (1991)

$\{x_i\}_{i=1}^{M} \quad x \in R^N \qquad M < N$

$$\mu = \frac{1}{M}\sum_{i=1}^{M} x_i$$

$$S = \sum_{i=1}^{M} (x_i - \mu)(x_i - \mu)^T$$

$$S = ULU^T$$

$$y = U^T(x - \mu)$$

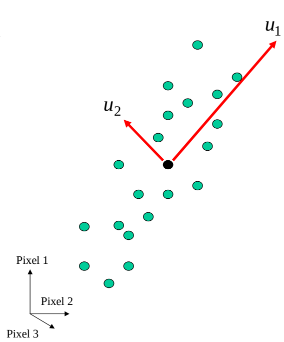


Figure 4: Standard Eigenfaces.

The benefit of eigenfaces over nearest neighbor

$$|\vec{y}_1-\vec{y}_2|^2=(\vec{y}_1-\vec{y}_2)^T(\vec{y}_1-\vec{y}_2)$$

image differences

eigenvalues

basis functions

$$=(U\vec{x}_1-U\vec{x}_2)^T(U\vec{x}_1-U\vec{x}_2)$$

$$=\left(\vec{x}_1^TU^T-\vec{x}_2^TU^T\right)\left(U\vec{x}_1-U\vec{x}_2\right)$$

$$=\vec{x}_1^T\vec{x}_1-\vec{x}_2^T\vec{x}_1-\vec{x}_1^T\vec{x}_2+\vec{x}_2^T\vec{x}_2$$

$$=\left(\vec{x}_1^T-\vec{x}_2^T\right)\left(\vec{x}_1-\vec{x}_2\right)$$

$$=|\vec{x}_1-\vec{x}_2|^2$$

eigenvalue differences

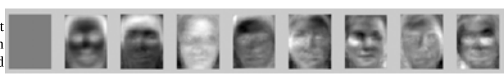
Matlab experiments

- Pca
- Spectrum of eigen faces
- eigenfaces
- Reconstruction
- Face detection
- Face recognition

Matlab example

- Effect of subtraction of the mean

Without mean subtracted



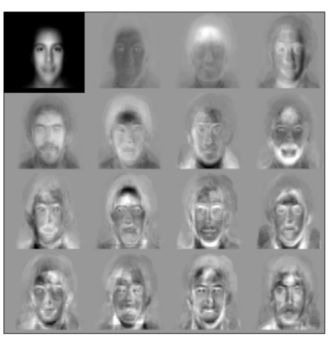
With mean subtracted

Eigenfaces

- Efficient ways to find nearest neighbors
- Can sometimes remove lighting effects
- What you really want to do is use a Bayesian approach…

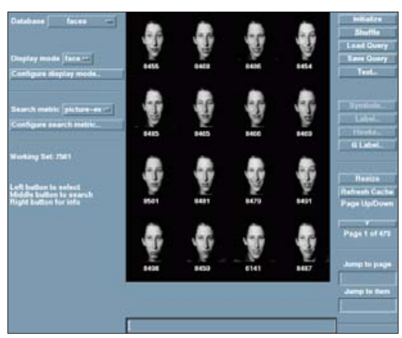
Eigenfaces

Turk & Pentland (1992)



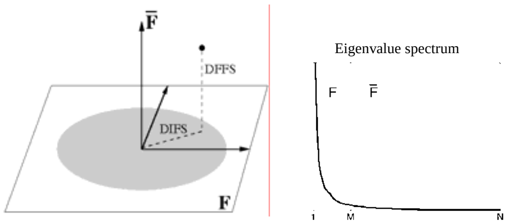
Eigenfaces

Photobook (MIT)



Subspace Face Detector

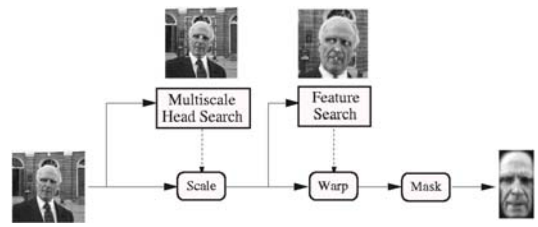
- PCA-based Density Estimation p(x)
- Maximum-likelihood face detection based on DIFS + DFFS



Moghaddam & Pentland, "Probabilistic Visual Learning for Object Detection," ICCV'95.
http://www-white.media.mit.edu/vismod/publications/techdir/TR-326.ps.Z

Subspace Face Detector

- Multiscale Face and Facial Feature Detection & Rectification



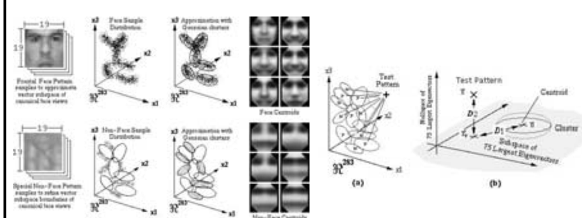
Moghaddam & Pentland, "Probabilistic Visual Learning for Object Detection," ICCV'95.

References

- Reading: Forsyth & Ponce: chapter 22.
- Slides from Baback Moghaddam are marked by reference to Moghaddam and Pentland.
- Slides from Rowley manuscript are marked by that reference.
- Slides from Viola and Jones are marked by reference to their CVPR 2001 paper.

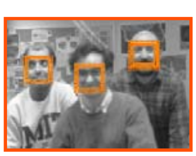
Distribution-Based Face Detector

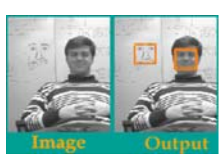
- Learn face and nonface models from examples [Sung and Poggio 95]
- Cluster and project the examples to a lower dimensional space using Gaussian distributions and PCA
- Detect faces using distance metric to face and nonface clusters



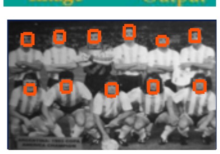
Distribution-Based Face Detector

- Learn face and nonface models from examples [Sung and Poggio 95]



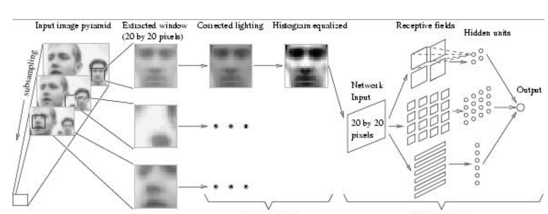


Training Database
1000+ Real, 3000+ *VIRTUAL*
50,0000+ Non-Face Pattern



Neural Network-Based Face Detector

- Train a set of multilayer perceptrons and arbitrate a decision among all outputs [Rowley et al. 98]



http://www.ius.cs.cmu.edu/demos/facedemo.html

CMU's Face Detector Demo

This is the front page for an interactive WWW demonstration of a face detector developed here at CMU. A detailed description of the system is available. The face detector can handle pictures of people (roughly) facing the camera in an (almost) vertical orientation. The faces can be anywhere inside the image, and range in size from at least 20 pixels hight to covering the whole image.

Since the system does not run in real time, this demonstration is organized as follows. First, you can submit an image to be processed by the system. Your image may be located anywhere on the WWW. After your image is processed, you will be informed via an e-mail message.

After your image is processed, you may view it in the gallery (gallery with inlined images). There, you can see your image, with green outlines around each location that the system thinks contains a face. You can also look at the results of the system on images supplied by other people.

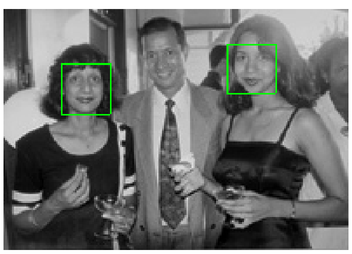
Henry A. Rowley (har@cs.cmu.edu)
Shumeet Baluja (baluja@cs.cmu.edu)
Takeo Kanade (tk@cs.cmu.edu)

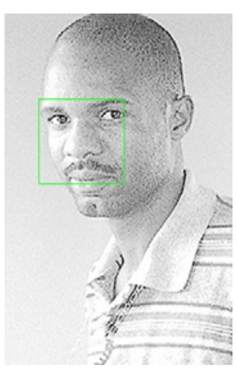
Example CMU face detector results

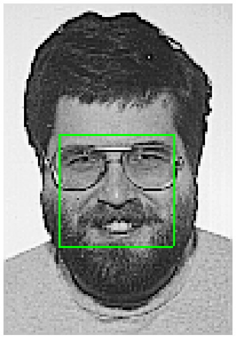
input

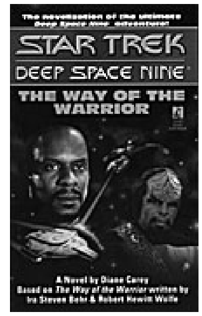
All images from: http://www.ius.cs.cmu.edu/demos/facedemo.html

output

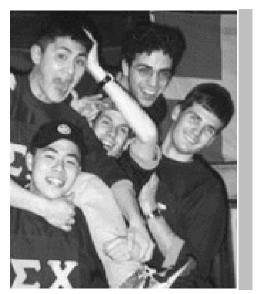


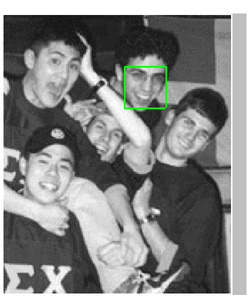


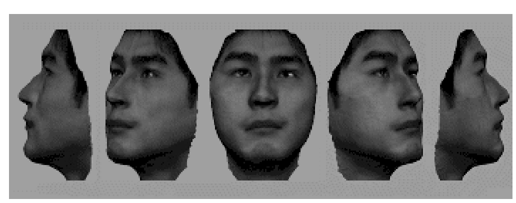


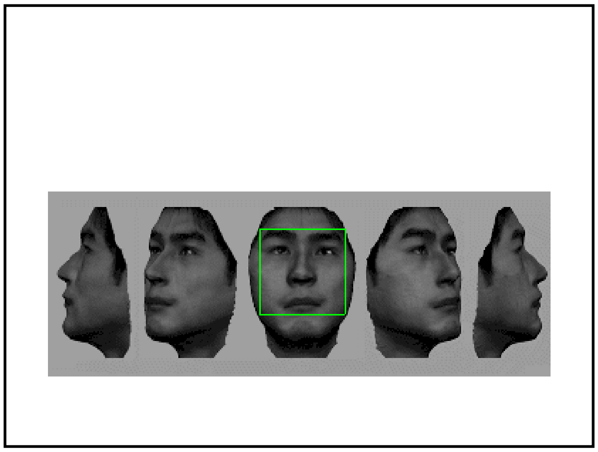

STAR TREK
DEEP SPACE NINE
THE WAY OF THE WARRIOR
A Novel by Diane Carey
Based on The Way of the Warrior written by
Ira Steven Behr & Robert Hewitt Wolfe

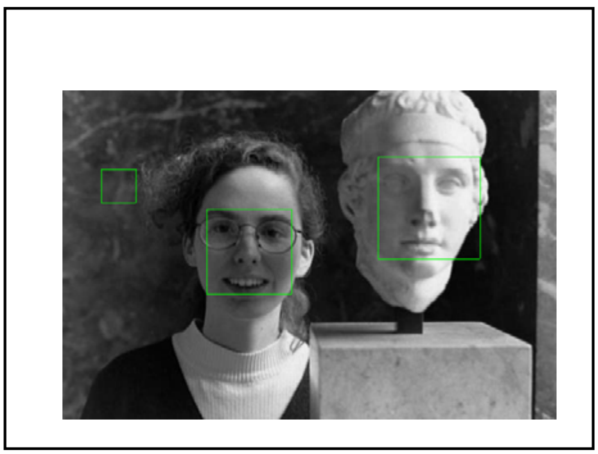

STAR TREK
DEEP SPACE NINE
THE WAY OF THE WARRIOR
A Novel by Diane Carey
Based on The Way of the Warrior written by
Ira Steven Behr & Robert Hewitt Wolfe

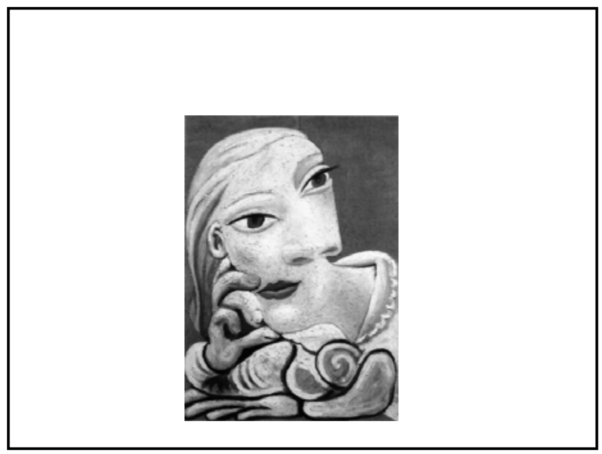


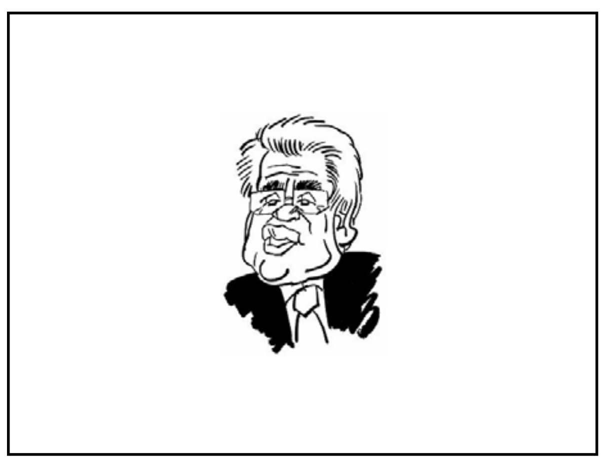


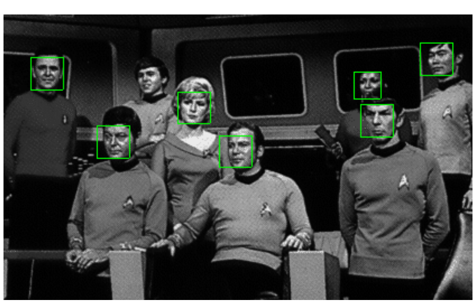




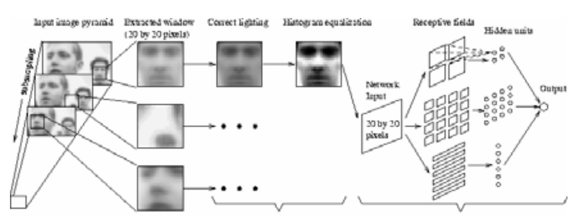




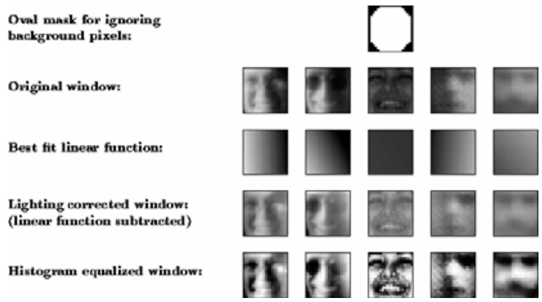




The basic algorithm used for face detection



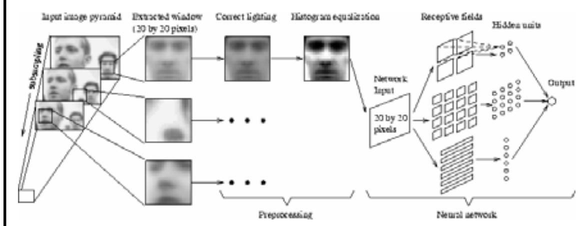
From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/



The steps in preprocessing a window. First, a linear function is fit to the intensity values in the window, and then subtracted out, correcting for some extreme lighting conditions. Then, histogram equalization is applied, to correct for different camera gains and to improve contrast. For each of these steps, the mapping is computed based on pixels inside the oval mask, while the mapping is applied to the entire window.

From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

The basic algorithm used for face detection



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Backprop Primer - 1

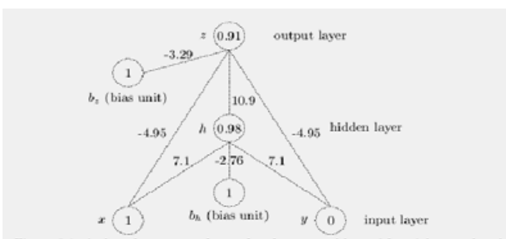


Figure 2.1: A three-layer network to solve the *xor* problem with weights produced by back-propagation.

Backprop Primer - 2

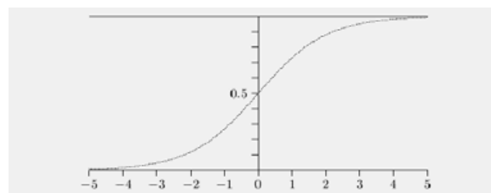


Figure 2.2: A plot of the most commonly used back-propagation activation function, $1/(1+e^{-x})$

Backprop Primer - 3

We will now look at the formulas for adjusting the weights that lead into the output units of a back-propagation network. The actual activation value of an output unit, k, will be o_k and the target for unit, k, will be t_k. First of all there is a term in the formula for δ_k, the *error signal* :

$$\delta_k = (t_k - o_k)f'(net_k) \quad (2.3)$$

where f′ is the derivative of the activation function, f. If we use the usual activation function:

$$\frac{1}{1+e^{-net_k}}$$

the derivative term is:

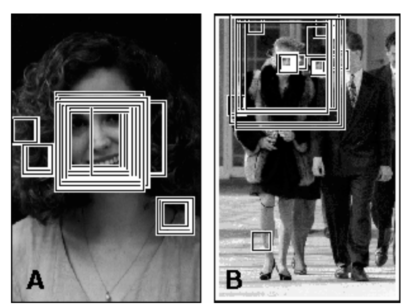
$$o_k(1-o_k) \quad (2.4)$$

The formula to change the weight, w_{jk} between the output unit, k, and unit j is:

$$w_{jk} \leftarrow w_{jk} + \eta\delta_k o_j \quad (2.5)$$

where η is some relatively small positive constant called the *learning rate* . With the network in 2.3 with η =

Images with all the above threshold detections indicated by boxes.

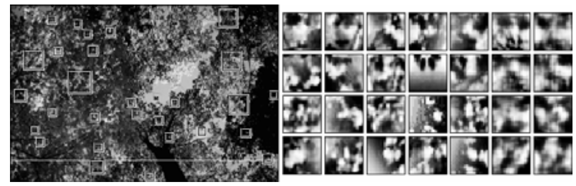


From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

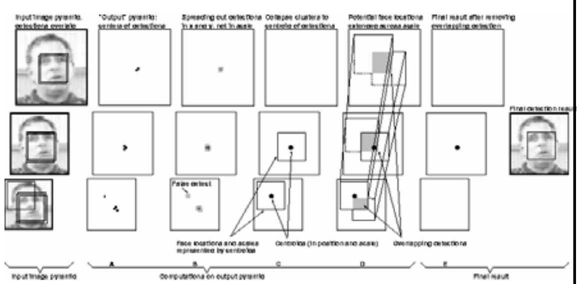
Example face images, randomly mirrored, rotated, translated, and scaled by small amounts (photos are of the three authors).

From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

During training, the partially-trained system is applied to images of scenery which do not contain faces (like the one on the left). Any regions in the image detected as faces (which are expanded and shown on the right) are errors, which can be added into the set of negative training examples.

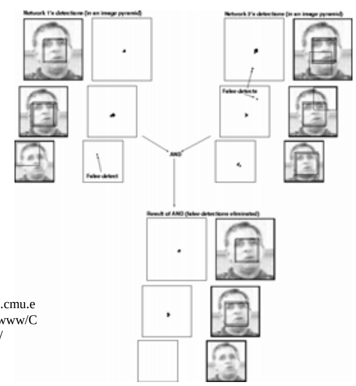


From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/



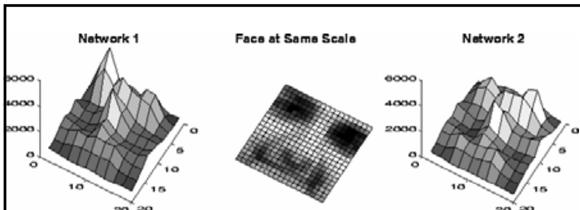
The framework used for merging multiple detections from a single network: A) The detections are recorded in an image pyramid. B) The detections are ``spread out'' and a threshold is applied. C) The centroids in scale and position are computed, and the regions contributing to each centroid are collapsed to single points. In the example shown, this leaves only two detections in the output pyramid. D) The final step is to check the proposed face locations for overlaps, and E) to remove overlapping detections if they exist. In this example, removing the overlapping detection eliminates what would otherwise be a false positive.

From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

ANDing together the outputs from two networks over different positions and scales can improve detection accuracy.



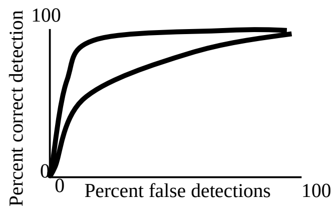
Error rates (vertical axis) on a small test resulting from adding noise to various portions of the input image (horizontal plane), for two networks. Network 1 has two copies of the hidden units shown in Figure 1 (a total of 58 hidden units and 2905 connections), while Network 2 has three copies (a total of 78 hidden units and 4357 connections).

The networks rely most heavily on the eyes, then on the nose, and then on the mouth (Figure 9). Anecdotally, we have seen this behavior on several real test images. Even in cases in which only one eye is visible, detection of a face is possible, though less reliable, than when the entire face is visible. The system is less sensitive to the occlusion of features such as the nose or mouth.

From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

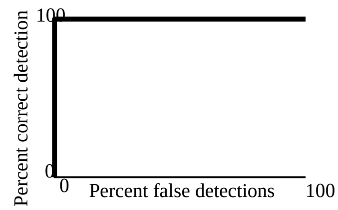
ROC (receiver operating characteristic) curve

ROC (receiver operating characteristic) curve

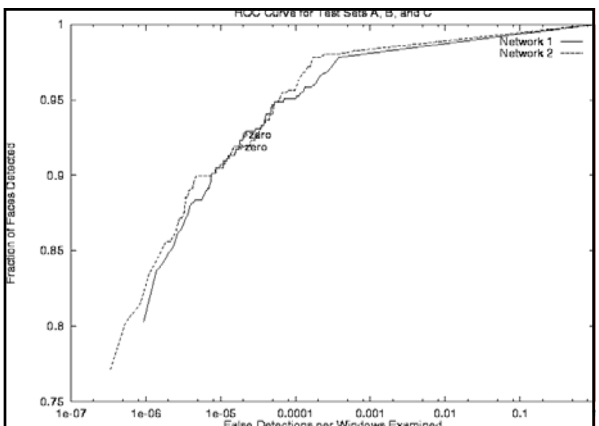


Realistic examples

ROC (receiver operating characteristic) curve



ideal



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Today (April 5, 2005)

- Face detection
  - Subspace-based
  - Distribution-based
  - Neural-network based
  - Boosting based

Some slides courtesy of: Baback Moghaddam, Trevor Darrell, Paul Viola

Rapid Object Detection Using a Boosted Cascade of Simple Features

Paul Viola Michael J. Jones
Mitsubishi Electric Research Laboratories (MERL)
Cambridge, MA

Most of this work was done at Compaq CRL before the authors moved to MERL

Face Detection Example

Many Uses
- User Interfaces
- Interactive Agents
- Security Systems
- Video Compression
- Image Database Analysis

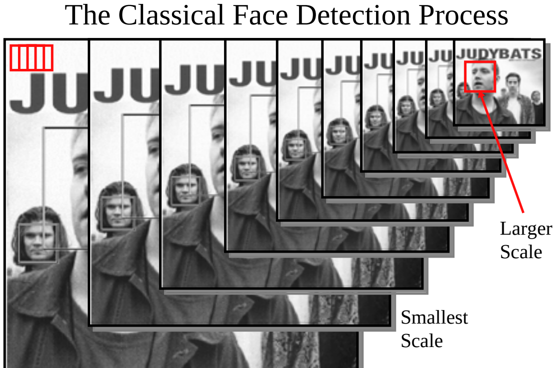
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Related Work

- Face detectors:
  - Sung and Poggio '98 (MIT)
  - Rowley, Baluja and Kanade '98 (CMU)
  - Schneiderman and Kanade '00 (CMU)
  - Many others: Cal Tech, UIUC, MIT Media Lab
- Feature-based approach to detection
  - Papageorgiou and Poggio '98 (MIT)
- AdaBoost for feature selection
  - Tieu and Viola '00 (MIT)
- Hierarchy of classifiers
  - Romdhani, Torr, Scholkopf, Blake '01 (Microsoft)

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

The Classical Face Detection Process



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Classifier is Learned from Labeled Data

- Training Data
  - 5000 faces
    - All frontal
  - 10^8 non faces
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose (rotation both in plane and out)

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

What is novel about this approach?

- Feature set (… is huge about 16,000,000 features)
- Efficient feature selection using AdaBoost
- New image representation: Integral Image
- Cascaded Classifier for rapid detection
  - Hierarchy of Attentional Filters

The combination of these ideas yields the fastest known face detector for gray scale images.

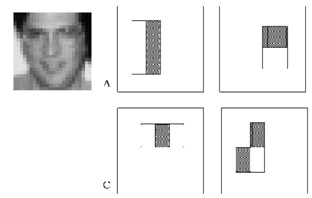
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Image Features

"Rectangle filters"

Similar to Haar wavelets

Differences between sums of pixels in adjacent rectangles



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

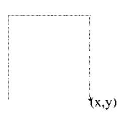
$160{,}000 \times 100 = 16{,}000{,}000$
Unique Features

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Integral Image

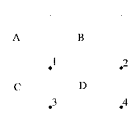
- Define the Integral Image

$$I'(x,y) = \sum_{\substack{x' \le x \\ y' \le y}} I(x',y')$$



- Any rectangular sum can be computed in constant time:

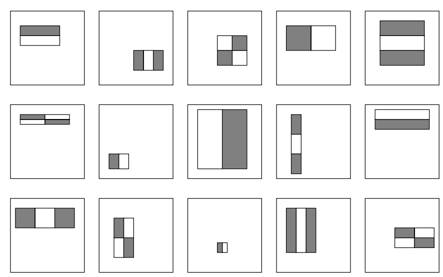
$$\begin{aligned} D &= 1 + 4 - (2 + 3) \\ &= A + (A + B + C + D) - (A + C + A + B) \\ &= D \end{aligned}$$



- Rectangle features can be computed as differences between rectangles

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Huge "Library" of Filters



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Constructing Classifiers

- Perceptron yields a sufficiently powerful classifier

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

- Use AdaBoost to efficiently choose best features

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

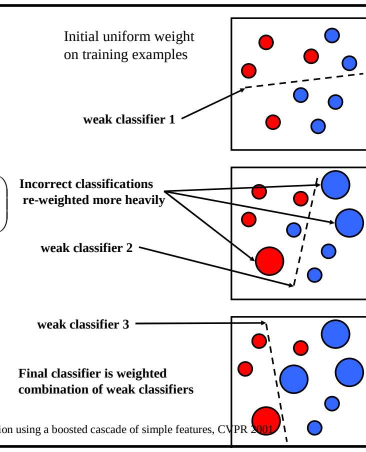
AdaBoost

(Freund & Shapire '95)

$$f(x) = \theta\left(\sum_t \alpha_t h_t(x)\right)$$

$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t}\right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Beautiful AdaBoost Properties

- Training Error approaches 0 exponentially
- Bounds on Testing Error Exist
  - Analysis is based on the Margin of the Training Set
- Weights are related the margin of the example
  - Examples with negative margin have large weight
  - Examples with positive margin have small weights

$$f(x) = \sum_i \alpha_i h_i(x) \qquad \min \sum_i e^{-y_i f(x_i)} \geq \sum_i \left(1 - y_i C(x_i)\right)$$

$$C(x) = \theta\left(f(x)\right)$$

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Ada-Boost Tutorial

- Given a Weak learning algorithm
  - Learner takes a training set and returns the best classifier from a weak concept space
    - required to have error < 50%
- Starting with a Training Set (initial weights 1/n)
  - Weak learning algorithm returns a classifier
  - Reweight the examples
    - Weight on correct examples is decreased
    - Weight on errors is decreased

$$\sum_{i \in Errors} w_i = \sum_{j \in Correct} w_j$$

- Final classifier is a weighted majority of Weak Classifiers
  - Weak classifiers with low error get larger weight

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Review of AdaBoost (Freund & Shapire 95)

- Given examples $(x_1, y_1), \ldots, (x_N, y_N)$ where $y_i = 0,1$ for negative and positive examples respectively.
- Initialize weights $w_{t=1,i} = 1/N$
- For $t=1, \ldots, T$
  - Normalize the weights, $w_{t,i} = w_{t,i} / \sum_{j=1}^{N} w_{t,j}$
  - Find a weak learner, i.e. a hypothesis, $h_t(x)$ with weighted error less than .5
  - Calculate the error of h_t : $e_t = \sum w_{t,i} \mid h_t(x_i) - y_i \mid$
  - Update the weights: $w_{t,i} = w_{t,i} B_t^{(1-d_i)}$ where $B_t = e_t / (1- e_t)$ and $d_i = 0$ if example x_i is classified correctly, $d_i = 1$ otherwise.
- The final strong classifier is

$$h(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x) > 0.5 \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log(1/ B_t)$

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

adaBoost live demo

AdaBoost for Efficient Feature Selection

- Our Features = Weak Classifiers
- 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)
    - Sorted list can be quickly scanned for the optimal threshold
  - Select best filter/threshold combination
  - Weight on this feature is a simple function of error rate
  - Reweight examples
  - (There are many tricks to make this more efficient.)

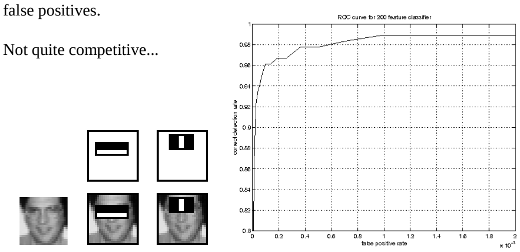
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Example Classifier for Face Detection

A classifier with 200 rectangle features was learned using AdaBoost

95% correct detection on test set with 1 in 14084 false positives.

Not quite competitive...

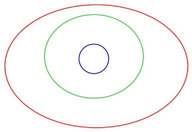


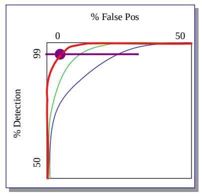
ROC curve for 200 feature classifier

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

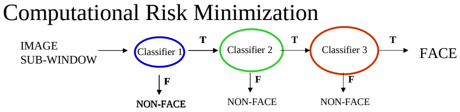
Trading Speed for Accuracy

- Given a nested set of classifier hypothesis classes



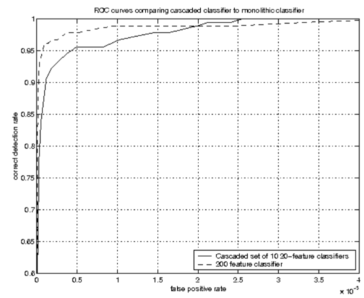


- Computational Risk Minimization



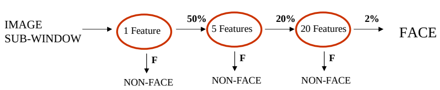
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Experiment: Simple Cascaded Classifier



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Cascaded Classifier



- A 1 feature classifier achieves 100% detection rate and about 50% false positive rate.
- A 5 feature classifier achieves 100% detection rate and 40% false positive rate (20% cumulative)
  - using data from previous stage.
- A 20 feature classifier achieve 100% detection rate with 10% false positive rate (2% cumulative)

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

A Real-time Face Detection System

Training faces: 4916 face images (24 x 24 pixels) plus vertical flips for a total of 9832 faces

Training non-faces: 350 million sub-windows from 9500 non-face images

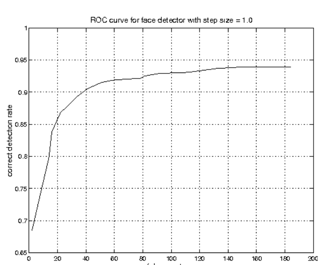
Final detector: 38 layer cascaded classifier
The number of features per layer was 1, 10, 25, 25, 50, 50, 50, 75, 100, …, 200, …

Final classifier contains 6061 features.

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Accuracy of Face Detector

Performance on MIT+CMU test set containing 130 images with 507 faces and about 75 million sub-windows.



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Comparison to Other Systems

Detector \ False Detections	10	31	50	65	78	95	110	167
Viola-Jones	76.1	88.4	91.4	92.0	92.1	92.9	93.1	93.9
Viola-Jones (voting)	81.1	89.7	92.1	93.1	93.1	93.2	93.7	93.7
Rowley-Baluja-Kanade	83.2	86.0				89.2		90.1
Schneiderman-Kanade				94.4				

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Speed of Face Detector

Speed is proportional to the average number of features computed per sub-window.

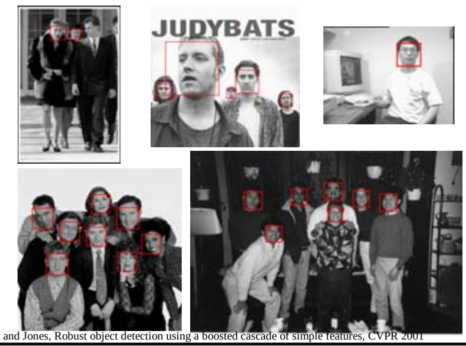
On the MIT+CMU test set, an average of 9 features out of a total of 6061 are computed per sub-window.

On a 700 Mhz Pentium III, a 384x288 pixel image takes about 0.067 seconds to process (15 fps).

Roughly 15 times faster than Rowley-Baluja-Kanade and 600 times faster than Schneiderman-Kanade.

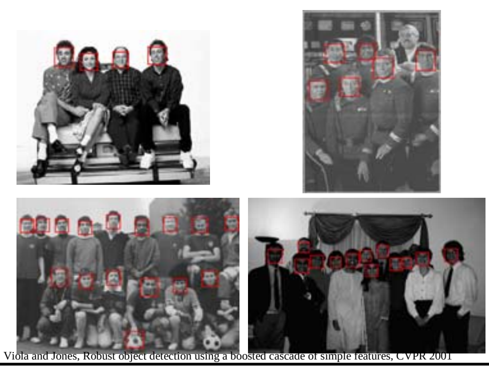
Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Output of Face Detector on Test Images



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

More Examples



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Video Demo

Conclusions

- We [they] have developed the fastest known face detector for gray scale images
- 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

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

end