??: James has question
JJ: James modifies it

Robust Real-time Object Detection

- IJCV, pp. 57(2), pp. 137-154, 2004.
- P. Viola and M.J. Jones

Keywords: Face Detection,
Feature Extraction, Integral Image,
AdaBoost , Cascade.

曾郁凱 Yu-Kai Jseng
paulll709@gmail.com
2011/8/16

由王基鎮學長指導
簡報承自
侯建州、朱建州、張峻豪學長

Outline

1. Introduction
2. Features
   2.1 Features Extraction
   2.2 Integral Image
3. AdaBoost
   3.1 Training Process
   3.2 Testing Process
4. The Attentional Cascade
5. Experimental Results
6. Conclusion
7. Reference

1. Introduction

- Frontal face detection system achieves:
  1. High detection rates
  2. Low false positive rates
- Three main contributions:
  1. Integral image
  2. AdaBoost*:* Select a small number of important features.
  3. Cascaded structure

2. Features ??

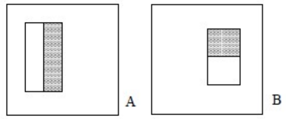
- Method
  1. Feature-based (Local ?): Use extraction features like eye, nose pattern.
  2. Knowledge-based: Use rules of facial feature.
  3. Image-based (Global ?): Use face segments and predefined face pattern.
- Use Feature-based method
- Reason:
  1. Knowledge-based system is difficult to learn using a finite quantity of training data.
  2. Much faster than Image-based system.

[3] A.S.S.Mohamed, Y. Weng, S. S Ipson, and J. Jiang, "Face Detection based on Skin Color in Image by Neural Networks", ICIAS 2007, pp. 779-783, 2007.

2.1 Feature Extraction (1/2)

- **Filter:**

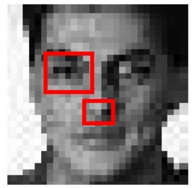
  Filter type: Ex. Parameters -1, +1.

  Ex: Haar-like filter 

- **Feature:**

  a. Filter 的座標位置 (position)

  b. Filter 的大小 (size)

  Ex: Eye , nose 

- **Feature Value:**

  Feature Value =
  Filter $*$ Feature(Position,Size)

  Convolution: Linear combination

  Ex: 

2.1 Feature Extraction (2/2)

- Haar-Like Filter:

  The sum of the pixels which lie within the white rectangles (+1) are subtracted from the sum of pixels in the grey rectangles (-1).

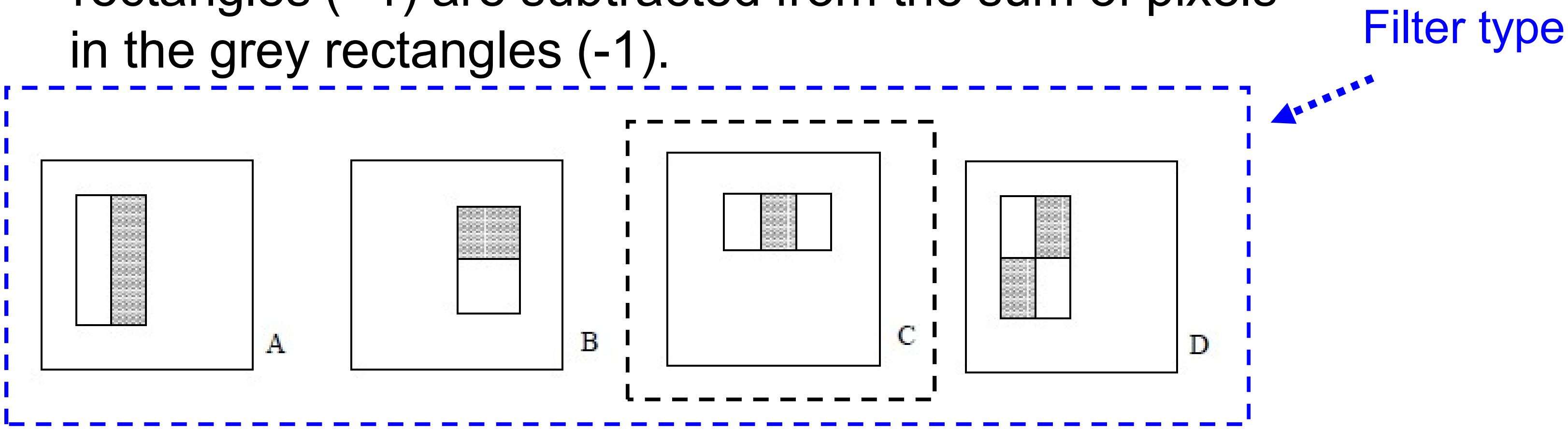
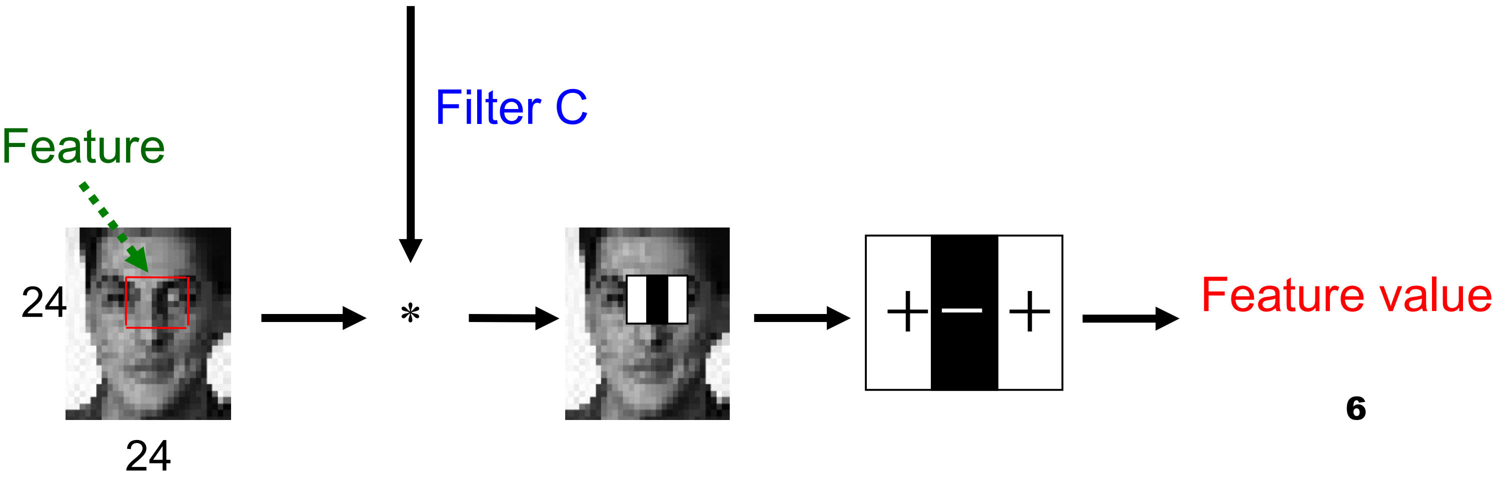


Figure 1

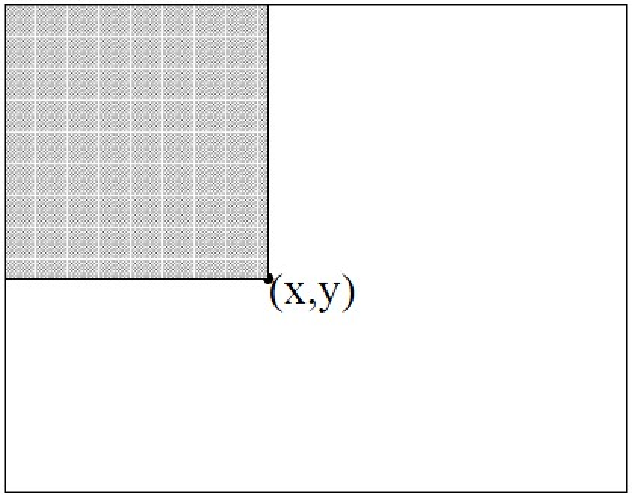


2.2 Integral Image (1/7)

- Integral Image
  1. Rectangle features
  2. Computed very rapidly
- *ii*(*x* , *y*) : Sum of the pixels above and to the left of *(x , y)*.

$$ii(x,y) = \sum_{x' \leq x,\, y' \leq y} i(x',y')$$

Grayvalue at position (x’,y’)



2.2 Integral Image (2/7)

Known:

A: Sum of the pixels within rectangle *A*.
B: Sum of the pixels within rectangle *B*.
C: Sum of the pixels within rectangle *C*.
D: Sum of the pixels within rectangle *D*.
Location 1 value is *A*.
Location 2 value is *A*+*B*.
Location 3 value is *A*+*C*.
Location 4 value is *A*+*B*+*C*+*D*.

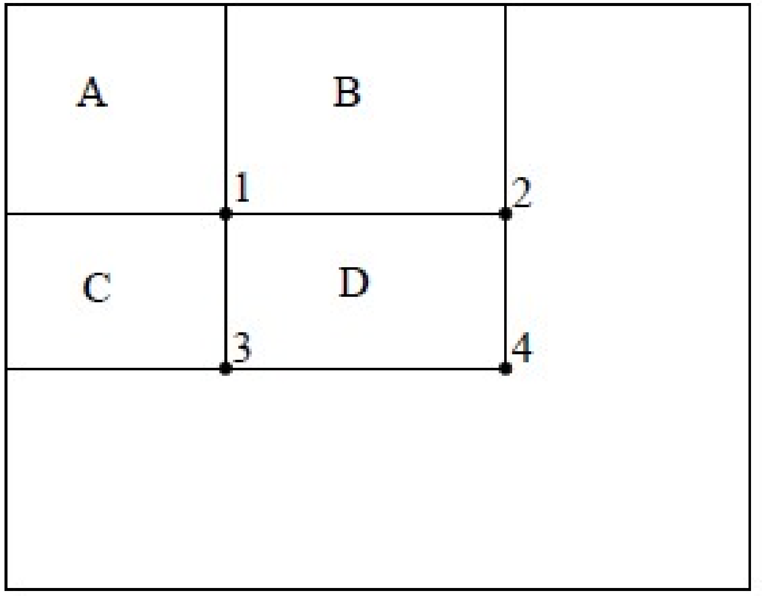


Figure 2: Integral image

Equation:

$1=A \quad 2=A+B \quad 3=A+C \quad 4=A+B+C+D$

Q : The sum of the pixels within rectangle *D* = ?

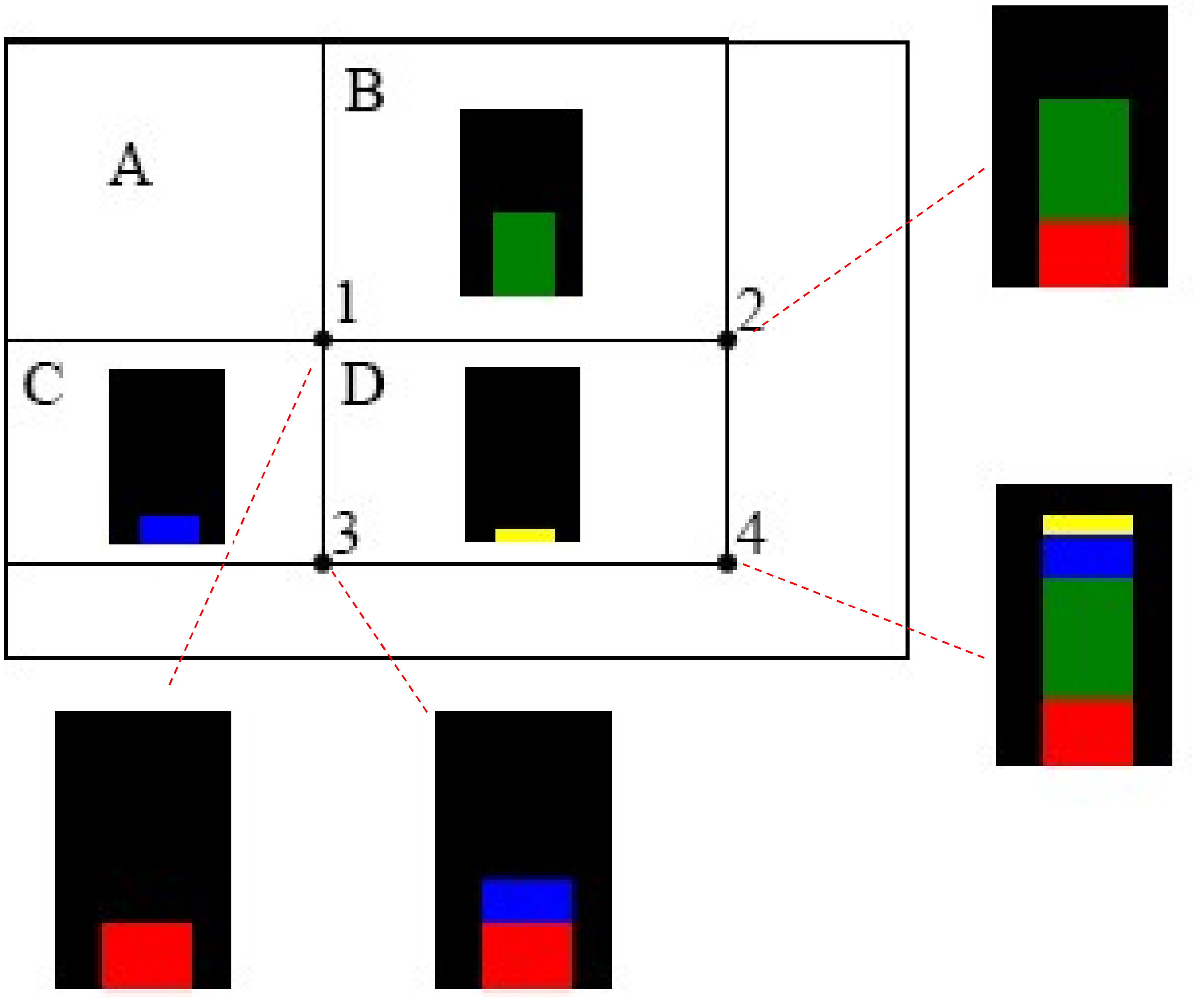
A : $4 = A + B + C + D \Rightarrow D = 4 - A - B - C$

$\Rightarrow D = 4 - (A + B) - C \Rightarrow D = 4 - 2 - C$

$\Rightarrow D = 4 - 2 - (3 - A) \Rightarrow D = 4 - 2 - (3 - 1) \Rightarrow 4 + 1 - (2 + 3)$

The sum within *D* can be computed as $\boxed{D = 4 + 1 - (2 + 3).}$

2.2 Integral Image (3/7)



2.2 Integral Image - Integral Histogram

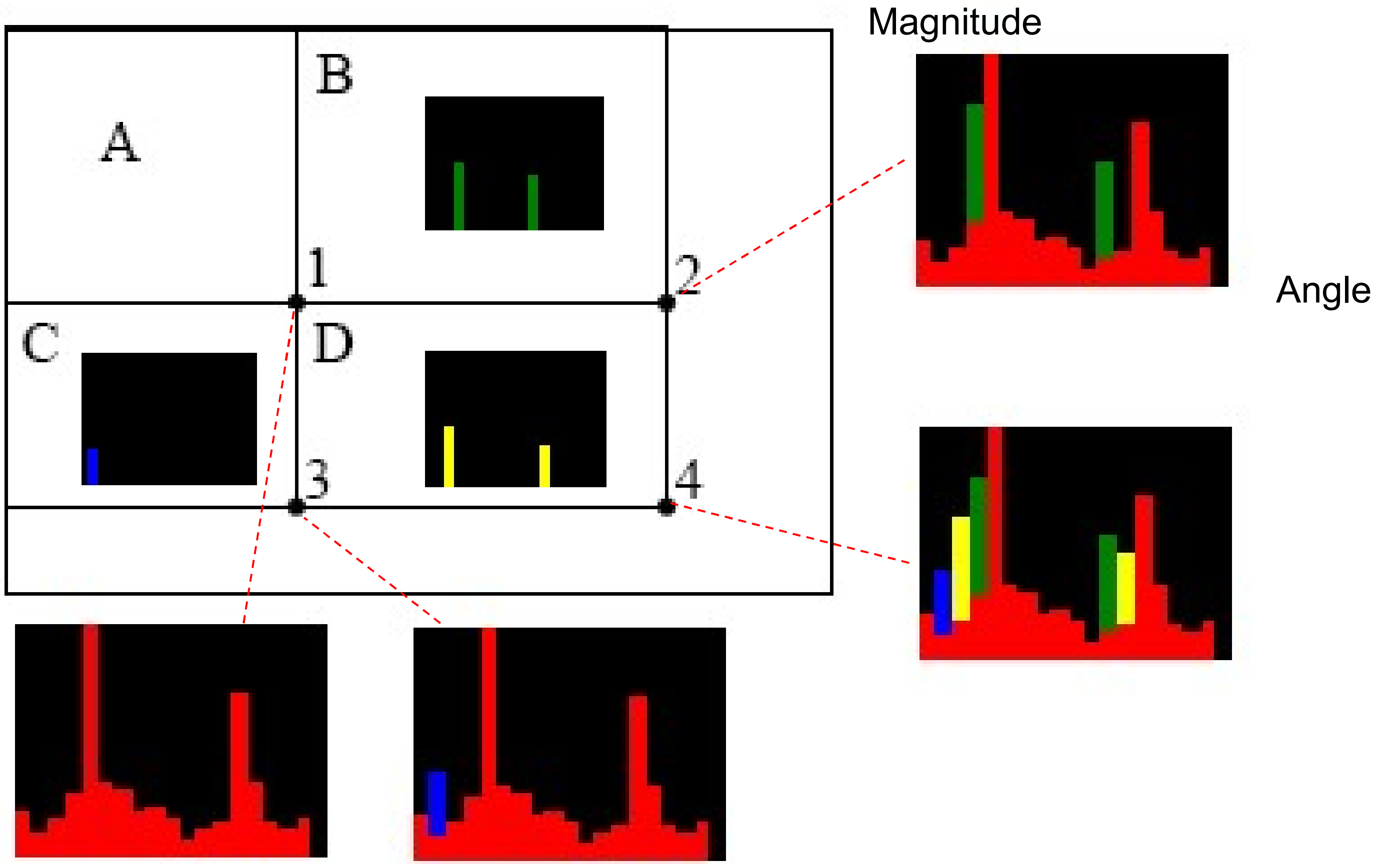


Fig. A: The information stored in integral histogram. The HOG information stored in each pixel is the sum of all the pixels above and to the left. Then HOG feature in region D can be easily compute by D = 4 + 1 – 2 - 3.

2.2 Integral Image (4/7)

- Using the following pair of recurrences to get integral image $ii(x, y)$:

$$s(x,y) = s(x,y-1) + i(x,y) \qquad (1) \qquad \text{Row Sum}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \qquad (2) \qquad \text{Column Sum}$$

$ii(x, y)$ is the integral image. $\qquad ii(-1, y) = 0$

$i(x, y)$ is the original image.

$s(x, y)$ is the cumulative row sum. $\qquad s(x, -1) = 0$

$$ii(x,y) = \sum_{x' \le x} \sum_{y' \le y} i(x', y')$$

2.2 Integral Image (5/7)

$i(x,y)$ original image	$s(x,y)$ cumulative row sum	$ii(x,y)$ integral image

$i(x,y)$ (original image):

x \ y	0	1	2
0	1	2	3
1	1	2	1
2	4	3	1

row sum

$s(x,y)$ (cumulative row sum):

x \ y	0	1	2
0	1	3	6
1	1	3	4
2	4	7	8

column sum

$ii(x,y)$ (integral image):

x \ y	0	1	2
0	1	3	6
1	2	6	10
2	6	13	18

$$s(x,y) = s(x,y-1) + i(x,y) \qquad (1)$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \qquad (2)$$

2.2 Integral Image (6/7)

$i(x, y)$ original image	$s(x, y)$ cumulative row sum	$ii(x, y)$ integral image

$i(x, y)$ (3*3):

x \ y	0	1	2
0	1	2	3
1	1	2	1
2	4	3	1

→

$s(x, y)$ (3*3):

x \ y	0	1	2
0	1	3	6
1	1	3	4
2	4	7	8

→

$ii(x, y)$ (3*3):

x \ y	0	1	2
0	1	3	6
1	2	6	10
2	6	13	18

$s(x,y) = s(x,y-1) + i(x,y)$

$s(0,0) = s(0,-1) + i(0,0)$ =0+1=1

$s(1,0) = s(1,-1) + i(1,0)$ =0+1=1

$s(2,0) = s(2,-1) + i(2,0)$ =0+4=4

$s(0,1) = s(0,0) + i(0,1)$ =1+2=3

$s(1,1) = s(1,0) + i(1,1)$ =1+2=3

⋮

$s(2,2) = s(2,1) + i(2,2)$ =7+1=8

$ii(x,y) = ii(x-1,y) + s(x,y)$

$ii(0,0) = ii(-1,0) + s(0,0)$ =0+1=1

$ii(1,0) = ii(0,0) + s(1,0)$ =1+1=2

$ii(2,0) = ii(1,0) + s(2,0)$ =2+4=6

$ii(0,1) = ii(-1,1) + s(0,1)$ =0+3=3

$ii(1,1) = ii(0,1) + s(1,1)$ =3+3=6

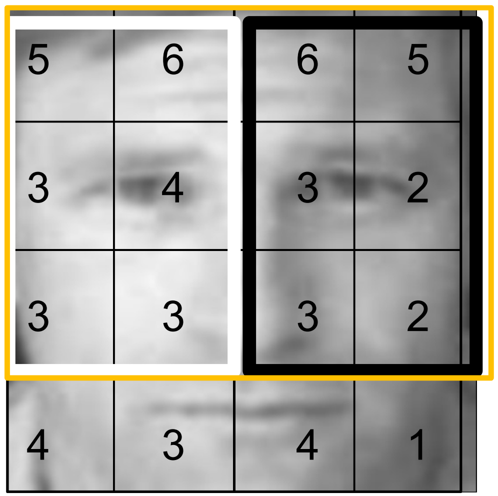
⋮

$ii(2,2) = ii(1,2) + s(2,2)$ =10+8=18

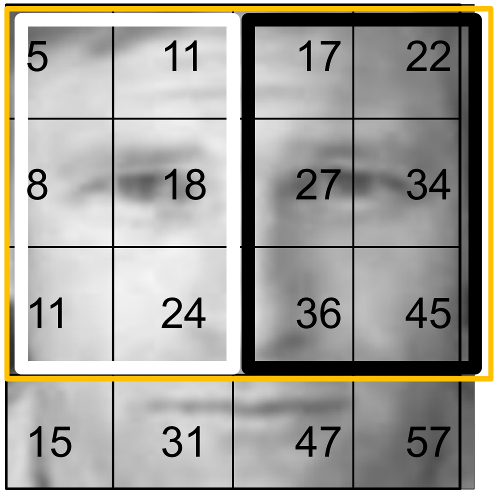
2.2 Integral Image with feature value (7/7)JJ

Haar-Like Filter

original image



integral image



original image Feature Value =

$$(5+6+3+4+3+3)-(6+5+3+2+3+2)=3$$

integral image Feature Value =

$$\{(5+24)-(11+11)\}-\{(17+45)-(22+36)\}=3$$

2.2 Integral Image: OpenCV (1/3)

An integral image sum has the form:

$$\text{sum}(X,Y) = \sum_{x \le X} \sum_{y \le Y} \text{image}(x,y)$$

The optional `sqsum` image is the sum of squares:

$$\text{sum}(X,Y) = \sum_{x \le X} \sum_{y \le Y} (\text{image}(x,y))^2$$

and the `tilted_sum` is like the sum except that it is for the image rotated by 45 degrees:

$$\text{tilt\_sum}(X,Y) = \sum_{y \le Y} \sum_{\text{abs}(x-X) \le y} \text{image}(x,y)$$

Using these integral images, one may calculate sums, means, and standard deviations over arbitrary upright or "tilted" rectangular regions of the image. As a simple example, to sum over a simple rectangular region described by the corner points ($x1$, $y1$) and ($x2$, $y2$), where $x2 > x1$ and $y2 > y1$, we'd compute:

$$\sum_{x1 \le x \le x2} \sum_{y1 \le y \le y2} [\text{image}(x,y)]$$

$$= [\text{sum}(x2,y2) - \text{sum}(x1-1,y2) - \text{sum}(x2,y1-1) + \text{sum}(x1-1,y1-1)]$$

In this way, it is possible to do fast blurring, approximate gradients, compute means and standard deviations, and perform fast block correlations even for variable window sizes.

2.2 Integral Image: OpenCV (2/3)

To make this all a little more clear, consider the 7-by-5 image shown in Figure 6-18; the region is shown as a bar chart in which the height associated with the pixels represents the brightness of those pixel values. The same information is shown in Figure 6-19, numerically on the left and in integral form on the right. Integral images (I') are computed by going across rows, proceeding row by row using the previously computed integral image values together with the current raw image (I) pixel value $I(x, y)$ to calculate the next integral image value as follows:

$$I'(x,y)=I(x,y)+I'(x-1,y)+I'(x,y-1)-I'(x-1,y-1)$$

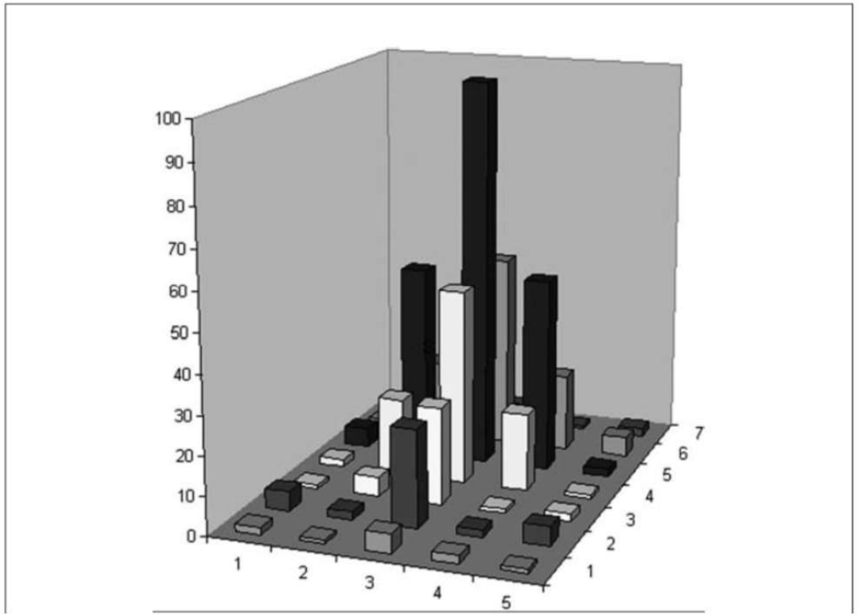


Figure 6-18. Simple 7-by-5 image shown as a bar chart with x, y, and height equal to pixel value

2.2 Integral Image: OpenCV (3/3)

The last term is subtracted off because this value is double-counted when adding the second and third terms. You can verify that this works by testing some values in Figure 6-19.

When using the integral image to compute a region, we can see by Figure 6-19 that, in order to compute the central rectangular area bounded by the 20s in the original image, we'd calculate 398 – 9 – 10 + 1 = 380. Thus, a rectangle of any size can be computed using four measurements (resulting in $O(1)$ computational complexity).

1	2	5	1	2
2	**20**	**50**	**20**	5
5	**50**	**100**	**50**	2
2	**20**	**50**	**20**	1
1	5	25	1	2
5	2	25	2	5
2	1	5	2	1

1	3	8	9	11
3	**25**	**80**	**101**	108
8	**80**	**235**	**306**	315
10	**102**	**307**	**398**	408
11	108	338	430	442
16	115	370	464	481
18	118	378	474	492

Figure 6-19. The 7-by-5 image of Figure 6-18 shown numerically at left (with the origin assumed to be the upper-left) and converted to an integral image at right

3. AdaBoost (1/2)

- **AdaBoost** (Adaptive Boosting) is a machine learning algorithm.
- AdaBoost works by choosing and combining weak classifiers together to form a more accurate strong classifier !
  - Weak Classifier:

Feature value

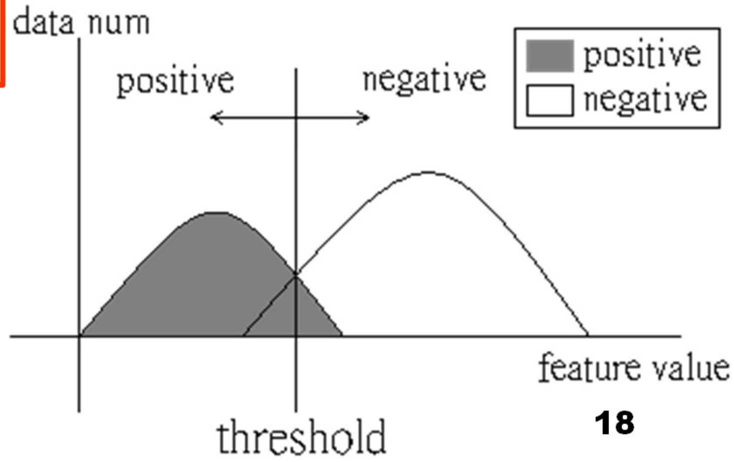
Threshold (from selected feature value)

$$h(X) = \begin{cases} \text{positive}, & \text{if } \underline{\text{filter}(X)} < \theta \\ \text{negative}, & \text{otherwise} \end{cases}$$

Image set

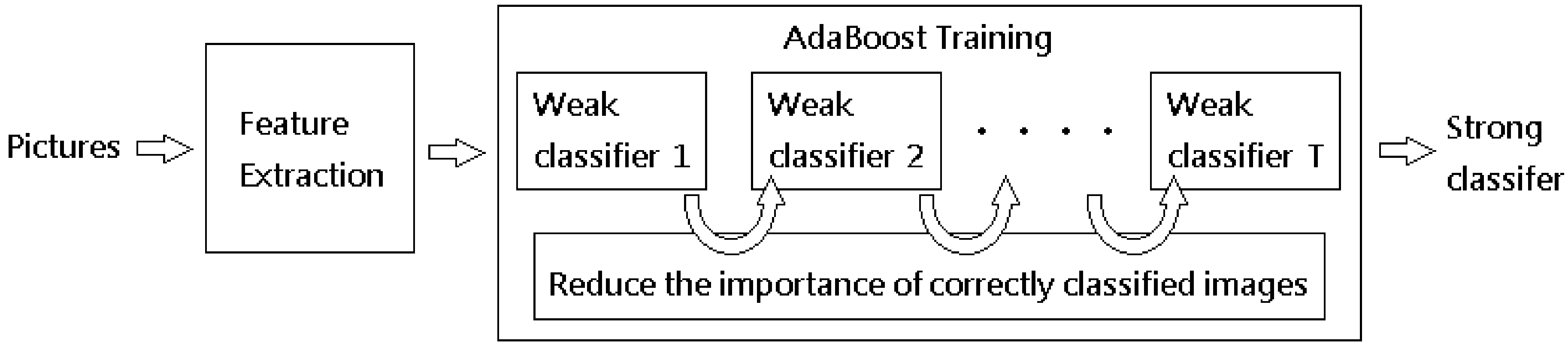
Weak Classifier: Consider —> Feature Value:
f(x) = Filter $\ast$ feature(Position, Size)

h(X) is weak classifier, not feature value



3. AdaBoost (2/2)

- Subsequent classifiers built are tweaked in favor of those instances misclassified by previous classifiers. [4]
- The goal is to minimize the number of features that need to be computed when given a new image, while still achieving high identification rates.



[4] AdaBoost - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/AdaBoost

3.1 Training Process Flowchart jj

Feature value = f(x)

1. Input:
Training image set *X*

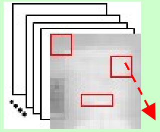
Face (24x24) *l* 張

Non-Face (24x24) *m* 張

$$y_i = \begin{cases} 1, & \text{postive/face} \\ 0, & \text{negative/non-face} \end{cases}$$

m>>l, n=m+l

2. Feature Extraction:
Using haar-like filter



feature

設每張 image 可
extract 出 *N* 個 feature value
共有*N**(*l*+*m*) =*N***n*個 feature value

Candidate threshold θ

3. AdaBoost Algorithm:

$$h_{i,j}(x_k) = \begin{cases} 1, & \text{if } p_{i,j} f_j(x_k) < p_{i,j}\theta_{i,j} \\ 0, & \text{otherwise} \end{cases}$$

$$p_{i,j} = \begin{cases} -1, & \text{if } \varepsilon_{i,j} > 0.5 \\ 1, & \text{otherwise} \end{cases}$$

3.0. Image weight initialization
$$w_i = \begin{cases} \frac{1}{2l}, & \text{for postive} \\ \frac{1}{2m}, & \text{for negative} \end{cases}, \quad i = 1, \ldots, n$$

t=0

$$\text{where } h_{i,j}(x_k) = \begin{cases} \text{positive, if filter}(x_k) < \theta \\ \text{negative, otherwise} \end{cases}$$

3.1. Normalize image weight
$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}, i = 1, \ldots, n$$

$i \in n$: *Total # of images*
$j \in N$: *Total # of features / image*

t=t+1

Summation of all weights having incorrect classifications

| |= 0 correct or 1 incorrect

3.2. Error calculation
$$\varepsilon_{i,j} = \sum_{k=1}^{n} w_{t,k} \left| h_{i,j}(x_k) - y_k \right|, i = 1, \ldots n, j = 1, \ldots N$$

label 0 or 1

3.3. Select a weak classifier h_t with the lowest error ε_t (= $\varepsilon_{i,j}$) and corresponding threshold $(=\theta_{i,j})$

One of feature value

How do you decide T?

t=T 個 weak classifiers?

3.4. Image weight adjusting
$$w_{t+1,i} = \begin{cases} w_{t,i}\beta_t, & \text{if } x_i \text{ is classified correctly} \\ w_{t,i}, & \text{otherwise} \end{cases} \quad \text{where } \beta_t = \frac{\varepsilon_t}{1-\varepsilon_t} = \frac{Error}{Correct} < 1.0$$

T 個 weak classifiers

h_t: filter, feature(Posit, Size)

$$\frac{\sum_{t=1}^{T} \alpha_t h_t(x)}{\sum_{t=1}^{T} \alpha_t} \geq \frac{1}{2}$$

4. Output:
A strong classifier
(24x24)

Weak classifier: filter, feature(Position, Size)

$$H_T(x) = \begin{cases} \text{positive/face}, & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ \text{negative/non-face}, & \text{otherwise} \end{cases}$$

$$\alpha_t = \log \frac{1}{\beta_t} = \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) = \log \frac{correct}{error}$$

Weak classifier weight

3.1.1 Training Process - Input

- Input:

  Training data set 以 ***X*** 表示 . 設有 ***l*** 張 positive image , m 張 negative image , 共 *n* (*n*=*l*+*m*) 張 image.

$$X = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}, \; y_i = \begin{cases} 1 \;, & \text{postive/face} \\ 0 \;, & \text{negative/non-face} \end{cases}$$

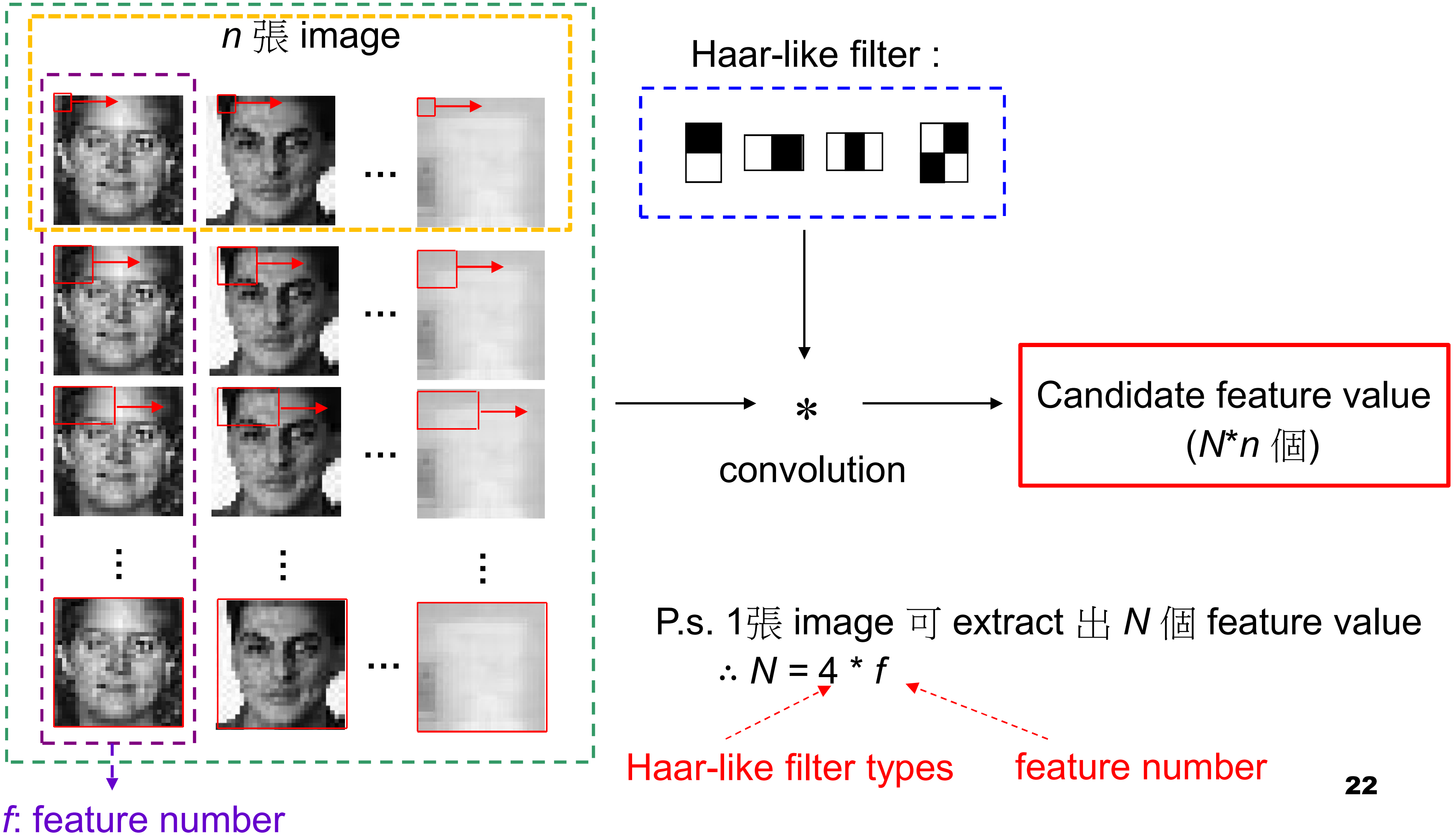
$\{$  $\}$

y_i 1 1 0 1 0 1

設每張 image 可以 extract 出 *N* 個 local feature , $f_j(x_i)$ 表示 image x_i 裡的第 *j* 個 local feature value , 共有 *N* * *n* 個 local feature value.

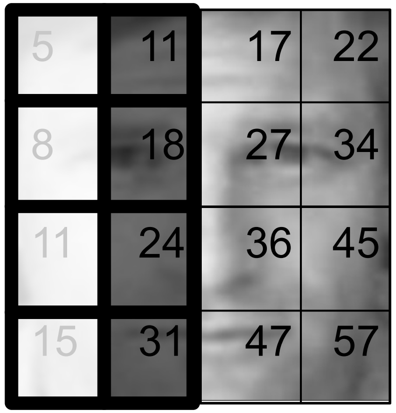
3.1.2 Training Process - Feature Extraction (1/2) JJ

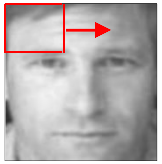
- Feature Extraction: Using haar-like filter



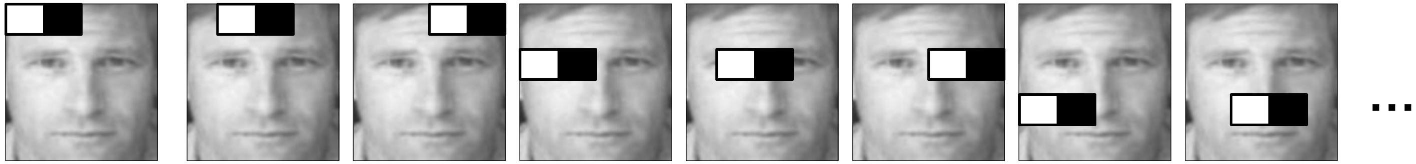
3.1.2 Training Process - Feature Extraction

Integral Image

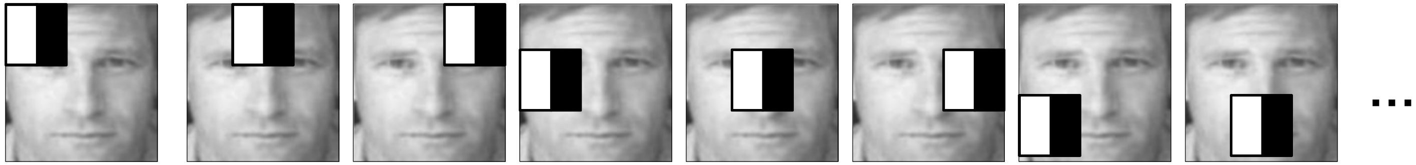




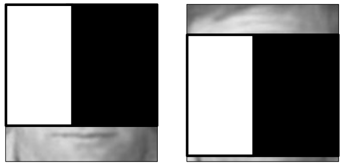
Position
(pixel by pixel shift)



Feature Value = -6 -6 -5 -10 -9 -7 -13 -12 …



Feature Value = 2 1 2 -1 1 2 1 -1 …



Feature Value = 3 3

… *f* 個 feature value

∵1種 filter 每種可能的大小、比例、位置可 extract 出 *f* 個 feature value，若有4種 filter，一張 image 可 extract 出 *N* 個 feature value。

∴ $N = 4 * f$

3.1.2 Training Process - Feature Extraction (2/2)

Define weak classifiers $h_{i,j}$:

Face

$$h_{i,j}(x_k) = \begin{cases} 1, & \text{if } p_{i,j} f_j(x_k) < p_{i,j}\theta_{i,j} \\ 0, & \text{otherwise} \end{cases}$$

Non-face

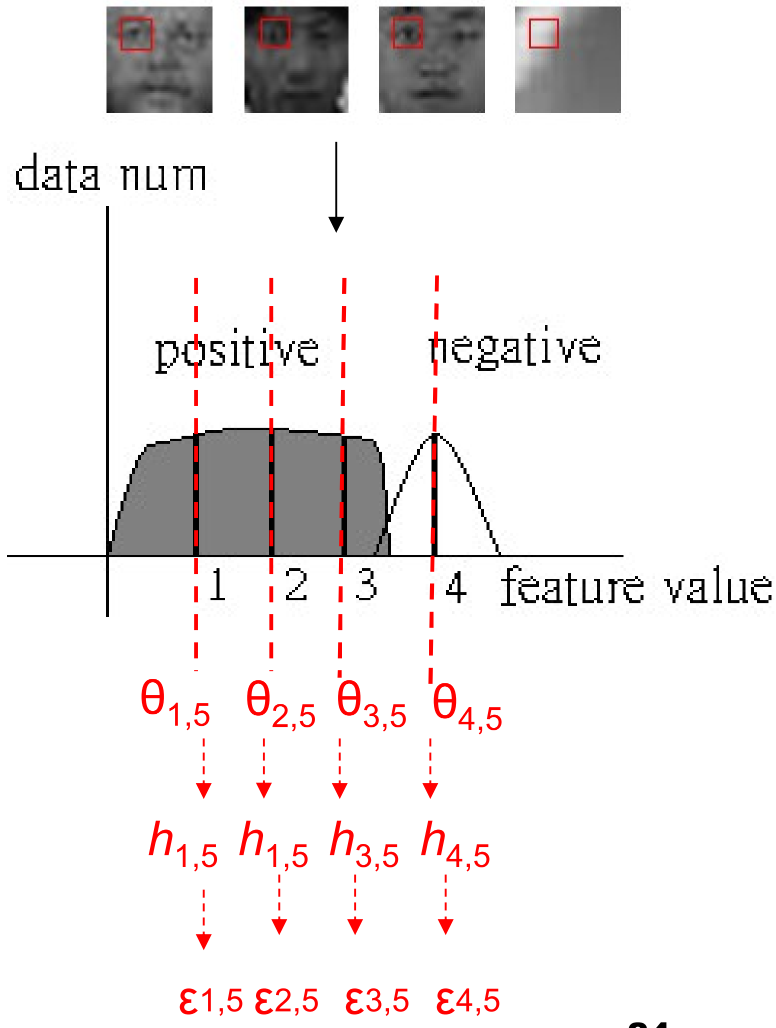
$\theta_{i,j}$: 即 image *i* 的第 *j* 個 local feature value

$f_j(x_k)$: 即 image *k* 的第 *j* 個 local feature value

Polarity: $$p_{i,j} = \begin{cases} -1, & \text{if } \varepsilon_{i,j} > 0.5 \\ 1, & \text{otherwise} \end{cases}$$

If total number of error images are the majority, then the sign p_{ij} for corresponding image (actually it is for each image) should change from -1 to +1.

Ex : 3 face & 1 non-face image extract by 5th feature



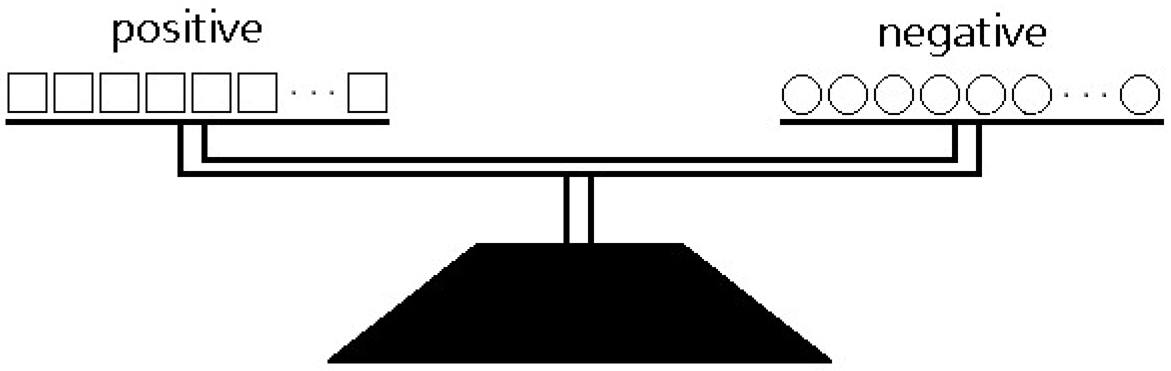
3.1.3 Training Process - AdaBoost Algorithm (1/5)

- AdaBoost Algorithm:

3-0. Image weight initialization :

$$w_i = \begin{cases} \dfrac{1}{2l} & \text{, for positive/face image} \\ \dfrac{1}{2m} & \text{, for negative/non-face image} \end{cases} \quad , \; i = 1, \ldots, n$$

l is the quantity of positive images.
m is the quantity of negative images.



3.1.3 Training Process - AdaBoost Algorithm (2/5)

Iterative: $t = 1, \ldots, T$ T : weak classifier number

3-1. Normalize image weight:

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}, i = 1, \ldots, n$$

Training data set X

3-2. Error calculation :

error rate

Candidate weak classifier: 0 or 1

$$\varepsilon_{i,j} = \sum_{k=1}^{n} w_{t,k} \left| h_{i,j}(x_k) - y_k \right|, i = 1, \ldots n, j = 1, \ldots N$$

positive or negative, 0 or 1

3-3. Select a weak classifier h_t with the lowest error rate ε_t .

3-4. Image weight adjusting :

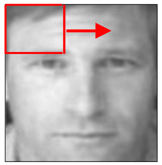
$$w_{t+1,i} = \begin{cases} w_{t,i}\beta_t & , \text{ if } x_i \text{ is classified correctly} \\ w_{t,i} & , \text{ otherwise} \end{cases}$$

$$where\ \beta_t = \frac{\varepsilon_t}{1-\varepsilon_t} = \frac{ErrorRate}{CorrectionRate} < 1.0$$

3.1.3 Training Process – select weak classifier (3/5)

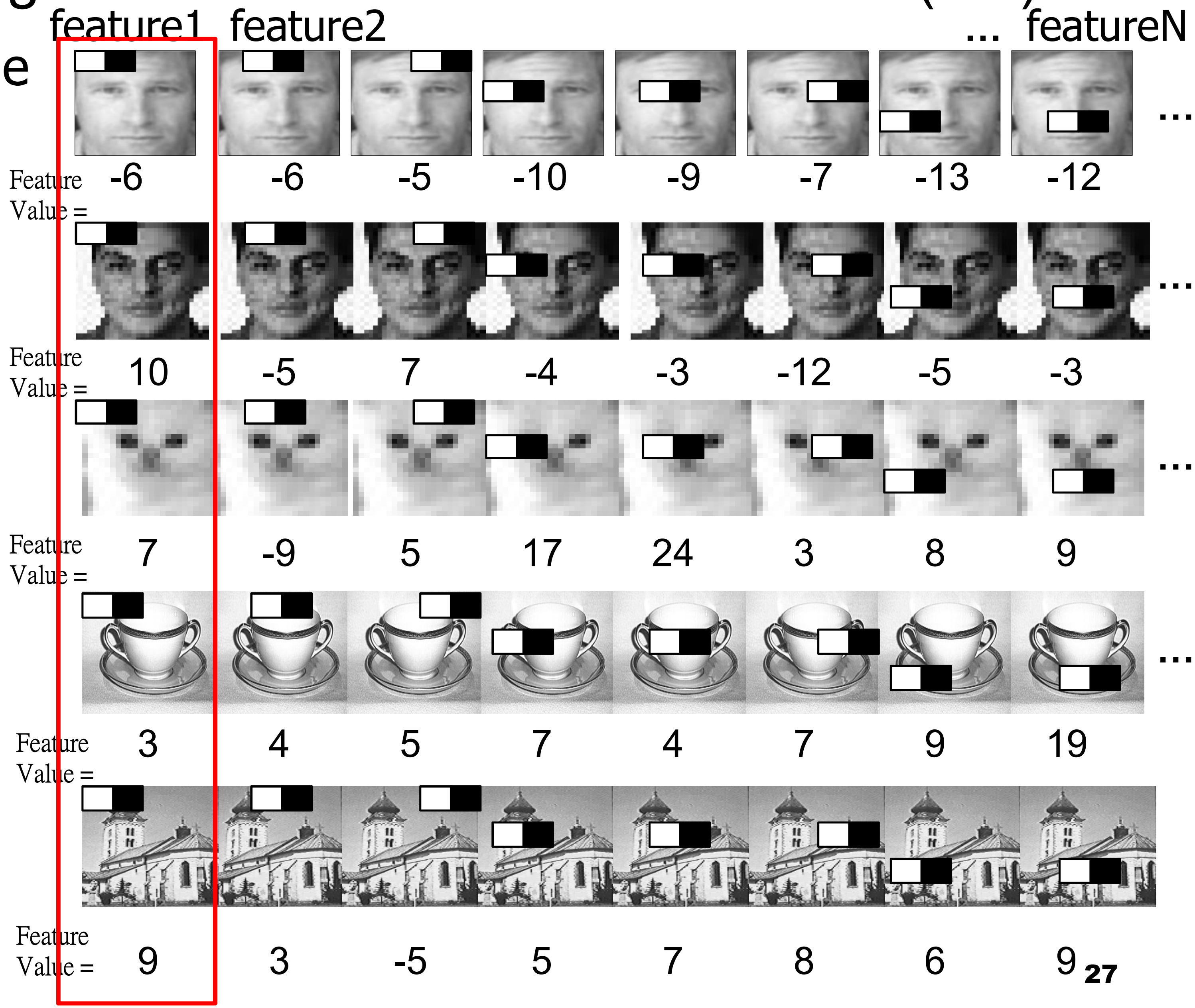
Integral Image

5	11	17	22
8	18	27	34
11	24	36	45
15	31	47	57



$$w_i = \begin{cases} \frac{1}{4}, & \text{face image} \\ \frac{1}{6}, & \text{non-face image} \end{cases}$$

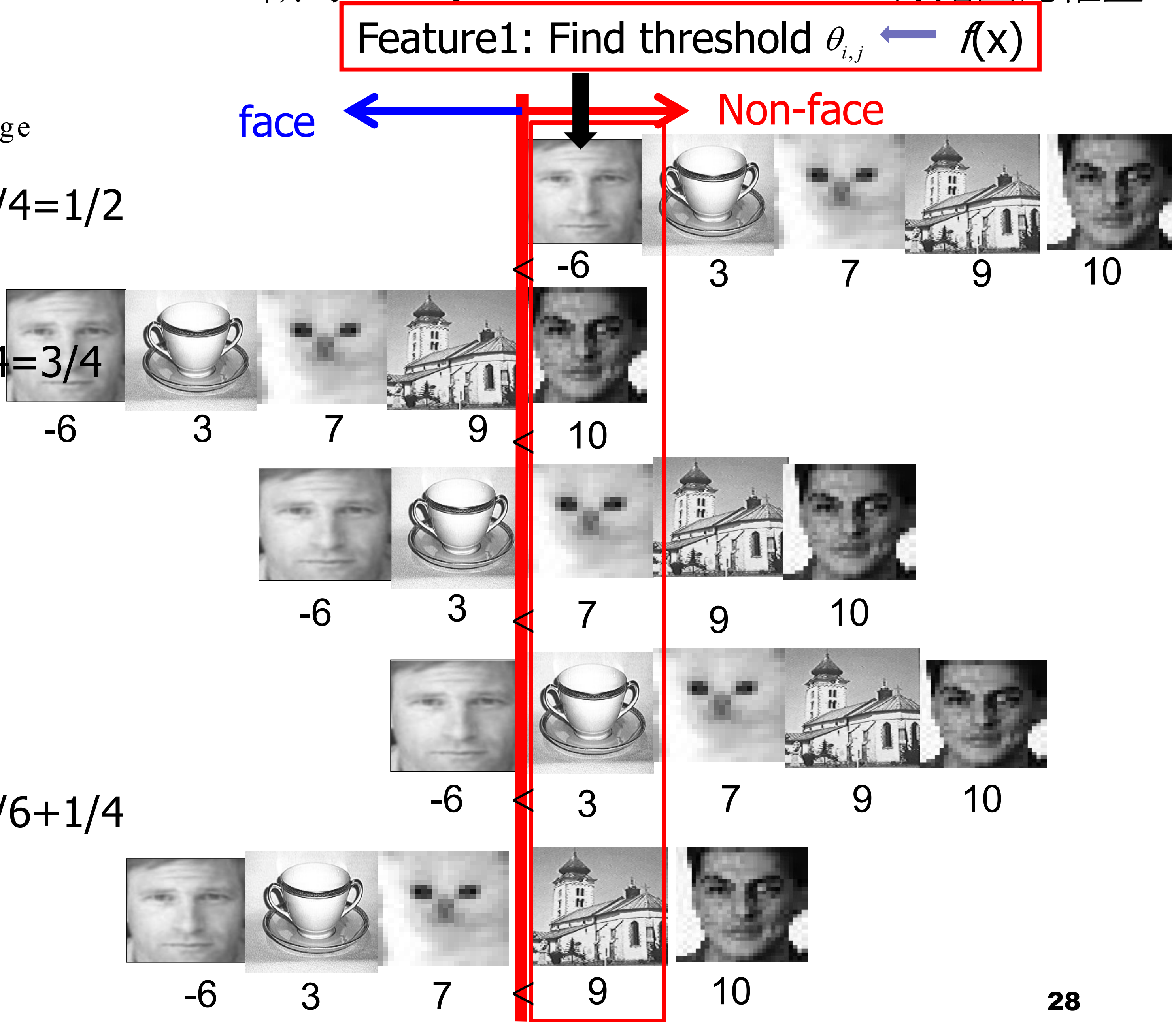
只有同樣 feature 的 feature value 可以互相比較，選擇最適合的 threshold



3.1.3 Training Process – select weak classifier(4/5) ??

選error rate最低的 feature value 做為filter的 threshold，error rate=分錯圖總權重

$$w_i = \begin{cases} \frac{1}{4}, & \text{face image} \\ \frac{1}{6}, & \text{non-face image} \end{cases}$$



Error rate: 1/4+1/4=1/2

??Error rate:
1/6+1/6+1/6+1/4=3/4
∵error rate > 0.5

Error rate:
1/6+1/4=5/12

Error rate: 1/4

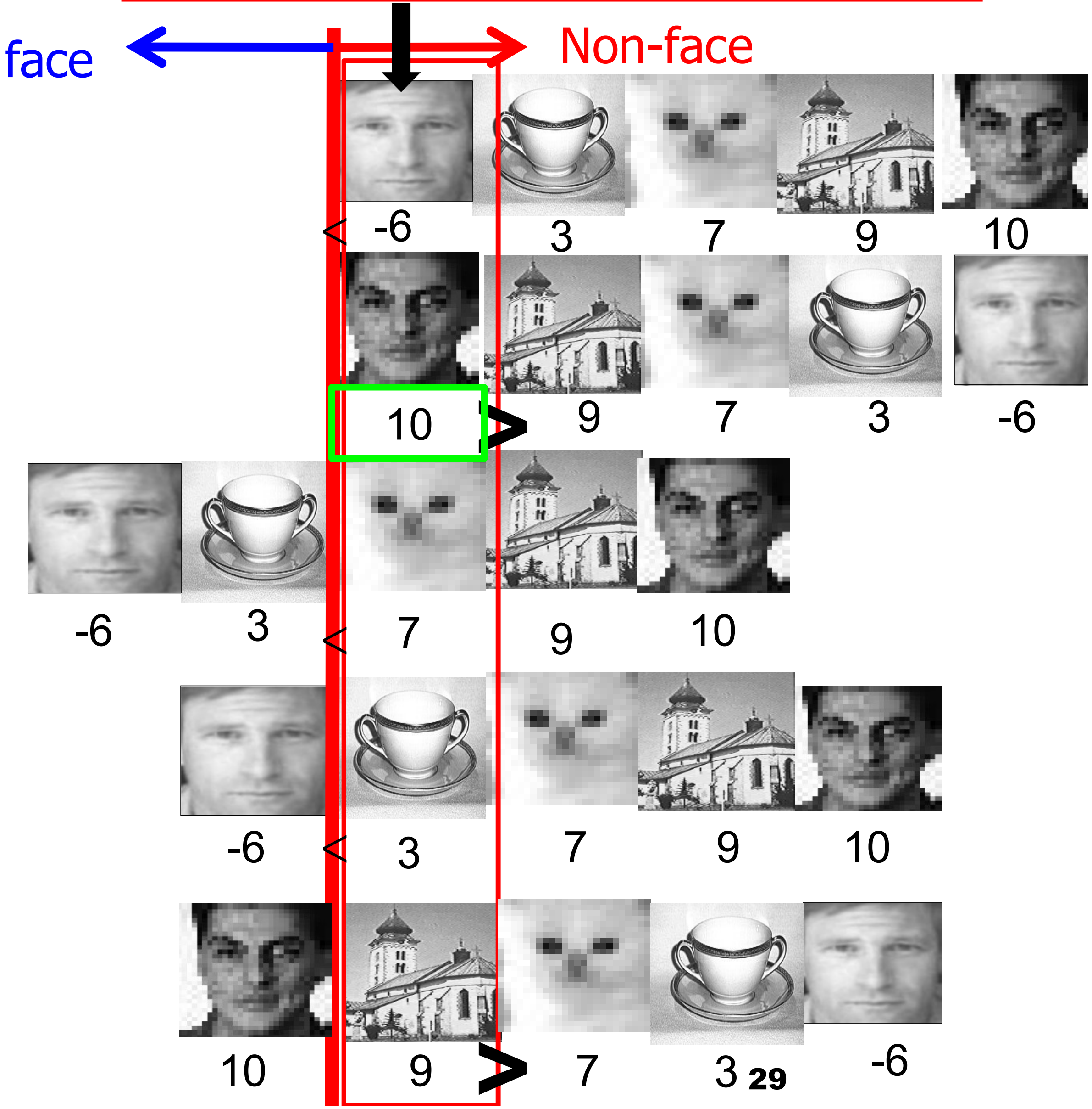
Error rate: 1/6+1/6+1/4
=7/12
∵error rate > 0.5

3.1.3 Training Process – select weak classifier (4/5) ??

選error rate最低的 feature value 做為filter的 threshold，error rate=分錯圖總權重

$$w_i = \begin{cases} \frac{1}{4}, & \text{face image} \\ \frac{1}{6}, & \text{non-face image} \end{cases}$$

Feature1: Find threshold $\theta_{i,j}$ ⟵ $f(x)$



Error rate: 1/4+1/4=1/2

$\because$ error rate > 0.5

Min Error rate: 1/4

with threshold value $\theta_{i,j}$ = 10

Error rate:
1/6+1/4=5/12

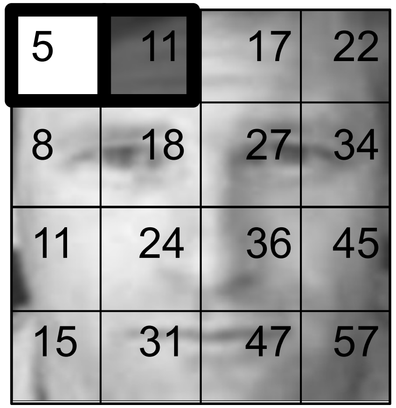
Error rate: 1/4

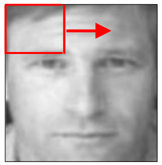
$\because$ error rate > 0.5

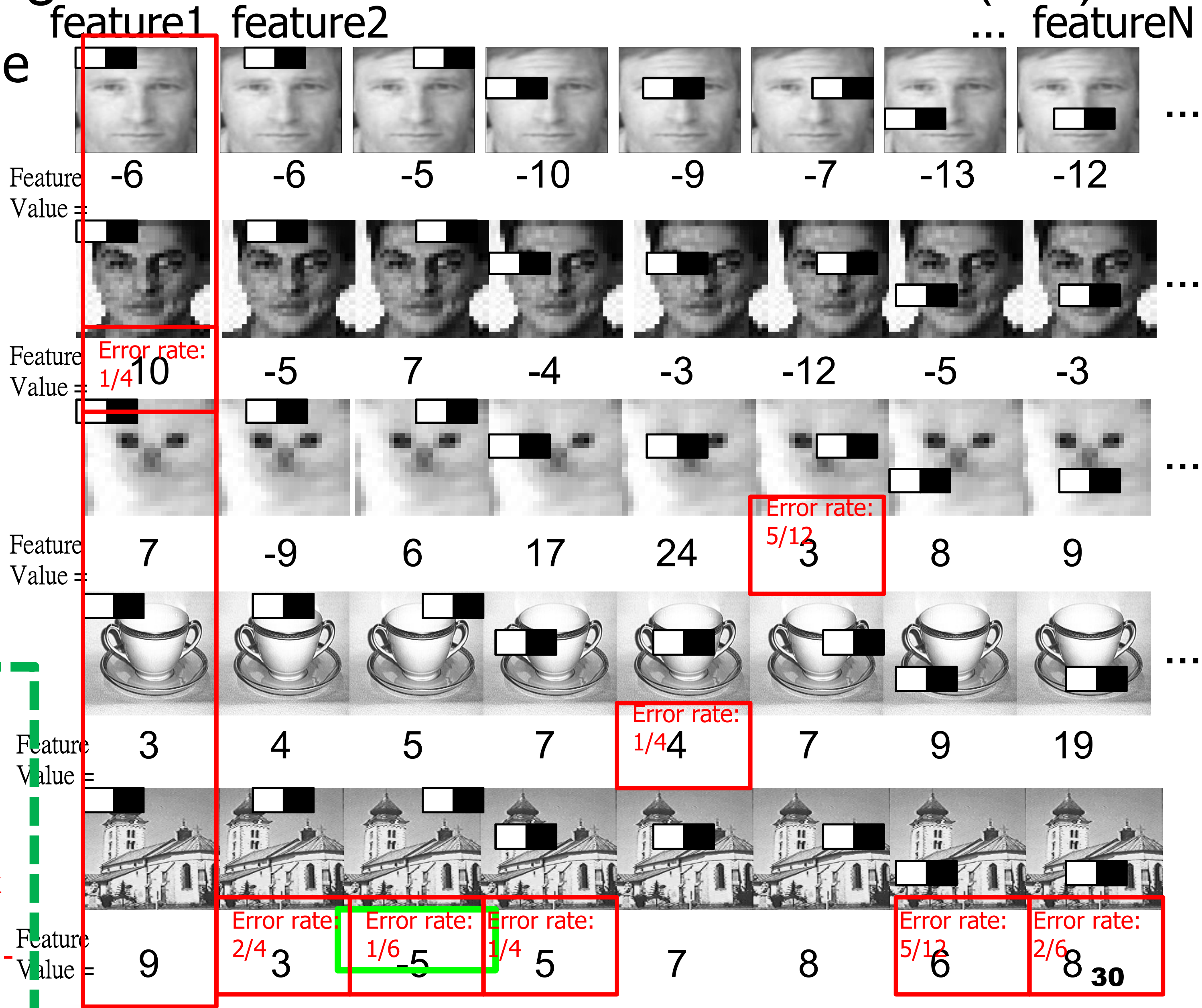
Error rate: 1/6+1/4
=5/12

3.1.3 Training Process – select a weak classifier(5/5) JJ

Integral Image







- 每組feature產生一個threshold $\theta_{i,j}$，各feature的threshold間再比較error rate，error rate最小的 (ex. 1/6) 做為該次的weak classifier with threshold $\theta_{i,j}$ (ex, = -5)

3.1.4 Training Process – Output (1/3)

- Output:

The finally strong classifier:

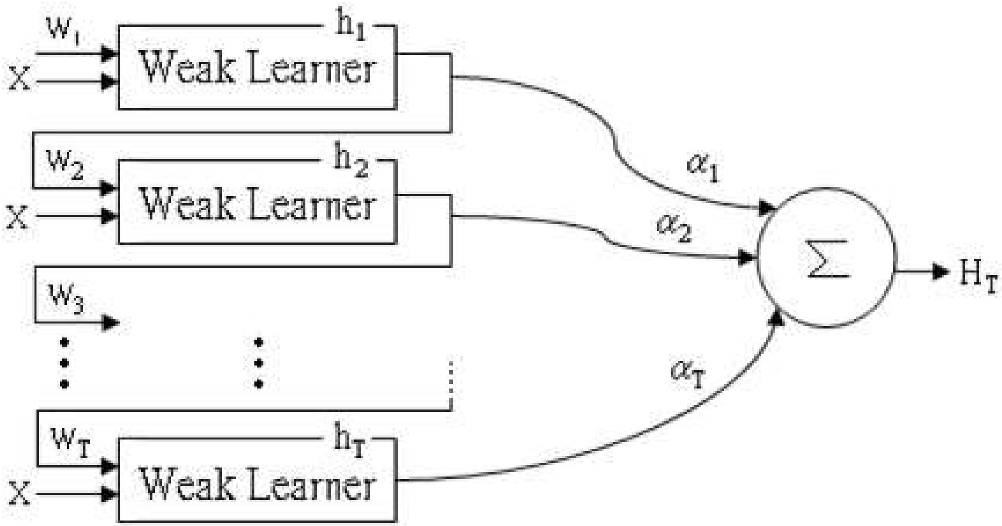
$$H_T(x) = \begin{cases} \text{positive/face} \ , & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \boxed{\frac{1}{2}} \sum_{t=1}^{T} \alpha_t \\ \text{negative/non-face} \ , & \text{otherwise} \end{cases}$$

$$\frac{\sum_{t=1}^{T} \alpha_t h_t(x)}{\sum_{t=1}^{T} \alpha_t} \geq \frac{1}{2}$$

strong classifier threshold

Weak classifier weight

where $$\alpha_t = \log \frac{1}{\beta_t} = \log(\frac{1-\varepsilon_t}{\varepsilon_t}) = \log(\frac{CorrectionRate}{ErrorRate})$$

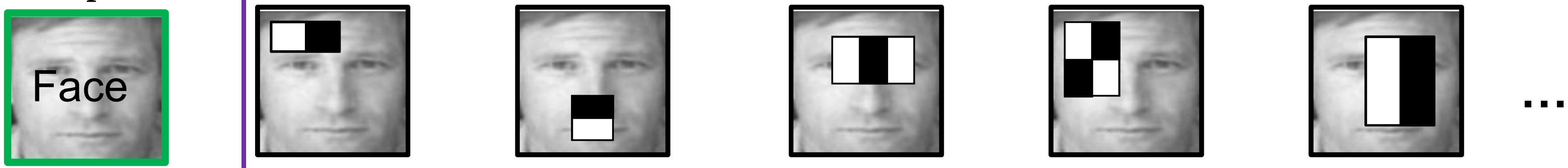


3.1.4 Training Process – Output (2/3)

JJ

Strong classifier: 分類人臉與非人臉

Input



Threshold value $\theta_{i,j}$ = 3 > -2   7 > -6   5 > 3   15 < 23   -2 < 5

Weak classifier weight= α_1 α_2 α_3 α_4 α_5

Input image feature value ，並假設 α 均為1

$$H_T(x) = \begin{cases} \text{positive/face} \ , & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ \text{negative/non-face} \ , & \text{otherwise} \end{cases}$$

Strong classifier: 若feature 的 feature value 小於weak classifier的threshold 則h(x) =1， 加總 α_t

$$\frac{\sum_{t=1}^{T} \alpha_t h_t(x)}{\sum_{t=1}^{T} \alpha_t} \geq \frac{1}{2}$$

左式表示:

若定 strong classifier 的 threshold 為1/2，則input image 在weak classifiers選定的 weak classifier weight的總和佔超過總weight有一半，即判定為人臉。

So this case $H_T(x) = \frac{1+1+1+0+0}{1+1+1+1+1}$ =0.6 > 0.5

3.1.4 Training Process – Output Analysis (3/3)

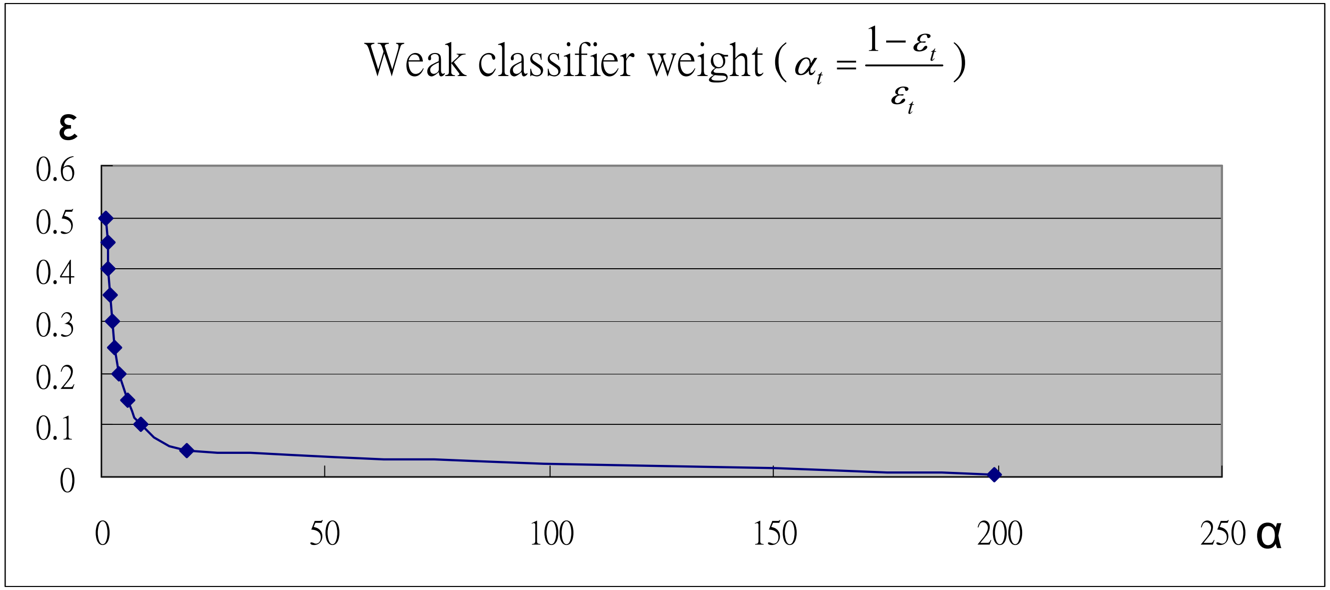


Fig. B

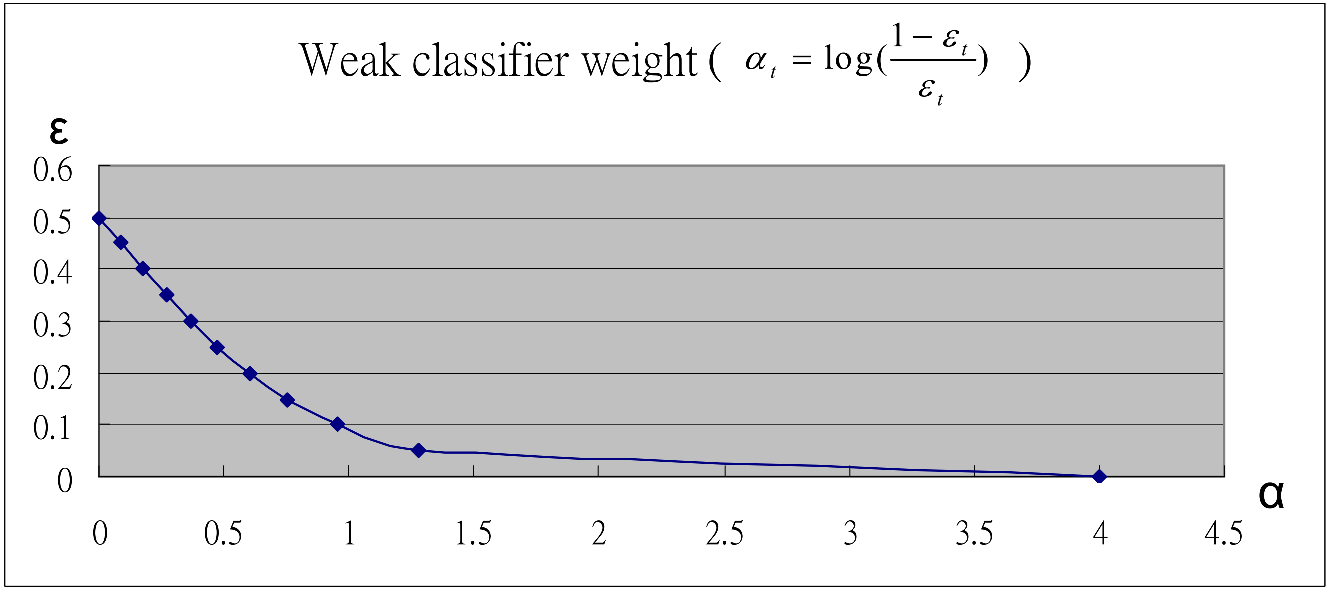


Fig. C

如 Fig. B，當ε (error rate)在 0 ~ 0.1 區間內與其他如 0.1 ~ 0.5 區間內即使ε有相同的變化量，所對應到的α (weak classifier weight)變化量差異也相當大，如此一來當ε越趨近於 0 時，即使ε只有些微改變，在 strong classifier 中其比重也會劇烈加大。因此，取 log 是為了縮小 weight 彼此間差距，使 strong classifier 中的各個 weak classifiers 均佔有一定比重。

ε	$\frac{1-\varepsilon}{\varepsilon}$	$\log(\frac{1-\varepsilon}{\varepsilon})$
0.001	999	2.99
0.005	199	2.29
0.101	8.9	0.94
0.105	8.52	0.93

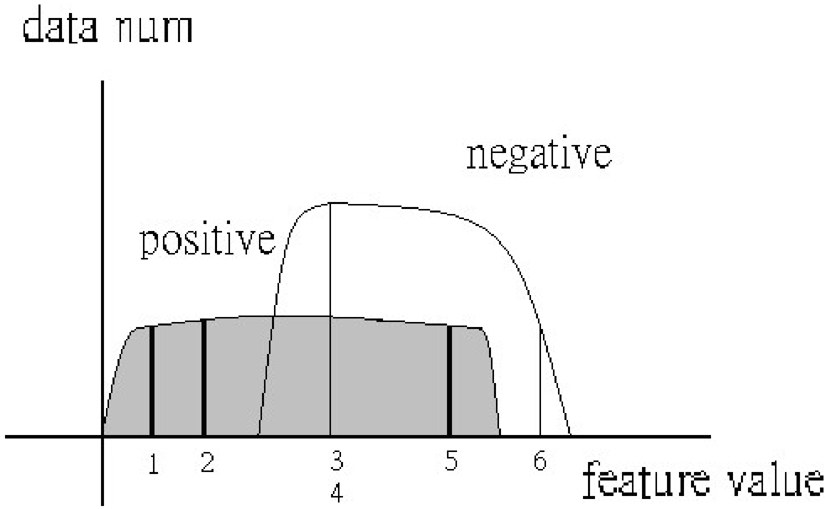
800 0.7

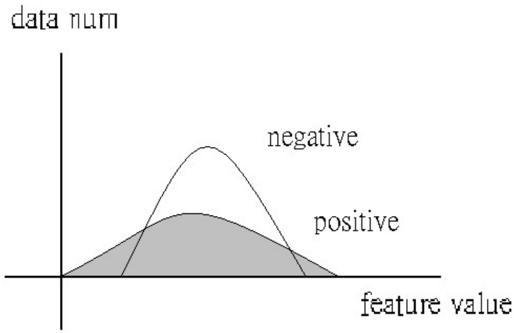
0.38 0.01

3.1.5 AdaBoost Algorithm – Image Weight Adjusting Analysis: Example (1/2)

Ex.:設有 6 筆 training image data ,3 筆為 positive (1、2、5),
3 筆 negative (3、4、6), 每張 image data 均有一個 weight $(w_1 \sim w_6)$

t =1 時(第一個weak classifier)





…

	feature1	feature2	…	featureN
以 1 當 classifier :	$\varepsilon_{1,1} = w_1 + w_2 + w_5$	$\varepsilon_{2,1}$	…	$\varepsilon_{N,1}$
以 2 當 classifier :	$\varepsilon_{1,2} = w_2 + w_5$	$\varepsilon_{2,2}$	…	$\varepsilon_{N,2}$
以 3 4 當 classifier :	$\varepsilon_{1,3} = \varepsilon_{1,4} = w_5$	.	…	.
以 5 當 classifier :	$\varepsilon_{1,5} = w_3 + w_4 + w_5$	.	…	.
以 6 當 classifier :	$\varepsilon_{1,6} = w_3 + w_4$	$\varepsilon_{2,6}$	…	$\varepsilon_{N,6}$

3.1.5 AdaBoost Algorithm – Image Weight Adjusting Analysis: Example (2/2)

If $\varepsilon_{1,3}$ 取最小，則 $\beta = \dfrac{\varepsilon_{1,3}}{1-\varepsilon_{1,3}} = \dfrac{0.167}{1-0.167} = 0.2$ 　　t =1 時

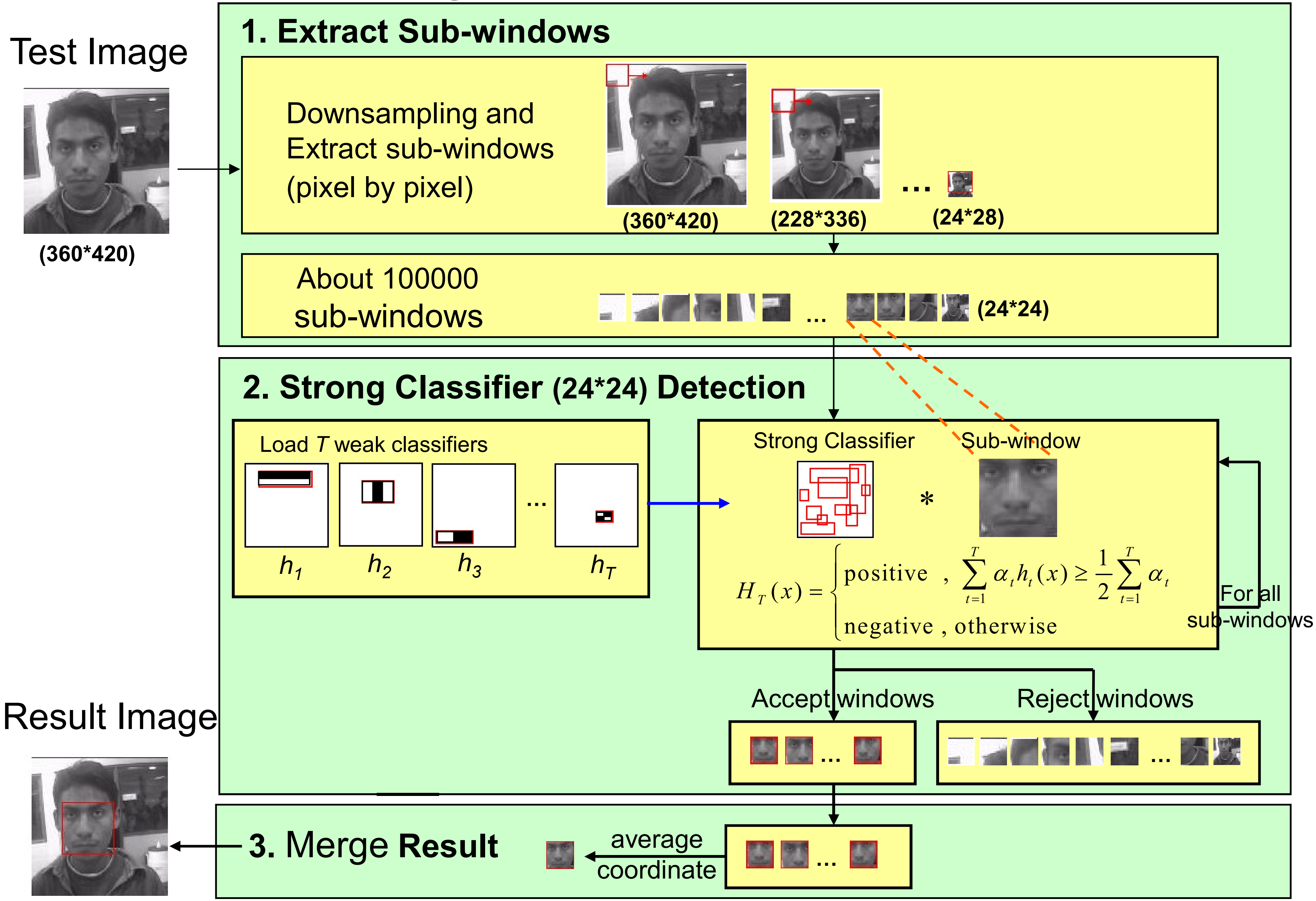
	W_1	W_2	W_3	W_4	W_5	W_6
初始值	0.167	0.167	0.167	0.167	0.167	0.167
經分類後	O	O	O	O	X	O
Update w_{t+1}	0.167*0.2	0.167*0.2	0.167*0.2	0.167*0.2	0.167	0.167*0.2
Normalize	0.1	0.1	0.1	0.1	0.5	0.1
Weight 變化	↘	↘	↘	↘	↗	↘

$$w_{t+1,i} = \begin{cases} w_{t,i}\beta_t & \text{, if } x_i \text{ is classified correctly} \\ w_{t,i} & \text{, otherwise} \end{cases}$$

That is the marginal problem!!

每一輪都將分對的 image 調低其weight ，經過 Normalize 後，分錯的 image的 weight 會相對提高，如此一來，常分錯的 image 就會擁有較高 weight。如果一張 image 擁有較高 weight 表示在進行分類評估時，會著重在此 image。

3.2 Testing Process - Flowchart



4. The Attentional Cascade (1/5)

- Advantage: Reducing testing computation time.
- Method: Cascade stages. Each stage corresponds to one strong classifier.
- Idea: Reject as many negatives as possible at the earliest stage. More complex classifiers were then used in later stages.
- The detection process is that of a degenerate decision tree, so called "cascade".

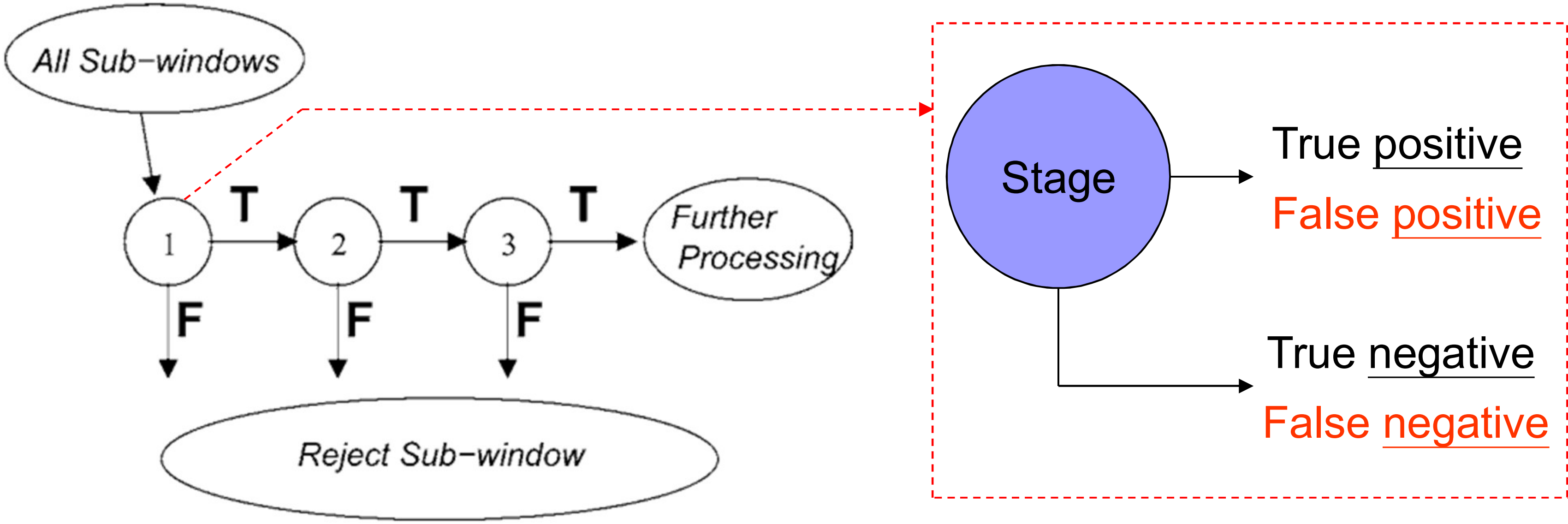


Figure 4: Cascade Structure

4. The Attentional Cascade (3/5)

- True positive rates (detection rates):

  將 positive 判斷為 positive 機率

  $$\frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} = \frac{Face}{AllFace}$$

- False positive rates (FP): (False Alarm)

  將 negative 判斷為 positive 機率

  $$\frac{\text{False Positive}}{\text{False Positive} + \text{True Negative}} = \frac{1 - NonFace}{AllNonFace}$$

- True negative rates:

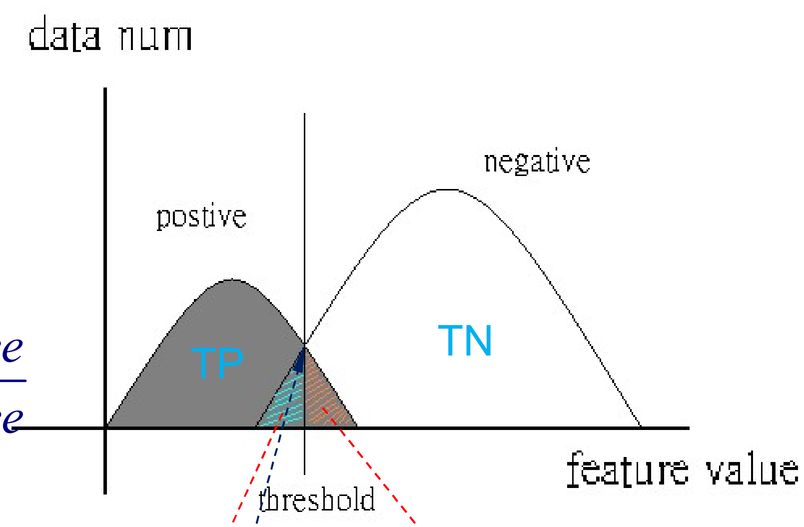
  將 negative 判斷為 negative 機率

  $$\frac{\text{True Negative}}{\text{False Positive} + \text{True Negative}} = \frac{NonFace}{AllNonFace}$$

- False negative rates (FN):

  將 positive 判斷為 negative 機率

  $$\frac{\text{False Negative}}{\text{True Positive} + \text{False Negative}} = \frac{1 - Face}{AllFace}$$



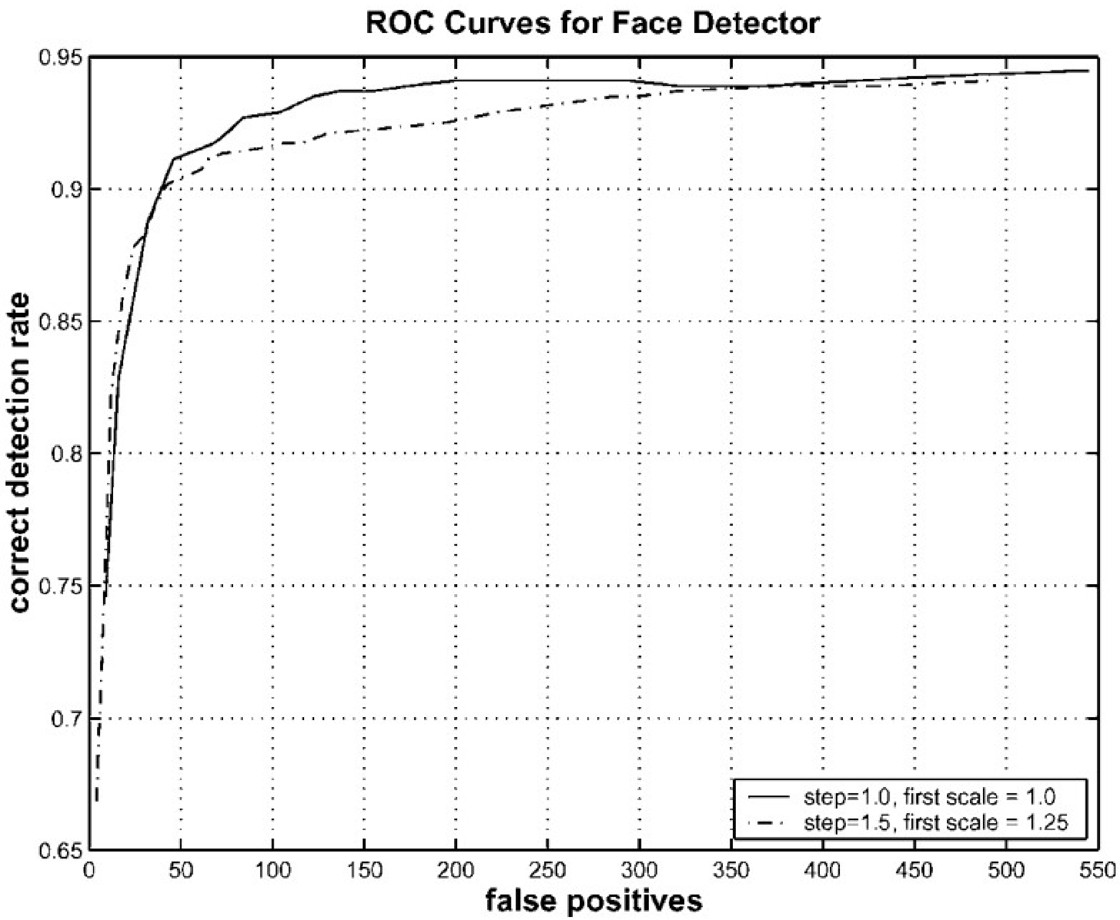
FP + FN => Error Rate

- EER (Equal Error Rate):
  - Both positive and negative errors are equal.
  - The lower the EER, the more accurate is considered to be.[5]

[5] Biometrics - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Biometrics

4. The Attentional Cascade (4/5)

- ROC Curve: Detection Rate Vs. False Positive (Numbers)



- Higher the detection rate, more false positive will happen.

False Positive = False Alarm

The number of false positive were detected.

4. The Attentional Cascade (4/5)

- Training a cascade of classifiers:
  - Involves two types of tradeoffs :
    1. Higher detection rates
    2. Lower false positive rates
  - More features will achieve higher detection rates and lower false positive rates. But classifiers require more time to compute.
  - Define an optimization framework:
    1. The number of stages (strong classifiers)
    2. The number of features (weak classifier: filter, feature(Posit., Size) in each stage
    3. The strong classifier $\boxed{\text{threshold}}$ in each stage

$$H_T(x) = \begin{cases} \text{positive/face} , & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \boxed{\frac{1}{2}} \sum_{t=1}^{T} \alpha_t \\ \text{negative/non-face} , & \text{otherwise} \end{cases}$$

4. The Attentional Cascade - Algorithm (1/4)

- User selects values for f, the maximum acceptable false positive rate per layer and d, the minimum acceptable detection rate per layer.
- User selects target overall false positive rate, F_{target}.
- P = set of positive examples
- N = set of negative examples
- $F_0 = 1.0$; $D_0 = 1.0$
- $i = 0$
- while $F_i > F_{target}$
  - $i \leftarrow i + 1$
  - $n_i = 0$; $F_i = F_{i-1}$
  - while $F_i > f \times F_{i-1}$
    - $n_i \leftarrow n_i + 1$
    - Use P and N to train a classifier with n_i features using AdaBoost
    - Evaluate current cascaded classifier on validation set to determine F_i and D_i.
    - Decrease threshold for the ith classifier until the current cascaded classifier has a detection rate of at least $d \times D_{i-1}$ (this also affects F_i)
  - $N \leftarrow \emptyset$
  - If $F_i > F_{target}$ then evaluate the current cascaded detector on the set of non-face images and put any false dectections into the set N

4. The Attentional Cascade - Algorithm (2/4)

- *f* : Maximum acceptable false positive rate. (最大 negative 辨識成 positive 錯誤百分比)
- *d*: Minimum acceptable detection rate. (最小辨識出 positive 的百分比)
- F_{target}: Target overall false positive rate. (最後可容許的 false positive rate)

- **Initial value:**

  P: Total positive images

  N: Total negative images

  f = 0.5

  d = 0.9999

  $F_{target} = 10^{-6}$

  $F_0 = 1.0$ 初始 False positive rate.

  $D_0 = 1.0$ 初始 Detection rate.

  Threshold = 0.5 AdaBoost threshold

  Threshold_EPS = 10^{-4} Threshold adjust weight

  i = 0 The number of cascade stage

4. The Attentional Cascade - Algorithm (3/4)

- **Iterative:**

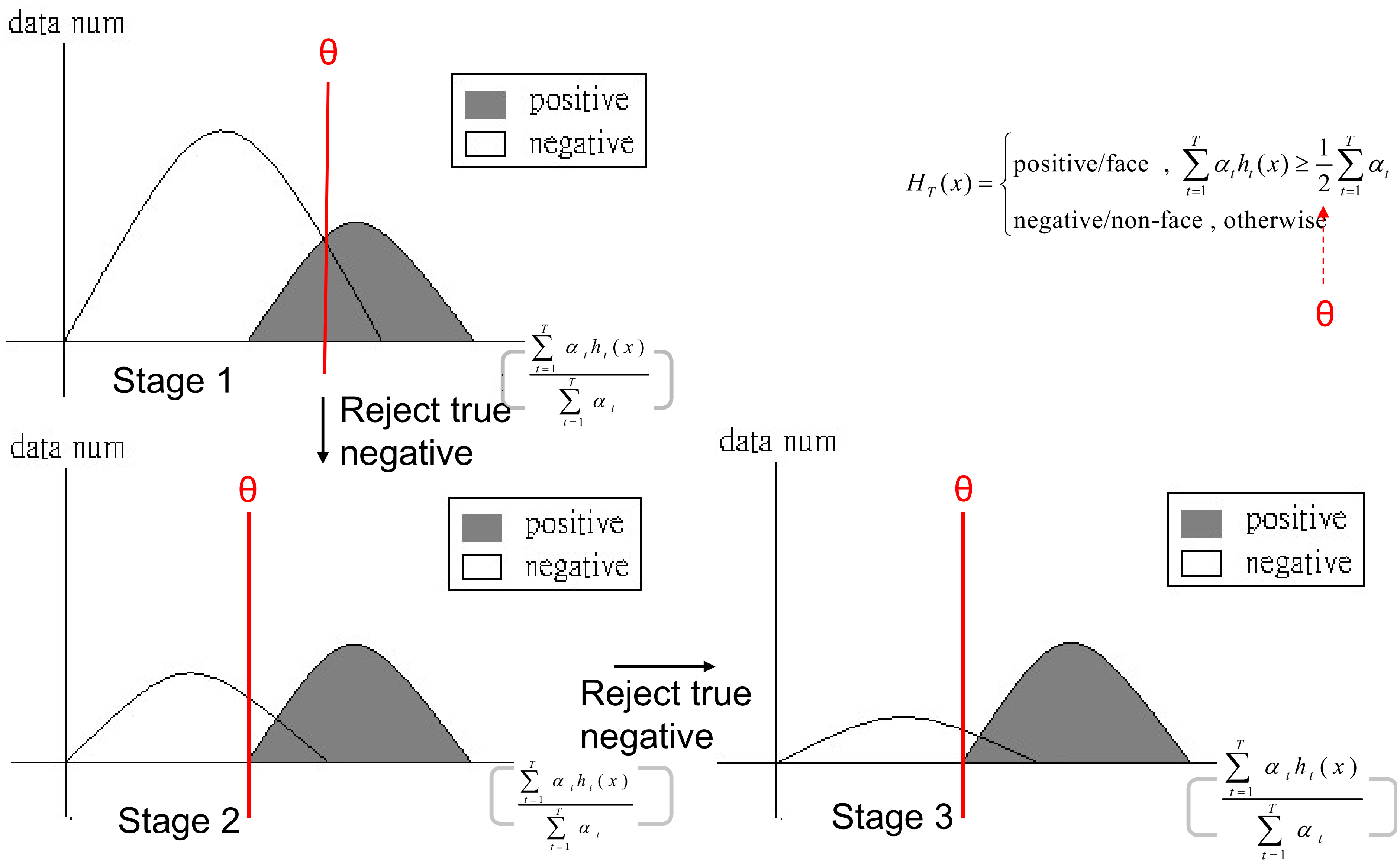
While($F_i > F_{target}$)
{
i=i+1 ← Add Stage
$n_i = 0, F_i = F_{i-1}$
While($F_i > f * F_{i-1}$)
{
$n_i = n_i + 1$ ← Add Feature
(F_i, D_i) = AdaBoost(*P*,*N*, n_i) ← Get New D_i , F_i ↓
While($D_i \le d * D_{i-1}$)
{
Threshold = *Threshold* – *Threshold_EPS*
D_i = Re-computer current strong classifier detection rate with Threshold (this also affects F_i)
}
}
Threshold ↓ , 則 D_i ↑ , F_i ↑
If($F_i > F_{target}$)
N = false detections with current cascaded detector on the N
}
$N = F_i * N$

f : Maximum acceptable false positive rate
d : Minimum acceptable detection rate
F_{target} : Target overall false positive rate
P : Total positive images
N : Total negative images
i : The number of cascade stage
F_i : False positive rate at ith stage
D_i : Detection rate at ith stage
n_i : The number of features at ith stage

4. The Attentional Cascade (4/4)

$$H_T(x) = \begin{cases} \text{positive/face}, & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ \text{negative/non-face}, & \text{otherwise} \end{cases}$$

θ



5. Experimental Results (1/4)

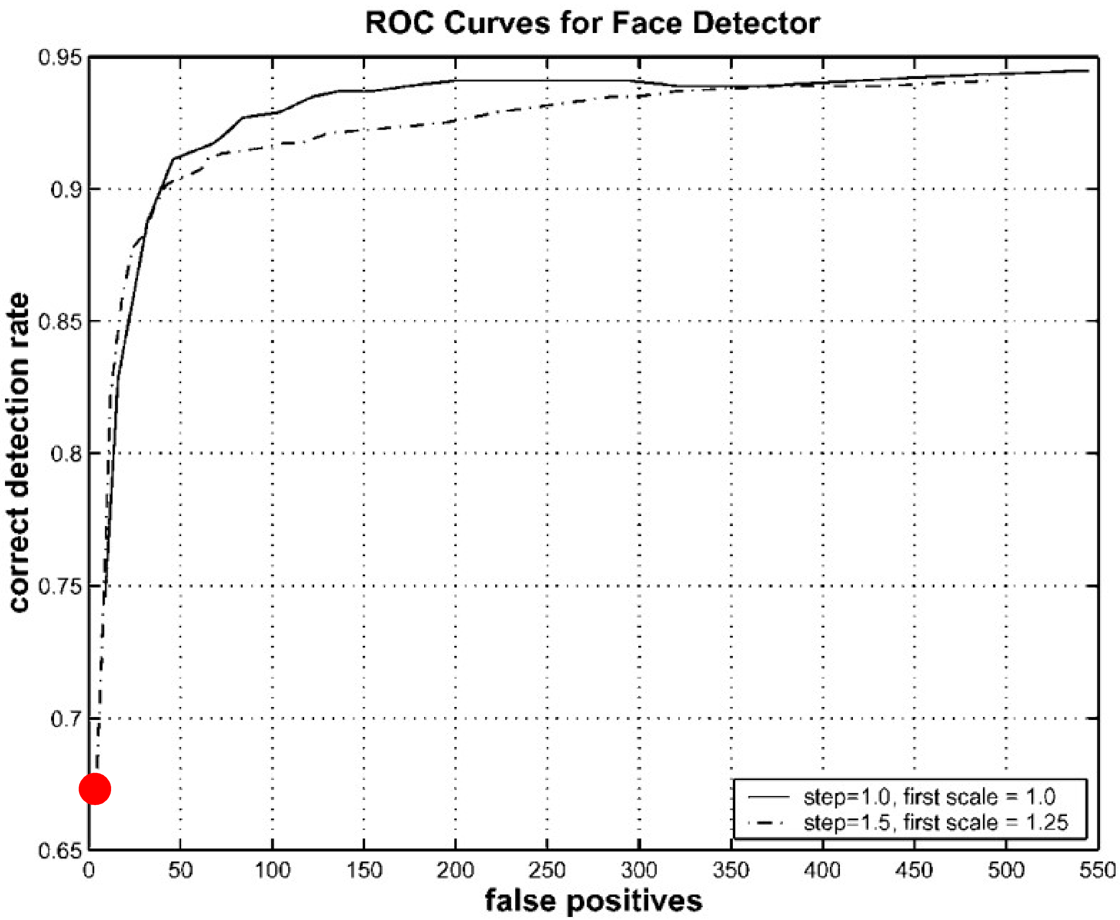
- Face training set:
  - Extracted from the world wide web.
  - Use face and non-face training images.
  - Consisted of 4916 hand labeled faces.
  - Scaled and aligned to base resolution of 24 by 24 pixels.
- The non-face sub-windows come from 9544 images which were manually inspected and found to not contain any faces.

Fig. 5: Example of frontal upright face images used for training

5. Experimental Results (2/4)

- In the cascade training:
  - Use 4916 training faces.
  - Use 10,000 non-face sub-windows.
  - Use the AdaBoost training procedure.
- Evaluated on the MIT+CMU test set:
  - An average of 10 features out of a stage are evaluated per sub-window.
  - This is possible because a large majority of sub-windows are rejected by the first or second stage in the cascade.
  - On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds .

5. Experimental Results (3/4)



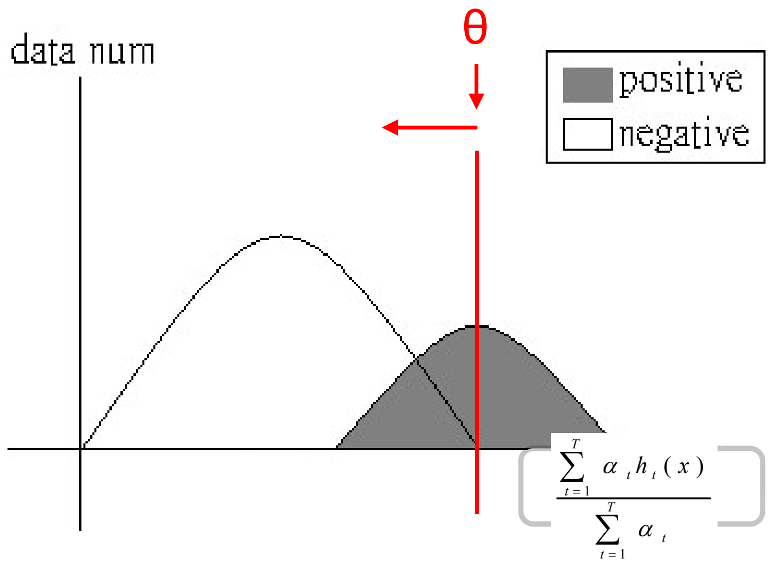


Fig. 6: Create the ROC curve (**Receiver Operating Characteristic**) the threshold of the final stage classifier is adjusted from $-\infty$ to ∞ .

5. Experimental Results (4/4)

	False detections							
Detector	10	31	50	65	78	95	167	422
Viola-Jones	76.1%	88.4%	91.4%	92.0%	92.1%	92.9%	93.9%	94.1%
Viola-Jones (voting)	81.1%	89.7%	92.1%	93.1%	93.1%	93.2%	93.7%	–
Rowley-Baluja-Kanade	83.2%	86.0%	–	–	–	89.2%	90.1%	89.9%
Schneiderman-Kanade	–	–	–	94.4%	–	–	–	–
Roth-Yang-Ahuja	–	–	–	–	(94.8%)	–	–	–

Table 2: Detection rates for various numbers of false positives on the MIT+CMU test set containing 130 images and 507 faces.

- Detection rate is lower than other approach.
- 15 times faster than Rowley-Baluja-Kanade detector (Rowley et al., 1998).
- 600 times faster than the Schneiderman-Kanade detector (Schneiderman and Kanade, 2000).

6. Conclusion

- We have presented an approach for object detection.
  1. Minimize computation time.
  2. Achieve high detection accuracy.
- The approach was used to construct a face detection system which is approximately 15 times faster than any previous approach (Rowley-Baluja-Kanade detector (Rowley et al., 1998)).

7. Reference (1/3)

[1] P. Viola and M. Jones, “Rapid Object Detection Using A Boosted Cascade of Simple Features”, Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol.1, pp. 511-518, 2001

[2] P. Viola and M. Jones, “Robust Real-time Object Detection”, IEEE International Journal of Computer Vision, vol.57, no.2, pp.137-154, 2004.

[3] A.S.S.Mohamed, Y. Weng, S. S Ipson, and J. Jiang, ”Face Detection based on Skin Color in Image by Neural Networks”, ICIAS 2007, pp. 779-783, 2007.

[4] AdaBoost - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/AdaBoost

[5] Biometrics - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Biometrics

7. Reference (2/3) JJ

References

Amit, Y. and Geman, D. 1999. A computational model for visual selection. *Neural Computation*, 11:1691–1715.

Crow, F. 1984. Summed-area tables for texture mapping. In *Proceedings of SIGGRAPH*, 18(3):207–212.

Fleuret, F. and Geman, D. 2001. Coarse-to-fine face detection. *Int. J. Computer Vision*, 41:85–107.

Freeman, W.T. and Adelson, E.H. 1991. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906.

Freund, Y. and Schapire, R.E. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Eurocolt 95*, Springer-Verlag, pp. 23–37.

Greenspan, H., Belongie, S., Gooodman, R., Perona, P., Rakshit, S., and Anderson, C. 1994. Overcomplete steerable pyramid filters and rotation invariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Itti, L., Koch, C., and Niebur, E. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Patt. Anal. Mach. Intell.*, 20(11):1254-–1259.

John, G., Kohavi, R., and Pfeger, K. 1994. Irrelevant features and the subset selection problem. In *Machine Learning Conference Proceedings*.

Osuna, E., Freund, R., and Girosi, F. 1997a. Training support vector machines: An application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Osuna, E., Freund, R., and Girosi, F. 1997b. Training support vector machines: an application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Papageorgiou, C., Oren, M., and Poggio, T. 1998. A general framework for object detection. In *International Conference on Computer Vision*.

Quinlan, J. 1986. Induction of decision trees. *Machine Learning*, 1:81–106.

Roth, D., Yang, M., and Ahuja, N. 2000. A snowbased face detector. In *Neural Information Processing* 12.

Rowley, H., Baluja, S., and Kanade, T. 1998. Neural network-based face detection. *IEEE Patt. Anal. Mach. Intell.*, 20:22–38.

Schapire, R.E., Freund, Y., Bartlett, P., and Lee, W.S. 1997. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of the Fourteenth International Conference on Machine Learning*.

Schapire, R.E., Freund, Y., Bartlett, P., and Lee, W.S. 1998. Boosting the margin: A new explanation for the effectiveness of voting methods. *Ann. Stat.*, 26(5):1651–1686.

7. Reference (3/3) JJ

- Schneiderman, H. and Kanade, T. 2000. A statistical method for 3D object detection applied to faces and cars. In *International Conference on Computer Vision*.
- Simard, P.Y., Bottou, L., Haffner, P., and LeCun, Y. (1999). Boxlets: A fast convolution algorithm for signal processing and neural networks. In M. Kearns, S. Solla, and D. Cohn (Eds.), *Advances in Neural Information Processing Systems*, vol. 11, pp. 571–577.
- Sung, K. and Poggio, T. 1998. Example-based learning for view-based face detection. *IEEE Patt. Anal. Mach. Intell.*, 20:39–51.
- Tieu, K. and Viola, P. 2000. Boosting image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Tsotsos, J., Culhane, S., Wai, W., Lai, Y., Davis, N., and Nuflo, F. 1995. Modeling visual-attention via selective tuning. *Artificial Intelligence Journal*, 78(1/2):507–545.

Webb, A. 1999. Statistical Pattern Recognition. Oxford University Press: New York.

4. The Attentional Cascade - Algorithm (3/4) ??

Q: 為什麼可以用$F<F_{i-1}*f$ 所train出來的新一層stage的F會小於f?

A:以下列為例

如果希望第一個strong classifier(stage) 的F (false positive rate)=0.2，且第二個stage亦為F=0.2，表示整個系統的F=0.04。

strong classifier $F_1=0.2$	strong classifier $F_2=0.2$

$F=0.04$

F= false positive rate= non-face image 被誤判 face的機率。

若此時有100張non-face image，F=0.2，表示第一層篩掉80張non-face image，而有20張被誤判為face，由於**train strong classifier**時會篩掉所有分對的**non-face**，因此若第二層F=0.2，表示第二層篩掉16張non-face image，剩下4張被誤判的non-face，因此就整個系統來看，100張的non-face經過兩層stage，只剩下4張被分錯，F=0.04

strong classifier $F_{i-1}=0.2$	strong classifier F_i

$F<F_{i-1}*f$