Formation et Analyse d'Images

James L. Crowley

ENSIMAG 3 — Premier Semestre 2009/2010

Lesson 11 — 11 January 2010

Face Detection using a Cascade of Boosted Classifiers

Outline:

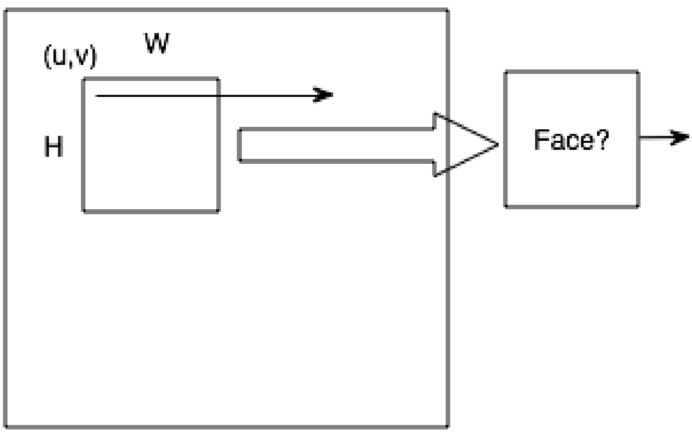
1. Processes Overview

The Detection Process

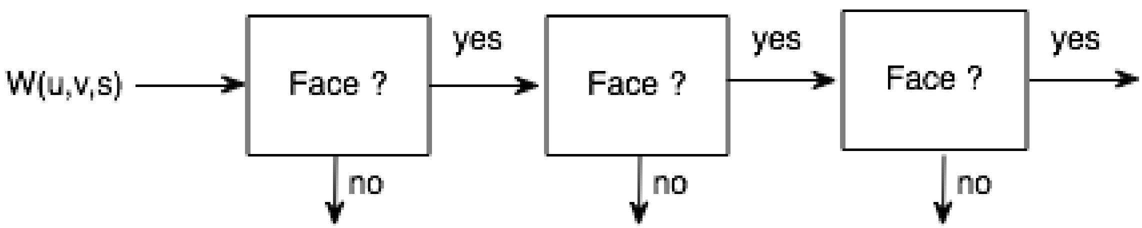
A Cascade of Classifiers detects faces with a window scanning approach.

An image window, W(i,j; u, v, s) is an (sW x sH) pixel window with its upper left corner at pixel (u, v). We will note this as W(u,v,s), leaving (i,j) as implicit.

The window is texture mapped (transformed) so that it fits into a standard size window of a given size (W, H). For faces, the window size is typically approximately (24, 24) pixels.



The decision of whether the window W(u,v,s) contains a face is provided by a cascade of boosted linear classifiers.



The algorithm requires a large number of local "features" to classify the window. This can be provided by the Scaled Gaussian Derivatives seen in lectures 7 and 8.

They can also be provided by Haar wavelets computed using a Difference of Boxes.

2. Image Description with Difference of Boxes

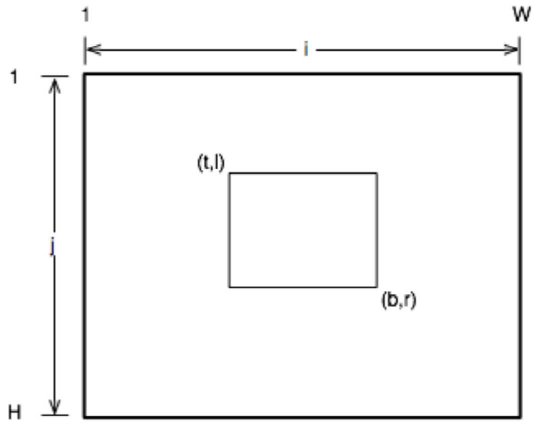
An image rectangle is defined by the top-left and bottom-right corner.
This may be represented by a vector (t, l, b, r). The sum of pixels in a rectangle defines a box feature.

Box Features

A box feature is the sum of pixels over a rectangle from top (t) to left (l) and bottom (b) to right (r)

With the constraints : top < bottom and right > left.

$$b(t,l,b,r) = \sum_{i=l}^{r}\sum_{j=t}^{b} p(i,j)$$



For an window of size WxH there are $N=W^2H^2/4$ possible boxes.

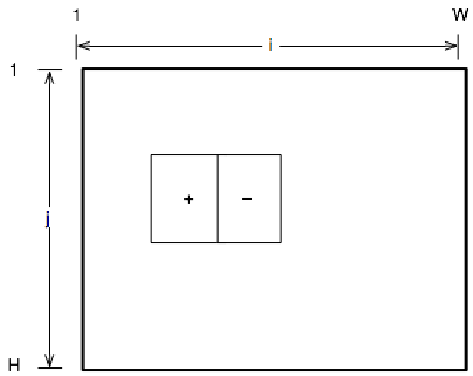
$$N = \frac{W^2}{2}\cdot\frac{H^2}{2}$$

Difference of Boxes

A first order Difference of Boxes (DoB) feature is a difference of two boxes $box(t1,l1,b1,r1)$.

$$DoB(t1,l1,b1,r1,t2,l2,b2,r2) = box(t1,l1,b1,r1) - box(t2,l2,b2,r2)$$

There are N^2 possible 1st order (2 box) DoB features in an image
There are N^3 possible 2nd order (3 box) DoB features in an image.
Not all DoBs are useful.

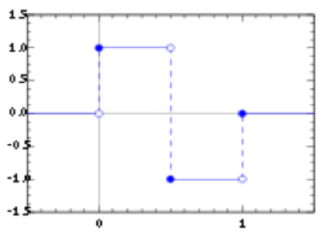


An interesting subclass are Difference of Adjacent Boxes where the sum of pixels is 0. These are Haar wavelets.

Haar Wavelets:

Haar A. Zur Theorie der orthogonalen Funktionensysteme, Mathematische Annalen, 69, pp 331–371, 1910.

The Haar wavelet is a difference of rectangular Windows.



$$h(t) = \begin{cases} 1 & \text{for } 0 \le t < 0.5 \\ -1 & \text{for } 0.5 \le t < 1 \\ 0 & \text{for } t < 0 \text{ and } t \ge 1 \end{cases}$$

The Haar wavelet may be shifted by d and scaled by s

$$h(t;s,d) = h(t/s - d)$$

Note that the Haar Wavelet is zero gain (zero sum).

$$G = \int_{-\infty}^{\infty} h(t)dt = 0$$

The Digital (discrete sampled) form of Haar wavelet is

$$h(n;d,k) = \begin{cases} 1 & \text{for d} \le n < d + k/2 \\ -1 & \text{for d} + \text{k}/2 \le n < d + k \\ 0 & \text{for } n < d \text{ and } n \ge d + k \end{cases}$$

Haar wavelets can be used to define an orthogonal transform analogous to the Fourier basis. This can be used to define an orthogonal transform (the Walsh-Hadamard Transform). The basis is

$$H_0 = +1$$

$$H_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_2 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

…

$$H_m = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{bmatrix}$$

Haar Functions, and the Walsh-Hadamard transform have been used in Functional Analysis and signal processing for nearly a century.

In the 1980s the Wavelet community re-baptized the Haar functions as "wavelets" and demonstrated that the Walsh-Hadamard transform is the simplest form of wavelet transform.

A 2-D form of Walsh-Hadamard transform may be defined using DoB features. These can be calculated VERY fast using an algorithm known as Integral Images.

3. Fast 2D Haar Wavelets using Integral Image

Integral Images.

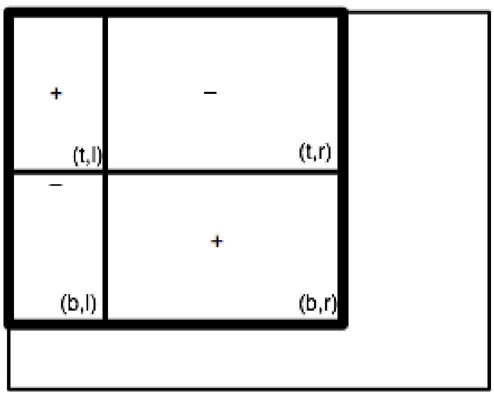
An integral image is an image where each pixel contains the sum from the upper left corner:

$$ii(u,v) = \sum_{i=1}^{u} \sum_{j=1}^{v} p(i,j)$$

An integral image provides a structure for very fast computation of 2D Haar wavelets.

Any box feature can be computed with 4 operations (additions/subtractions).

box(t,l,b,r)=ii(b,r)–ii(t,r)–ii(b,l)+ii(t,l)

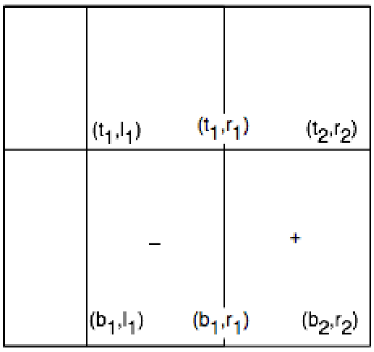


An arbitrary 1st order difference of boxes costs 8 ops.

$$DoB(t_1,l_1,b_1,r_1,t_2,l_2,b_2,r_2) = ii(b_1,r_1)-ii(t_1,r_1)-ii(b_1,l_1)+ii(t_1,l_1)$$
$$- ii(b_2,r_2)-ii(t_2,r_2)-ii(b_2,l_2)+ii(t_2,l_2) -$$

However, a 1st order Haar wavelet costs only 6 ops because $r_1=l_2$ and thus

$ii(t_1,r_1) = ii(t_2,l_2)$ and $ii(b_1,r_1)= ii(b_2,l_2)$



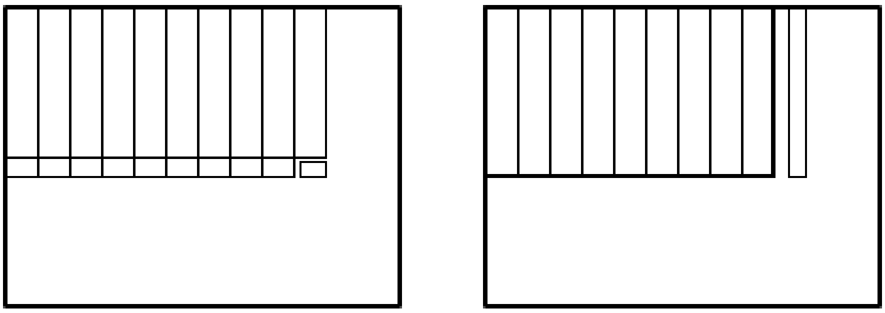
$$Haar(t_1,l_1,b_1,r_1,b_2,r_2) = ii(b_2,r_2)–2ii(b_1,r_1)+ii(b_1,l_1)–ii(t_2,r_2)+2ii(t_1,r_1)–ii(t_1,l_1)$$

Fast Integral Image algorithm.

Integral images have been used for decades to compute local energy for normalization of images. The fast algorithm involves a row buffer that contains the sum of each row

```
For j = 1 to I
For i=1 to J
        r(i) := r(i) + p(i,j)
        ii(i,j) = ii(i-1,j)+r(i)
```

Cost = 2IJ ops.



In 2001, Paul Viola and Mike Jones at MERL (Misubishi Research Labs) showed that Haar wavelets could be used for real time face detection using a cascade of linear classifiers.

They computed the Haar Wavelets for a window from integral images.

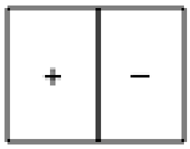
4. Linear Classifiers for Face Detection

The innovation in the Viola-Jones face detector resulted from

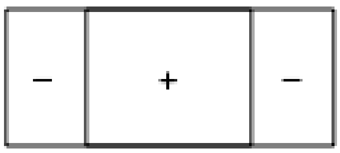
1) A very large number of very simple features (Haar wavelets).
2) The use of the Ada boost algorithm to learn an arbitrariy good detector.

HAAR wavelets are computed using difference of Boxes, with Integral Images.

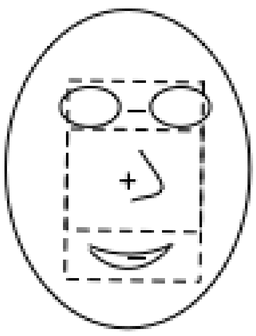
A WxH image contains $N = W^2H^2/4$ possible 1st order Haar wavelets.
(difference of adjacent boxes of same size).



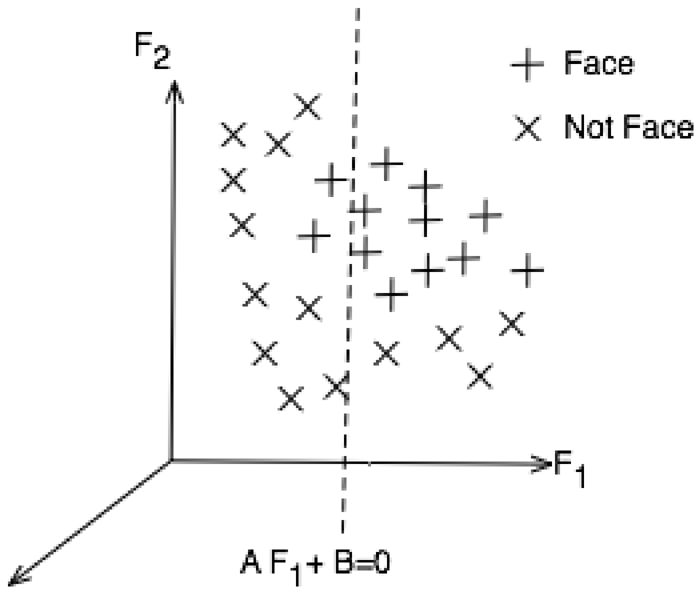
Similarly, any 2nd Haar wavelet can be computed with 8 ops.



A WxH window contains $W^2H^2/8$ possible 1st and 2nd order Haar wavelets.
These provide a space of $N = W^2W^2/8$ features for detecting Faces.
Each feature, F_n is defined as a difference of boxes.



For a give position (u,v) and scale (s) any window W(u, v, s) that contains a face is a point "+" in this very N-dimensional space.



Each feature can be used to define a hyper-plane $<W(u,v), F_n> + B_n = 0$.

where $\langle W(u,v), F_n \rangle = \sum_{i=1}^{W}\sum_{j=1}^{H} W(i,j), F_n(i,j)$

The problem is to choose B_n so that most non-face windows are on one side of the hyperplane and most face windows are on the other.

Training a single classifier:

Assume a very large training set of M face windows $\{W_m\}$ that have been labeled by a set of labels $\{y_m\}$ such that $y=+1$ if face and $y=-1$ if not face,

For a give feature n: Compute $B_n = E\{ <W_m, F_n> \cdot y_m\}$

Then for a given window, W, we can classify it as Face or Not face using;

if $<W, F_n> + B_n > 0$ then Face else not Face.

This is the minimum error linear classifier for the feature F_n.

For the training set $\{W_m\}$, the error rate for the feature F_n is

$$E_n = \mathrm{Card}\{(<W_m, F_n> + B_n) \cdot y_m < 0\}$$

For a set $\{(W_m, y_m)\}$, $\{(F_n, B_n)\}$ the best classifier is

$\text{Arg-Min}_n\{E_n\}$

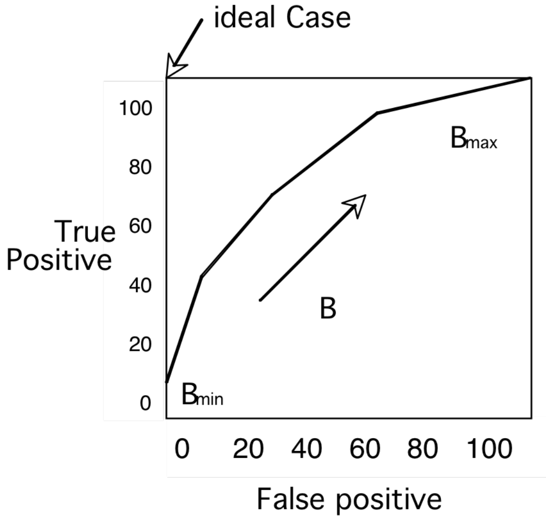
We want to learn the set of *i* best classifiers.
We can improve learning with Boosting.

Note that there are two parts to the errors:
False Positives (FP) and False Negatives (FN)

Wecan trade FPs for FNs by adding a Bias B,

For a classifier F_n

$$FP = \text{Card}\{(\langle W_m, F_n\rangle + B_n + B) > 0 \text{ and } y_m = -1\}$$
$$FN = \text{Card}\{(\langle W_m, F_n\rangle + B_n + B) < 0 \text{ and } y_m = +1\}$$



Boosted Learning
To boost the learning, after selection of each "best" classifier, (F_n, B_n)
we re-weight the labels y_n to increase the weight of incorrectly classed windows:

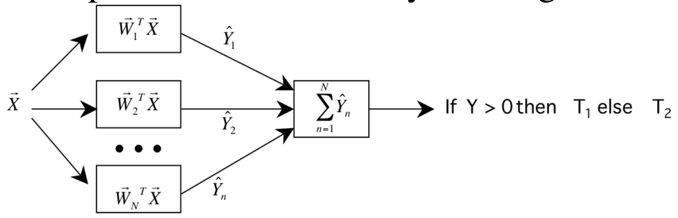
$$\text{For all } m = 1 \text{ to } M \text{ if } (\langle W_m, F_n\rangle + B_n) \cdot y_m^{(i-1)} < 0 \text{ then } y_m^{(i)} = 2\, y_m^{(i-1)}$$

We then learn the i^{th} classifier from the re-weighted set

$E_{min} = M$
For n=1 to N do
  $B_m = E\{ \langle W_m, F_n\rangle \cdot y_m^{(i)}\}$
  $E_n = \text{Card}\{(\langle W_m, F_n\rangle + B_n) \cdot y_m < 0\}$
  if $E_n < E_{min}$ then $E_{min} := E_n$

5. Learning a Committee of Classifiers with Boosting

We can improve classification by learning a committee of the best I classifiers.



The decision is made by voting. A window W is determined to be a Face if the majority of classifiers vote > 0.

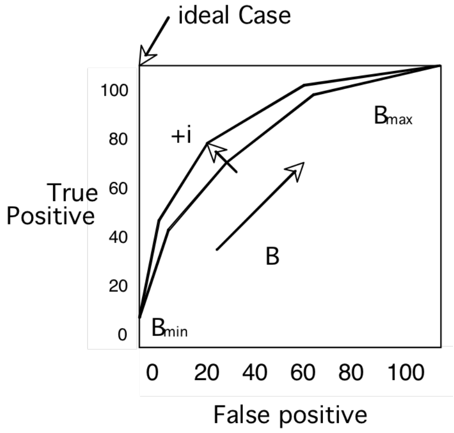
$$\text{If } \sum_{i=1}^{I} < W_m F_n > + B_n \text{ then Face else Not-Face.}$$

ROC Curve

We can describe a committee of classifiers with an ROC curve, but defining a global bias, B. The ROC describes the number of False Positives (FP) and False Negatives (FN) for a set of classifier as a function of the global bias B.

$$FP = \mathrm{Card}\{(<W_m, F_n> + B_n + B) > 0 \text{ and } y_m = -1\}$$
$$FN = \mathrm{Card}\{(<W_m, F_n> + B_n + B) < 0 \text{ and } y_m = +1\}$$

The Boosting theorem states that adding a new boosted classifier to a committee always improves the committee ROC curve. We can continue adding classifiers until we obtain a desired rate of false positives and false negatives.



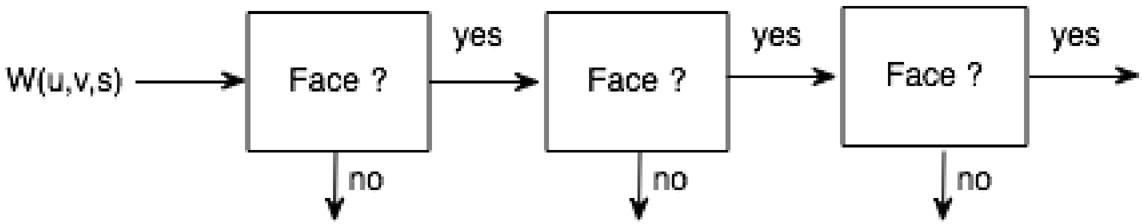
6. Learning a Multi-Stage Cascade of Classifiers

We can optimize the computation time by using a multistage cascade.

Algorithm:

1) Set a desired error rate for each stage j : (FP_j, FN_j).
2) For j = 1 to J

For all windows labeled as Face by j-1 stage, learn a boosted committee of classifiers that meets (FP_j, FN_j).



Each stage acts as a filter, rejecting a grand number of easy cases, and passing the hard cases to the next stage.