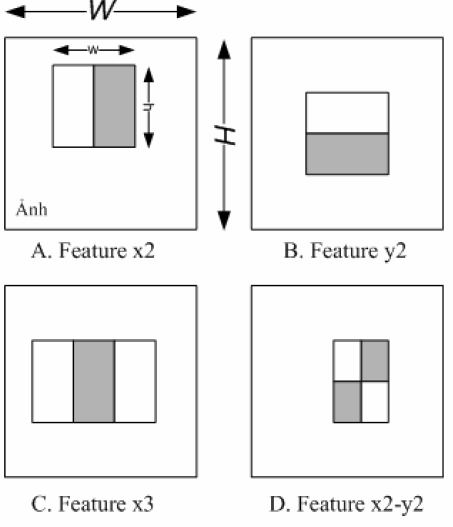
# Phần 1 Viola Jones Face dectection

## 1 Haar Feature

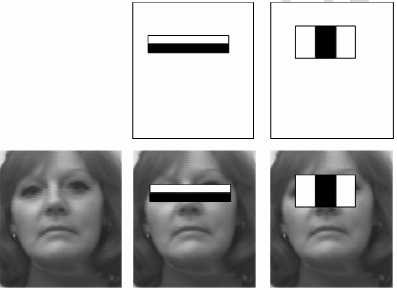
### 1.1 Haar Feature

Haar Feature là một loại đặc trưng thường được dùng cho bài toán nhận dạng trên ảnh. Haar Feature được xây dựng từ các hình chữ nhật có kích thước bằng nhau, dùng để tính độ chênh lệch giữa các giá trị điểm ảnh trong các vùng kề nhau.

Giá trị feature bằng tổng các điểm ảnh nằm trong vùng hình chữ nhật màu tối trừ cho tổng các điểm ảnh nằm trong hình chữ nhật màu sáng.

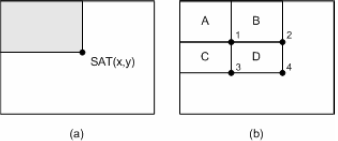


Lợi ích của Haar Feature là nó diễn đạt được tri thức về các đối tượng trong ảnh (VD: vùng hai mắt sẽ tối hơn vùng má và vùng chán, vùng giữa sống mũi tối hơn vùng hai bên mũi), điều mà bản thân từng điểm ảnh không biểu diễn được.



### 1.2 Integral Image

Trong quá trình huấn luyện, số lượng xử lý trên các Haar Feature là rất lớn, việc tính tổng các điểm ảnh cho bởi từng feature làm cho thời gian xử lý tăng đáng kể. Để khắc phục điều này, Viola và Jones đã đưa ra khái niệm Integral Image để tính toán nhanh cho khác feature cơ bản



Với định nghĩa integral image tại điểm (x,y) là:

Mỗi phần tử của bảng SAT có thể được tính như sau:

Với

Khi đó, tổng các điểm ảnh trong một hình chữ nhật bất kì có thể tính nhanh dựa trên integral image tại 4 đỉnh của nó

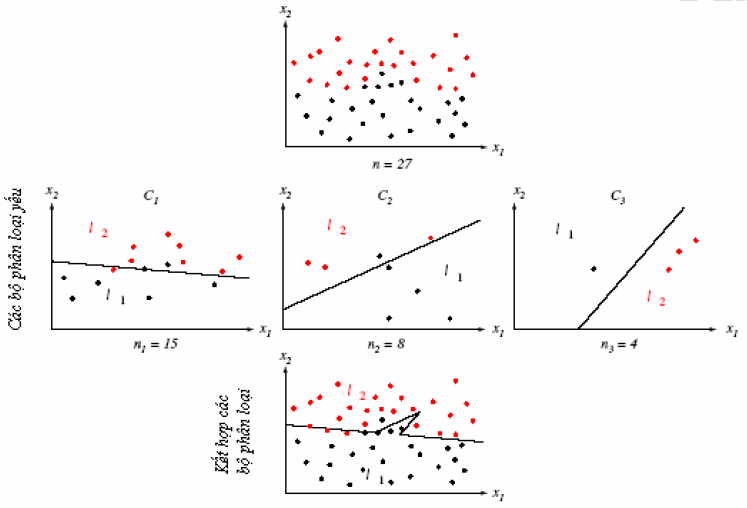
## 2 AdaBoost

### 2.1 Kỹ thuật Boosting

Boosting là kỹ thuật dùng để tăng độ chính xác cho các thuật toán học (Learning algorithm). Nguyên lý cơ bản của nó là kết hợp các weak classifiers thành một strong classifier. Trong đó, weak classifier là các bộ phân loại đơn giản chỉ cần có độ chính xác trên 50%. Bằng cách này, chúng ta nói bộ phân loại đã được “boost”.

Xét một bài toán phân loại 2 lớp (mẫu cần nhận dạng sẽ được phân vào 1 trong 2 lớp) với D là tập huấn luyện gồm có n mẫu. Trước tiên, chúng ta sẽ chọn ngẫu nhiên ra n1 mẫu từ tập D ( n1 < n ) để tạo tập D1. Sau đó, chúng ta sẽ xây dựng weak classifier đầu tiên C1 từ tập D1. Tiếp theo, chúng ta xây dựng tập D2 để huấn luyện bộ phân loại C2. D2 sẽ được xây dựng sao cho một nửa số mẫu của nó được phân loại đúng bởi C1 và nửa còn lại bị phân loại sai bởi C1. Bằng cách này, D2 chứa đựng những thông tin bổ sung cho C1. Bây giờ chúng ta sẽ xây huấn luyện C2 từ D2. Tiếp theo, chúng ta sẽ xây dựng tập D3 từ những mẫu không được phân loại tốt bởi sự kết hợp giữa C1 và C2: những mẫu còn lại trong D mà C1 và C2 cho kết quả khác nhau. Như vậy, D3 sẽ gồm những mẫu mà C1 và C2 hoạt động không hiệu quả. Sau cùng, chúng ta sẽ huấn luyện bộ phân loại C3 từ D3 .

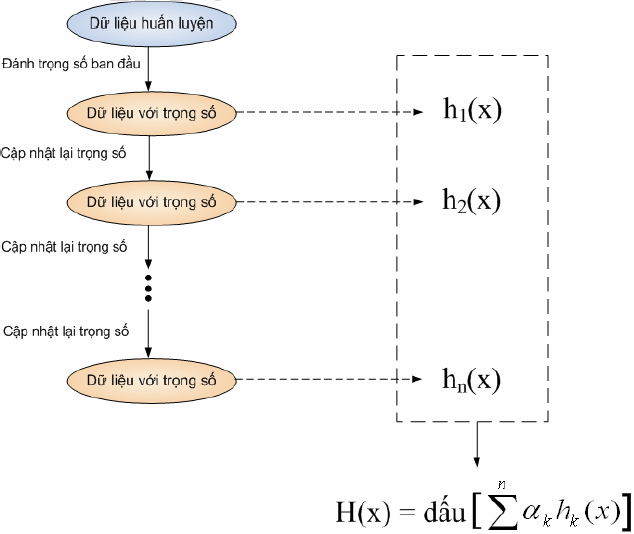
Bây giờ chúng ta đã có một strong classifier: sự kết hợp C1, C2 và C3. Khi tiến hành nhận dạng một mẫu X, kết quả sẽ được quyết định bởi sự thỏa thuận của 3 bộ C1 , C2 và C3: Nếu cả C1 và C2 đều phân X vào cùng một lớp thì lớp này chính là kết quả phân loại của X; ngược lại, nếu C1 và C2 phân X vào 2 lớp khác nhau, C3 sẽ quyết định X thuộc về lớp nào.



### 2.2 AdaBoost

Adaboost cũng hoạt động trên nguyên tắc kết hợp tuyến tính các weak classifiers để có một strong classifier. Là một cải tiến của tiếp cận boosting, Adaboost sử dụng thêm khái niệm trọng số (weight) để đánh dấu các mẫu khó nhận dạng. Trong quá trình huấn luyện, cứ mỗi weak classifier được xây dựng, thuật toán sẽ tiến hành câp nhật lại trọng số để chuẩn bị cho việc xây dựng weak classifier kế tiếp: tăng trọng số của các mẫu bị nhận dạng sai và giảm trọng số của các mẫu được nhận dạng đúng bởi weak classifier vừa xây dựng. Bằng cách này, các weak classifier sau có thể tập trung vào các mẫu mà các weak classifiers trước nó chưa làm tốt. Sau cùng, các weak classifiers sẽ được kết hợp tùy theo mức độ “tốt” của chúng để tạo dựng nên strong classifier.

Có thể hình dung một cách trực quan như sau: để biết một ảnh có phải là mặt người hay không, ta hỏi T người (tương đương với T weak classifiers xây dựng từ T vòng lặp của boosting), đánh giá của mỗi người (tương đương với một weak classifier) chỉ cần tốt hơn ngẫu nhiên một chút (tỉ lệ sai dưới 50%). Sau đó, ta sẽ đánh trọng số cho đánh giá của từng người (thể hiện qua hệ số α), người nào có khả năng đánh giá tốt các mẫu khó thì mức độ quan trọng của người đó trong kết luận cuối cùng sẽ cao hơn những người chỉ đánh giá tốt được các mẫu dễ. Việc cập nhật lại trọng số của các mẫu sau mỗi vòng boosting chính là để đánh giá độ khó của các mẫu (mẫu càng có nhiều người đánh giá sai là mẫu càng khó).



Strong classifier H(x) xây dựng bằng AdaBoost

Các weak classifier hk(x) là các bộ phân loại yếu được biểu diễn như sau:

Trong đó:

* x: cửa sổ con cần xét
* θk: ngưỡng
* fk: giá trị của đặc trưng Haar-like
* pk: hệ số quyết định chiều của phương trình

Nếu giá trị vectơ đặc trưng của mẫu cho bởi hàm lượng giác của bộ phân loại vượt qua một ngưỡng cho trước thì mẫu là object (đối tượng cần nhận dạng), ngược lại mẫu là background (không phải đối tượng).

Adaboost sẽ kết hợp các bộ phân loại yếu thành bộ phân loại mạnh như sau:

H(x) = sign(a1h1(x) + a2h2(x) + … + anhn(x) = alpha)

Với an >= 0 là hệ số chuẩn hóa cho các bộ phân loại yếu

Mỗi bộ phân loại yếu sẽ quyết định kết quả cho một đặc trưng haar-like, được xác định ngưỡng đủ nhỏ sao cho có thể vượt được tất cả các bộ dữ liệu mẫu trong tập dữ liệu huấn luyện (số lượng ảnh khuôn mặt trong tập huấn luyện lớn). Trong quá trình xác định khuôn mặt người, mỗi vùng ảnh con sẽ được kiểm tra với các đặc trưng trong chuỗi các đặc trưng haar-like, nếu có một đặc trưng haar-like nào cho ra kết quả là khuôn mặt người thì các đặc trưng khác không cần xét nữa. Thứ tự xét các đặc trưng trong chuỗi các đặc trưng haar-like sẽ được dựa vào trọng số (weight) của đặc trưng đó do Adaboost quyết định dựa vào số lần và thứ tự xuất hiện của các đặc trưng haar-like.

**Thuật toán AdaBoost**

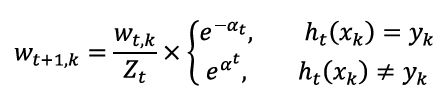
1. Cho một tập huấn luyện gồm N mẫu có đánh dấu (x1, y1), (x2, y2), …, (xn, yn). Với xk X = (x1k, x2k, …,xmk) là vector đặc trưng và yk (-1,1) là nhãn của mẫu (1 ứng với object, -1 ứng với background).
2. Khởi tạo trọng số ban đầu cho tất cả các mẫu: với m là số mẫu đúng (ứng với object và y = 1) và l là số mẫu sai (ứng với background và y = -1).
3. Xây dựng T weak classifier

Lặp t = 1…T

* Với mỗi đặc trưng trong vector đặc trưng, xây dựng một weak classifier hj với ngưỡng θj và lỗi εj
* Chọn ra hj với εj nhỏ nhất, ta được ht:

ht : X →{1; -1}

* Cập nhật lại trọng số:



Trong đó:

Zt: hệ số dùng để đưa Wt+1 về đoạn [0, 1] (normalization factor)

1. Strong classifier xây dựng được:

* Quá trình huấn luyện bộ phân loại được thực hiện bằng một vòng lặp mà ở mỗi bước lặp, thuật toán sẽ chọn ra weak classifier ht thực hiện việc phân loại với lỗi nhỏ nhất (do đó sẽ là bộ phân loại tốt nhất) để bổ sung vào strong classifier. Mỗi khi chọn được một bộ phân loại ht Adaboost sẽ tính giá trị αt theo công thức ở trên. αt cũng được chọn trên nguyên tắc làm giảm thiểu giá trị lỗi εt.
* Trong công thức của bộ phân loại H(x):

Ta thấy tất cả bộ phân loại ht đều có đóng góp vào kết quả phân loại của H(x), và mức độ đóng góp của chúng phụ thuộc vào giá trị αt tương ứng: ht với αt càng lớn thì nó càng có vai trò quan trọng trong H(x).

* Trong công thức tính αt  
  Dễ thấy giá trị αt tỉ lệ nghịch với εt. Bởi vì ht được chọn với tiêu chí đạt εt nhỏ nhất, do đó nó sẽ bảo đảm giá αt trị lớn nhất. Công thức này do Freund và Schapire đưa ra.
* Sau khi tính được giá trị αt, AdaBoost tiến hành cập nhật lại trọng số của các mẫu: tăng trọng số các mẫu mà ht phân loại sai, giảm trọng số các mẫu mà ht phân loại đúng. Bằng cách này, trọng số các mẫu phản ánh được độ khó nhận dạng của mẫu đó và ht+1 sẽ ưu tiên học cách phân loại các mẫu này.
* Vòng lặp xây dựng strong classifier sẽ dừng lại sau T lần lặp. Trong thực tế cài đặt, người ta ít sử dụng giá trị T vì không có công thức nào đảm bảo tính được giá trị T tối ưu cho quá trình huấn luyện. Thay vào đó người ta sử dụng giá trị max false positive hay max false alarm (tỉ lệ nhận dạng sai tối đa các mẫu background – false positive). Tỉ lệ này của bộ phân loại cần xây dựng không được phép vượt quá giá trị này. Khi đó, qua các lần lặp, false alarm của strong classifier Ht(x) xây dựng được (tại lần thứ t) sẽ giảm dần, và vòng lặp kết thúc khi tỉ lệ này thấp hơn max false alarm.

## 3 Cascade

### 3.1 Cascade of classifiers

Các bộ phân loại tốt thường tốn rất nhiều thời gian để cho ra kết quả phân loại bởi vì nó phải xét rất nhiều đặc trưng của mẫu. Tuy nhiên, trong các mẫu đưa vào, không phải mẫu nào cũng thuộc loại khó nhận dạng, có những mẫu background rất dễ nhận ra (ta gọi đây là những mẫu background đơn giản). Đối với những mẫu này, ta chỉ cần xét một hay vài đặc trưng đơn giản là có thể nhận diện được chứ không cần xét tất cả các đặc trưng. Nhưng đối với các bộ phân loại thông thường thì cho dù mẫu cần nhận dạng là dễ hay khó thì nó vẫn sẽ xét tất cả các đặc trưng mà nó rút ra được trong quá trình học. Do đó, chúng tốn thời gian xử lý một cách không cần thiết.

Cascade of Classifiers được xây dựng chính là nhằm rút ngắn thời gian xử lý, giảm thiểu false alarm cho bộ phân loại. Cascade tree gồm nhiều stage (hay còn gọi là layer), mỗi stage của cây sẽ là một stage classifier. Một mẫu để được phân loại là đối tượng thì nó cần phải đi qua hết tất cả các stages của cây. Các stage classifiers ở stage sau được huấn luyện bằng những mẫu negative mà stage classifier trước nó nhận dạng sai, tức là nó sẽ tập trung học từ các mẫu background khó hơn, do đó sự kết hợp các stage classifiers này lại sẽ giúp bộ phân loại có false alarm thấp. Với cấu trúc này, những mẫu background dễ nhận diện sẽ bị loại ngay từ những stages đầu tiên, giúp đáp ứng tốt nhất đối với độ phức tạp gia tăng của các mẫu đưa vào, đồng thời giúp rút ngắn thời gian xử lý.

### 3.2 Cascade of Boosted Classifiers

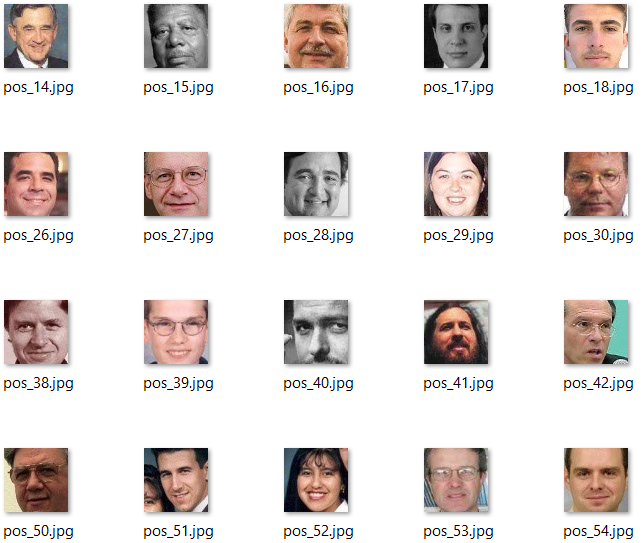
Cascade of Boosted Classifiers là mô hình Cascade of Classifers với mỗi classifier được xây dựng bằng AdaBoost sử dụng Haar Feature. Mô hình này đã được Viola và Jones sử dụng rất thành công trong bài toán nhận dạng mặt người (Face Detection). Với tập huấn luyện gồm 4196 hình mặt người được đưa về ảnh grayscale kích thước 24x24 và 9500 hình background, Viola và Jones đã xây dựng cấu trúc cascade tree gồm 38 stage với tổng cộng 6060 haar features. Thực nghiệm đã cho thấy classifier ở stage đầu tiên sử dụng 2 features và loại được khoảng 50% mẫu background (non-face) và có detection rate là 100%. Classifier ở stage thứ 2 sử dụng 10 features loại được 80% mẫu background vẫn với 100% detection rate. Hệ thống này được so sánh với hệ thống của Rowley-Baluja-Kanade (sử dụng neural network), Schneiderman-Kanade (sử dụng phương pháp thống kê), và cho thấy tỉ lệ nhận dạng là ngang nhau, trong khi hệ thống của Viola và Jones chạy nhanh hơn đến 15 lần so với hệ thống của Rowley-Baluja-Kanade và nhanh hơn 600 lần hệ thống của Schneiderman-Kanade.

# Phần 2 Nhận diện khuôn mặt với C-SVM

# Phần 3 Cài đặt triển khai

### Tập huấn luyện

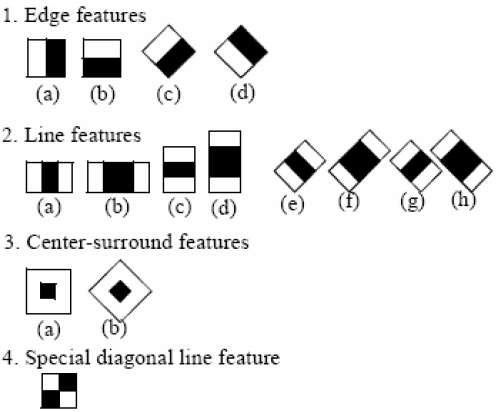
Tập huấn luyện bao gồm các mẫu positive: 1800 ảnh (là mặt người) và negative: 3000 ảnh (mẫu không chứa mặt người). Các mẫu positive được resize với kích thước 32x32 và được chuyển về ảnh grayscale. Các mẫu negative là các hình background không chứa mặt người: nhà cửa, cây cối,…



Các mẫu negative trong tập huấn luyện có thể có kích thước tùy ý nhưng phải lớn hơn kích thước mẫu positive. Trong quá trình huấn luyện, các weak classifier sẽ học từ các mẫu positive trong tập huấn luyện và các mẫu negative là các vùng ảnh (sub window) trích ra từ các mẫu negative trong tập huấn luyện.

### Đặc trưng

Nhóm sử dụng các đặc trưng Haar Feature sau:



### Gentle Adaboost

Với 15 loại feature sử dụng và với kích thước mẫu positive là 24x24, tập feature (Feature Pool) của chúng ta có khoảng 160000 feature. Tuy nhiên, không phải feature nào cũng thực hiện tốt việc phân loại mà chỉ có một số lượng nhỏ trong số 160000 feature này là thực sự hữu dụng. Nhiệm vụ của bộ phân loại là phải tìm ra được các feature này.

Mỗi weak classifier gồm có 1 feature và 1 ngưỡng, ngưỡng này chính là giá trị của một mẫu cụ thể cho bởi feature này (vấn đề của thuật toán là phải tìm được mẫu nào dùng làm ngưỡng). Như vậy, với N mẫu và M features thì số weak classifier có thể có là NxM, tức là với 1800 mẫu cho một cử chỉ và 160000 features thì hệ thống sẽ phải chọn được 1 weak classifier trong số 1800 x 160000 weak classifiers trong mỗi vòng lặp boosting.

AdaBoost được thiết kế để có thể chọn nhanh các features, cũng là chọn nhanh các weak classifiers. Thuật toán sử dụng ở đây là Gentle AdaBoost, một biến thể của AdaBoost. Gentle AdaBoost để xây dựng bộ nhận dạng cho khuôn mặt như sau:

Triển khai CreatClassifier

1. Cho một tập huấn luyện n mẫu (x,y) với x X, y {-1. 1}
2. Chọn min detection rate và max false alarm
3. Xây dựng tập feature và tính toán các bảng SAT và RSAT cho tất cả các mẫu trong tập huấn luyện.
4. Khởi tạo trọng số ban đầu cho tất cả các mẫu
5. Lặp

* Với mỗi đặc trưng trong tập đặc trưng, xây dựng một weak classifier hj với hàm phân loại ft , ngưỡng θj và tính giá trị lỗi εj của bộ phận này
* Từ các hj đã có, chọn ra hj có εj nhỏ nhất, ta đươc ht :
* Cập nhật lại trọng số:

Với là normalization factor

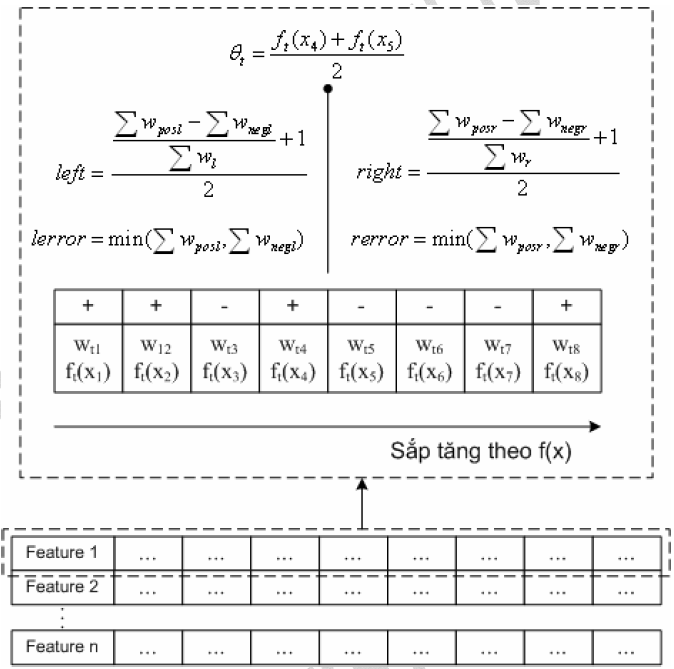
* , tính ngưỡng θ

Cho đến khi

1. Bộ nhận dạng H(x)

Với F(x) = Ft (x)

Mỗi weak classifier ht(x) sẽ được chọn bằng cách tính giá trị của từng feature cho tất cả các mẫu trong tập huấn luyện f(xk) và sắp xếp các mẫu theo thứ tự tăng dần của giá trị này.



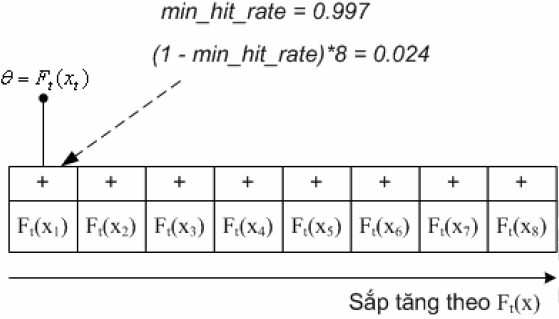
Hình 11: Cách chọn weak classifier của AdaBoost

Feature được chọn là feature thực hiện phân đôi tập huấn luyện với lỗi

nhỏ nhất

Sau mỗi vòng lặp của boosting, ta xây dựng thêm được một weak classifier, ta có thể xấy dựng được một strong classifier mới từ sự kết hợp của các weak classifier có được cho tới lần boost hiện tại:

Giá trị ngưỡng θ được chọn nhờ vào giá trị min detection rate (hay min hit rate) . *min detection rate* là tỉ lệ nhận dạng đúng tối thiểu các mẫu positive mà bộ phân loại phải đạt, giá trị này do chúng ta xác lập khi bắt đầu quá trình huấn luyện. Bằng cách sắp xếp các mẫu positive xk theo Ft(xk) tăng dần:



Hình 12: Chọn ngưỡng θ dựa vào min detection rate

Trong hình 12, min detection rate được chọn là 0,997 , nghĩa là tỉ lệ nhận dạng đúng các mẫu positive không được thấp hơn 99.7%. Do đó, phần tử được chọn làm ngưỡng là mẫu đầu tiên trong danh sách sắp tăng. Giá trị ngưỡng θ này đảm bảo có ít nhất 99.7% các mẫu positive sẽ có Ft(x) >= θ. Nếu ta chọn min detection rate là 0.5, khi đó (1-min detection rate)\*số\_mẫu = 4, phần tử thứ 4 của danh sách các mẫu sẽ được chọn làm ngưỡng, như vậy sẽ chỉ có 4 mẫu trong số 8 mẫu positive là được nhận dạng đúng. Ngưỡng có ý nghĩa rất quan trọng, ngưỡng càng nhỏ thì hit rate sẽ càng cao nhưng false alarm cũng tăng theo và ngược lại

Sau khi có được Ht(x), vấn đề cần xem xét là liệu bộ phân loại này có đủ tốt chưa, nghĩa là giá trị lỗi của nó đã thấp hơn max false alarm hay chưa. Max false alarm là một giá trị được xác lập trước khi tiến hành huấn luyện. False alarm (hay còn gọi là false positive) là tỉ lệ nhận dạng sai các mẫu negative. Ví dụ, nếu max false alarm = 0.5 thì trên 100 mẫu negative, nó phải nhận đúng ít nhất là 50 mẫu (50 mẫu còn lại bị phân loại nhầm thành positive). Vòng lặp xây dựng strong classifier sẽ kết thúc khi giá trị false alarm của strong classifier này thấp hơn max false alarm.

### Cascade of boosted classifiers

Trong phần cài đặt của bài toán nhận diện khuôn mặt, max false alarm và min detection rate được xác lập bằng nhau ở tất cả các stages. Khi đó, max false alarm và min detection rate của stage thứ k lần lượt là: và

Thuật toán xây dựng cascade tree:

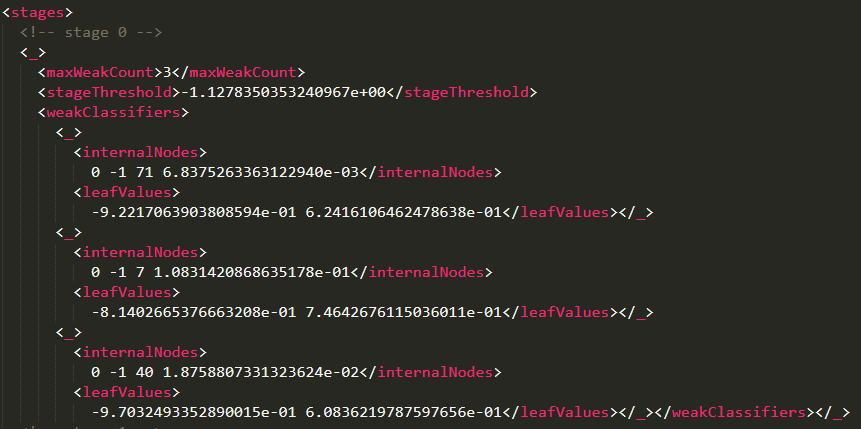
* Xác lập max false alarm f, min detection rate d cho các bộ phân loại ở mỗi tầng và số stagee tối đa cascade tree sẽ có
* Tính false alarm F cho bộ phân loại chính (cây phân lớp)
* F0 = 0, i = 0
* p, n là số lượng mẫu positive và negative
* P0 , N0 là tập positive và negative cho bộ phân lớp ở tầng đầu tiên
* Trong khi Fi > F
* Huấn luyện bộ phân loại Hi từ tập Pi và Ni với detection rate d và max false alarm f (thủ tục CreatClassifier đã đề cập ở phần trên)
* Thêm Hi và cây phân lớp
* Dùng cây phân lớp hiện có để tính Fi+1 : Duyệt qua N mẫu negative cho đến khi tìm đủ n mẫu mà cây phân lớp hiện có phân loại sai
* Nếu Fi+1 > F, đưa n mẫu negative trên vào Ni+1 : xây dựng Pi+1 với tối đa p mẫu positive mà cây phân lớp hiện có phân loại đúng.
* i = i+1

Qua thuật toán trên ta thấy, các mẫu negative trong tập huấn luyện của stage classifier sau sẽ là những mẫu negative mà stage classifier trước nó nhận dạng sai. Như vậy, nó có điều kiện tập trung học những mẫu background khó.

### Cài đặt tham số và kết quả triển khai

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Loại tham số | Tên tham số | Mô tả | Giá trị |
| 1 | Common arguments | data<cascade\_dir\_name> | Thư mục chứa kết quả học (file xml) |  |
| 2 | vec <vec\_file\_name> | File .vec chưa thông tin về các positive sample | pos\_samples.vec |
| 3 | numPos <number\_of\_positive\_samples> | Số lượng các positive sample được sử dụng để train cho các stage. Nhóm chọn 80% số ảnh positive | 1800\*80%=1440 |
| 4 | numNeg <number\_of\_negative\_samples> | Số lượng các negative sample được sử dụng để train cho các stage. | 3000 |
| 5 | numStages <number\_of\_stages> | Số lượng các cascade stage được train | 15 |
| 6 | acceptanceRatioBreakValue <break\_value> | Tham số xác định độ chính xác của mô hình có học tiếp hay dừng lại. Thường dùng 10e-5 để tránh overtrain trên tập học. Sử dụng -1 để bỏ tham số này | -1 |
| 7 | Cascade parameters | stageType <BOOST(default)> | Kiểu Stages | BOOST |
| 8 | featureType<{HAAR(default), LBP}> | Kiểu feature | HAAR |
| 9 | h <sampleHeight> | Chiều cao của traning sample (pixel) | 24 |
| 10 | w <sampleWidth> | Chiều rộng của training sample (pixel) | 24 |
| 11 | Boosted classifer parameters | bt <{DAB, RAB, LB, GAB(default)}> | Các kiểu boost classifier: DAB - Discrete AdaBoost, RAB - Real AdaBoost, LB - LogitBoost, GAB - Gentle AdaBoost. | GAB |
| 12 | minHitRate <min\_hit\_rate> | Giá trị hit rate tối thiểu cho mỗi stage | 0.995 |
| 13 | maxFalseAlarmRate <max\_false\_alarm\_rate> | Giá trị false alarm rate tối đa cho mỗi stage | 0.5 |
| 14 | maxDepth <max\_depth\_of\_weak\_tree> | Chiều cao tối đa của của cây phân lớp yếu (1- cây bậc 1) | 1 |
| 15 | maxWeakCount <max\_weak\_tree\_count> | Số lượng weak trees tối đa trong mỗi cascade stage | 100 |
| 16 | Haar-like feature parameters | mode <BASIC (default) | CORE | ALL> | Các kiểu haar feature được sử dụng để train. BASIC use only upright features, while ALL uses the full set of upright and 45 degree rotated feature set | ALL |

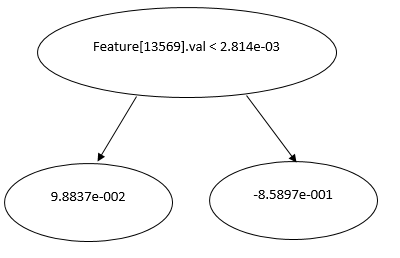
**Kết quả thu được:** file **cascade.xml** chứa thông tin về các stage và features:

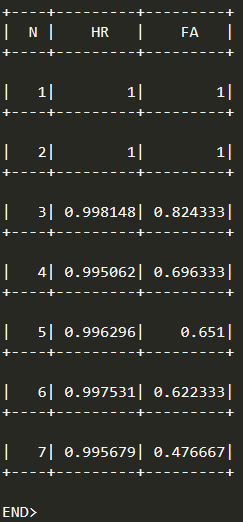


Thông tin về stage 0 trong kết quả train

Mô tả một stage trong đó cặp nhãn <internalNodes> và <leafValues> mô tả một weak classifier trong stage đó. Cây có chiều cao tối đa bằng 1.

* Các giá trị 0 và -1 là các index của con trái và phải của node gốc.
* Giá trị 71 là index của feature trong tập <feature>
* Giá trị 6.8375e-3 là ngưỡng của node đó
* Leaf values chứa trọng số của node lá trong cây cascade



Ví dụ: Weak feature có index[13569] có ngưỡng là 2.814e-03. Nếu giá trị của feature h(x) < threshole thì feature sẽ có weight là giá trị của leaf node trái, và ngược lại.

**Mô tả log khi train:**

Làm rõ hơn quá trình kết hợp các weak classifier thành một strong classifier

* N thứ tự thêm các weak classifier
* HR: hit rate
* FA: false alarm

Các weak classifier sẽ được thêm lần lượt để tạo ra một strong classifier. Weak classifier đầu tiên đảm bảo HR=1. Các weak classifier sau đó được thêm theo nguyên tắc giảm dần HR sao cho HR >= min\_hit\_rate, giảm dần FA. Khi nào FA < max\_false\_alarm thì quá trình thêm sẽ dừng. Trong hình 15: khi thêm đếm weak classifier thứ 7 thì FA=0.47<0.5(max\_alarm\_rate) nên quá trình thêm sẽ dừng.

**Hình minh họa**: kết quả chạy của thuật toán Viola Jones trong nhận diện khuôn mặt:

