A New Method for Haar-Like Features Weight Adjustment Using Principal Component Analysis for Face Detection

Ramiro Pereira de Magalhães and Cabral Lima
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil
e-mail: ramiro.p.magalhaes@gmail.com, cabrallima@ufrj.br

Abstract—**This paper proposes a new weight assignment method for Haar-like features. The method uses principal component analysis (PCA) over the positive training instances to assign new weights to the features. Together with the method, a particular Haar-like feature that uses statistics extracted from positive training instances is employed. The method and the Haar-like feature were designed to verify if the distribution of points produced from the negative instances in the single rectangle feature space (SRFS) of each Haar-like feature could be modeled as an uniform distribution. Although negative instances may spread themselves in very different and chaotic ways through the SRFS, experiment with the method and the Haar-like feature has shown that the negative instance cannot be properly modeled as an uniform distribution.**

Keywords-pattern detection; Viola-Jones framework; Adaboost; Haar wavelet; principal component analysis

I. INTRODUCTION

Object detection is the task of automatically discovering the presence and location of a particular object in an image. It is usually the first step for additional processing over the target object. The detection of human faces in complex scenes, required for several applications, is a complex problem and has been the main subject of several researches, many of them surveyed by Zhang and Zhang [1]. The Viola-Jones framework [2] is probably the most known method to deal with this problem. The use of Haar-like features to develop an accurate frontal face classifier makes the process fast enough to detect objects in real time video. Some researches tried to improve the speed, accuracy and robustness of such approach [3] [4] [5] [6]. For instance, Pavani et al. [7] established that it is possible to assign better weights to Haar-like features by interpreting them as the inner product of the weights versus the rectangular areas average values. Their experiments showed superior results if compared to many other relevant works.

This paper, based on the approach of Pavani and colleagues, presents a new method for assigning weights to Haar-like features. PCA is used to find a vector of weights that befittingly identifies the positive training instances. This simple and fast method may be complementary to Pavani's approach and has the advantage that it can be applied on a step preceding the classifier boosting. This provides some relief to the boosting process, known as a lengthy phase. During the PCA processing, some statistics are extracted from the positive instance dataset in order to be employed in a new classifier similar to the one suggested by Landesa-Vazquez and Alba-Castro [5].

In this paper, Section II introduces the processes and frameworks used in the development of a face detector as shown in [2]. Section III reviews some recent researches that brought interesting ideas and enhancements to such detectors. Section IV details the main contributions of this paper. In Section V some experiments using the proposed method are detailed. Section VI concludes this paper and presents some future works.

II. THE VIOLA-JONES FACE DETECTOR

In this section, Viola-Jones' face detector building blocks are described. Notions of boosting and the structure of the face detector along with the functions used to extract features are also presented.

A. *Boosting*

Boosting is a machine learning technique based on the idea that it is possible to form an accurate classification rule (named strong classifier) by merging many inaccurate classification rules (the weak classifiers). The most known boosting algorithm is Adaboost [8]. Roughly, a boosting algorithm must show the input, a labeled dataset with positive and negative instances, to another algorithm (generically named weak learner) pointing out that some instances are more important to be accurately classified than others. With this information, the weak learner must choose a weak classifier that best labels the input dataset while considering the importance of each instance. After that, the importance of each instance is updated with aid of the recently produced weak classifier, which is then pushed into the strong classifier along with a rating of its classification capability. This whole loop then is repeated for a certain amount of iterations until the strong classifier with its boosted weak classifiers reaches some stop criteria.

A boosting algorithm usually receives as input a set of m labeled instances $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X$ represents the objects to be classified; and $y_i \in Y = \{-1, +1\}$ is the set of possible classes, where $+1$ indicates that the object belongs to the desired class and -1 the opposite. The main objective of a boosting algorithm is to generate a strong classifier $H : X \mapsto Y$ composed by some weak classifiers $h_t(x)$, where $t = 1, \dots, T$ means the iteration in which the weak classifier was generated. For each iteration the boosting algorithm invokes another algorithm, generically referred as weak learner, that is responsible to produce the weak classifiers

```
Input: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X$ e
       $y_i \in \{-1, +1\}$
begin
    Set $D_1(i) = 1/m$ where $i = 1, \dots, m$;
    for $t = 1, \dots, T$ do
        Provide $D_t$ to the weak learner and from it get
        $h_t : X \mapsto \{-1, +1\}$, such that $h_t$ minimizes
        $\epsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
        Let $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
        for $i$ = 1, ..., $m$ do
            $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$
            $= \frac{D_t(i) exp(-\alpha_t y_i h_i(x_i))}{Z_t}$
            where $Z_t$ is a normalization factor chosen
            so that $D_{t+1}$ is a distribution.
        end
    end
    return $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$.
end
```

Figure 1. The Adaboost algorithm used to boost a set of weak classifiers into a strong classifier.

$h_t(x)$ which will be added to the strong classifier. Figure 1 shows Adaboost, slightly adapted from [9].

B. Face detector

The goal of a face detector is to determine the presence and location of faces in an arbitrary image. If they exist, then the detector should also be able to determine the region they occupy in the image [1]. This has been seen as a challenging task for a machine due to the enormous variety of human skin, hair, eye colors, texture, facial features, accessories, expressions, rotations, and even environment lighting conditions.

The powerfull and fast face detector proposed by Viola and Jones in [2], and revised in [10], operates by classifying the contents found inside a window positioned over the image. This window slides in the vertical and horizontal directions until the whole image has been scrutinized. This process may repeat with windows having different sizes as first shown by Rowley et al. [11]. Such "sub-windows" form an overcomplete set of the examined image, but very few of them contain a face. Therefore, a detector that thoroughly inspects every sub-window consumes a lot of time evaluating background scenes in order to find a single face. To deal with this problem, Viola and Jones proposed that the final classifier should be built like a chain of increasingly complex strong classifiers. Each node in this chain should reject many background objects (around 50% or more) while rejecting very few faces (preferably none). This "rejection cascade", originally proposed by Baker and Nayar [12], allows the quick discarding of uninteresting sub-windows because it is enough that a single node rejects the input for it to be classified as background (Figure 2).

The whole chain is the result of a bootstrapping process. To each node a threshold of maximum false positive and true positive rates are set. Similarly, a maximum false positive rate is also set to the whole chain. Positive and negative training instances are provided to Adaboost that will iterate as much as needed to reach the node thresholds. Once a node is ready it is added to the chain that is then tested for its maximum false positive threshold. If the threshold of the chain has not yet been reached, a number of false detections made by the chain over the negative instance set are then used to boost the next node. The set of positive instances always remains the same. Through this process the nodes closer to the end of the classifier will be trained with "harder" instances, hence they will be more complex, i.e., they will have more weak classifiers and be more precise.

C. Haar wavelets as a weak classifiers

A Haar wavelet is a function proposed be Alfred Haar [13] to transform a signal in a simpler (or more meaningful) representation to certain analysis procedures. Papageorgiou et al. [14] created a feature extractor that uses Haar wavelets to encode local differences of pixels in images. This Haar-like feature is a value in $\mathbb{R}$ obtained from the weighted sum of pixel intensities contained in the d rectangular regions of the Haar wavelet, where each region is associated with a weight $v \in \mathbb{R}, v \neq 0$. Usually, the weights of a Haar-like feature add up to 0, and are proportional to the amount of pixels contained in the rectangle that they refer to. Considering w a Haar wavelet, r a rectangular region of w, and l the pixels contained in r, it is possible to establish:

$$f(w) = \sum_{i=1}^{d} v_i (\sum_{l \in r_i} l). \qquad (1)$$

The weak classifiers proposed by Viola and Jones [2] use such features. In fact, they are a function $h(x, f(w), p, \theta) \mapsto \{-1, +1\}$, where the value $+1$ means that the object belongs to the class of interest, and -1 the opposite. Considering $p \in \{-1, +1\}$ the polarity (or parity), θ a threshold, and x an image sub-window, [2] established:

$$h(x, f, p, \theta) = \begin{cases} +1 & \text{if } pf(x) < p\theta \\ -1 & \text{otherwise} \end{cases} . \qquad (2)$$

p simply affects the orientation of the comparison. Sometimes, $h(x, f, p, \theta)$ is defined to return 0 instead of -1. The value used in this paper keeps (2) consistent with Algorithm 1.

Since its proposition, researchers are trying to improve the Haar-like features. Besides extending Papageorgiou and

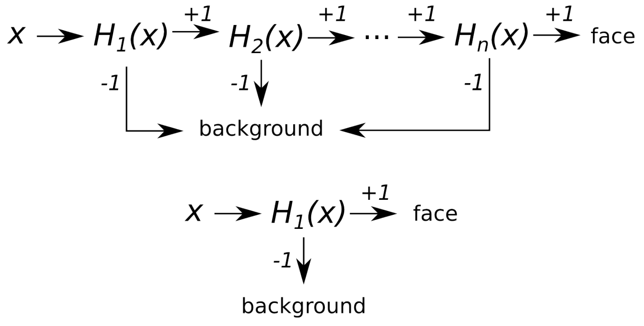


Figure 2. Rejection cascade composed of strong classifiers $H_i(x)$ (above) compared with a monolithic classifier (below).

colleagues set of features, Viola and Jones [2] developed a method to calculate any Haar-like feature value in constant time. Lienhart and Maydt [15] proposed $45°$ rotated features, also calculated in constant time. Disjoint rectangles, a more general way to produce features, were introduced by Li et al. [16]. Later, Viola and Jones [17] showed a set of "diagonal features". Figure 3 shows some examples of such features. A collection of other researches about this topic can be found in [1].

III. Related works

This section presents a review of some researches dealing with accuracy, performance and training time of strong classifiers through manipulation of Haar-like features.

Dembski [3] presents the result of some experiments carried out with the Lienhart and Maydt's extended feature set [15]. The main goal of this research was to verify if there is some pattern in the contribution of the features found in a strong classifier, i.e., to find if there is a set of features more useful than others. He demonstrated that the line features (rotated or upright) provide a better generalization error than both the center-surround and the border ones. Dembski compared the horizontal and rotated features and established that the latter generalize better than the former. He observed that larger features perform better than the smaller ones. Nevertheless, the generalization differences among the compared features are small, so it is possible that Dembski's results do not hold in other experiments.

In Baumann's work [4], a modification in the Adaboost algorithm to explore the human face symmetry was proposed. In their experiments, the time taken to boost a strong classifier was reduced by almost 40%, due to the selection of two weak classifiers per round instead of just one as usually occurs. The first classifier is chosen using the normal procedure [8] and a second symmetric feature is chosen and placed in a symmetric region of the sub-window, but its final position will still be target of a search in the close neighbouring area.

Landesa-Vázquez and Alba-Castro [5] developed a weak classifier slightly different from the one proposed by Viola and Jones [2]. Motivated by physiological studies on human vision, they modeled an apolar weak classifier that considers the Haar-like feature's absolute value. They compared their strong classifier with Viola and Jones and, although they did not observe any change in the detector precision, their final cascade had much less weak classifiers.

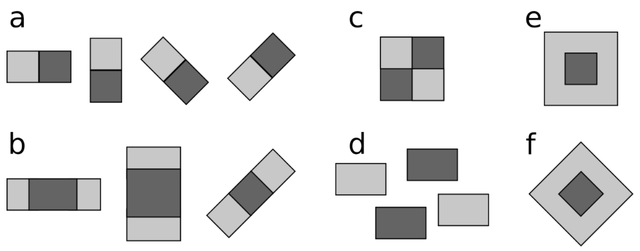


Figure 3. Examples of Haar-like features: (a) border; (b) line; (c) 4-dimensional feature proposed in [14]; (d) disjoint rectangles proposed in [16]; (e) e (f) center-surround features. Darker regions have different weight than the lighter ones.

Vural and colleagues [6] proposed a new set of Haar-like features with a very different composition of rectangular regions, and able to rotate in six angles. The upright features that serve as template for the rotated versions, were automatically generated through an iterative procedure that first adds a single rectangle and then evaluates the feature performance. Only those with the smallest error rate participated on the classifier boosting rounds. As a result, only a quite small amount of features, if compared to other researches, was used in the boosting rounds. This not only sped up the boosting procedure but also reduced the amount of features found in the final detector.

Pavani et al. [7] argued that the weights typically assigned to rectangles of a feature are suboptimal. They demonstrated this through the introduction of the SRFS, where vectors s of d dimensions contain the averages $s_i, i = \{1, \dots, d\}$ of the pixels contained in each one of the Haar-like feature's rectangles. This is the linear algebra interpretation of the feature value calculation, as shown in (3):

$$f(w) = \sum_{i=1}^{d} v_i (\sum_{l \in r_i} l) = \sum_{i \in w} v_i s_i. \tag{3}$$

Therefore, $f(w)$ is the result of the inner product of s by the weights vector v, i.e., a Haar-like feature projects s in the direction of v.

Pavani evaluated the distribution of vectors in the SRFS. For some Haar-like features w they generated a set of vectors S_w^+ using only the positive training instances, and did the same to the negative instances, creating the set S_w^-. He established that S_w^+ results in a very concentrated point cloud, while S_w^- shows a much more varied spread. Since those classes spread over the SRFS in very particular ways, Pavani and collaborators observed that the projections made with typical values of v may not help to discern between classes as they should. For instance, consider a Haar-like feature w' with two rectangles ($d = 2$) and weight vector $v' = \{-1, 1\}$, and assume that the SRFS $S_{w'}^+$ of w' spreads like a bivariate Gaussian distribution, with the highest variance axis parallel to v', as seen in Figure 4. In this case, points in $S_{w'}^+$ are projected in the direction of v' mixing themselves more often with the $S_{w'}^-$ set. If v' is perpendicular to the highest variance axis formed by $S_{w'}^+$ distribution, then this mixture will be less frequent, and the Haar-like feature will be more discriminative. Pavani proposed then the optimization of vector v of all candidate Haar-like features, and used three different methods to this effect: brute-force search, genetic algorithms and Fisher's linear discriminant analysis (FLDA). These methods not only optimize v, but also select during boosting the parameters p and θ with the smallest classification error.

The three resulting detectors were tested and compared among themselves and with the ones of [2] [18] [19] [20]. The most accurate (given by the area over the ROC curve) was the genetic algorithm optimized one, although it took 20 days to be trained [7]. This detector was also considered the fastest since, in average, rejects more negative samples using less nodes of the classifier cascade.

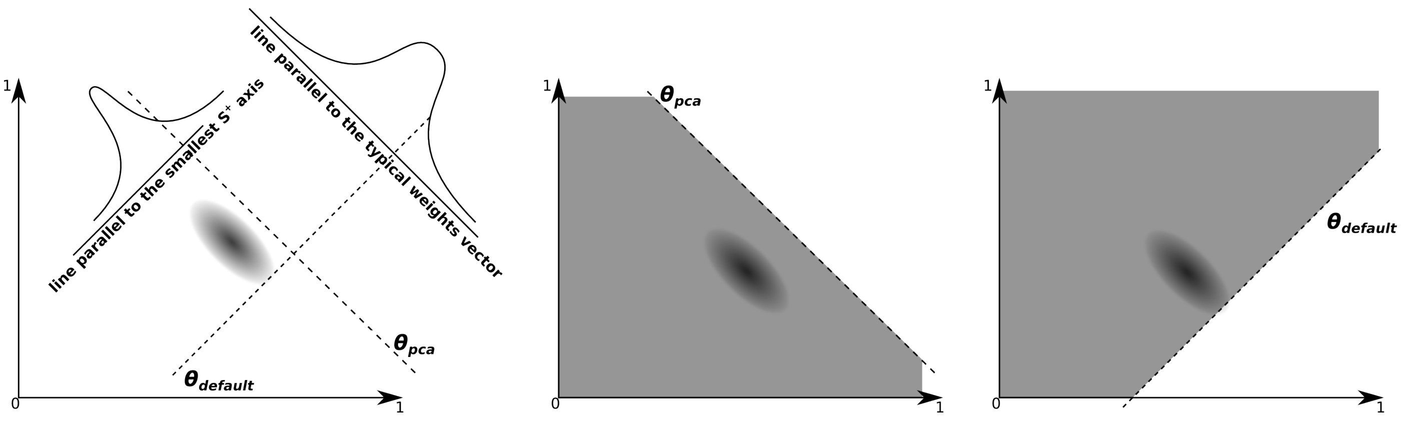

line parallel to the smallest S^+ axis
line parallel to the typical weights vector
θ_{pca}
$\theta_{default}$
θ_{pca}
$\theta_{default}$
0
1
1

Figure 4. The left image shows a set of points in the SRFS formed only by faces. Projecting a point in the SRFS in a parallel direction to the optimal v yields a higher concentration of points.The application of the classifier threshold θ splits the SRFS in very distinct ways, as the remaining images show.

IV. A NEW OPTIMIZATION METHOD

In this section, an optimization method complementary to those shown in [7] is proposed. Additionally, Pavani's results are discussed in order to properly motivate the ideas used in the new method.

As shown in Section III, Pavani's results suggest that (a) the distribution formed by the face and background feature points in the SRFS do not have the same covariance; or (b) the spread of S^+ or S^- is not properly described by the Gaussian distribution. It is possible to reach such conclusions by recalling that FLDA is particularly effective when both classes it tries to separate behave as Gaussian distributions with the same covariance [21, p. 120]. In addition, through the analysis of data shown in [7], even though it looks like that S^+ spread can be modeled through the Gaussian distribution, the same seems hard to be stated about S^-. These observations were described in [7]. In fact, up to the moment this paper is written, only a few other researches considered Pavani's findings [22] [23]. Therefore, it is hard to state for sure how face and background feature values produced from the most used Haar-like features are spread over their respective SRFSs.

This research aims to provide some additional information about how features values are laid out in the SRFS. Admitting that a Gaussian distribution befittingly models S_w^+ distributions, it is intended to verify if a uniform distribution better models the S^- spread. This assumption might seem naive, but it should be verified because a simpler and faster Haar-like feature weight assignment procedure could be employed if the assumption holds true. Hence, the method proposed here uses PCA to assign weights to each Haar-like feature w from S_w^+'s principal component of least variance. To explore even further the fact that S_w^+ is highly concentrated around its mean, a weak classifier similar to the one proposed in [5] is used. The method and the classifier are detailed in Section IV-A.

The proposed method indeed reduces the total training time. While FLDA is a fast method if compared to brute-force search or genetic algorithms, it needs to estimate the average and variance of both sets S_w^+ e S_w^- in order to obtain the inter and intra-class spread. PCA is less complex than FLDA and, in this case, is applied only over S_w^+. Another important aspect that would also reduce the total training time is the moment when v optimization occurs. In [7], an optimization method must be invoked at least once per node of the chain of classifiers, after all, as shown in Section II-B, the negative instances set change between each cascade node boosting run. In the particular case of the FLDA optimization, each feature must be optimized once per node. The method proposed here allows the pre-optimization of the weak classifiers prior to boosting them. This is possible because the set of positive instances remains the same throughout the whole chain of classifiers construction process, so it is unnecessary to recalculate each feature's weight when the construction of a new cascade node begins.

Vural et al. [6] described an iterative construction technique of features. In the method proposed here, the features are chosen following Pavani's rules, as described in Section V-B. Through these rules, neither the rotated features, neither the center-surround are used, what conforms with Dembski's [3] considerations (see Section III).

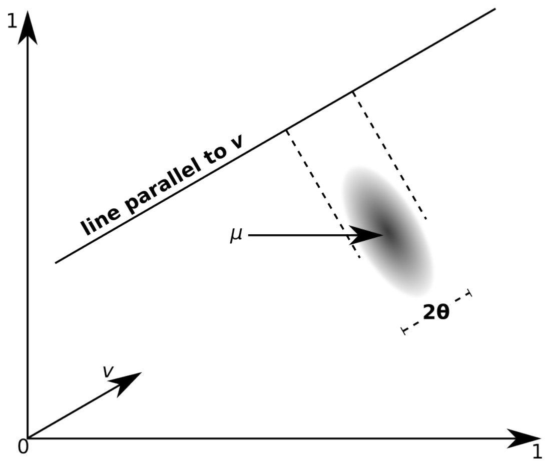
A. *The proposed feature*

Let μ be S^+ mean. The following feature is proposed:

$$f'(w) = |\sum_{i \in w} v_i(s_i - \mu_i)|. \quad (4)$$

Combined with Viola-Jones' weak classifier threshold, this feature effectively creates a "band" over the SRFS perpendicular to v, that passes through μ, and has 2θ width. Figure 5 illustrates this.

The insight behind this features and classifier combination is very simple: the "band" should cover the maximum amount of points of S^+ and the minimum of S^-. It is important to note that S^- is assumed to be uniformly distributed, and S^+ spreads like a multivariate Gaussian distribution (Section IV). Hence, by projecting S^+ in the direction parallel to its axis of smallest variance, it is possible to get the highest concentration of projections of points of this set, therefore the smallest possible θ value. μ's value is set during the PCA execution together with v. Similarly to [10], p and θ is assigned by the weak learner during the boosting phase.



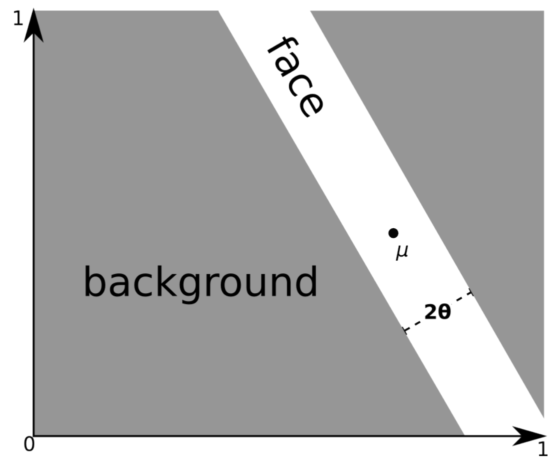


Figure 5. Representation of the effect caused by the proposed feature when combined with a weak classifier in an hypothetical SRFS. To the left, it is represented by a point cloud S^+, its mean μ, and θ. To the right, the effect of the classifier creates on that space if $p = +1$.

B. Feature parameters selection

While running PCA, both v and μ are set for each Haar wavelet. To achieve this, the first step is to produce the SRFS S_k^+ for each Haar-like feature k. Then, for each feature, the mean and the covariance matrix Σ_k^+ are estimated. While the mean is attributed to μ, Σ_k^+'s eigenvector of smallest eigenvalue is calculated and assigned to v_k. v_k must be normalized. This procedure is shown in Figure 6.

V. Experiments

This research hypothesis was experimentally verified. In order to do this, three monolithic strong classifiers (each one with 200 weak classifiers) were boosted. The first of them is similar to Viola and Jones' classifier; the second had only its weights assigned via PCA; and the third used the feature proposed in this paper, as seen in Section IV-A, with the relevant parameters set with the procedure shown in Section IV-B. Although the weights must be different, the rectangle templates used for each Haar wavelet were the same. Additionally, the same face and background images were used to boost all classifiers.

```
Input: $x_i \in X^+$, the set of positive instances.
Input: $w_k, k = 1, \ldots, K$, the Haar-like features.
Output: $v_k$, the new weight vectors $w_k$.
Output: $\mu_k$, the mean of $S_k^+$.
begin
    foreach $w_k$ do
        Use $w_k$ to produce $S_k^+$ for every $X^+$
        Estimate $\mu_k$ and $\Sigma_k$, respectively mean and
        covariance matrix of $S_k$
        Find $\Sigma_k$ eigenvalues and corresponding
        eigenvectors.
        Assign to $v_k$ the eigenvector of smallest
        eigenvalue of $\Sigma_k$
    end
end
```

Figure 6. Assignment of v_k and μ_k to Haar-like feature k using PCA.

A. Positive and negative instances

Four different available face databases were used to create the positive instances dataset needed for all training procedures: the MIT-CBCL Face Database #1 [24]; the BioId face database [25]; the FEI Face Database [26]; and the AR Face Database [27]. A total of 4,938 faces were automatically extracted with programs especially designed to adequate each image to this requirements of the proposed method. Figure 7 shows some faces present in the trainning dataset.

The MIT-CBCL Face Database #1 contains 2,429 images in grayscale and width and height of 19 pixels of faces looking straight forward. It is the dataset that best fits this requirements for the present experiments because it only needed to be rescaled to 20 pixels.

The BioID Face Database contains 1,521 grayscale images with 384 pixels wide and 284 high. In general, subjects in this database are looking to the camera, but show some variations in facial expressions and face rotation, tilt and yaw. A file describing the position of the eyes accompanies each image. By using such descriptor, a program automatically estimated the position and rotation of the face and then aligned it with the horizontal, cut and rescaled the face region.

The original version of FEI Face Database contains color photographs of 640×480 pixels of 200 people (100 of each

Figure 7. Excerpt from the positive instances dataset mixing faces extracted from publicly available datasets.

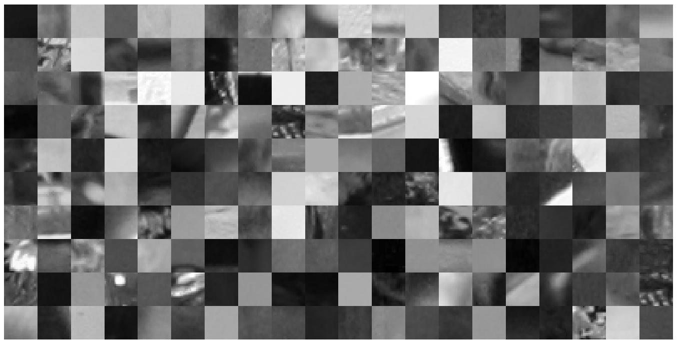


Figure 8. Excerpt of 200 samples from the negative instances dataset used to boost the three strong classifiers.

sex) in 14 different poses and lighting conditions looking straight to the camera in a controlled environment. Some other works were already made over FEI's images, so there are derivations of the original database. For the present experiments, the chosen version contains images in grayscale 250 pixels wide and 300 high of the same 200 individuals looking straight forward but in two poses: relaxed and smiling. The eyes were already aligned as described in [28].

The AR Face Database contains pictures of 126 faces taken on two different days and in controlled conditions. From a single person in a single day, 13 different photographs were taken, each with a particular facial expression, or with the subject wearing a particular accessory, or under certain lighting conditions. All subjects were looking straight forward. Only the subsets 1, 2, 3, 4, 7 and 8 were used. The extraction process is very similar to the one employed in the FEI Database, since these images from the dataset also had the eyes aligned.

Concerning the negative instances, a total of 114,865 samples were taken from around 2,000 digital colored pictures of nature, animals, landscapes, buildings, architecture, paintings, sculptures and people. From those samples, 6,000 were randomly chosen to be used in the classifier boosting procedure. Many of the original pictures had faces which were manually removed with the aid of an application specially designed for this purpose. Other parts of the human body, including hands, hair, feet, clothing and accessories, were not removed. Samples from rough artistic reproductions of the human face were also left in this dataset. Examples of the negative instances can be seen in Figure 8.

B. Haar wavelet set

The Haar wavelets rectangles size and position were chosen according to the same rules mentioned in [7]:

1) only 2 to 4 rectangles can be combined in a Haar wavelet;
2) the template of each Haar wavelet must fit in 20×20 pixels window;
3) rotated features like those proposed in [15] must not be used;
4) distances dx and dy between rectangles, as described in [16], are integer numbers multiples of the rectangle size in the respective directions;
5) all rectangles of a Haar wavelet have the same height and width;
6) the minimum sizes of any rectangle is 3×3 pixels.

By strictly following these rules, a total of 1,641,107 Haar wavelets were generated. From this set, only 218,544 were selected through the application of some additional rules.

C. The face detector operation

The face detector examines the test images, as shown in [10]: moving $[\Delta s]$ pixels in the vertical or horizontal direction, where s is a factor that scales the size of the detector itself. After scanning the whole image, the detector window size is increased by 25%, and the image is scanned again. This repeats until the detector is bigger than one of the image's sides. The initial value of those parameters are: $\Delta = 1.5$ and the initial scale is 1.5. The initial detector sub-window size is 20×20 pixels.

There are some ways to determine if a sub-window had been correctly classified. In [29], a face is considered correctly detected if the detected region contains all face annotations (eyes, nose and mouth) and the size of the detector is smaller than four times the distance between the eyes. In [7], a detection is considered correct when the size of the detected region is $\pm 10\%$ of the annotated face, and when the distance from the center of the detected region to the center of the annotated region is at most 10% of the size of the annotation. In the method proposed here, Pavani's approach is applied. The face region is calculated from the annotated eyes that comes with the test dataset. The height and width of a face region is 1.9402 times the annotated distance between the eyes. The region's top left point was positioned 0.2423 times the region width to the right of the annotated right eye, and 0.25 times the width of the window above the right eye. These were the same parameters used to extract faces from the BioId database. No detection sub-window integration was made.

The detector also does some pre-processing of the image. Viola and Jones' detector performed variance normalization on the image sub-window prior to evaluating them [10], but the authors did not clearly mention the parameters they used. Hence, the normalization implemented for the traditional Viola-Jones' detector was based in [15]. In [7], the images were intensity normalized, i.e., each pixel value was divided by the maximum value they could admit. The detectors that had their v vectors assigned through PCA employed this normalization procedure.

D. Results

The tools, training and testing software developed for this research were written in C++. All image manipulation operations were made with OpenCV [30], although certain image loading procedures were written with OpenImageIO [31]. Some algorithms were easily parallelized with the Intel Threading Building Blocks library [32]. The PCA optimization uses a slightly modified version of libpca [33] and the Armadillo [34] library. Some modules of Boost C++ Libraries [35] were also used for many different tasks.

Three monolithic detectors with 200 features were trained with the same datasets and under very similar conditions. The Haar wavelets rectangles' layouts were all the same, even though its weights were different. More specifically:

1) the first detector used the typical weights assigned to rectangles and was trained and operated exactly as described in [10];
2) the second detector had its weighs set via PCA, but did not use S^+ means for any purpose. Also, the images it scanned were intensity normalized;
3) the third detector was trained as proposed here, with every weight of the Haar-like features set as shown in the Algorithm 6. The feature values were calculated as described in Section IV-A) and images were also intensity normalized.

Detector (1) works as the control experiment while detectors (2) and (3) serve as means to verify the hypothesis. Monolithic classifiers were used instead of a cascade because they suffice to the present research's intention to further investigate the SRFS.

It took 4 minutes to run Algorithm 6 for all the 218,544 Haar-like features, each one consuming the 4,938 positive instances on a Intel Core i7 machine with 4 GiB of RAM memory. This algorithm ran in parallel using all the processor's cores. In the same machine, each boosting procedure took from 9 to 10 hours with the weak learner also running in parallel.

The detectors were tested against the MIT + CMU A, B and C datasets [18] [36], which contain pictures of many subjects whose faces are generally looking forward. A total of 19,024,094 sub-windows were scanned, and the detection acceptance criteria turned the 511 face annotations in 2,571 possible true positive sub-windows. Section V-C describes in details both the scanning method and the acceptance criteria.

The 200 feature detector ROC curve created by altering the detector threshold from $-\infty$ to $+\infty$, as shown in [37], is plotted in Figure 9. Considering the following: a) the three detectors performances; b) all weak classifiers were candidates to be part of the strong classifier in every iteration round; and c) the assumption that the Gaussian distribution is a good model for S^+'s distribution; then it is reasonable to conclude that the uniform distribution does not model adequately how S^- spread over the SRFS. This is an interesting observation since the data available about S^- suggest that it can spread itself in very different and chaotic ways.

The detectors using the proposed method have overfitted: their ROC curves near the perfect classification when tested against the training dataset. An exception to this occurs if the detector had its weights set via PCA and used the proposed weak classifier. In this case, the weak classifiers θ parameter was set to very small values, causing the "band" to be too thin, and allowing too many misclassifications to occur. The main cause of this problem is probably the lack of information about the negative instances in the weak classifiers. Indeed, when assuming that the negative instances behave as uniform distributions in the SRFS, one makes it impossible to use any additional information about the negative instances. On the other hand, the simplicity and speed of the training method proposed here are surely too compelling to be left untested. These observations points future researches towards the usage of weak classifiers that carries with them more information about the distributions of both classes in the SRFS.

It also seems that the rotations made in the BioId dataset and the choice of the AR datasets made the positive instances be too similar to each other, aggravating the overfitting. It is possible that the background instances used to boost the classifiers was relatively small. This observation comes from the comparison of the performance of the original Viola-Jones monolithic detector with the one produced in this work. Additional evaluations are being made in order to create a more diverse face database and to fine-tune the boosting parameters.

VI. Conclusion and future work

In this paper, a new weight adjustment method for Haar-like features (complementary to the ones shown by Pavani and collaborator's [7]) was proposed. This method uses PCA over the positive training instances to assign new weights to the features. It was also proposed the employment of a Haar-like feature that operates with parameters estimated from the same positive instances.

Both the weight assignment method and the Haar-like feature were designed to verify if the distribution of points produced from the negative instances in the SRFS of each Haar-like feature could be modeled as an uniform distribution. The motivation behind this, besides shedding light in a somewhat still unexplored concept, was the simplicity and speed of the proposed weight assignment method.

The weight adjustment method as well as the Haar-like feature were tested in the face detection task and compared with the Viola-Jones detector. Although the negative instances may spread themselves in very different and chaotic ways through the SRFS, the obtained results suggest that they cannot be properly modeled as an uniform distribution. This interesting finding is the most important contribution of this paper.

The future works concern to evolve the methods and classifiers to more complex ones while still exploring and bringing understanding about the SRFS. Experiments with other processes of feature weight adjustments using information of both positive and negative training instances are ongoing.

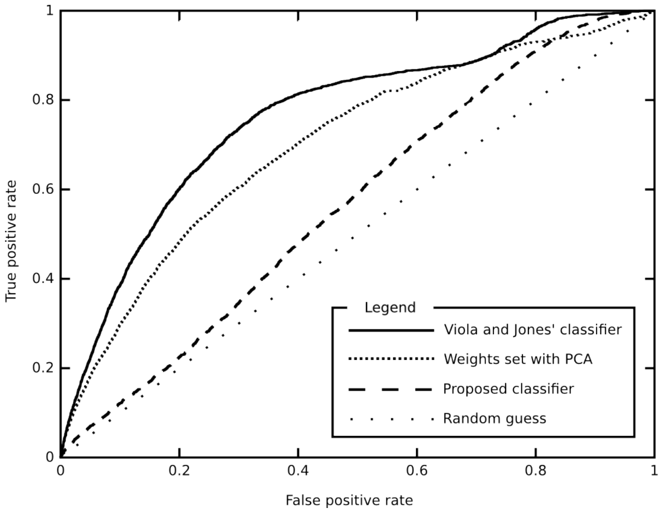


Figure 9. ROC curves of the detectors when tested with the MIT + CMU dataset. The vertical axis shows the true positive rate and the horizontal axis shows the false negative rate.

References

[1] C. Zhang and Z. Zhang, "A survey of recent advances in face detection," Microsoft Research, Technical Report MSR-TR-2010-66, 2010.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1, 2001, pp. I–511–I–518 vol.1.

[3] J. Dembski, "Feature type and size selection for adaboost face detection algorithm," in Image Processing and Communications Challenges 2, R. Choraś, Ed. Springer Berlin Heidelberg, 2010, vol. 84, ch. Advances in Intelligent and Soft Computing, pp. 143–149.

[4] F. Baumann, K. Ernst, A. Ehlers, and B. Rosenhahn, "Symmetry enhanced adaboost," in Advances in Visual Computing, G. Bebis, R. Boyle, B. Parvin, D. Koracin, R. Chung, R. Hammoud, M. Hussain, T. Kar-Han, R. Crawfis, D. Thalmann, D. Kao, and L. Avila, Eds. Springer Berlin Heidelberg, 2010, vol. 6453, ch. Lecture Notes in Computer Science, pp. 286–295.

[5] I. Landesa-Vazquez and J. L. Alba-Castro, "The role of polarity in haar-like features for face detection," in Proceedings of the 2010 20th International Conference on Pattern Recognition, ser. ICPR '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 412–415.

[6] S. Vural, Y. Mae, H. Uvet, and T. Arai, "Multi-view fast object detection by using extended haar filters in uncontrolled environments," Pattern Recognition Letters, vol. 33, no. 2, 2012, pp. 126–133.

[7] S.-K. Pavani, D. Delgado, and A. F. Frangi, "Haar-like features with optimally weighted rectangles for rapid object detection," Pattern Recognition, vol. 43, no. 1, Jan 2010, pp. 160–172.

[8] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in Proceedings of the Second European Conference on Computational Learning Theory, ser. EuroCOLT '95. London, UK: Springer-Verlag, 1995, pp. 23–37.

[9] R. E. Schapire, "The boosting approach to machine learning: An overview," Lecture Notes in Statistics, 2003, pp. 149–172.

[10] P. Viola and M. J. Jones, "Robust real-time face detection," International Journal of Computer Vision, vol. 57, no. 2, May 2004, pp. 137–154.

[11] H. A. Rowley, S. Member, S. Baluja, and T. Kanade, "Neural network-based face detection," IEEE Transactions On Pattern Analysis and Machine intelligence, vol. 20, 1998, pp. 23–38.

[12] S. Baker and S. K. Nayar, "Pattern rejection," in Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on, 1996, pp. 544–549.

[13] A. Haar, "Zur theorie der orthogonalen funktionensysteme [On the Theory of Orthogonal Function Systems]," Mathematische Annalen, vol. 71, no. 1, 1911, pp. 38–53.

[14] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in Computer Vision, 1998. Sixth International Conference on, ser. ICCV '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 555–562.

[15] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in IEEE ICIP 2002, vol. 1, 2002, pp. 900–903.

[16] S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum, "Statistical learning of multi-view face detection," in In Proceedings of the 7th European Conference on Computer Vision, 2002, pp. 67–81.

[17] M. Jones and P. Viola, "Fast multi-view face detection," Mitsubishi Electric Research Lab TR-20003-96, vol. 3, 2003, p. 14.

[18] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," in Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on, 1996, pp. 203–208.

[19] H. Schneiderman and T. Kanade, "A statistical method for 3d object detection applied to faces and cars," in Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, vol. 1, 2000, pp. 746–751 vol.1.

[20] M. Yang, D. Roth, and N. Ahuja, "A SNoW-based face detector," in Advances in Neural Information Processing Systems 12. MIT Press, 2000, pp. 855–861.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2nd ed. New York: Wiley-Interscience, 2001.

[22] S.-K. Pavani, D. Delgado Gomez, and A. Frangi, "Gaussian weak classifiers based on haar-like features with four rectangles for real-time face detection," in Computer Analysis of Images and Patterns, ser. Lecture Notes in Computer Science, X. Jiang and N. Petkov, Eds. Springer Berlin Heidelberg, 2009, vol. 5702, pp. 91–98.

[23] J. Shen, C. Sun, W. Yang, Z. Wang, and Z. Sun, "A novel distribution-based feature for rapid object detection," Neurocomputing, vol. 74, no. 17, Oct 2011, pp. 2767–2779.

[24] MIT Center for Biological and Computation Learning, "CBCL database # 1," http://www.ai.mit.edu/projects/cbcl, retrieved: June, 2013.

[25] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the hausdorff distance." Springer, 2001, pp. 90–95.

[26] L. L. de Oliveira Junior and C. E. Thomaz, "Captura e alinhamento de imagens: Um banco de faces brasileiro [Capture and Alignment of Images: a Brazilian Face Database]," Departamento de Engenharia Elétrica, FEI, São Bernardo do Campo, São Paulo, Brazil, Tech. Rep., Jun 2006.

[27] A. Martínez and R. Benavente, "The AR face database," CVC, Tech. Rep. 24, Jun 1998.

[28] V. Amaral and C. E. Thomaz, "Normalização espacial de imagens frontais de face [Spatial Normalization of Frontal Face Images]," Departamento de Engenharia Elétrica, FEI, São Bernardo do Campo, São Paulo, Brazil, Tech. Rep. 1, 2008.

[29] M. Castrillón, O. Déniz, D. Hernández, and J. Lorenzo, "A comparison of face and facial feature detectors based on the Viola–Jones general object detection framework," Machine Vision and Applications, vol. 22, no. 3, 2011, pp. 481–494.

[30] Itseez, "OpenCV," http://opencv.org/.

[31] L. Gritz, "OpenImageIO," https://sites.google.com/site/openimageio/home.

[32] Intel Corporation, "Intel Threading Building Blocks," https://www.threadingbuildingblocks.org/, retrieved: August, 2013.

[33] C. Blume, "libpca C++ library," http://sourceforge.net/projects/libpca/, retrieved: September, 2013.

[34] C. Sanderson, "Armadillo: An open source C++ linear algebra library for fast prototyping and computationally intensive experiments," NICTA, Technical Report, 2010.

[35] Boost Community, "Boost C++ libraries," http://www.boost.org/, retrieved: September, 2013.

[36] K.-K. Sung and T. Poggio, "Example-based learning for view-based human face detection," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 20, no. 1, 1998, pp. 39–51.

[37] T. Fawcett, "An introduction to ROC analysis," Pattern Recognition Letters, vol. 27, no. 8, Jun. 2006, pp. 861 – 874.