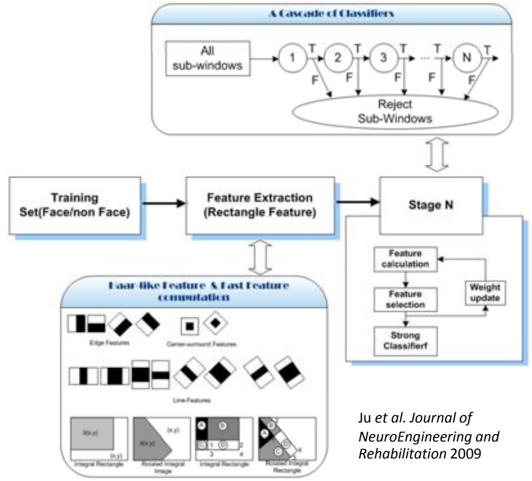
Viola & Jones Object Detection Framework

Overview

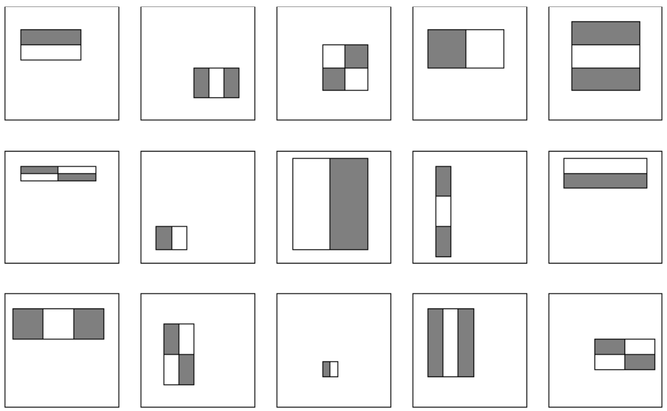
Rapid Object Detection using a Boosted Cascade of Simple Features
Viola & Jones, 2001



Ju et al. Journal of NeuroEngineering and Rehabilitation 2009

- **Feature Extraction**
  - Haar wavelet response (Haar-like Feature)
  - Integral Image
- **Classification**
  - Weak Classifier
  - Adaboost Learning
  - Feature Selection
  - Optimal Threshold
  - Cascade Classifier

Haar-like Feature



Considering all possible filter parameters: position, scale, and type:

180,000+ possible features associated with each 24 x 24 window

Use AdaBoost both to select the informative features and to form the classifier

Integral Image

1	2	3	4	3	0	6	7	9
4	5	6	2	1	5	4	5	2
1	4	8	6	3	2	9	0	0
2	1	4	5	7	0	2	0	2
1	1	1	0	1	4	4	2	5
1	2	4	0	0	0	2	5	1
9	8	2	5	7	8	9	1	0
1	2	7	5	0	2	1	0	0

Original Image

$I(x, y)$

If we want to calculate the summation in red area, we need 14 operations

5+7+0+2+0+0+1+4+4+2+0+0+0+2+5 = 32

We can reduce this computation by using integral image

Integral Image

Summation

1	2	3	4	3	0	6	7	9
4	5	6	2	1	5	4	5	2
1	4	8	6	3	2	9	0	0
2	1	4	5	7	0	2	0	2
1	1	1	0	1	4	4	2	5
1	2	4	0	0	0	2	5	1
9	8	2	5	7	8	9	1	0
1	2	7	5	0	2	1	0	0

Original Image

$I(x,y)$

1	3	6	10	13	13	19	26	35
5	12	21	27	31	36	46	58	69
6	17	34	46	53	60	79	91	102
8	20	41	58	72	79	100	112	125
9	22	44	61	76	87	112	126	144
10	25	51	68	83	94	121	140	159
19	42	70	92	114	133	169	189	208
20	45	80	107	129	150	187	207	226

Integral Image

$$S(x,y) = \sum_{p \le x, q \le y} I(p,q)$$

Integral Image

1	2	3	4	3	0	6	7	9
4	5	6	2	1	5	4	5	2
1	4	8	6	3	2	9	0	0
2	1	4	5	7	0	2	0	2
1	1	1	0	1	4	4	2	5
1	2	4	0	0	0	2	5	1
9	8	2	5	7	8	9	1	0
1	2	7	5	0	2	1	0	0

Original Image

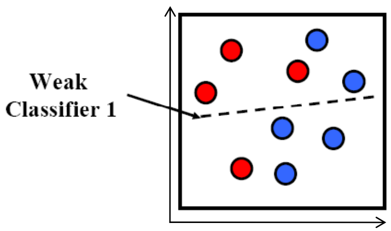
1	3	6	10	13	13	19	26	35
5	12	21	27	31	36	46	58	69
6	17	34	46	53	60	79	91	102
8	20	41	58	72	79	100	112	125
9	22	44	61	76	87	112	126	144
10	25	51	68	83	94	121	140	159
19	42	70	92	114	133	169	189	208
20	45	80	107	129	150	187	207	226

Integral Image

= 140-91-51+34 = 32

Can be compute using 3 operators

Adaboost [Adaptive Boosting]



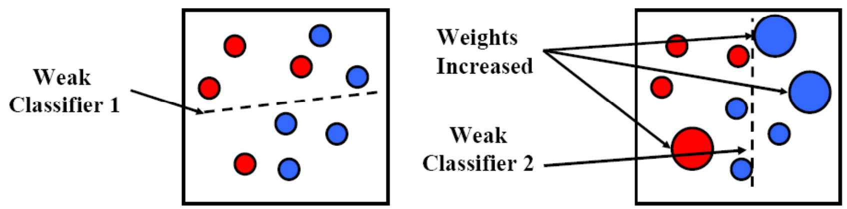
Consider a 2-d feature space with positive and negative examples.

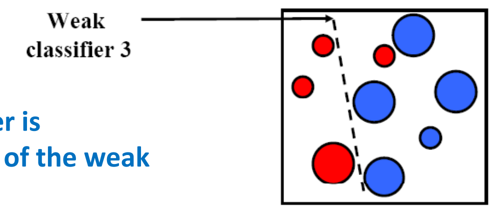
Each weak classifier splits the training examples with at least 50% accuracy.

Examples misclassified by a previous weak learner are given more emphasis at future rounds.

Figure adapted from Freund and Schapire

Adaboost [Adaptive Boosting]



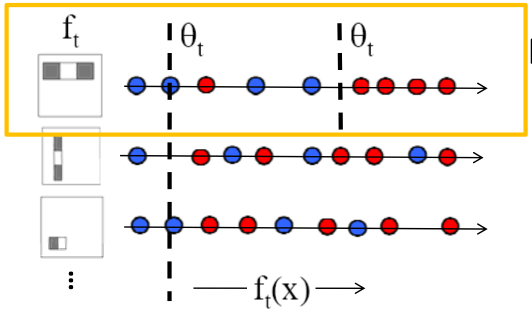


Final classifier is combination of the weak classifiers

Figure adapted from Freund and Schapire

Adaboost

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



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

Resulting weak classifier:

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

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Adaboost

Start with uniform weights on training examples

Evaluate *weighted* error for each feature, pick best.

Incorrectly classified -> more weight

Correctly classified -> less weight

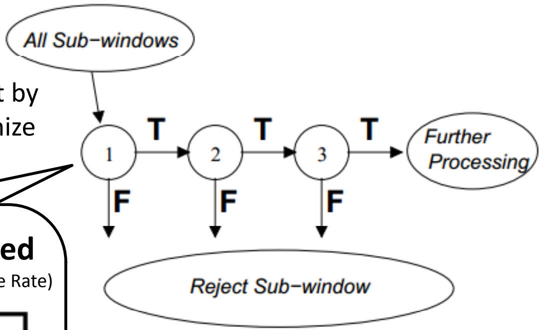
Final classifier is combination of the weak ones, weighted according to error they had.

- Given example images $(x_1, y_1), \dots, (x_n, y_n)$ 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 $t = 1, \dots, T$:
  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, $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
  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 $\alpha_t = \log \frac{1}{\beta_t}$

Cascade Classifier

- Reducing Computation Time
- Each stage trained by Adaboost by adjusting the threshold to minimize false negatives

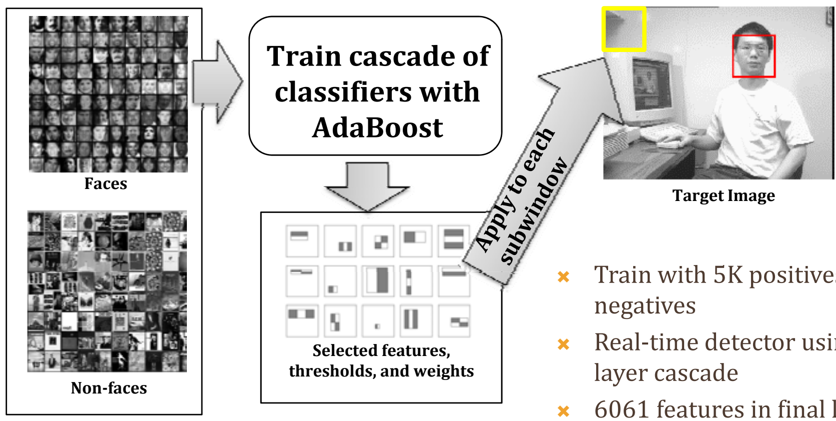


First two features selected

(100% Detection rate with 40% False Positive Rate)

- Each stage is trained by adding features until the target detection and false positives rates are met
- Stages are added until the overall target for false positive and detection rate is met.

Summary



- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- Implementation available in OpenCV

K. Grauman, B. Leibe