Object Recognition

Chapter 7: Window-Based Object Detection

Prof. Dr. Johannes Maucher

HdM CSM

Version 1.1
12.05.2020

Document History

Version Nr.	Date	Changes
1.0	04.03.2013	Initial Version
1.1	12.05.2020	Adaptations for SS 20

Chapter 7: Window-Based Object Detection

Viola-Jones Face Detection

- Before face recognition can be applied to a general image, the locations and sizes of any faces must first be found
- Frontal Faces well suited for window-based representation, due to the high regularity in the 2D texture.
- Viola-Jones face detection was introduced in [2].
- *Of all the face detectors currently in use, the one introduced by Viola and Jones is probably the best known and most widely used* [1].
- Detection runs in real time
- Training is quite expensive

Face Detection Example

Key Elements of Vioala-Jones Face Detection

- Window-Based Representation: For each subwindow of proposed size (24×24), the classifier detects either *face* or *non-face*.
- Features are the responses of rectangular filters. For each subwindow an extremely large set of features are calculated.[1]
- Before the calculation of the features the grey-scale input image is converted into an integral image. The integral image allows a very efficient computation of the features.
- The Adaboost classifier is applied.
  - This classifier is a weighted combination of weak classifiers.
  - Each weak classifier depends on a single feature.
  - During the supervised training, the weights of the weak classifiers is learned. Each weight describes the relevance of the corresponding feature.
- Cascading Classifiers to achieve increased detection performance.

[1] In general in order to achieve good classification performance one should have a moderate set of well selected informative features or a huge set of simple features. The last option is choosen by VJ-approach. It is feasible because they found a fast method to calculate these features

Features calculated from rectangular filters

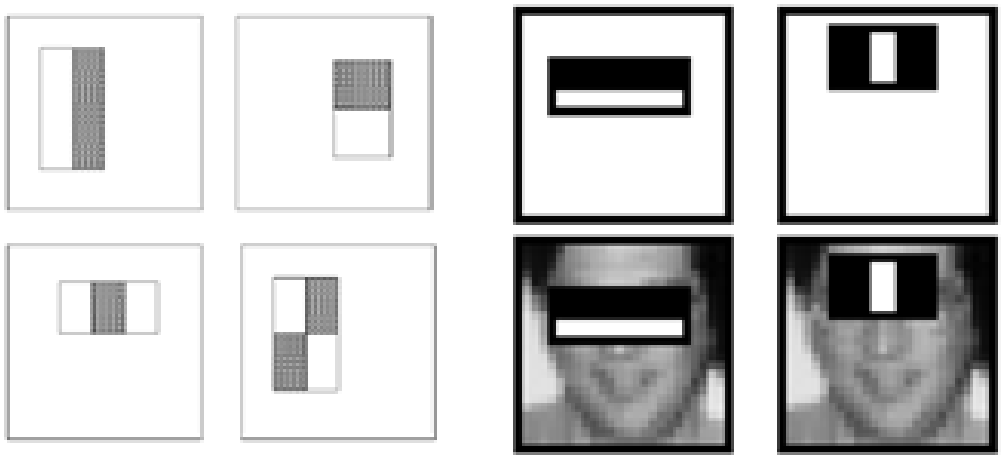
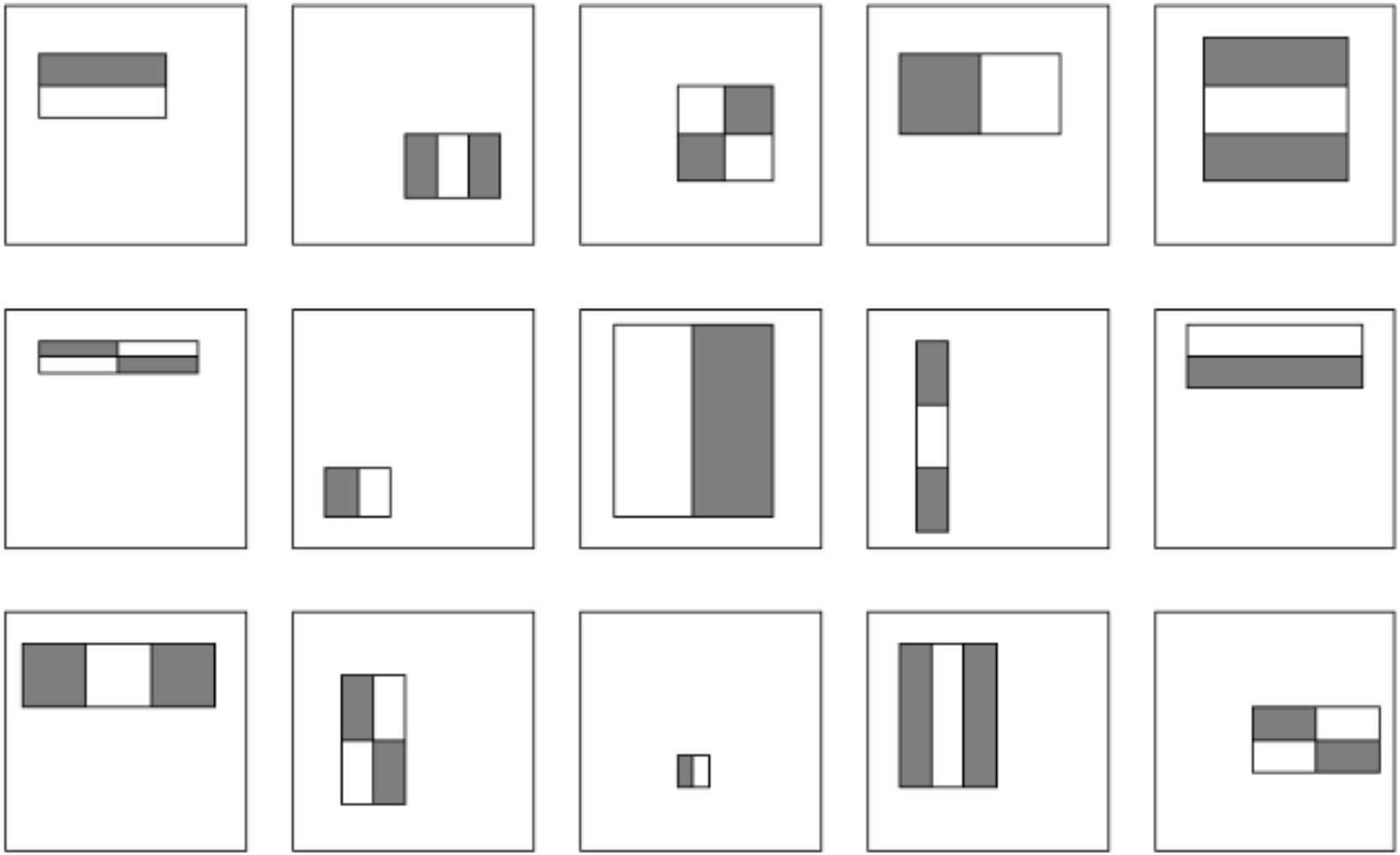


Abbildung: Rectangular filters applied to a sub-window

- Features are the responses of rectangular filters
- The response is calculated by summing the pixel values in the black region and subtracting the sum of the pixel values in the white region of the filter.
- The rectangular filters vary in
  - relative location in the sub-window
  - scale
  - type

Some more rectangular filters



Features calculated from rectangular filters

- In [2] a sub-window size of (24×24) pixels is proposed.
- In each sub-window more than 180000 rectangular filter responses (features) are calculated.
- Much more features than pixels?
- Unrealistic complexity?
- No! Because,
  - Efficient filter response calculation based on integral images
  - For classification only a few most relevant features are applied. Most relevant features are determined by Adaboost supervised learning.

Integral Image by Example

3	2	7	2	3
1	5	1	3	4
5	1	**3**	5	1
4	3	2	1	6
2	4	1	4	8

(a) S = 24

3	5	12	14	17
4	*11*	**19**	24	31
9	**17**	28	38	46
13	24	37	48	62
15	30	44	59	81

(b) s = 28

3	5	12	*14*	17
4	11	19	24	31
9	17	28	38	46
13	24	37	**48**	62
15	30	44	59	81

(c) S = 24

Figure 3.17 Summed area tables: (a) original image; (b) summed area table; (c) computation of area sum. Each value in the summed area table $s(i, j)$ (red) is computed recursively from its three adjacent (blue) neighbors (3.31). Area sums S (green) are computed by combining the four values at the rectangle corners (purple) (3.32). Positive values are shown in **bold** and negative values in *italics*.

Source: [1]

Integral Image: Formal description

- Let $f(k, l)$ be the pixel intensity in row k, column l of a grey-scale image. $f(0, 0)$ is the pixel intensity at the upper left corner (origin).
- The value $s(i, j)$ at row i, column j of the integral image is the sum of all pixel intensities $f(k, l)$ in the rectangle between the origin $(0, 0)$ and the position (i, j):

$$s(i,j) = \sum_{k=0}^{i} \sum_{l=0}^{j} f(k,l)$$

- Starting from the origin this sum can be computed efficiently by

$$s(i,j) = s(i-1,j) + s(i,j-1) - s(i-1,j-1) + f(i,j)$$

- The sum of all pixel values inside the rectangular area $[i_0, j_0] \times [j_0, j_1]$ can be calculated from only 4 entries in the integral image:

$$S(i_0 \ldots i_1, j_0, \ldots, j_1) = s(i_1, j_1) - s(i_1, j_0 - 1) - s(i_0 - 1, j_1) + s(i_0 - 1, j_0 - 1)$$

- For calculating the responses of rectangular filters it is enough to calculate sums like this and add (black filter regions) and subtract them (white regions).

AdaBoost (Adaptive Boosting) in General

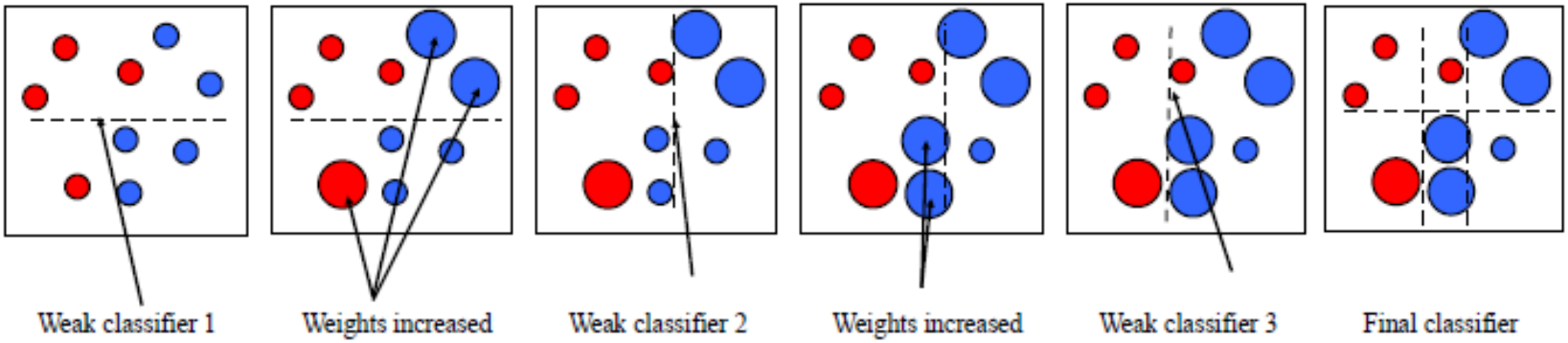


Abbildung: Source [1]

- Classifier is a weighted combination of weak classifiers
- Weak classifiers must only be better than chance
- The weight of a weak classifier depends on its error on training data - smaller error means higher weight.
- In each iteration the training samples are weighted. Samples that have been misclassified in the previous iterations get larger weights, i.e. succeesive classifiers priorize the correct classifications of these samples.

General Adaboost Training Algorithm

General Adaboost Training Algorithm

Let $T = \{\mathbf{x}_t, r_t\}_{t=1}^{N}$ be the set of N labeled training samples.

1. For each training sample $(\mathbf{x}_t, r_t)$ initialize the weight $w_t^{(1)} = \frac{1}{N}$. Define number of iterations (weak classifiers) to be M.
2. For $m = 1 \ldots M$
   1. Fit a weak classifier $h_m(\mathbf{x})$ to the training data by minimizing the weighted error function
   $$J_m = \sum_{t=1}^{N} w_t^{(m)} I(h_m(\mathbf{x}_t) \neq r_t),$$
   where $I(h_m(\mathbf{x}_t) \neq r_t)$ equals 1 if $h_m(\mathbf{x}_t) \neq r_t$ and 0 otherwise.
   2. Evaluate
   $$\alpha_m = \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right) \quad \text{where } \epsilon_m = \frac{J_m}{\sum_t w_t^{(m)}}$$
   3. For all training samples $(\mathbf{x}_t, r_t)$ update weights:
   $$w_t^{(m+1)} = w_t^{(m)} \exp\left(\alpha_m I(h_m(\mathbf{x}_t) \neq r_t)\right)$$

General Adaboost

- The trained weak classifiers $h_m(\mathbf{x})$ and the corresponding coefficients α_m constitute the final classifier

$$h(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m h_m(\mathbf{x}) \tag{1}$$

- For novel input data $\mathbf{x}$ the classification decision is $sgn\,(h(\mathbf{x}))$

Adaboost applied to Face Detection

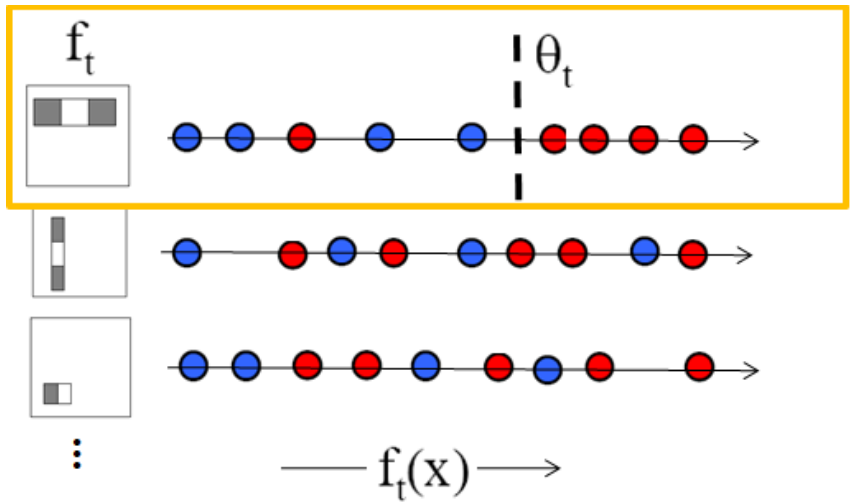
- In VJ-Face Detection AdaBoost is applied to find a relatively small set of most informative features out of a large set (about 180000) rectangular filter responses per sub-window.
- This can be done by associating only a single feature f_m to each classifier $h_m(x)$.
- A weak classifier $h_m(x, f_m, p_m, \Theta_m)$ is the combination of parameters (f_m, p_m, Θ_m) which yields the best weighted error function in iteration m.
  - f_m is a single feature, i.e. the best rectangular filter in iteration m.
  - Θ_m is the threshold applied to feature f_m, which best separates the set of weighted traning samples such that the weighted error function J_m is minimal.
  - $p_m \in \{-1, 1\}$ is the polarity, which indicates on which side of the threshold the positive (face) and negative (non-face) examples are located.
  - The definition of p_m allows that the decision for *face* is always 1 and non-face 0.

$$h_m(x, f_m, p_m, \Theta_m) = \begin{cases} 1 & \text{if } p_m f_m(x) < p_m \Theta_m \\ 0 & \text{otherwise} \end{cases}$$

- Note that the weak classifier has 1-dimensional input (response of only a single rectangular filter)! But in each iteration of Adaboost another filter can be the best.

Weak Classifiers for VJ Face Detection

Training of weak classifier $h_m(x, f_m, p_m, \Theta_m)$: Find parameter combination (f_m, p_m, Θ_m) which best separates red (faces) and blue (non-faces) circles. $p_m = 1$ if faces are below threshold and $p_m = -1$ if faces are above. threshold.



Outputs of a possible rectangle feature on faces and non-faces.

Adaboost Training in VJ-Face Detection

Adaboost Training in VJ-Face Detection

Let $T = \{\mathbf{x}_t, r_t\}_{t=1}^{N}$ be the set of N labeled training samples, with $r_t = 1$ for faces and $r_t = 0$ for non-faces.

1. Initialize the weights for all training samples with $r_t = 1$ by $w_t^{(1)} = \frac{1}{\sum_t r_t}$ and for all others by $w_t^{(1)} = \frac{1}{\sum_t (1-r_t)}$. Choose a number M of weak classifiers.
2. For $m = 1 \ldots M$
   1. Normalize weights:
   $$w_t^{(m)} := \frac{w_t^{(m)}}{\sum_t w_t^{(m)}}$$
   2. Fit a weak classifier $h_m(\mathbf{x}, f_m, p_m, \Theta_m)$ to the training data by determining the combination (f_m, p_m, Θ_m), which minimzes the weighted error function
   $$J_m = \sum_{t=1}^{N} w_t^{(m)} I(h(\mathbf{x}_t, f, p, \Theta) \neq r_t),$$
   where $I(h_m(\mathbf{x}_t, f, p, \Theta) \neq r_t)$ equals 1 if $h_m(\mathbf{x}_t) \neq r_t$ and 0 otherwise.
   3. Evaluate
   $$\beta_m = \frac{J_m}{1 - J_m} \quad \text{and} \quad \alpha_m = \ln\left(\frac{1}{\beta_m}\right)$$
   4. For all training samples $(\mathbf{x}_t, r_t)$ update weights:
   $$w_t^{(m+1)} = w_t^{(m)} \beta_m^{1 - I(h_m(\mathbf{x}_t, f, p, \Theta) \neq r_t)}$$

VJ-Adaboost Classification

- The trained weak classifiers $h_m(\mathbf{x}, f_m, p_m, \Theta_m)$ and the corresponding coefficients α_m constitute the final classifier

$$C(\mathbf{x}) = \begin{cases} 1 & \sum_{m=1}^{M} \alpha_m h_m(\mathbf{x}, f_m, p_m, \Theta_m) \geq \frac{1}{2} \sum_{m=1}^{M} \alpha_m \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

- In the experiments of [2] the number of weak classifiers has been set to $M = 200$

The 2 main features

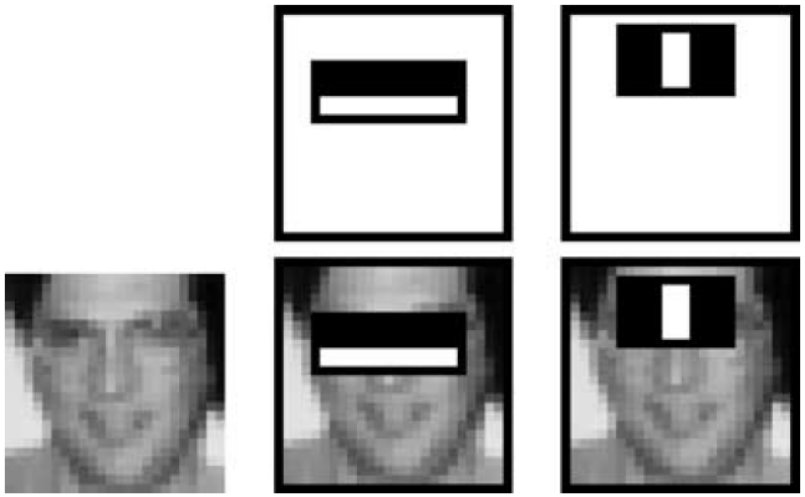
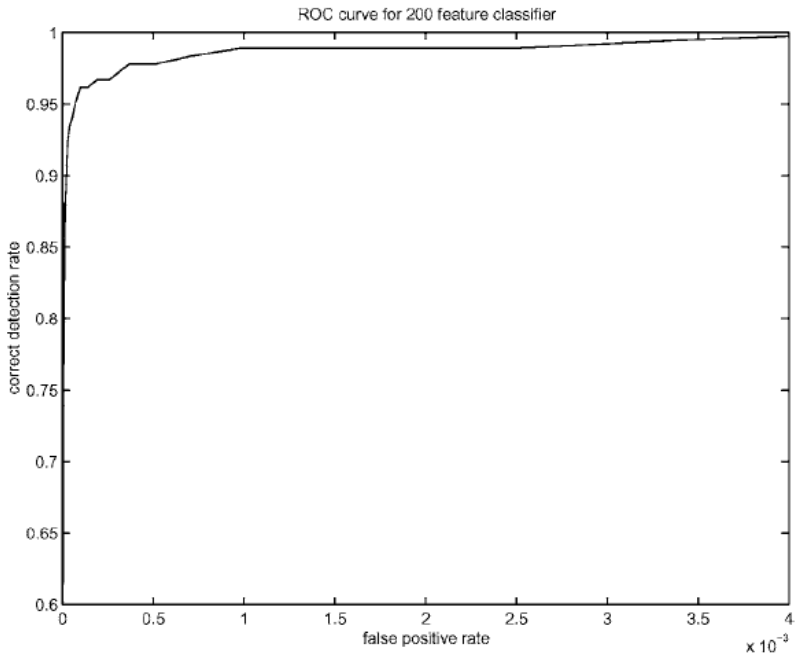


Figure 5. The first and second features selected by AdaBoost. The two features are shown in the top row and then overlayed on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

Source: [2]

Receiver Operate Curve (ROC) for face detection tests in [2]

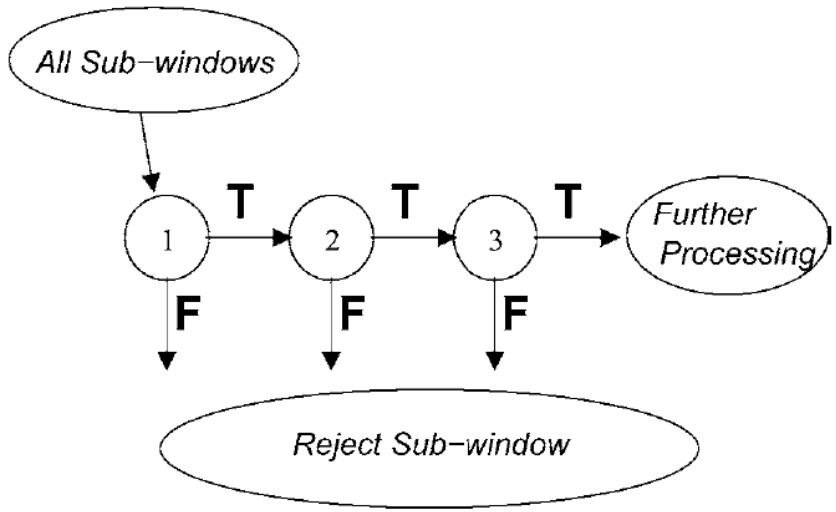


Source: [2]

At a face detection rate of 95% the false positve rate is 1/14084. The rate of false positives can be further decreased by the cascading approach.

Cascading Adaboost Classifiers

- Series of classifiers applied to every sub-window.
- Initial classifier: Eliminate large number of negative examples (non-faces) with little processing.
- Subsequent classifiers: Eliminate additional negative examples with increasing processing effort.
- Last classifier: Requires most processing power but must be applied for relatively small number of sub-windows.
- Effect: Increased detection performance and reduced computation time.



Source: [2]

Training and Test in [2]

Training:

- 4916 hand labeled faces scaled and aligned to base resolution of (24×24). Images have been downloaded during random web-crawl.
- 350 Million non-face subwindows collected from 9600 non-face images.
- About 6000 non-face subwindows have been used to train each stage of the cascade.
- Final detector is 38 layer cascade, included a total of 6060 features:
  - 2 features (shown in slide 4) recjects 50% non-faces and detects all faces
  - 10 features and rejects 80% of non-faces.
  - next 2 layers have 25 features ...
- Training time about one day

Detection Speed:

- Since a large majority of the sub-windows are discarded by the first two stages of the cascade, an average of 8 features out of a total of 6060 are evaluated per sub-window.
- 0.067 seconds to scan a (384×288) image (Pentium III, 700 Mhz)

Test

- Applied Testset: MIT + CMU frontal face test set: 130 images with 507 labeled frontal faces.

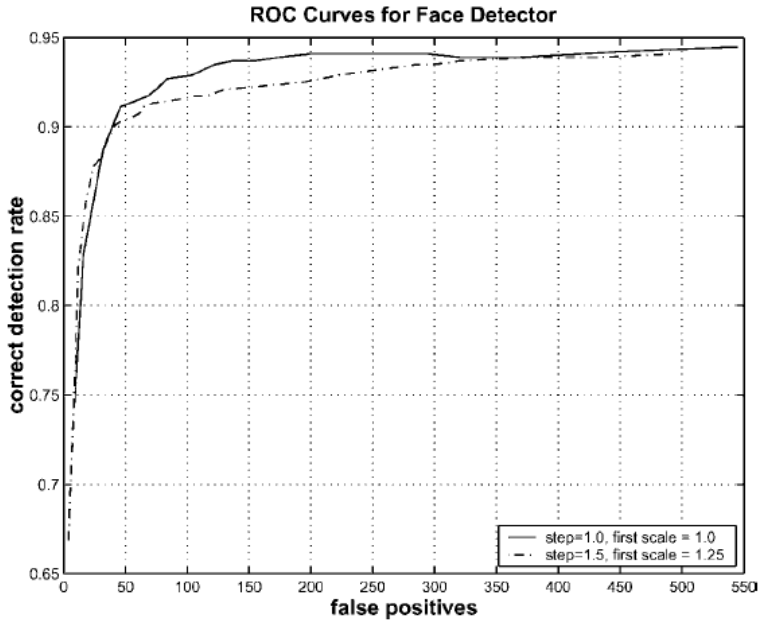


Figure 9. ROC curves for our face detector on the MIT + CMU test set. The detector was run once using a step size of 1.0 and starting scale of 1.0 (75,081,800 sub-windows scanned) and then again using a step size of 1.5 and starting scale of 1.25 (18,901,947 sub-windows scanned). In both cases a scale factor of 1.25 was used.

Detection Failures

- Reliable detection if in plane rotation is less than $\pm 15^\circ$ and out of plane rotation (towards profile) is less than $\pm 45^\circ$
- Detection problems in case of strong backlighting in which faces are dark and background is relatively light.
- Detection fails for significantly occluded faces - in particular if eyes are occluded.

Detection Failures

Out of Plane Rotation

What you train is what you get

References I

[1] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[2] P. A. Viola and M. J. Jones. Robust real-time face detection. In *ICCV*, page 747, 2001.