1. Robust Real-Time Face Detection [1]

在本篇 Paper 裡 Viola 等人利用了三個演算法，來快速找到人臉。第一個是 Integral Image，第二個是 Ada boost，第三個是 Cascade classifier。這些我們在下面將一一解釋。對於 Ada boost 由於前面幾篇 Paper 已經有深入講解過了，這邊就簡單略述。

- **Integral Image**

所謂的Integral Image的概念，如Figure 1所示，點(x, y)位置的值爲左上角所有灰色方塊範圍內的pixel的值的加總。

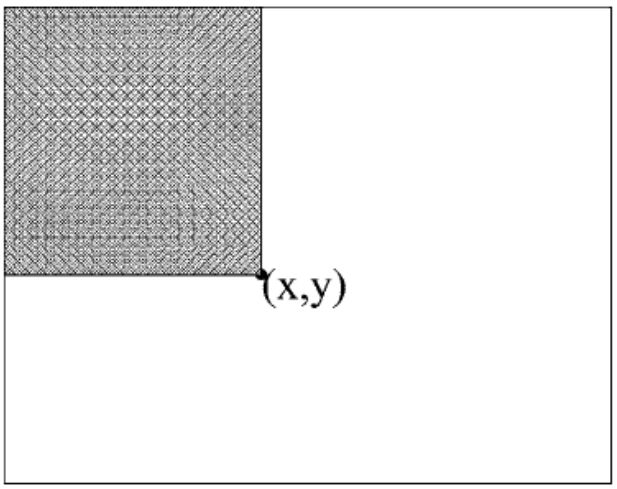

Figure 1. Integral Image

有了Integral Image值的定義以後，對於一張Image我們會有小小的類似Figure 2的Feature，這些Feature我們稱之爲rectangle feature，這些rectangle feature的大小不固定，但是白色的方塊一定和灰色的方塊一樣大。這些方塊會像Filter一樣，在一張Image之中移動。而對於每一個Feature值的計算，就是將白色的Integral的值減掉灰色方框Integral的值。

對於這樣的方式，爲何可以找到 Image 中想要的 Feature，可以利用 Simard et al 等人的證明來解釋。這個部份可以直接參閱 paper 第四頁的部份。

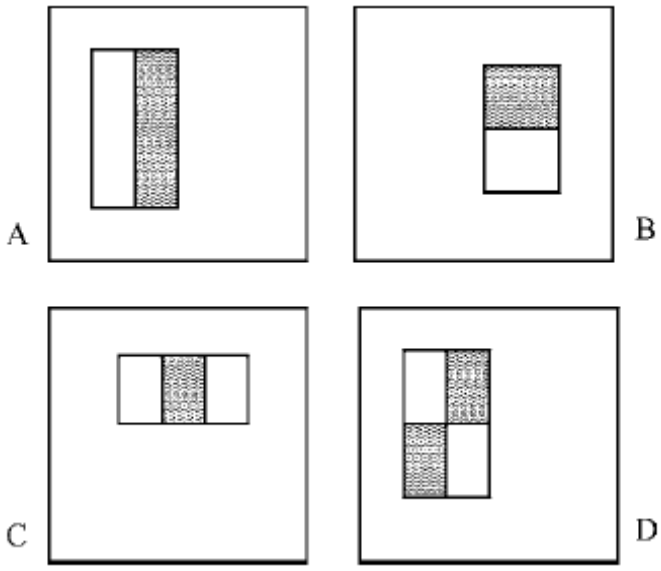

Figure 2. Rectangle Feature

- **Ada Boost**

Ada Boost結合Rectangle feature在人臉辨識的algorithm如Algorithm 1所示。首先會有一堆training的image，分別標示著人臉*m*張以及非人臉*l*張。對於每一張image我們一開始分別給他們 *1/m*或是 *1/l*的weight，端看它們是人臉或是非人臉。

接著我們要從一大堆 rectangle feature 裡面取出 T 個 feature 來。因此對下面的步驟，我們會重複 T 次。

1. 首先將所有的 weight normalize 成加起來爲 1。
2. 根據hypotheses function，選出一個error值最小的feature。關於這個function如Algorithm 1所示。
3. 紀錄可以使 error 值最小的參數。
4. 將weight根據Algorithm 1的公式update。

最後我們求出來的 classifier 就是由 T 個 weak feature 所組成的。因此對於丟入的一張 image，就會由這 T 個 feature 來投票，每一個 feature 的投票的權重不太一樣。但只有當加權值大於一半以上的所有分數時，才會認可這一張 image 爲人臉。

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ 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 $t = 1, \ldots, T$:

  1. Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
  2. Select the best weak classifier with respect to the weighted error

     $$\epsilon_t = \min_{f,p,\theta} \sum_i w_i \, |h(x_i, f, p, \theta) - y_i| \, .$$

     See Section 3.1 for a discussion of an efficient implementation.
  3. Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where f_t, p_t, and θ_t are the minimizers of 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:

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

Algorithm 1. Ada Boost in face detection

- **Cascade Classifier**

對於Cascade classifier的概念，就如Figure 3所示。我們一開始將feature分成好幾個classifier。最前面的classier辨識率最低，但是可以先篩選掉很大一部份不是人臉的圖片；接下來的Classier處理比較難處理一點的case篩選掉的圖片也不如第一個classier多了；依此下去，直到最後一個classier爲止。最後留下來的就會是我們想要的人臉的照片。

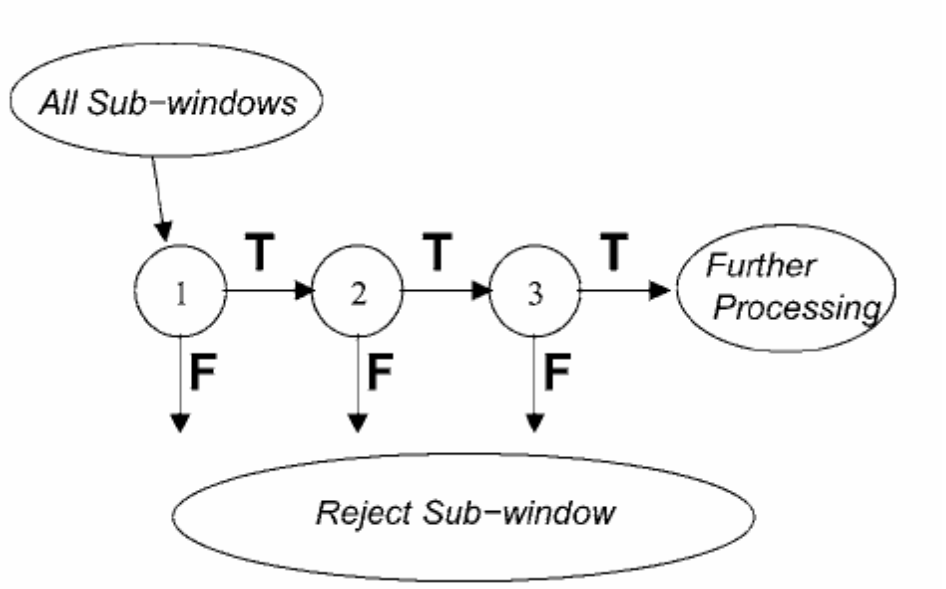

Figure 3. Cascade classifier

然而應該要決定多少個 classifier 呢？這個問題決定於我們所設定的 false positive rate 以及 detection rate 而定。所謂的 false positive rate 就是我們將人臉的圖片誤判成不是人臉的圖片的機率。而所謂的 detection rate 則正確找到人臉的準確率。通常這兩個之間會有 trade off，如果我們想要達到比較高的 detection rate，那麼 false positive rate 可能就會比較高一點；而如果想達到比較低的 false positive rate，那麼正確率難免就會下降。

整個選取cascade classifier的演算法如Algorithm 2所示。首先我們要決定每一個階層的classifier的false positive rate以及detection rate。然後我們要決定一個target的false positive rate以及target detection rate，當所有的整體的false positive rate以及detection rate達到設定的值以後才會停止。因此對於每一個階層，我們就要選足夠多的feature來達到false positive rate以及detection rate。

- 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 detections into the set N

Algorithm 2. Cascade classifier in face detection.

2. Detecting Pedestrian Using Pattern of Motion Appearance [2]

在這篇 paper 裡面 Viola 等人利用上一篇 paper 所發展出來的演算法來並增加了 motion 的 feature 來偵測一張圖片裡面的路人。所以，同樣的也用到了 Integral Image 的概念，也用到了 Ada boost 以及 Cascade classifier，唯一增加的部份就是 motion feature。

對於 training 的 image 他們取了六段 video 的 sequence 來做他們的 training set。

每一段裡面大約有 2000 張 frame，對於每一張 frame 裡面有路人的地方，他們就用手動的方式將它們框選出來。

- **Integral Image & Motion Filter**

Figure 4. Trained static rectangle features.

Integral Image的定義就如同前面的paper所述。Figure 4爲所找到的靜態rectangle feature，而Figure 5則是所找到的dynamic rectangle feature。Dynamic rectangle feature是利用motion filter方式所找到，下面會做解釋。

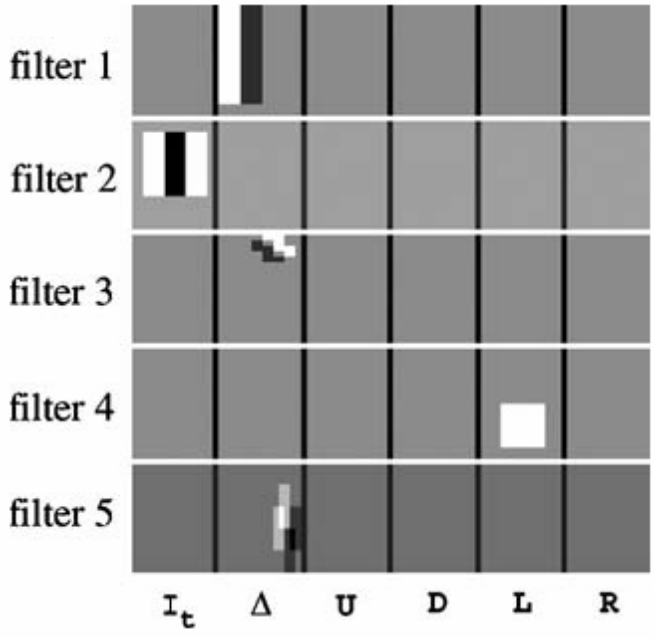

Figure 5. Trained dynamic rectangle feature.

在找rectangle feature時，我們有利用到frame之間的motion，以Figure 6爲例，對於Frame 1 和Frame 2 之間，我們會做一些motion的比較的運算。運算的方式如Algorithm 3所示。原本的Frame 1 的值I_t減掉Frame 2 的值I_{t+1}就定義爲Δ; Frame 1 的值I_t減掉Frame 2 的往上位移一個pixel的值I_{t+1}就定義爲U; Frame 1 的值I_t減掉Frame 2 的往下位移一個pixel的值I_{t+1}就定義爲D; Frame 1 的值I_t減掉Frame 2 的往左位移一個pixel的值I_{t+1}就定義爲L; Frame 1 的值I_t減掉Frame 2 的往右位移一個

pixel的值I_{t+1}就定義爲R。

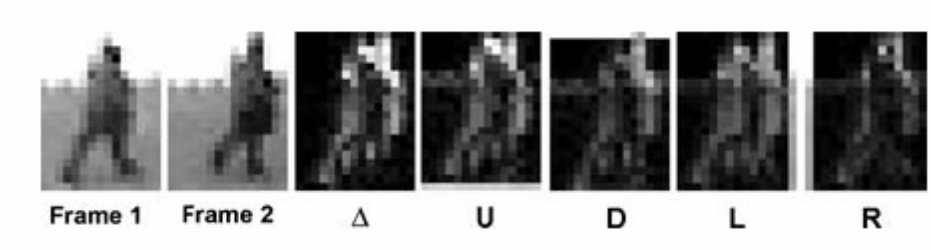

Figure 6. Motion shift between two frame.

$$\Delta = abs(I_t - I_{t+1})$$
$$U = abs(I_t - I_{t+1} \uparrow)$$
$$L = abs(I_t - I_{t+1} \leftarrow)$$
$$R = abs(I_t - I_{t+1} \rightarrow)$$
$$D = abs(I_t - I_{t+1} \downarrow)$$

Algorithm 3. Motion filter

有了這些運算的 operator 以後我們就可以定義我們 feature 的計算方式了。

$$f_i = r_i(\Delta) - r_i(S)$$

對於 f_i 所計算的是兩個 frame 之間的 motion 關係爲何，其中 S 爲{U, L, R, D}。

$$f_j = \phi_j(S)$$

f_j則是對於同一個motion stage裡面找尋feature，所用的integral image feature類似Figure 4，而 ϕ_j 所代表的是integral image feature。

$$f_k = r_k(S)$$

f_k 只是單純的計算 S operator 對於圈選出來的 rectangle 裡面的總值，這個值代表著 motion 的位移強度。

$$f_m = \phi(I_t)$$

而 f_m 則是計算靜態的 integral image 的 feature。

有了以上定義的 feature 以後我們就可以利用 Ada Boost 去 train 出能夠尋找出路

人的 feature 了。對於這些尋找出來的 feature 我們將 test image 丟進去運算之後會得到一個值，如果這個值大於 t_i 則會回傳 α 值不然會回傳一個 β 值，關於這些值會在 Ada Boost training 階段時決定。

$$F_i(I_t, I_{t+1}) = \begin{cases} \alpha & \text{if } f_i(I_t, \Delta, U, L, R, D) > t_i \\ \beta & \text{otherwise} \end{cases}$$

對於每一個 feature 我們都做如上述算式的運算。然後將所有 feature 所得到的值全部加總之後，會得到一個總值，如果這個總值大於 θ 值我們就會將這個方塊視爲路人。其算式如下:

$$C(I_t, I_{t+1}) = \begin{cases} 1 & \text{if } \sum_{i=1}^{N} F_i(I_t, \Delta, U, L, R, D) > \theta \\ 0 & \text{otherwise} \end{cases}$$

- **Cascade Classifier**

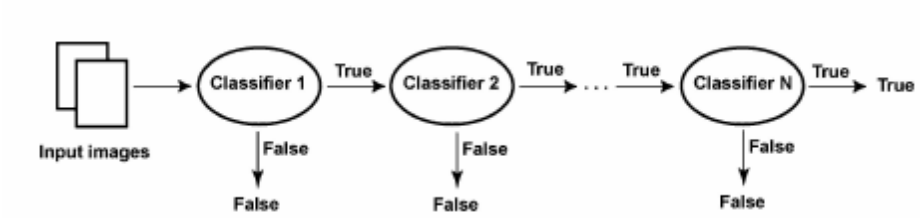

Figure 7. Cascade classifier in pedestrian detection

關於 Cascade classifier 運作的方式，跟前面一篇 face detection 的方式一樣，這邊就不再重新贅述了。

3. Ensemble Tracking [3]

這一篇 paper 是描述如何用 Ada Boost 來達到 tracking 的效果，雖然其準確率很高，不過不適用於 Gray level image，還有沒有辦法做到及時的 tracking，每秒只能 track 幾張 image 而已。

在一開始之前，使用者必須先將要 tracking 的東西用 rectangle 框起來，只需要一開始做這個動作就可以了。然後 Ada Boost 就會根據被框起來的範圍來 training 11 個 feature，這些 feature 包括 8 bins 的 5*5 local orientation histogram 以及另外三個 RGB color，在 Ada Boost training 時會自動給這些 feature weight。

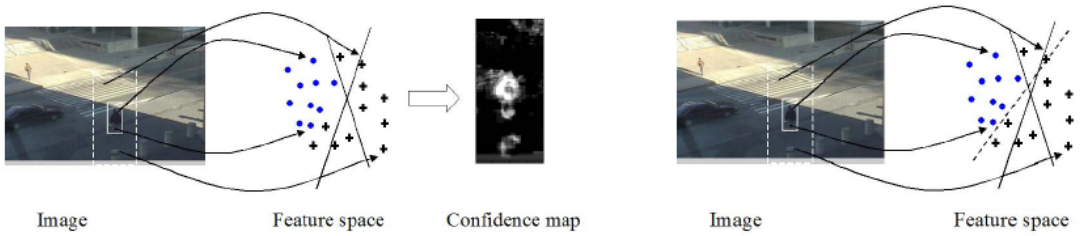

Figure 8. Ensemble update and test

找feature的方式就是利用least square的方式來尋找，如Figure 8所示。然後在找下一張image裡的feature的方式，就先用mean shift找到下一張裡面可能是人的方框。Training時會針對內框部份也就是框住人的部份當成是正確的部份的training set，而不是人的部份的training set就利用以內框爲中心向外長出的外框和內框之間的image content爲非人的training set。

每一組training出來的classifier都會有權重，如Figure 9所示，最新的classifier會在最上層，每一組classifier都有 11 個feature，而每一個classifier也都有權重，越新的classifier權重越重，而最舊的classifier會被淘汰掉。如這張圖爲例，在(d)之中，後面三個bin代表著RGB2 的權重明顯較高，是因爲圖(a)裡面的人可以用RGB就可以辨認出來了。然而在圖(b)裡面當人跟車子重疊時，而人的顏色和車子的顏色又很接近時，就必須利用到local orientation的feature了。

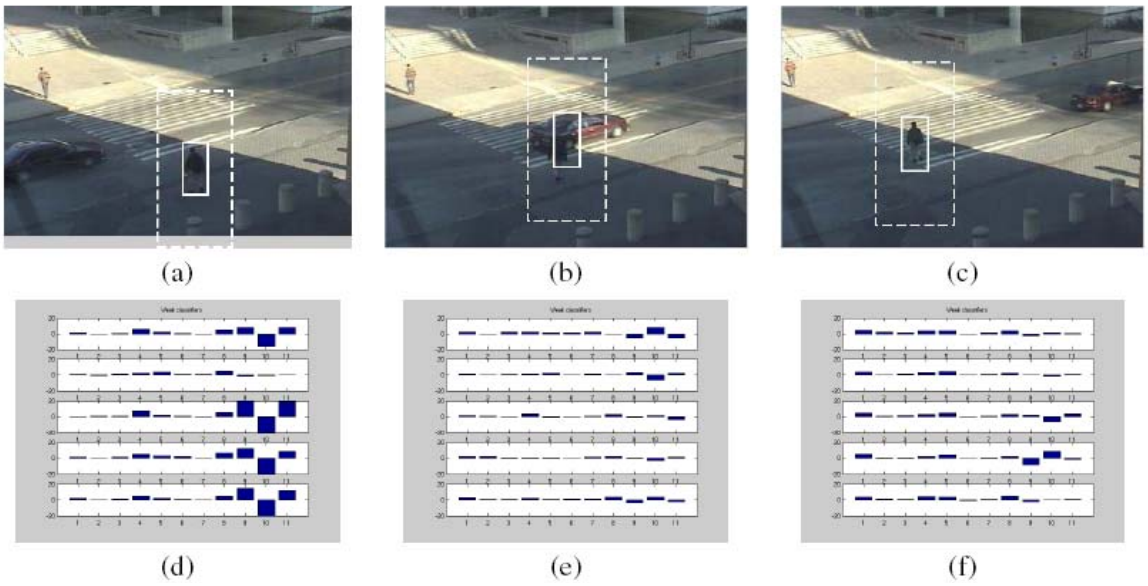

Figure 9. Adapting the weak classifiers.

Reference

[1] Paul Viola, Michael Jones, Robust Real-Time Face Detetion, International Journal of Computer Vision, 2004.

[2] Paul Viola, Michael Jones, Detecting Pedestrians Using Patterns of Motion and Appearance, International Journal of Computer Vision, 2005.

[3] Shai Avidan, Ensemble Tracking, CVPR, 2005.