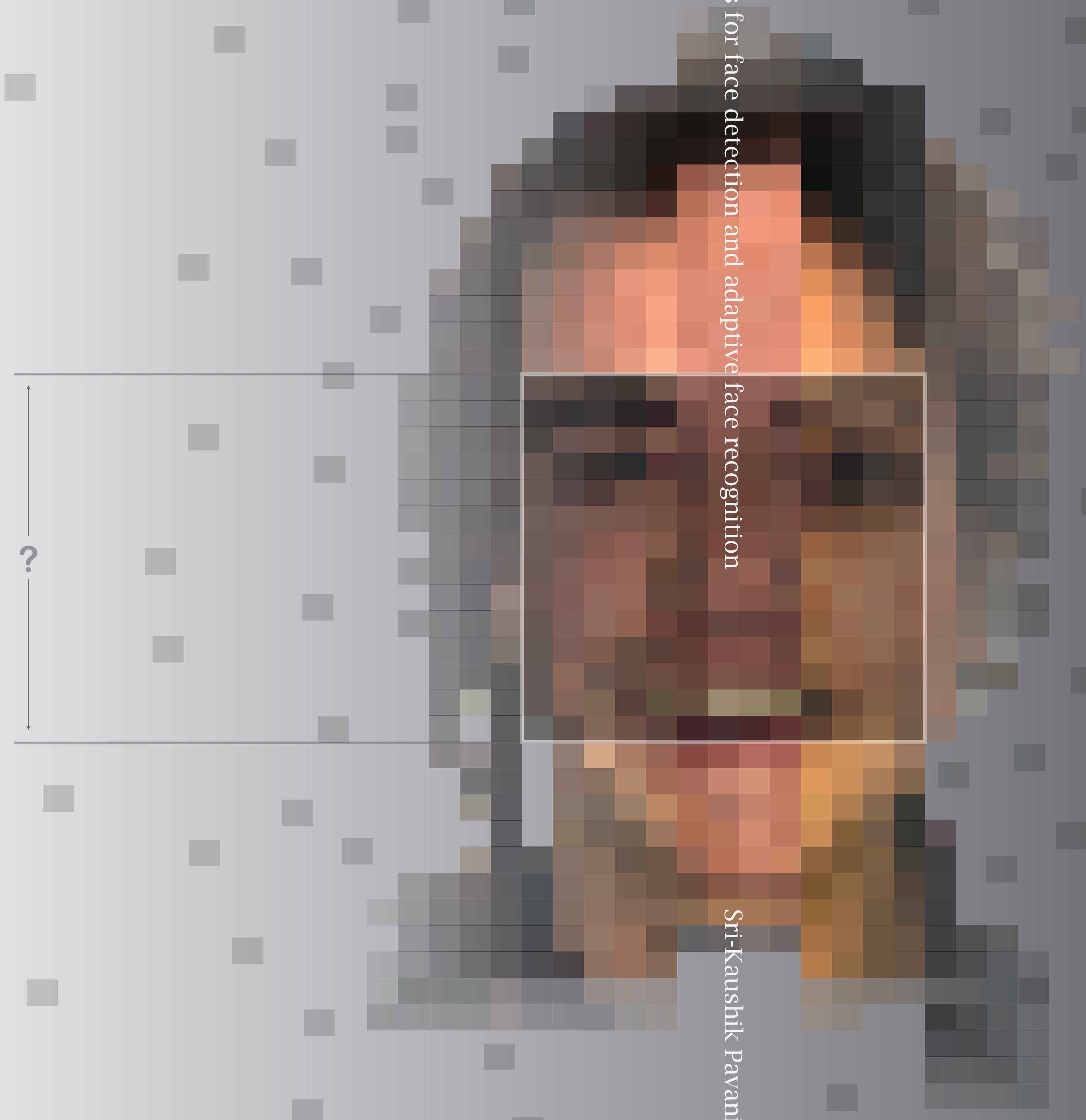
# Methods for face detection and adaptive face recognition

Sri-Kaushik Pavani

Methods for face detection and adaptive face recognition Sri-Kaushik Pavani 2010



ISBN 9789081059237

# Methods for face detection and adaptive face recognition

Sri-Kaushik Pavani

Methods for face detection and adaptive face recognition
Sri-Kaushik Pavani

ISBN: 9789081059237
Depósito legal: DLB-29799-2010
Typeset in $\LaTeX\,2_\varepsilon$, generated by pdfTeX ver. 3.1415926-1.40.10-2.2.
Printed by Agpograf S.A. Barcelona, Spain.

Thesis class courtesy of Dr. Avan Suinesiaputra.

# Methods for face detection and adaptive face recognition

A thesis submitted by Sri-Kaushik Pavani in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Departament de Tecnologies de la Informació i les Comunicacions
Universitat Pompeu Fabra

2010

Supervisors: Dr. Alejandro F. Frangi
Universitat Pompeu Fabra

Dr. David Delgado-Gomez
Universidad Carlos III de Madrid

This work was carried out in the Center for Computational Imaging and Simulation Technologies in Biomedicine (CISTIB).
Dissertation series number 4.

This work was partially funded by the grants *TIC2002-04495-C02* & *TEC2006-03617/TCM* from the Spanish Ministry of Education & Science, *FIT-360000-2006-55* & *FIT-360005-2007-9* from the Spanish Ministry of Industry, and TIN2009-14536-C02-01 from Spanish Ministry of Science & Innovation.
Financial support for the publication of this thesis was kindly provided by CISTIB and by UPF.

# ABSTRACT / RESUM

The focus of this thesis is on facial biometrics; specifically in the problems of face detection and face recognition. Despite intensive research over the last 20 years, the technology is not foolproof, which is why we do not see use of face recognition systems in critical sectors such as banking. In this thesis, we focus on three sub-problems in these two areas of research. Firstly, we propose methods to improve the speed-accuracy trade-off of the state-of-the-art face detector. Secondly, we consider a problem that is often ignored in the literature: to decrease the training time of the detectors. We propose two techniques to this end. Thirdly, we present a detailed large-scale study on self-updating face recognition systems in an attempt to answer if continuously changing facial appearance can be learnt automatically.

L'objectiu d'aquesta tesi és sobre biometria facial, específicament en els problemes de detecció de rostres i reconeixement facial. Malgrat la intensa recerca durant els últims 20 anys, la tecnologia no és infalible, de manera que no veiem l'ús dels sistemes de reconeixement de rostres en sectors crítics com la banca. En aquesta tesi, ens centrem en tres sub-problemes en aquestes dues àrees de recerca. En primer lloc, es proposa mètodes per millorar l' equilibri entre la precisió i la velocitat del detector de cares d'última generació. En segon lloc, considerem un problema que sovint s'ignora en la literatura: disminuir el temps de formació dels detectors. Es proposen dues tècniques per a aquest fi. En tercer lloc, es presenta un estudi detallat a gran escala sobre l'auto-actualització dels sistemes de reconeixement facial en un intent de respondre si el canvi constant de l'aparença facial es pot aprendre de forma automàtica.

# CONTENTS

# List of Figures

# List of Tables

# PREFACE AND ACKNOWLEDGEMENTS

Everything that can be invented has been invented

Attributed to *Charles H. Duell*

So, why another thesis on facial biometrics? Did you see new Sony camera, how does your detector compare with it? How can you compete with the industry? Some of the questions I have had to face over the last four and a half years ... Indeed, Biometrics is a challenging and an active field; new papers are churned out faster than pancakes, *literally*. The state-of-the-art is not only pushed to its limits by research labs, but also by the cash-rich industry. Research has been going on for decades, and at first glance, it does seem all that needs to be invented has been invented. The Biometrics consortium web site sums it all up:

> *"In the early days of biometrics, a publication or article on anything 'biometric' was cause for celebration. Recently, we have stopped listing individual publications and articles on biometrics as the sheer number of items would quickly overwhelm our site."*

Looking at the other side of the coin, the very reason there is so much funding and research in biometrics is a testament to the amount of research that is left to do. As my supervisor would say, research is all about critically analyzing the state-of-the-art, and solutions to improve it will emerge as a result. In a sense, this has been the story of this thesis.

Thanks to Dr. Alejandro F. Frangi for inviting me to work as a software developer in his lab, where I was initiated to Biometrics. Working for a year on implementing state-of-the-art facial biometric algorithms gave us sufficient confidence to identify a niche where improvements could be made. As his student, I have benefited from his knowledge, his critical eye and most importantly, his strive for perfection.

Dr. David Delgado-Gomez has been my day-to-day discussions guy. His patience in revising my first paper was phenomenal. I sincerely think, without his help, I would still have difficulty in publishing it. He has made it so easy for me to discuss even some of my craziest ideas, and we more often than not, knock the crazy out of it.

With Dr. Federico M. Sukno, Dr. Constantine Butakoff and Xavier Planes, there was a effective work flow in the Biometrics section. We had fun making facial biometrics demonstrator, *el mono*. Thanks to them for fighting for the life of Biometrics section in the lab, without their consistent effort, this thesis would not exist. A special mention of thanks to Dr. Sukno and Dr. Butakoff, they have been instrumental in writing new projects and securing funds. Their work ethic is contagious.

Thanks to all the current and the past members of CISTIB; working with them has been fun and intellectually challenging. The informal brain storming sessions we have had has basically transformed itself into this thesis. The management section in CISTIB, contrary to popular jokes, has been very useful. Be it booking a flight ticket in the last minute, or renewing the work permit, they have provided me with immediate support. Thanks to my SOS guy, Milton Hoz de Vila, for his immediate help on numerous occasions. Right from the day one, when he helped me to get the NIE card, to the day when we had to call him at 12 in the night for a lost password, and until last week when he immediately replaced a burnt graphics card. Thanks for everything!

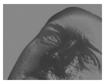
For me, a productive week starts with a good weekend. And, as most would agree, a good weekend does not exist without good friends. The laughs I have shared with my *kuzhandhai*, Renuka Kandhaya Pillai and Dr. Chandrashekaran Gurunathan have made writing the thesis enjoyable.

I would run out of space to thank my *amma* and *nana*, who have basically provided all that I wanted. This thesis is for both of you. My brother, Dr. Sri Rama Prasanna Pavani, was a great source of inspiration as well.

Thanks to the grants *TIC2002-04495-C02* & *TEC2006-03617/TCM* from the Spanish Ministry of Education & Science, *FIT-360000-2006-55* & *FIT-360005-2007-9* from the Spanish Ministry of Industry, and TIN2009-14536-C02-01 from Spanish Ministry of Science & Innovation. They have partially funded this work. Thanks to the Biosecure (*IST-2002-507534*, European NoE), which provided opportunity to work with researchers from other labs.

# 1

# Introduction



Racist Camera! No, I did not blink... I'm just Asian!

*Joz Wang*

Privacy advocates may complain, but in the post 9/11 world, it is hardly possible to commute to work without being captured by a CCTV[1] camera in most major cities. They have been quite effective as a crime-deterrent and to prosecute criminals. A Google search on "CCTV footage" lists the success stories. Their effectiveness in stopping an ongoing crime is, however, limited by one factor: the human operator monitoring the video. Although humans are well adept in identifying intruders, their ability is limited by their attentiveness. Here is where computer-based techniques can help an operator in answering *Who is the individual*? or even the watered-down question *Is there anyone*?

Nature has made each one of us unique in one way or the other. Biometric algorithms try to exploit the uniqueness in order to make a computer identify an individual under *observation*. An individual can be observed in different forms - through the now-ubiquitous cameras, through his writings or through voice recorders. The challenge is to make the computer reliably and repeatedly make correct decisions.

Depending on the characteristic observed, biometrics can be categorized into several modalities: face, palm, iris, handwriting, speech, voice, gait, fingerprint or ear. Facial biometrics, the focus of this thesis, is an intuitive identification modality, as humans are well adept in recognizing people using their faces. A major advantage in using faces is that a facial image can be captured from far away, and therefore, it does not need the cooperation of the individuals under observation. This makes facial biometrics, in principle, ideal for surveillance applications.

Apart from applications in surveillance, facial biometric techniques have been in widely adopted in consumer electronics over the past 2-3 years. Even the low-end digital cameras come pre-installed with perfectly working face detectors. They *click* automatically when people smile. Japanese advertising boards, having taken a cue from the movie *Minority Report*, have started serving gender/age-specific ads by making an educated guess about who is currently looking at the advertisement board (`www.telegraph.co.uk`, March 10, 2010). Cigarette vending machines, by deciding the approximate age of the buyer, autonomously decide if the cigarette should be sold or not (`www.japanprobe.com`, August 07, 2008). Not to be left out, the multi-billion dollar gaming industry is set to cash in on facial biometrics based controllers. The first games are already in the market (`www.engadget.com`, July 18, 2009).

If success in the market is an indicator of the maturity of the technology, has the research in biometrics reached a steady-state? The answer is a clear *no*. Parts of the technology are far from being marketable. For example, face recognition is not yet used in critical applications, such as in banks, where valuable data might be stored. The reason being the technology is not in foolproof yet. The face recognition technology in the popular Picasa albums is known to make occasional mistakes (`www.brighthub.com`, March

[1]Closed-circuit television

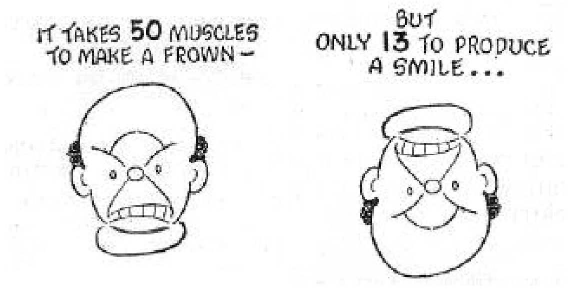

IT TAKES 50 MUSCLES TO MAKE A FROWN –
BUT ONLY 13 TO PRODUCE A SMILE...

Figure 1.1: If perception of a face can change drastically to a well trained human eye with a simple rotation of an image, can we make a computer interpret the image correctly?

25, 2010). Facial biometry based computer login programs from popular vendors such as Lenovo®, Asus®and Toshiba®- each set to its highest security level - were hacked easily with phony photos of the legitimate user and gain access to the laptops (BlackHat DC Security Conference, 2009).

So, why is it that computers are still bad in identifying faces...? Humans seem to have incredibly powerful "facial transformation-invariant filters" that make us recognize known people under variety of conditions. It does not matter if a good part of a friend's face is occluded by a sunglass and a bandana, his/her name pops up immediately. Similarly, it does not matter if he/she is in a discotheque with variety of colors reflected on the face, or in the middle of the night in dark lighting conditions. Add to that the fact that we can recognize quite effectively negating the effects of in-plane/out-of-plane rotations, effects of expression changes, ageing or the size of the face. Obviously, if the computers are to convince a human with their recognition capabilities, they need effective filters that basically mimic the performance of the human brain. Mimicking the brain turns out to be a big ask, if only we knew how they work! Generally, in computer based approaches, the process of recognizing a face is split into two sub-tasks:

- Firstly, we need to determine if there is a face in the image or not. This seemingly easy problem has been a hot topic of research for well over 20 years [15] - without being able to generate a commercially usable detector. The difficulty was in achieving an accurate detector which would work in real time. Methods based on skin color detection were fast, however, their accuracy was poor [4][2]. Appearance based methods that used Neural networks [11] and Support vector machines [8] were accurate, but slow. The first commercially usable detector was invented by Viola and Jones in 2001 [12]. Their detector worked in real-time while achieving good accuracy rates of up to 92% correct detection rates in the challenging MIT+CMU database [11]. After Viola and Jones's work, many modifications were proposed,

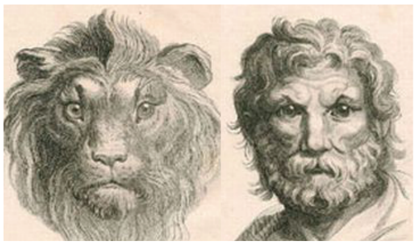


Figure 1.2: How dissimilar are these two faces?

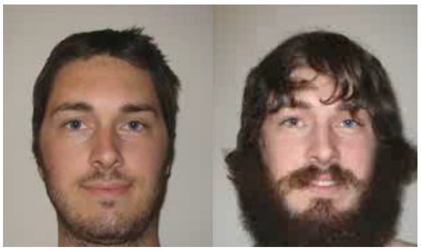


Figure 1.3: How similar are these two faces?

mainly to improve the speed-accuracy trade-off of their detector. Till this day, numerous patents on techniques based on Viola and Jones's proposal are being filed by leading camera companies, which is a strong indicator of the popularity of their approach. Despite the popularity, there is avenue of further research in this area. Challenges include improving accuracy and making the detector work in low power consuming processors. A recent viral hit in YouTube (December 11, 2009) "HP computers are racist", shows clearly where the face detectors could be improved: In its ability to correctly detect people from different ethnicities.

- Secondly, if there are face(s) in the image, the computer should be able to answer the question *who is the person in the image?* Currently, this problem seems far from being solved. Although, research has been going on for more than 30 years [16], the progress achieved is hardly applicable in real-life scenarios. Mostly, the proposed algorithms are tested in image databases captured in controlled conditions. This is achieved by controlling person's facial expression, his/her distance from the camera, the camera angle, and the scene's lighting. Now, even under these simplified conditions, recognition performance of computers is not perfect. Independent studies performed using FRVT[2] 2006 database have shown that even the best recognition algorithms make recognition errors on high-resolution image acquired in controlled conditions [9]. The disconcerting fact is that these tests were performed on a small database with as low as 366 users. It can only be assumed that with more number of users, the recognition performance will worsen. Progress has made performed in normalizing illumination [14][1], in normalizing pose [3], removing occlusions [13][5] or to determine liveliness of a face [6][7]; however, the challenge still lies in successfully integrating all the blocks together. There are two principal schools of thought to improve the identification performance. Firstly, researchers are working on improving the invariance filters such as reducing the effects on illumination, pose, size of the face, and the recognition algorithm itself.

[2]Face Recognition Vendor Test, www.frvt.org/FRVT2006/

Secondly, multi-modal recognition techniques [10] are being implemented, where inputs from two or more modalities, say face and hand, are input to the computer and then a decision is made using both the inputs, thus reducing probability of making error.

In this thesis, we propose improvements to the state-of-the-art that address some of the open issues in face detection and in recognition. The proposals made this thesis have been grouped into three chapters.

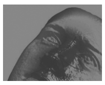
1. The first chapter focuses on optimizing three important performance indicators of the face detection system: the testing speed, the true-positive rate and the false-positive rate. In general, one can say that the speed is inversely proportional to the accuracy, *i.e.,* obtaining high true-positive rate (TPR) and low false-positive rate (FPR). For a given technique, the trick is to operate the detector at the point where one gets the optimal speed-accuracy trade-off. In this chapter, we propose two modifications to the Viola and Jones's object detection system, that makes it possible to choose a better trade-off between speed and accuracy. The first proposal we make is detailed in Section 2.1, where we propose assigning optimal weights to Haar-like features in order to obtain more powerful features. In Section 2.2, we propose an optimal way of fusing Joint Haar-like features to achieve more powerful weak classifiers. Our results, were tested on the standard MIT+CMU face test database, showed that, in comparison to Viola and Jones's technique, better speed-accuracy tradeoff could be obtained.

2. In the second chapter, we propose two techniques that, along with speed, TPR and FPR, optimizes the training speed of the Viola and Jones's type detectors. This is a fairly ignored problem in the literature. Though Viola and Jones's technique has been very popular and widely used, training the detector is a very time-consuming task. One may need to wait for days, if not weeks, to get the "face" model. Of course, once it is generated, the benefits of the system are many. There are some applications when such long training times are prohibitive. For example, in an emergency, if the security guard needs to retrieve image frames with a particular object of interest, the guard cannot wait two or three days to get the "object" model, and then use the detector. He/she would do better is manual search is done through the video frames. The proposed techniques (Section 3.1 and Section 3.2), reduces the training time from the order of days and weeks to a matter of minutes, thus provides an almost instant search capability.

3. In the third chapter, we study self-updating face recognition systems - a topic which has attracted much interest in the research community. The need for making a face recognition system "self-updating" arises because people's facial appearance changes with time due to factors such as ageing, hair-growth (Fig. 1.3), sun-tan, health factors *etc.* When a face starts looking differently, the recognition system may wrongly label the face as some other user, or it may deny access by labeling a genuine user as an impostor - both of which are undesirable. One can manually enroll the user again, but: can computers learn the appearance changes gradually? If so,

under what conditions? Can we guarantee autonomous learning without human supervision? These are the questions that are answered in this chapter. Conclusions are reached by making a large scale study using two large and challenging databases, and using three popular face recognition algorithms.

Each chapter is made self-contained, therefore, some of the basic concepts might be repeated in this manuscript. Happy reading!



## 1.1 References

[1] T Chen, W Yin, X S Zhou, D Comaniciu, and T S Huang. Illumination normalization for face recognition and uneven background correction using total variation based image models. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2005.

[2] J L Crowley and F Berard. Multi-modal tracking of faces for video communications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 640–645, Washington, DC, USA, 1997. IEEE Computer Society.

[3] H Gao, H K Ekenel, and R Stiefelhagen. Pose normalization for local appearance-based face recognition. In *Proceedings of the International Conference on Advances in Biometrics*, pages 32–41, 2009.

[4] T S Jebara and A Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 144–150, 1997.

[5] H Jia and A M Martinez. Support vector machines in face recognition with occlusions. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 136–141, 2009.

[6] K Kollreider, H Fronthaler, and J Bigun. Evaluating liveness by face images and the structure tensor. In *Proceedings of the Fourth Workshop on Automatic Identification Advanced Technologies*, pages 75–80, 2005.

[7] K Kollreider, H Fronthaler, and J Bigun. Non-intrusive liveness detection by face images. *Image and Vision Computing*, 27(3):233–244, 2009.

[8] E Osuna, R Freund, and F Girosi. Training support vector machines: an application to face detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 130–136, Washington, DC, USA, 1997.

[9] P J Phillips, W T Scruggs, A J O'Toole, P J Flynn, K W Bowyer, C L Schott, and M Sharpe. FRVT 2006 and ICE 2006 large-scale experimental results. *IEEE Transactions on Pattern Analysis and Machine Intelligence, In press*, 2009.

[10] A Ross and A K Jain. Multimodal biometrics: An overview. In *Proceedings of the European Signal Processing Conference*, pages 1221–1224, 2004.

[11] H A Rowley, S Baluja, and T Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

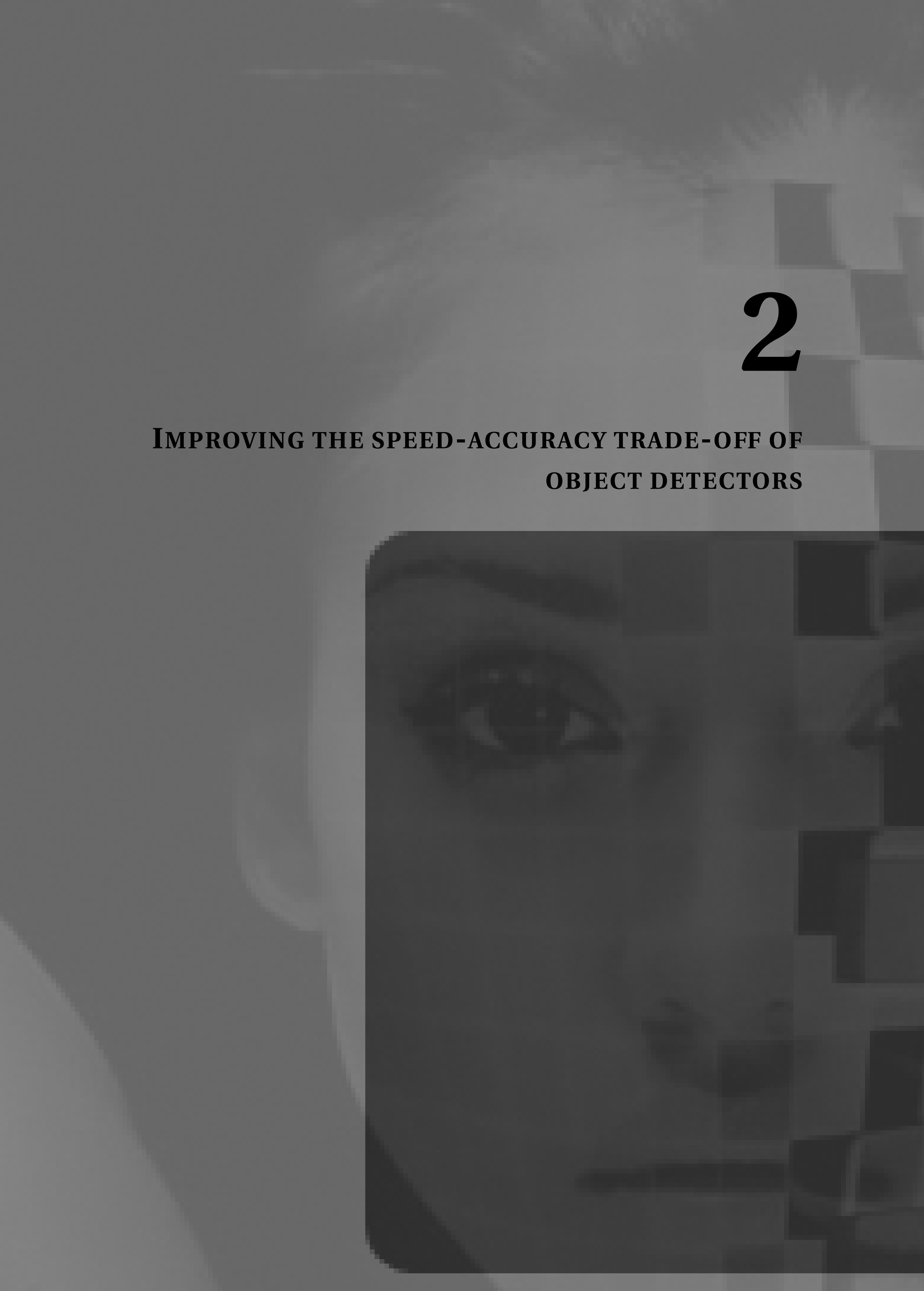
[12] P Viola and M J Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[13] J Wright, A Y Yang, A Ganesh, S S Sastry, and Y Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[14] X Xie and K-M Lam. An efficient illumination normalization method for face recognition. *Pattern Recognition Letters*, 27(6):609–617, 2006.

[15] M H Yang, D J Kriegman, and N Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[16] W Zhao, R Chellappa, P J Phillips, and A Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–458, 2003.



# 2 Improving the speed-accuracy trade-off of object detectors

If it is not fast, we lose interest, after all if we can not become a millionaire in 365 days, we may lose interest in becoming one at all ...

NOT!

*Christopher Jansen*

Viola and Jones's (VJ's) object detection system was not the most accurate detection procedure available when it was proposed. Detectors proposed by Rowley-Baluja-Kanade [29], Roth-Yang-Ahuja [28] and Schneiderman-Kanade [32] were more accurate. Yet, VJ's technique quickly gained popularity because it provided the best speed-accuracy trade-off. VJ's technique works at real time, while, techniques proposed in [29][28][32] were at least 15 times slower. In this chapter, we investigate two methods that improve the speed-accuracy trade-off of the Viola and Jones's object detection systems. Firstly, we propose assigning optimal weights to Haar-like features to maximize their ability to discriminate objects from clutter (non-objects). Secondly, we propose feature-level fusion of co-occurring, or multiple, Haar-like features to construct potentially more discriminative weak classifiers.

Adapted from:

S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, *Haar-like Features with Optimally Weighted Rectangles for Rapid Object Detection*, Pattern Recognition, 43(1): 160-172, 2010

S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, *Gaussian Weak Classifiers based on Co-occurring Haar-like Features for Face Detection.* Submitted, 2010.

S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, *Gaussian Weak Classifiers Based on Haar-Like Features with Four Rectangles for Real-time Face Detection.* In Proc. Computer Analysis of Images and Patterns. Lecture Notes in Computer Science vol. 5702, pages 91-98, 2009.

## 2.1 Haar-like Features with Optimally Weighted Rectangles for Rapid Object Detection

Object detectors aim at finding sub-regions of an image that contain instances of an object of interest. Many applications of object detection are challenging because high accuracy is required while images are evaluated at real-time speeds. Such applications include vehicle detection [34], where one needs to alert automobile drivers about possible accidents as soon as possible, and, surveillance tasks where every video frame needs to be checked in real-time for the presence of intruders [37].

Classifiers based on Haar-like features [25] have demonstrated to be considerably successful for use in object detection tasks. This is mainly due to the fact that they provide an attractive trade-off between accuracy and evaluation speed. Viola and Jones [36] proposed a popular object detection system based on these features. After Viola and Jones's contribution, many successful systems based on Haar-like features have been proposed [18, 22, 38, 40].

In this chapter, we propose modified Haar-like features whose rectangles are assigned optimal weights. These weights are optimal in the sense that they maximize their ability to discriminate object from clutter. Similar to the traditional Haar-like features, the proposed features can be used to form weak classifiers, which in turn can be boosted [30, 31] and arranged in a rejection cascade architecture [4] to form an object detection system. In our experiments, three different techniques, Brute-force search (BFS), Genetic algorithms (GA) [6, 8, 9], and Fisher's linear discriminant analysis (FLDA) [7] were used to determine optimal weights.

The proposed features were trained for two object detection problems: detection of human frontal faces in digital photographs, and detection of cardiac structures from Magnetic Resonance Imaging (MRI). The obtained results show that the object detectors based on the proposed features are more accurate and faster than the object detectors built with traditional Haar-like features.

The remainder of the chapter is organized as follows. In Section 2.1.1, weak classifiers based on Haar-like features are introduced. In Section 2.1.2, algorithms to find optimal rectangle weights using BFS, GA and FLDA are presented. Later, in Section 2.1.3, the boosting procedure to combine weak classifiers to form strong classifiers is described. Section 2.1.4 describes the framework and the training procedure of the proposed object detector. A human frontal face and a heart detector were built based on the proposed object detection system. The results of their evaluation on real-life datasets are presented in Section 2.1.6. The chapter is concluded in Section 2.1.7.

### 2.1.1 Haar-like features

Haar-like features are an over complete set of two-dimensional Haar functions, which can be used to encode local appearance of objects [25]. They consist of two or more rectangular regions enclosed in a template. The feature value $f$ of a Haar-like feature which has $k$ rectangles is obtained as in (2.1).

$$f = \sum_{i=1}^{k} w^{(i)} \cdot \mu^{(i)} \tag{2.1}$$

where $\mu^{(i)}$ is the mean intensity of the pixels in image $\mathbf{x}$ enclosed by the $i^{th}$ rectangle. Henceforth we will refer to the quantity $\mu$ as the *rectangle mean*. In (2.1), $w^{(i)}$ is the weight assigned to the $i^{th}$ rectangle. Traditionally, the weights assigned to the rectangles of a Haar-like feature are set to *default* integer numbers such that (2.2) is satisfied.

$$\sum_{i=1}^{k} w^{(i)} = 0 \tag{2.2}$$

For example, the rectangles of a Haar-like feature as in Fig. 2.1(a) are assigned default weights 1 and −1. Similarly, the rectangles of a Haar-like feature as in Fig. 2.1(c) are assigned default weights 1, −2 and 1.

One of the main reasons for the popularity of the Haar-like features is that they provide a very attractive trade-off between speed of evaluation and accuracy. With a simple weak classifier based on Haar-like features costing just 60 microprocessor instructions, Viola and Jones [36] achieved 1% false negatives and 40% false positives for the face detection problem. The high speed of evaluation is mainly due to the use of integral images [36], which once computed, can be used to rapidly evaluate any Haar-like feature at any scale in constant time.

Since the introduction of horizontally and vertically aligned Haar-like features by Papageogiou *et al.* [25], many different Haar-like features have appeared in the literature [16, 18, 36]. Some of the features are shown in Fig. 2.1. They mainly differ in the number of rectangles and the orientation of the rectangles with respect to the template. Jones and Viola [12] introduced diagonal features to capture diagonal structures in object's appearance. Lienhart [18] enriched the features of Viola and Jones [36] with efficiently computable rotated features. With the enriched set, they achieve a 10% lower false alarm rate for a given true positive rate. Li *et al.* [16] proposed Haar-like features with disjoint rectangles which were meant to characterize non-symmetrical features of an object's appearance.

#### 2.1.1.1 Haar-like Features with Optimally Weighted Rectangles

In this chapter, we argue that a Haar-like feature can be optimized for a given object detection problem by assigning optimal weights to its rectangles. The feature value of the proposed features is computed exactly as in (2.1), except that the default weights of its rectangles, $w^{(i)}$, are substituted with optimal values, $\hat{w}^{(i)}$. The proposed features maintain the simplicity of the traditional ones as in (2.1), while being more discriminative.

#### 2.1.1.2 Weak classifiers

Weak classifiers label a sub-region of an image, $\mathbf{x}$, as belonging to either the object or clutter class by comparing $f$ to a threshold $\theta$ as in (2.3). They are called weak because they are expected to perform only slightly better than a random guesser.

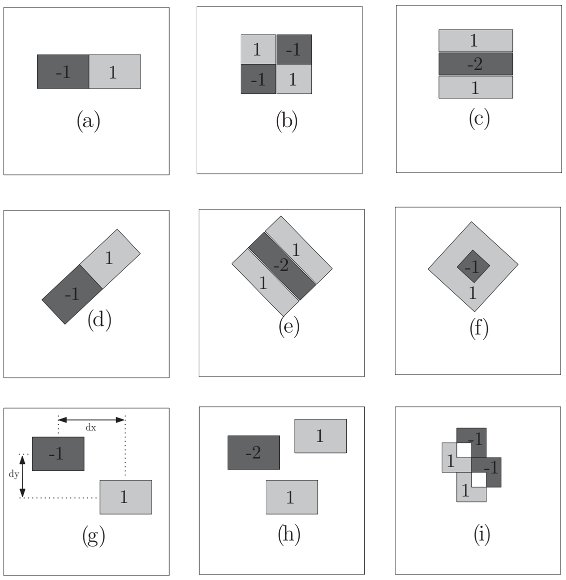


Figure 2.1: Haar-like features are shown with the default weights assigned to its rectangles. (a) and (b) show Haar-like features proposed by Papageogiou *et al.* (c) shows a Haar-like feature with three rectangles introduced by Viola and Jones. (d-f) show Leinhardt's rotated features. (g) and (h) show Li *et al.*'s disjoint Haar-like features. (i) shows Jones and Viola's diagonal feature designed to capture diagonal structures in the object's appearance.

$$h(\mathbf{x}) = \begin{cases} \text{1, object,} & f \cdot p \geq \theta \cdot p \\ \text{-1, clutter,} & \text{otherwise} \end{cases} \tag{2.3}$$

Here, $p \in \{1, -1\}$ is a polarity term, which can be used to invert the inequality relationship between $f$ and $\theta$.

#### 2.1.1.3 Single-rectangle feature space

When an image is evaluated with a Haar-like feature with $k$ rectangles, a vector of length $k$ containing rectangle mean measurements can be generated. Using these measurements, the image can be represented as a point in a $k$-dimensional feature space - which we call Single-rectangle Feature Space (SRFS). Let the representation of an image belonging to the object class be called an object point, and similarly, the representation of an image belonging to the clutter class be called a clutter point. In the following, a discussion on the distribution of object and clutter points in a SRFS is presented. Based on this discussion, a comparison of the performance of weak-classifiers constructed with traditional and the proposed Haar-like features is made.

For the purpose of this study, a set of images belonging to the object and clutter class were evaluated on Haar-like feature with two rectangles, thus generating a cloud of object and clutter points in a two dimensional (2D) SRFS. The number of rectangles in the Haar-like feature was restricted to two for clarity of visualization.

Two different object databases were chosen for this study. One contained human frontal face images from the AR [20] database and the other contained short-axis magnetic resonance images of the heart. The clutter points were generated from a clutter database containing images that neither belonged to the face class nor the heart class. Some of the typical images in the object and clutter databases are shown in Fig. 2.2.

Each image was intensity normalized by dividing every pixel value by $2^n - 1$, where $n$ is the number of bits used to represent each pixel value. This makes pixel values from any two images comparable as all of them vary from 0 to 1.

The distribution of object and clutter points in the 2D SRFS is shown in Fig. 2.3 and Fig. 2.4 respectively. As it can be observed, the object points form a more compact distribution than the clutter points irrespective of the size and the relative distance of the rectangles constituting the Haar-like feature. Compact distribution of object points results from making similar rectangle mean measurements. This can be understood from the fact that the images within the object class are strongly correlated to each other. The clutter points, however, are spread-out as the clutter images are very different from each other.

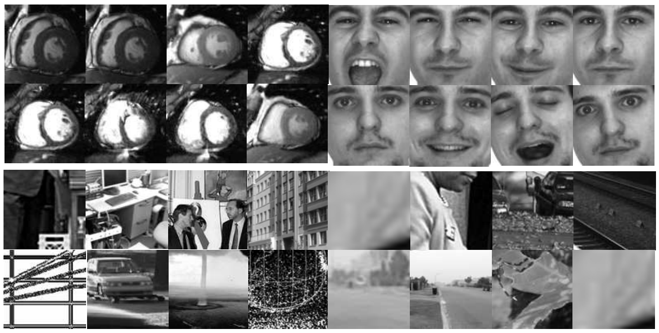


Figure 2.2: The first row shows images from the heart and human frontal face image databases. The second row contains typical images from the clutter database.

#### 2.1.1.4 Performance of weak classifiers in SRFS

Consider a hypothetical 2D SRFS as shown in Fig. 2.5(a). It shows a compact distribution of object points in the midst of clutter points. As discussed earlier, each point in the SRFS is a representation of an image through its rectangle mean measurements ($\boldsymbol{\mu}$). The feature

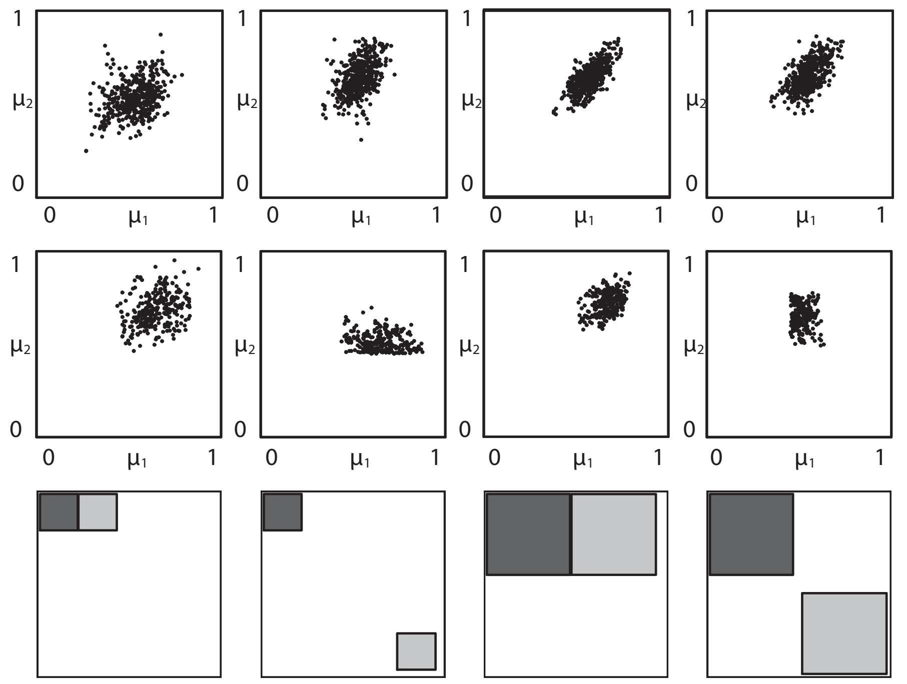


Figure 2.3: Distribution of object points for different Haar-like features. The first and the second row show object point distributions in SRFS from human frontal face images and the heart images respectively. The third row shows the Haar-like features that were used for the evaluation.

value of a Haar-like feature when evaluated on an image is the scalar product of the vector $\mathbf{w}$ that contains weights assigned to its rectangles, and $\boldsymbol{\mu}$. This has been mathematically expressed in (2.1). Geometrically, the computation of feature value can be understood as a product of magnitude of projection of $\boldsymbol{\mu}$ onto $\mathbf{w}$ and the magnitude of $\mathbf{w}$, which is a constant.

Traditionally, the rectangles of a Haar-like feature with two rectangles are assigned default weights of 1 and $-1$. This yields a projection line that is parallel to the vector $\mu_1 - \mu_2$. To train a weak classifier as in (2.3), a scalar value $\theta$ is found such that object and clutter points are best separated. As shown in the Fig. 2.5(b), $\theta$ divides the SRFS into two regions; the object region, coded white, and the clutter region, coded black.

Fig. 2.5(c,d) show classification using the optimal projection line. It is intuitive that by projecting object and clutter points to an optimal line, better classification performance can be achieved. Note that the proposed features have the capability to capitalize on the absolute difference in the brightness between the object class and the majority of

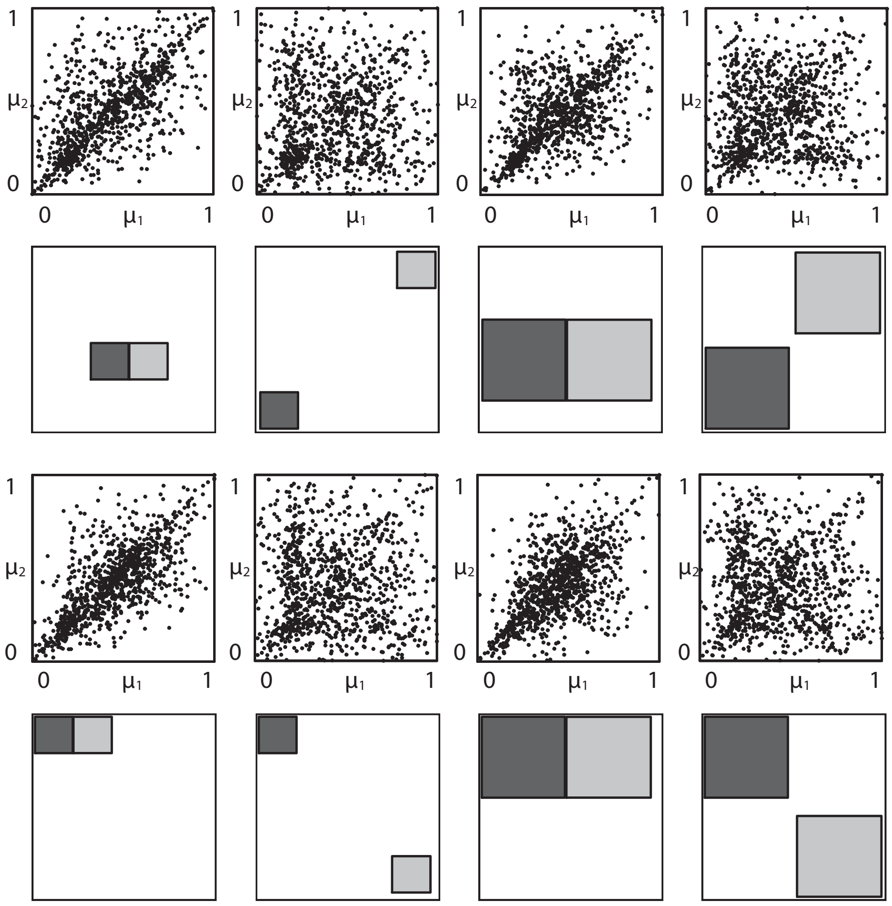


Figure 2.4: Distribution of clutter points for different Haar-like features.

clutter class images. For example, when the object points lie on the top-right or bottom-left corners of the 2D SRFS, better classification can be obtained by setting optimal line parallel to the vector $\mu_1 = \mu_2$. Traditional features, however, can produce good classifiers only when the majority of the object points would lie in left-top or the right-bottom of the 2D SRFS.

### 2.1.2 Training weak classifiers

Training a weak classifier involves determining the following three parameters that give the least error on the training set of object and clutter class images.

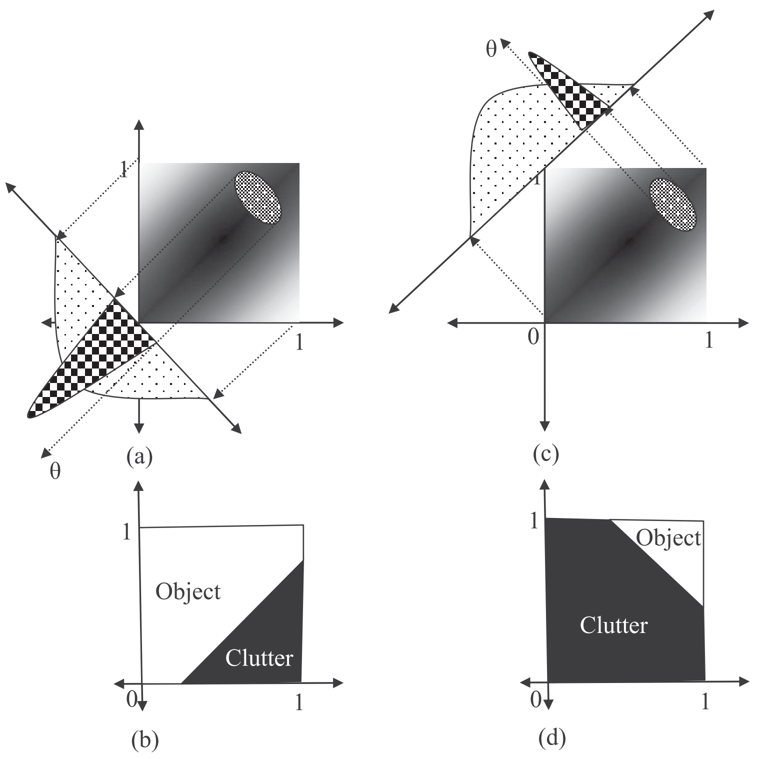


Figure 2.5: A geometrical view of a weak classifier performance. (a) weak classifiers built with traditional Haar-like features. (c) weak classifier constructed with modified Haar-like features whose rectangles are assigned optimal weights. (b) and (d) show their respective partitioned SRFS.

1. the threshold value $\theta$,
2. the polarity term $p$, and
3. the weight to be assigned to each rectangle of a Haar-like feature $\hat{w}$.

The two terms, $\theta$ and $p$, are optimized using a brute-force search as performed by Viola and Jones [36]. Three different techniques were used to find optimal weights of the rectangles of the Haar-like features. The description of these techniques and the motivation for their use is described in the following sections.

**Algorithm 1**: Training of a single weak classifier using BFS

```
  Input: Training images: $\mathbf{x}^{(i)}, i \in \{1, \ldots, n\}$
  Input: Training labels: $y^{(i)} \in \{-1, 1\}, i \in \{1, \ldots, n\}$
  Input: Weights for each training image: $z^{(i)} \in \Re, i \in \{1, \ldots, n\}$
  Input: Weak classifier with $k$-rectangle Haar-like feature: $h$
  Input: Possible weight values: $d$
1 Set $\epsilon_{\min}$ to $\infty$.
2 for $t = 1$ to $\frac{dP_{d-k}}{2}$ do
3     Set weights ($\hat{\mathbf{w}}$) to be assigned to the rectangles of $h$.
4     Find $\hat{\theta}$ and $\hat{p}$ that minimize the training error $\epsilon$.
          $[\hat{\theta}, \hat{p}] = \arg\min \epsilon$
      where,
          $\epsilon = \frac{1}{2} \sum_{i=1}^{n} z^{(i)} |h(\mathbf{x}^{(i)}) - y^{(i)}|$
5     if $\epsilon < \epsilon_{min}$ then
6         $\mathbf{w}^* = \hat{\mathbf{w}}, \theta^* = \hat{\theta}, p^* = \hat{p}, \epsilon_{\min} = \epsilon$
  Output: Trained weak classifier: $h \to \mathbf{w} = \mathbf{w}^*$, $h \to \theta = \theta^*$, $h \to p = p^*$
  Output: Error of the weak classifier: $\epsilon_{\min}$
```

#### 2.1.2.1 Brute-force search

One obvious way to guarantee an optimal solution to the problem of finding optimal weights would be a Brute-Force Search (BFS) through the possible weights. The search space could be restricted to a finite number of possible values by quantizing the weight values and limiting them to a predefined range. If the weights assigned to rectangles are restricted to $d$ discrete values, then the number of distinct weight combinations is $\frac{dP_{d-k}}{2}$, where $k$ is the number of rectangles and assuming $d > k$. For each of these weight combinations, optimal values of $\theta$ and $p$ need to be determined, which makes brute-force search extremely time consuming. The search space can be reduced by increasing the quantization step and restricting the range. Such steps that reduce the resolution of the search space, however, may lead to assigning sub-optimal weights to the rectangles. Finding optimal weights using BFS is outlined in **Algorithm 1**.

#### 2.1.2.2 Genetic Algorithms

Given a specific problem to solve, the input to the GA is a set of solutions to that problem, called *genomes*, and a metric called a *fitness function* that returns a quantitative measure of the quality of a genome. GA, through a series of iterations, called *generations*, tries to improve the genomes through genetic operations such as *crossover* and *mutation*. The main advantages of GA are that it does not require gradient information and therefore it is capable of solving nonlinear problems with multiple local optima. Further, GA has been demonstrated to produce substantial improvement over random and local search methods [13]. Since GA evolves its solutions over many iterations, it tends to be computationally expensive. Yet, in comparison with brute-force search, it is faster. In practice, it was found that evolving 30 genomes over 100 generations yielded stable solutions.

In our problem of finding optimal weights to rectangles, the genome is an array, representing the weights to be assigned to the rectangles. A valid genome for a Haar-like feature with $k$ rectangles would be an array of length $k$, whose elements can take real values. For a given genome and a Haar-like feature, our fitness function builds a weak classifier and returns the error $\epsilon$ made by the weak classifier on training examples. The lower the error returned by the fitness function, the better the genome represents the desired solution. The genetic optimization, as explained in **Algorithm 2**, is done over a series of $N$ generations. In each generation, the population of genomes are improved through the application of crossover and mutation operators. The genome, along with the threshold and polarity values, that produce the least training error are chosen as the parameters to be assigned to the weak classifier $h$.

#### 2.1.2.3 Fisher's linear discriminant analysis

FLDA provides a closed-form solution to find a linear classifier that best separates two or more classes. Although it provides an optimal solution only when the classes are Gaussian with equal covariances [5], it is reasonably quick and provides good approximations to the optimal solution in the general case. Our motivation for choosing FLDA is as follows. Since object detection is basically a two class problem, FLDA might provide quick answer as it comes with a closed-form solution. This procedure directly outputs the optimal weights and, therefore, the optimal values of $\theta$ and $p$ need to be computed only once. This makes FLDA much faster when compared to GA or brute force methods.

The **Algorithm 3** describes a solution to find optimal weights using FLDA. The algorithm is presented with a database of object and clutter class training images with their corresponding labels. FLDA outputs a linear projection vector whose slope corresponds to the optimal weights assigned to the rectangles of the Haar-like feature. Once the optimal weights are found, the optimal values for threshold ($\hat{\theta}$) and polarity terms ($\hat{p}$) are found by searching exhaustively [36] over all possible solutions.

### 2.1.3 Boosting weak classifiers

Boosting is a meta algorithm that refers to a method of producing a “strong" classifier by additively combining a set of weak classifiers. The predictions from many weak classifiers are combined through weighted majority voting to produce the prediction of the strong classifier. For binary classification problems, the strong classifier has the form:

$$H(\mathbf{x}) = \begin{cases} 1, \text{object}, & \sum_{i=1}^{k} \alpha^{(i)} h^{(i)}(\mathbf{x}) \geq \Theta \\ 0, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.4}$$

**Algorithm 2**: Training of a single weak classifier using GA

```
    Input: Training images: $\mathbf{x}^{(i)}, i \in \{1, \dots, n\}$
    Input: Training labels: $y^{(i)} \in \{-1, 1\}, i \in \{1, \dots, n\}$
    Input: Weights for each training image: $z^{(i)} \in \Re, i \in \{1, \dots, n\}$
    Input: Weak classifier to be trained: $h$
    Input: Number of generations: $N$
    Input: Size of each generation: $m$
 1  begin
 2      Select initial population of genomes $\mathbf{g}_1^{(l)}, l \in \{1, \dots, m\}$.
 3      Set $\epsilon_{\min}$ to $\infty$.
 4      for $t = 1$ to $N$ do
 5          forall $\mathbf{g}_t^{(l)}$ do
 6              Set current genome as the weights to be assigned to the rectangles of $h$
                    $$h \rightarrow \mathbf{w} = \mathbf{g}_t^{(l)}$$
 7              Find $\hat{\theta}$ and $\hat{p}$ that minimize the training error $\epsilon$.
                    $$[\hat{\theta}, \hat{p}] = \arg\min \epsilon$$
                where,
                    $$\epsilon = \frac{1}{2} \sum_{i=1}^{n} z^{(i)} |h(\mathbf{x}^{(i)}) - y^{(i)}|$$
 8              if $\epsilon < \epsilon_{min}$ then
 9                  $\mathbf{w}^* = \mathbf{g}_t^{(l)}, \theta^* = \hat{\theta}, p^* = \hat{p}, \epsilon_{\min} = \epsilon$
10          Reproduce genomes with the lowest training error in the population to form
            new ones through crossover and mutation.
11          Replace the worst genomes in the population with the best new genomes.
12  end
    Output: Trained weak classifier: $h \rightarrow \mathbf{w} = \mathbf{w}^*, h \rightarrow \theta = \theta^*, h \rightarrow p = p^*$
    Output: Error of the weak classifier: $\epsilon_{\min}$
```

In (2.4), $h^{(i)}, i = 1, \dots, k$ are the $k$ weak classifiers selected by boosting and $\alpha^{(i)}, i = 1, \dots, k$ are their corresponding weights. The *AdaBoost* [30] [31], one of the popular boosting procedures, has been used in our implementation. The pseudocode of *AdaBoost* adapted to the object detection problem is shown in **Algorithm 4**.

The algorithm takes as input the object class training images $\mathbf{x}^{(i)}, i = 1, \dots, n$, their labels $y^{(i)}, i = 1, \dots, n$ and a set of weak classifiers $h^{(j)}, j = 1, \dots, m$. The algorithm goes through a series of $N$ iterations, and in each iteration a weak classifier is selected. In the first iteration, all the weak classifiers are trained according to **Algorithm 2** or **3** and the weak classifier that produces the lowest error is selected. In each subsequent iteration,

**Algorithm 3**: Training of a single weak classifier using FLDA

```
  Input: Training images: $\mathbf{x}^{(i)}, i \in \{1,\dots,n\}$
  Input: Training labels: $y^{(i)} \in \{1,-1\}, i \in \{1,\dots,n\}$
  Input: Weights for each training image: $z^{(i)} \in \Re, i \in \{1,\dots,n\}$
  Input: Weak classifier to be trained: $h$
1 begin
2     Evaluate the Haar-like feature in $h$ over all object and clutter training images.
      Each evaluation gives a $k$-dimensional vector $\boldsymbol{\mu}^{(i)}$.
3     Compute between-class scatter [7]: $\boldsymbol{m}_1 - \boldsymbol{m}_2$ where $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ are the
      $k$-dimensional means of measurements made from $n_o$ object class and $n_c$
      clutter class training images respectively.
          $$\boldsymbol{m}_1 = \sum_{\forall i, \mathbf{y}^{(i)}=1} \frac{\boldsymbol{\mu}^{(i)}}{n_o}, \boldsymbol{m}_2 = \sum_{\forall i, \mathbf{y}^{(i)}=-1} \frac{\boldsymbol{\mu}^{(i)}}{n_c}$$
4     Compute within-class scatter [7]: $\boldsymbol{S} = \boldsymbol{S}_1 + \boldsymbol{S}_2$
          $$\boldsymbol{S}_1 = \sum_{\forall i, \mathbf{y}^{(i)}=1} (\boldsymbol{\mu}^{(i)} - \boldsymbol{m}_1)(\boldsymbol{\mu}^{(i)} - \boldsymbol{m}_1)^t$$
          $$\boldsymbol{S}_2 = \sum_{\forall i, \mathbf{y}^{(i)}=-1} (\boldsymbol{\mu}^{(i)} - \boldsymbol{m}_2)(\boldsymbol{\mu}^{(i)} - \boldsymbol{m}_2)^t$$
5     Compute optimal weights
          $$\hat{\mathbf{w}} = \boldsymbol{S}^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2)$$
6     Find $\hat{\theta}$ and $\hat{p}$ that minimize the training error $\epsilon$.
          $$[\hat{\theta}, \hat{p}] = \arg\min \epsilon$$
      where,
          $$\epsilon = \frac{1}{2} \sum_{i=1}^{n} z^{(i)} |h(\mathbf{x}^{(i)}) - y^{(i)}|$$
7 end
  Output: Trained weak classifier: $h \to \mathbf{w} = \hat{\mathbf{w}}$, $h \to \theta = \hat{\theta}$, $h \to p = \hat{p}$
  Output: Error of the weak classifier: $\epsilon$
```

a weak classifier is selected such that the combined error of all the previously selected weak classifiers is minimized. This is performed by maintaining weights $z^{(i)}, i = 1, \dots, n$ on the training images. The higher the magnitude of $z^{(i)}$, the greater is the need for $\mathbf{x}^{(i)}$ to be classified correctly. Initially, all the training images belonging to a class are assigned equal weights $z^{(i)}$. In each iteration, the value of $z^{(i)}$ is increased if $\mathbf{x}^{(i)}$ was misclassified by the selected weak classifier and decreased if $\mathbf{x}^{(i)}$ was classified correctly. This compels the algorithm to select a weak classifier that classifies the training samples with higher weights correctly in the subsequent iteration.

Once the weak classifiers and their corresponding weights determined through the boosting procedure, the threshold value of the strong classifier ($\Theta$) is set such that the strong classifier has a maximum false rejection rate, $\beta$, on a validation set of object class images. The threshold value $\Theta$ is found such that the sum of weighted decisions of the selected weak classifiers is greater than $\Theta$ for at least $(1-\beta)q$ images in the validation set. Here, $q$ is the total number of images in the validation set.

### 2.1.4 Training the object detector

This section describes how object detection systems can be built using boosted set of weak classifiers.

#### 2.1.4.1 Framework of the object detection system

The framework of the proposed object detector is based on Baker *et al.*'s rejection cascade architecture [4] [3]. A rejection cascade, as shown in Fig. 2.6, consists of serially connected nodes which label a test image as object or clutter. Each node contains a boosted set of weak classifiers. In Fig. 2.6, the *Node* 3 of the rejection cascade has been expanded to show the $k$ weak classifiers present in it.

The object detection proceeds as follows: a given test image is scanned at all positions and scales by the rejection cascade. When an image sub-region $\mathbf{x}$ is input to a node, it is classified by all the weak classifiers present in the node, and the weighted average of their decisions is output as the final decision of that node. Thus, each node labels an image sub-region as object or clutter. An image sub-region is labeled object only if it is labeled object by all the nodes of the cascade. On the other hand, if a sub-region is labeled clutter by a node, it is not further processed by any successive node. Thus, the rejection cascade tries to reject clutter as early as possible. Since a majority of image sub-regions belong to clutter [24] [39], an object detection system based on rejection cascade architecture will be fast in scanning the entire test image.

The rejection cascade $\mathscr{H}(\mathbf{x})$ can be expressed as:

$$\mathscr{H}(\mathbf{x}) = \begin{cases} 1, \text{object}, & \prod_{i=1}^{M} H^{(i)}(\mathbf{x}) = 1 \\ 0, \text{clutter}, & \text{otherwise} \end{cases}$$

where $H^{(i)}(\mathbf{x})$ is a boosted set of weak classifiers as in (2.4), and $M$ is the number of nodes.

### 2.1.5 Building the rejection cascade

The rejection cascade is built according to **Algorithm 5**. The algorithm takes in a database of object and clutter class images $\mathbf{x}^{(i)}, i = 1, \ldots, n$, their labels $y^{(i)}, i = 1, \ldots, n$ and a set of weak classifiers $h^{(j)}, j = 1, \ldots, m$. Given the training data, the algorithm iterates $M$ times to construct the rejection cascade, where $M$ is the desired number of nodes. In each

**Algorithm 4**: Boosting weak classifiers

```
  Input: Training images: $\mathbf{x}^{(i)}, i \in \{1, \ldots, n\}$
  Input: Training labels: $y^{(i)} \in \{-1, 1\}, i \in \{1, \ldots, n\}$
  Input: Validation object class images: $\mathbf{v}^{(i)}, i \in \{1, \ldots, q\}$
  Input: Maximum false rejection rate: $\beta$
  Input: Set of weak classifiers: $\mathbf{h} = \{h^{(j)}\}, j \in \{1, \ldots, m\}$
  Input: Number of weak classifiers per node: $N$
1 begin
2     Set weights to each training image:
          $$z_1^{(i)} = \begin{cases} \frac{0.5}{n_o}, & \mathbf{x}^{(i)} \in \text{object} \\ \frac{0.5}{n_c}, & \mathbf{x}^{(i)} \in \text{clutter} \end{cases}$$
      where, $n_o$ and $n_c$ are the number of object and clutter images, respectively.
3     for $t = 1$ to $N$ do
4         Apply Algorithm 1, 2, or 3 to train each weak classifier in $\mathbf{h}$ and find the
          weak classifier, $h_t$, with minimum error, $\epsilon_t$.
5         Compute the new weight for $h_t$:
              $$\alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t}$$
6         Update the weights for each training image:
              $$z_{t+1}^{(i)} = \frac{z_t^{(i)} e^{-\alpha_t h_t(\mathbf{x}^{(i)})}}{Z}$$
          where $Z$ is a normalization factor such that $z_{t+1}^{(i)}$ will be a distribution.
7     Evaluate the selected weak classifiers on the validation set and find $\Theta$ such that
      $\sum_{t=1}^{N} \alpha_t h_t(\mathbf{v}^{(i)}) \geq \Theta$ is true for $(1 - \beta)q$ validation set images.
8 end
  Output: Strong classifier, $H(\mathbf{p})$, where $\mathbf{p}$ is a test image:
      $$H(\mathbf{p}) = \begin{cases} 1, \text{object}, & \sum_{t=1}^{N} \alpha_t h_t(\mathbf{p}) \geq \Theta \\ 0, \text{clutter}, & \text{otherwise} \end{cases}$$
```

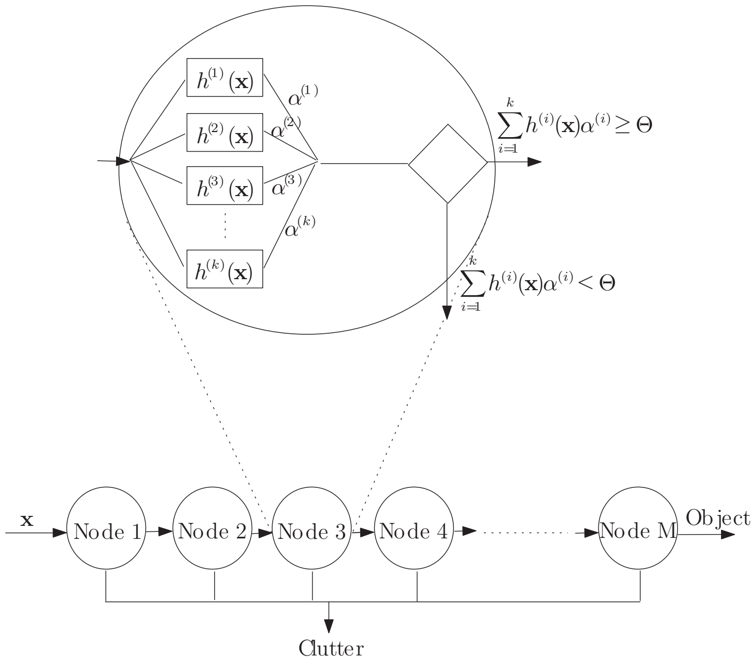


Figure 2.6: A cascaded classifier structure consists of multiple nodes arranged in a degenerated decision tree fashion. The output of the node is the weighted majority of decisions of the individual classifiers present at that node. If an image is classified as clutter by a node, then that image is not processed in the successive nodes.

iteration, a node $H^{(u)}(\mathbf{x})$ is built according to **Algorithm 4**. The clutter training images that were correctly classified by the current node are not considered to build the next node. This ensures that the next node to be built concentrates on rejecting the clutter images that were misclassified by the previously built nodes.

### 2.1.6 Experimental setup and results

The proposed features were trained for two object detection problems: detection of human frontal faces from digital photographs and detection of heart regions in short-axis cardiac magnetic resonance images. Detecting human frontal faces and heart regions is challenging as they exhibit high degree of intra-class appearance changes. Further, these objects are ideal for our study because applications involving detection of frontal faces and hearts benefit from real-time localization. Face detection is the first step in many computer vision systems and its output in real-time is indispensable for security systems or for Human-Computer Interaction systems. Heart detection can be used as a precursor

**Algorithm 5**: Building a rejection cascade of strong classifiers

```
Input: Desired number of nodes in the cascade: $M$
Input: Training images: $\mathbf{x}^{(i)}, i \in \{1, \dots, n\}$
Input: Training labels: $y^{(i)} \in \{-1, 1\}, i \in \{1, \dots, n\}$
Input: Set of weak classifiers: $\mathbf{h} = \{h^{(j)}\}, j \in \{1, \dots, m\}$
1 begin
2     for $u = 1$ to $M$ do
3         Build node $H^{(u)}$ according to Algorithm 4.
4         Remove $\mathbf{x}^{(i)}$ if $H^{(u)}(\mathbf{x}^{(i)}) = 0$ and $y^{(i)} = -1$.
5         Test the cascade with $u$ nodes on a test set of clutter images, and add the
          sub-windows that were misclassified to the training set.
6 end
Output: Rejection cascade, $\mathscr{H}(\mathbf{t})$, of strong classifiers where $\mathbf{t}$ is a test image:
```

$$\mathscr{H}(\mathbf{t}) = \begin{cases} 1, \text{object}, & \prod_{i=1}^{M} H^{(i)}(\mathbf{t}) = 1 \\ 0, \text{clutter}, & \text{otherwise} \end{cases}$$

to complex heart segmentation techniques which in turn may help detecting any abnormality in the cardiac function. Since an MRI scan of heart produces a sequence of 3D images, and each image typically contains dozens of slices, heart regions in lots of images need to be detected and segmented before the pathology could be understood. Therefore, quick detection and segmentation algorithms would help in reducing the diagnosis time.

In the following experiments, we compare the speed and the accuracy of the object detectors constructed with the proposed features (optimal weights) over the traditional ones (default weights). Before the results are presented, the databases on which the object detectors were trained and tested are described.

#### 2.1.6.1 The training datasets

To train the frontal face and heart detectors, two object databases (face and heart), and a clutter database were used. The frontal face database was composed of 5000 images. The images in this database were collected from publicly available facial image databases such as MIT-CBCL face database No. 1 [2], XM2VTS [21], Equinox [33], AR [20], and BioID [11]. The facial regions were cropped manually and resized to images of size $20 \times 20$ pixels.

The heart database consisted of 493 short-axis MR heart images. Similar to the face dataset, the heart regions were manually cropped and resized to images of size $20 \times 20$ pixels. To train the heart detector, 300 of the 493 images were used.

For the clutter image database, a total of 27,000 images were downloaded from the internet. These images did not contain either frontal face nor heart regions. For our experiments we generated approximately $4 \times 10^9$ sub-regions from these images.

#### 2.1.6.2 The test datasets

The face detectors were tested on the MIT+CMU frontal face database (test sets A, B, and C) [29]. This database consists of 130 images with 511 frontal faces. Two of the faces in this test set had to be flipped as they were initially upside-down. The remaining images were left untouched. Ground truth face location is available for every face in the database.

The heart detectors were tested on a set of 193 images. In all these images the heart regions were manually landmarked to obtain ground truth size and location.

In our experiments, we assumed that a face or a heart has been detected correctly if a positively detected sub-region satisfies the following two conditions:

1. The size of the detected region is within $\pm 10\%$ of the size of the annotated face/heart.

2. The distance between the center of the detected region and the annotated face/heart is not more than 10% of the size of the annotated face/heart.

These two conditions ensure that a positively detected sub-region of a test image contains a face/heart.

#### 2.1.6.3 Limiting the number of Haar-like features

The number of ways in which rectangles can be arranged in a template to form Haar-like features is (almost) infinite [18]. Therefore, for practical reasons, all implementations of object detection systems that use Haar-like features need to limit the number of features that are used for training . In order to limit the number of Haar-like features, the following measures were taken:

1. Only Haar-like features with two, three and four rectangles were considered.

2. The template size of the Haar-like features was set $20 \times 20$ pixels.

3. Lienhart and Maydt's [18] rotated rectangle features (see Fig. 2.1 (d,e,f)) were not considered.

4. The inter-rectangle distances $dx$ and $dy$ for Li *et al.*'s [16] disjoint features as shown in Fig. 2.1 (g) were discretized so that they were integer multiples of 100% of the rectangle size in corresponding directions.

5. All the rectangles contributing to a single Haar-like feature had the same size.

6. Rectangles with size less than $3 \times 3$ pixels were not allowed to form Haar-like features.

A total of 207,807 Haar-like features remained after removing the features that did not satisfy the above-mentioned conditions. Despite this reduction, we train more number of features than Viola and Jones [36], who used a total of 160,000 features.

#### 2.1.6.4 Training object detectors

For every object detection problem (face and heart), four detectors were constructed. These four detectors basically differed on the type of features used to construct them. The features used were:

- Default weights: Traditional Haar-like features whose rectangles are assigned default weights.
- Optimal weights (BFS): Proposed features with optimal rectangle weights computed using BFS.
- Optimal weights (FLDA): Proposed features with optimal rectangle weights computed using FLDA.
- Optimal weights (GA): Proposed features with optimal weights computed using GA.

Each face detector had 36 nodes with a total of $1,832$ weak-classifiers distributed among them. The heart detectors had 16 nodes with 172 weak-classifiers. Corresponding nodes in every object detector had the same number of weak classifiers. The first ten nodes of the face detector and the heart detector were assigned only one weak classifier in order to reject as many clutter regions as possible with minimum computational effort. Every node of the rejection cascade was trained so that their false rejection rate on a database of object class validation set is 0.01 or lower. The Haar-like features that were picked in the first iteration of the AdaBoost procedure for different object detectors are illustrated in Fig. 2.7. The weight values that produced the minimum training error, which are superimposed on the rectangles, seem to be different from the default weights.

To compute optimal weights using BFS, the weight values were restricted to the range $[-1, 1]$ and quantized with a step size of 0.1. The GA parameters were set to the following values: population size = 30, number of generations = 100, crossover probability = 0.8, and mutation probability = 0.1. In our implementation, single point crossover was used, and mutation was effected by changing a randomly selected element of the genome with a random value selected in the interval [-10 10].

The time required to training the face and heart detectors on a 2.33 GHz Intel Xeon Dual Core “Woodcrest" processor with 8 GB RAM is shown in Table 2.1. Viola and Jones [36] report that the time required to train a 38-node detector was in the order of weeks in a 466 MHz AlphaStation XP900.

#### 2.1.6.5 Comparison of accuracies of object detectors

The face detectors were tested on the MIT+CMU frontal face dataset, and Fig. 2.8 shows the obtained Receiver Operating Characteristics (ROC) curves. The ROC curves were generated as described in [36] by adjusting the threshold ($\Theta$) of the last two nodes of the object detector from $-\infty$ to $+\infty$. Note that in Fig. 2.8, the true positive rate is plotted against the number of false detections. The false positive rate can be obtained by dividing the number of false detections by $12,378,518$, which is the total number of searched sub-windows in the MIT+CMU database.

Table 2.1: Comparison of training time on a 2.33 GHz Intel Xeon Dual Core processor with 8 GB RAM

| Feature Type | Face | Heart |
|---|---|---|
| Default weights | ~ 2 days | ~ 5 hours |
| Optimal weights (BFS) | ~ 22 days | ~ 2 days |
| Optimal weights (GA) | ~ 20 days | ~ 2 days |
| Optimal weights (FLDA) | ~ 10 days | ~ 8 hours |

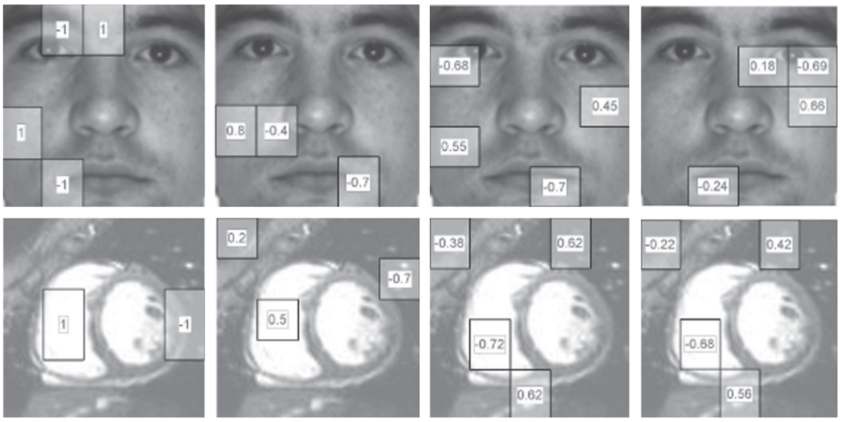


Figure 2.7: Illustration of the Haar-like features, superimposed on a typical training images, that were selected in the first iteration of the AdaBoost procedure for different object detectors. The columns, from left to right, show features selected using different rectangle weighting strategies: Default weights, Optimal weights (BFS), Optimal weights (GA) and Optimal weights (FLDA), respectively.

Face detection results on MIT+CMU database from five other popular face detection systems are superimposed in Fig. 2.8. Rowley-Baluja-Kanade [29] used an ensemble of neural-networks to learn face and clutter patterns. Their results are based on evaluating their detector on 130 images from MIT+CMU database. Roth-Yang-Ahuja [28] evaluated their SNoW based detector on all the images in MIT+CMU dataset except those containing line-drawn faces. Schneiderman-Kanade [32] proposed a probabilistic method to detect faces based on local appearance and principal component analysis. They report their results based on evaluating their detector on 125 images from MIT+CMU database.

Viola and Jones [36] report their results on evaluating their detector on 130 images from MIT+CMU database. Their detector had a cascade structure with 38 nodes, which in turn was built using Haar-like features with default weights. They also reported a modest improvement in detection results by using a simple majority voting scheme when each test image was evaluated using three differently built detectors. Note that the results reported under default weights refer to results from our own implementation of the Viola and Jones's object detection procedure. We believe that the ROC curves obtained for default weights and those reported in [36] differ because of different image databases and different feature pool used in the training process. The image databases used in [36] are not publicly available or accessible in any way. Also, parameters such as how small or how big the rectangles forming the Haar-like features should be are not specified, which makes it difficult to recreate the feature pool exactly.

Fig. 2.8 shows the ROC curves obtained for the heart detector. The false positive rates for the heart detector can be obtained by dividing the number of false detections by $9,842,870$, which is the number of searched sub-windows. As observed in the results from the face detector and the heart detectors constructed with optimal weights (BFS, GA and FLDA) were more accurate than that constructed with default weights. The only difference among the detectors was the strategy used to allocate the weights to the rectangles of Haar-like feature. All the other training parameters remained the same. Therefore, it can be concluded that the increase in accuracy was obtained by using the optimal weights alone.

For face and heart detection problems, it can be observed that the FLDA performs poorer than GA. We believe that there are two factors that contribute to this sub-optimal performance:

1. The non Gaussian nature of the object and clutter cluster of points in the SRFS (See Section 2.1.2.3).

2. When FLDA is used, optimization of the three parameters $\hat{w}$, $\theta$ and $p$ occur sequentially. That is, $\hat{w}$ is optimized first, then optimal values of $\theta$ and $p$ are found for this particular combination of weights. This is sub-optimal in comparison to GA, where all three parameters are optimized simultaneously.

The accuracy of object detectors built with BFS are worse than those built with GA. However, it is expected that the performance of BFS would tend towards that of GA if the search range is increased and the quantization step is decreased. The values for the range ($[-1, 1]$) and quantization step (0.1) were chosen such that the time required for training object detectors using BFS and GA are similar.

The output of the face detector over various test images in MIT+CMU database are shown in Fig. 2.9. The face detector tolerates approximately $15^\circ$ in-plane and out-of-plane rotations of face, and it is also tolerant to the race of the person. Fig. 2.10 shows the heart detection output on various test images. False positives occur when the underlying image is similar to that of the object of interest. For example, for the image on the top-left in Fig. 2.9, some of the collar regions have been labeled positive as they resemble a face. False negatives might have occurred due to extreme lighting conditions, poor image quality or due to insufficient resolution during search.

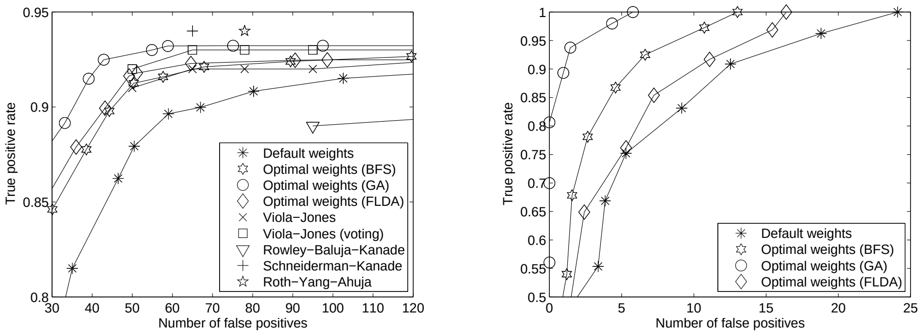


Figure 2.8: Comparison of ROC curves for various face (left) and heart (right) detectors

### 2.1.6.6 Comparison of speed of object detectors

As discussed in Sec. 2.1.4.1, the speed of an object detector is proportional to its efficiency in rejecting clutter images. The efficiency in rejecting clutter can be expressed as average number of nodes ($\bar{n}$) and the average number of weak classifiers ($\bar{c}$) required to reject a clutter image. The quantities $\bar{n}$ and $\bar{c}$ are computed as shown in (2.5) and (2.6) respectively.

$$\bar{n} = \sum_{i=1}^{n} p^{(i)} \text{ nodes} \tag{2.5}$$

$$\bar{c} = \sum_{i=1}^{n} c^{(i)} \cdot p^{(i)} \text{ weak classifiers} \tag{2.6}$$

Here, $p^{(i)}$ is the probability that a clutter image is evaluated by the $i^{th}$ node of rejection cascade. In (2.6), $c^{(i)}$ represents the number of weak classifiers in the $i^{th}$ node. An estimate of $p^{(i)}$ can be obtained as in (2.7) by testing the rejection cascade with $N$ number of clutter images, and recording the number of clutter images evaluated ($e^{(i)}$) by the $i^{th}$ node.

$$p^{(i)} = \frac{e^{(i)}}{N} \tag{2.7}$$

Note that $p^{(1)} = 1$ because all the clutter images will be evaluated by the first node. Fig. 2.11 shows the estimated value of $p^{(i)}$ at each node of the rejection cascade for the face and heart detectors. They were obtained by testing the rejection cascades with 10000

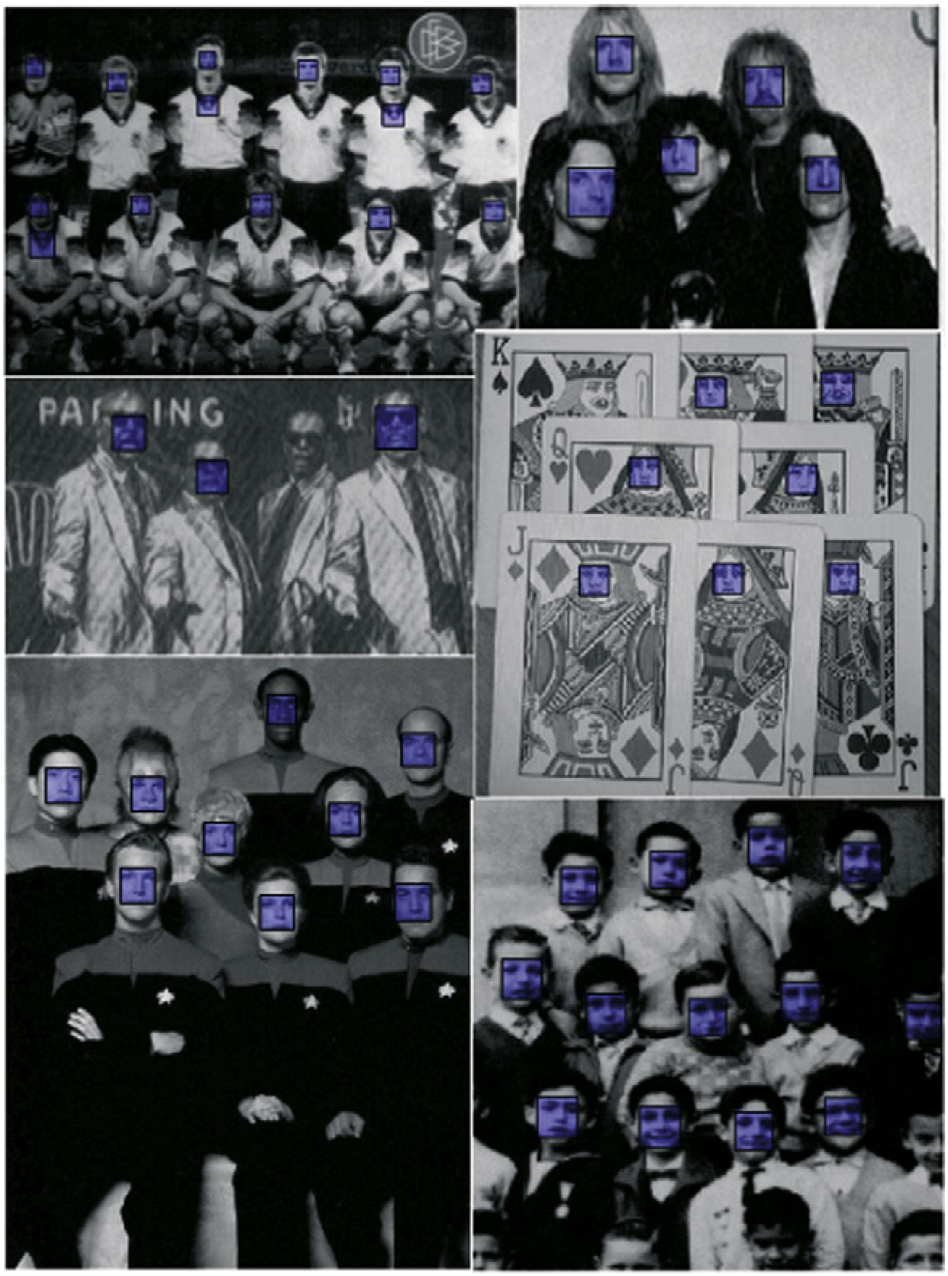


Figure 2.9: Face detector output on various test images.

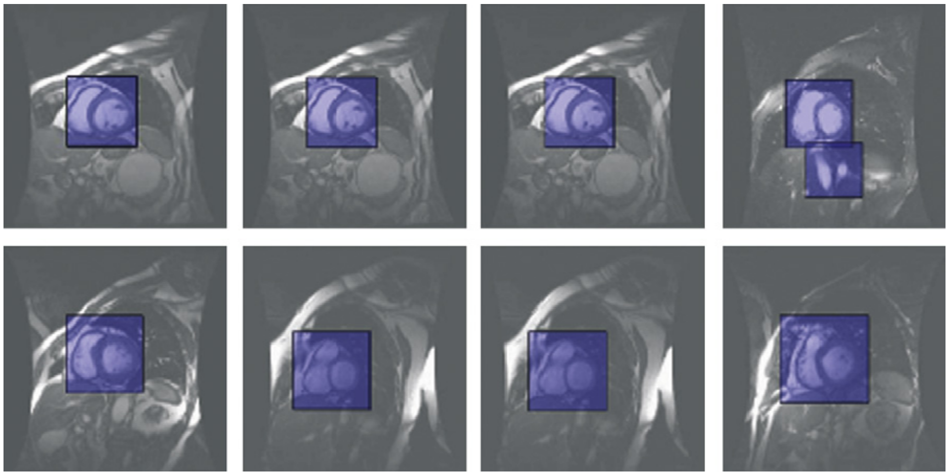


Figure 2.10: Heart detector output on various test images.

randomly selected clutter images. It can be noticed that the probability that a clutter image is evaluated in a node is consistently lower for the object detectors built with the proposed features than the traditional ones. This means that the object detectors built with the proposed features will better in rejecting clutter images and therefore, they will be faster in scanning through a test image. Table 2.2 tabulates $\bar{n}$ and $\bar{c}$ values for different rejection cascades. It can be observed that the rejection cascades built with the proposed features require less number of nodes and weak classifiers to reject an average clutter image than those built with traditional features.

The more number of weak classifiers ($\bar{c}$) required by the face detector to reject a clutter image on the average suggests that weak-classifiers selected for face detectors are less powerful than those selected for the heart detector. This, in turn, suggests that faces are more difficult class to model using Haar-like features than hearts.

The average speed, in frames per second (fps), of object detectors (when tested on images that have one instance of object) is tabulated in Table 2.2. The face detectors were tested on images (of resolution $320 \times 240$ images) proceeding from a camera installed in a office scenario. On each frame, 8858 regions were searched for the presence of a face. The heart detectors were tested on $640 \times 480$ images, and on each image, 42145 regions were searched. The reason why approximate times are reported in Table 2.2 is that the speed of the object detectors is not constant and it is dependent on the number of sub-regions that resemble an object. Sub-regions that resemble an object have high chances of being processed by most of the nodes of the detector and, therefore, they require more time to be processed. The speed of the object detectors increased approximately 1.5 times when the test images were uniformly white and contained no instances of object.

From (2.1), it can be noticed that the proposed features require additional floating point multiplications which make them slower to evaluate. However, since they reject

Table 2.2: Comparison of $\bar{n}$, $\bar{c}$ and speed for different object detectors

| Weak classifier type | $\bar{n}$ | | $\bar{c}$ | | fps | |
|---|---|---|---|---|---|---|
| | Face | Heart | Face | Heart | Face | Heart |
| Default weights | 2.14 | 1.22 | 5.06 | 1.74 | ~ 12 | ~ 15 |
| Optimal weights (BFS) | 1.55 | 1.09 | 3.01 | 1.37 | ~ 15 | ~ 20 |
| Optimal weights (FLDA) | 1.47 | 1.07 | 2.74 | 1.37 | ~ 17 | ~ 20 |
| Optimal weights (GA) | 1.38 | 1.05 | 2.35 | 1.23 | ~ 19 | ~ 20 |

more clutter than the traditional features, object detectors built with the proposed features are faster.

On a 700 Mhz Pentium III processor, Viola and Jones [36] report that their implementation of the face detector (using Haar-like features with default weights) processes 384 × 288 image at the rate of 15 fps. Rowley-Baluja-Kanade [29] and Roth-Yang-Ahuja [28] do not specify the speed of their detector, however, Viola and Jones [36] independently tested Rowley-Baluja-Kanade detector and concluded that it is about 15 times slower than theirs. Schneiderman and Kanade [32] report that their detector can evaluate a 240 × 256 image in about 90 seconds on average using a Pentium II 450 MHz processor.

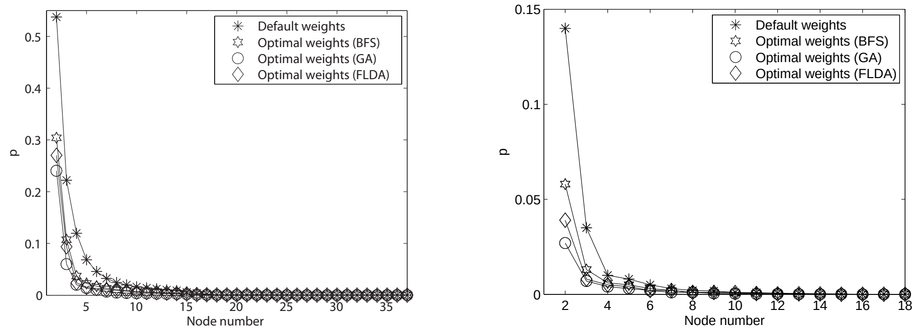


Figure 2.11: False positive rate at a given node of the face (left) and the heart (right) detectors

### 2.1.7 Conclusions

In this chapter, we propose assigning optimal weights to the rectangles of the Haar-like features so that the weak classifiers constructed based on them give best possible classification performance. The optimal weights were computed in a supervised manner using three different techniques. 1) Brute-force search, 2) Genetic Algorithms and 3) Fisher's linear discriminant analysis. This chapter presents detailed experiments on two difficult object detection problems (frontal faces and human hearts), and our results indicate that by using the proposed features, better object detectors, both in terms of accuracy and speed of detection, can be constructed.

## 2.2 Gaussian weak classifiers based on co-occurring Haar-like features for face detection

Haar-like features (HFs) have successfully been used in modeling complex objects such as human faces and pedestrian images [15] [25]. HFs are used to construct weak classifiers that provide an attractive trade-off between speed and accuracy. In the domain of face detection, Viola & Jones [36] achieved 1% false negatives and 40% false positives using a weak classifier which can be evaluated in approximately 60 microprocessor instructions. Their weak classifiers (VWCs), decide if an image belongs to face or clutter (non-face) class by comparing the feature value of a HF to a threshold. Using boosted ensembles [30] of weak classifiers arranged in a cascade architecture [4], Viola & Jones built the first reliable real-time face detector.

In a recent work, Mita *et al.* [23] proposed a new paradigm in constructing weak classifiers, wherein they build a weak classifier (MWC) based on observing outputs of two or more HFs (called co-occurring HFs). The decision of a MWC is obtained as follows. For each HF associated with a MWC, a VWC is trained. The weighted average decision of all the VWCs is the decision of a MWC. In other words, MWCs are obtained by fusing information obtained from a co-occurring HF at the decision level. Mita *et al.* experimentally demonstrated that boosted ensembles of MWCs give better speed-accuracy trade-off than boosted ensembles of VWCs.

In this chapter, we propose an alternative to Mita *et al.*'s method of forming weak classifiers based on co-occurring HFs. The proposed weak classifiers, called Gaussian weak classifiers (GWCs), fuse information obtained from co-occurring HFs at the feature level, and are potentially more discriminative than MWCs, which fuse information at the decision level. Preliminary results on GWCs [26], showed that face detectors based on GWCs simultaneously achieve both higher accuracy and higher speed than those constructed with MWCs. In this work, we formulate GWCs on a general framework, based on co-occurring HFs built by automatically selecting HFs with different configurations. Unlike Mita *et al.*, we train and test entire face detectors built as a cascade with boosted ensembles of weak classifiers. Our experiments on face detectors built with VWCs, MWCs and GWCs are designed to 1) quantify the speed and the accuracy of the detectors and 2) to determine the optimal number of HFs in a co-occurring HF to obtain the best speed-accuracy trade-off.

### 2.2.1 Weak classifiers based on HFs

Haar-like features [25], shown in Fig. 2.1, consist of two or more rectangular regions enclosed in a template. Such features, when evaluated on an image, produce a feature value as in (2.8).

$$f = \mathscr{F}(\mathbf{x}) = \sum_{i=1}^{q} w^{(i)} \cdot \mu^{(i)} \tag{2.8}$$

where $i$ iterates through all the $q$ rectangles of the HF. The quantity $\mu^{(i)}$ represents the mean intensity of the pixels in image $\mathbf{x}$ enclosed within the $i^{th}$ rectangle. Every rectangle in the HF is assigned a weight that is represented by $w^{(i)}$. The weights are set such that $\sum_{i=1}^{q} w^{(i)} = 0$ is satisfied.

VWCs compare the feature value $f$ to a threshold $\theta$ according to (2.9).

$$h_{vj}(\mathbf{x}) = \begin{cases} 1, \text{face}, & f \cdot p \leq \theta \cdot p \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.9}$$

Here, $p \in \{1, -1\}$ is a polarity term, which can be used to invert the inequality relationship between $f$ and $\theta$.

MWCs are constructed using $k$ HFs as shown in (2.10). For each HF, a VWC is constructed, and the weighted decision of the VWCs is the decision of the MWC. The weight assigned to each VWC is given by $2^{k-i}$, where $i \in \{1 \dots k\}$.

$$h_m(\mathbf{x}) = \begin{cases} 1, \text{face}, & \sum_{i=1}^{k} 2^{k-i} h_{vj}^{(i)}(\mathbf{x}) \geq \theta \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.10}$$

The $k$ VWCs are selected iteratively using Sequential Forward Selection (SFS) algorithm [10].

#### 2.2.1.1 Gaussian weak classifiers

GWCs, like MWCs, make their decision based on feature values from $k$ HFs $\boldsymbol{f} = [f_1, \dots, f_k]$. Firstly, as shown in (2.11), $\boldsymbol{f}$ is computed from a test image, and its Mahalanobis distance (d) to the mean of measurements obtained from face class training images ($\bar{\boldsymbol{f}}$) is computed. Then, the distance ($d$) is compared to a threshold ($d_t$) to decide on whether the test image belongs to face or clutter as in (2.12). The quantity $\boldsymbol{\Sigma}$ in (2.11) is the covariance matrix obtained from $\boldsymbol{f}$s computed on a database of face class training images. The computation of these quantities are shown in Algorithm 6. Conceptually, the distance $d$ measures how different a test image is from a mean instance of the images from the face class.

$$d = \sqrt{\left(\boldsymbol{f} - \bar{\boldsymbol{f}}\right)^T \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{f} - \bar{\boldsymbol{f}}\right)} \tag{2.11}$$

$$h(\mathbf{x}) = \begin{cases} 1, \text{face}, & d \cdot p \leq d_t \cdot p \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.12}$$

The HFs $\mathscr{F}_1, \dots, \mathscr{F}_k$ used in Algorithm 6, are selected using sequential forward selection (SFS) algorithm, which will be addressed later in Algorithm 7.

Comparing (2.10) to (2.11) and (2.12), it can be noted that GWCs are computationally more expensive owing to additional subtraction and matrix multiplication operations. The question of whether its accuracy is adequate to provide a better accuracy-speed trade-off still remains, and it is answered in Sec 2.2.4. It is to be noted that, to improve testing

**Algorithm 6**: Steps involved in the training of the weak classifier

```
Input: Training images ($m$ face class and $n$ clutter class): $\mathbf{x}^{(i)}, i \in \{1, \dots, m, \dots, m+n\}$
Input: Training labels: $y^{(i)} \in \{1, -1\}, i \in \{1, \dots, m, \dots, m+n\}$
Input: Weights assigned to images: $z^{(i)}, i \in \{1, \dots, m, \dots, m+n\}$, such that $\sum_{i=1}^{m+n} z^{(i)} = 1$
Input: Haar-like features: $\mathscr{F}_1, \dots, \mathscr{F}_k$
begin
    Compute normalized weights for object class images: $\beta^{(i)}, i \in \{1, \dots, m\}$ such
    that $\beta^{(i)} = \frac{z^{(i)}}{\sum_{i=1}^{m} z^{(i)}}$
    Evaluate the HFs on all images to get $\boldsymbol{f}^{(i)} = [f_1^{(i)}, \dots, f_k^{(i)}], i \in \{1, \dots, m, \dots, m+n\}$.
    Compute the weighted mean of feature values from face class:
    $\bar{\boldsymbol{f}} = [\bar{f}_1, \dots, \bar{f}_k] = \sum_{i=1}^{m} \beta^{(i)} \boldsymbol{f}^{(i)}$
    Compute the weighted covariance matrix of feature values from face class:
    $\boldsymbol{\Sigma}^{(a,b)} = \sum_{i=1}^{m} \beta^{(i)} \left(f_a^{(i)} - \bar{f}_a\right)\left(f_b^{(i)} - \bar{f}_b\right)$
    where $a, b \in \{1, 2, \dots, k\}$
    Compute distance $d^{(i)}, i \in \{1, \dots, m, \dots, m+n\} = \sqrt{\left(\boldsymbol{f}^{(i)} - \bar{\boldsymbol{f}}\right)^T \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{f}^{(i)} - \bar{\boldsymbol{f}}\right)}$
    Find $d_t$ and $p$ that minimize the training error: $[d_t, p] = \underset{[d_t \in \mathbb{R}, p \in \{-1,1\}]}{\arg\min} \quad \epsilon$
    where, $\epsilon = \sum_{i=1}^{m} z^{(i)} |h(\mathbf{x}^{(i)}) - y^{(i)}|$
end
Output: $\bar{\boldsymbol{f}}, \boldsymbol{\Sigma}, d_t, p, \epsilon$
```

speed, the square root symbol included in (2.11) can be omitted, and the distance $d$ obtained can be compared to $d_t^2$ in (2.12).

### 2.2.2 Motivation for using GWCs

In Fig. 2.12, the joint distribution of feature values from three co-occurring HFs are shown. For simplicity of visualization, each co-occurring HF illustrated in Fig. 2.12, has two HFs in them. To obtain the joint distribution of feature values from the HFs, the features were evaluated on a database of object and clutter class images. The co-occurring HFs used for this study have been superimposed on a typical training image. It can be observed that the distributions from the face class images are more compact than those from the clutter class. This is mainly because images within the face class are more correlated with each other than those within the clutter class, which can be any random pattern. Consider a hypothetical feature space as in Fig. 2.13a where the typical distribution of measurements from face and clutter class images are illustrated. The decision spaces of

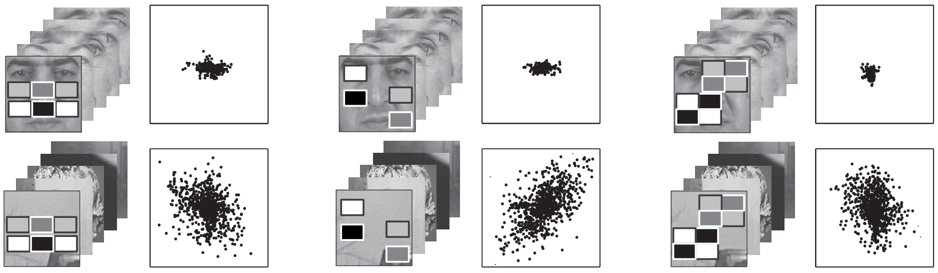


Figure 2.12: Joint distribution of feature values from co-occurring HFs when evaluated on face (top row) and clutter images (bottom row). For ease of visualization, 2D feature spaces were generated. The two HFs used to form a co-occurring HF have been super-imposed on typical images used for this study.

GWC and MWC are shown in Figs. 2.13b and 2.13d, respectively. It can be observed that, GWCs are potentially better adapted to the probability distribution function of the object class than MWCs.

### 2.2.3 Building the face detector

The cascade classifier architecture [4] is commonly used to build face detectors as it is conducive to fast scanning of a test image. As shown in Fig. 2.14, the cascade classifier consists of series of nodes that are arranged as a degenerate decision tree. Each node classifies an incoming image sub-region either as object or clutter. If an image sub-region is classified as clutter by a node, then it is not processed in the subsequent nodes. Thus, the setup intends to minimize the computational effort required to classify clutter images. Since clutter images form the majority of sub-regions that are searched for in a test image [24] [39], the cascade classifier architecture is suitable for scanning a test image quickly. Each node consists of multiple weak classifiers that are selected by using AdaBoost [30] procedure. The output of each node is a weighted sum of decisions of all of its constituent weak classifiers. The HFs used in each weak classifier are selected according to Algorithm 7.

### 2.2.4 Results

In the following, the speed-accuracy trade-off of face detectors built with VWCs, MWCs, and GWCs is compared.

To train the face detectors, two databases, face and clutter, were collected. The frontal face database was composed of 5,000 images. The facial regions were cropped manually

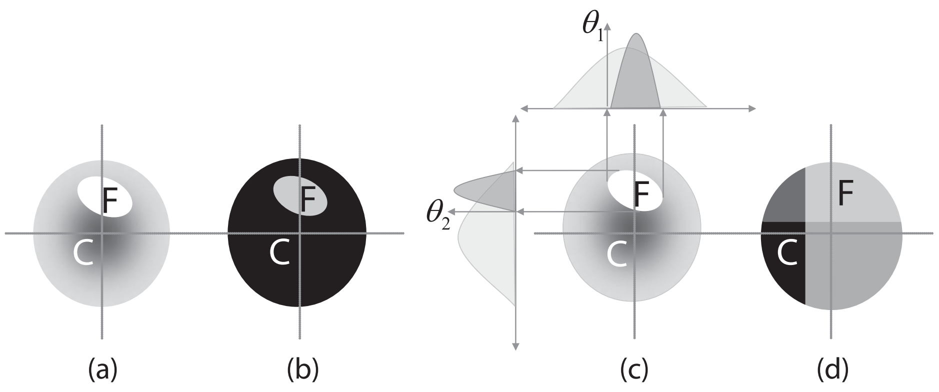


Figure 2.13: A geometrical view of the performance of GWCs and MWCs. (a) shows a hypothetical two dimensional joint feature space formed by the feature values, where F and C stand for Face and Clutter class respectively. The corresponding decision space formed by the GWC is shown in (b). MWC based on feature values from two HFs is illustrated in (c) and (d). The feature values $f_1$ and $f_2$ are used to construct independent VWC-type weak classifiers whose results are fused to generate the decision space shown in (d). The different shades of gray in (d) denote the weighted decision space. The whiter the region, the higher the probability that it belongs to face class. A threshold $\theta$ is used in (2.10) to obtain a binary classification space.

and resized to images of size $20 \times 20$ pixels. For the clutter image database, a total of 27,000 images were downloaded from the internet. These images did not contain faces.

A total of eleven face detectors were built for the following experiments. They differed only in the weak classifier used to build them. The weak classifiers used were: 1) VWCs, 2) MWCs, and 3) GWCs.

To choose the number of HFs in a weak classifier, we use two strategies. Firstly, the number of HFs were set to constants. In our experiments, combinations of 2, 3, 6 and 9 HFs were fused. The corresponding GWC-type weak classifiers are denominated GWC 2, GWC 3, GWC 6, and GWC 9, respectively. Similarly, the MWC-type weak classifiers are denominated MWC 2, MWC 3, MWC 6, and MWC 9, respectively.

In the second scheme, the number of HFs were determined automatically as performed by Mita *et al.*. In this scheme, features were added to the selected feature pool as in Algorithm 7 until the true positive rate of the fused features starts to decrease on a validation set of images. The corresponding GWC and MWC-type weak classifiers are denominated GWC A and MWC A, respectively. The average number of HFs fused to form a weak classifier for GWC A and MWC A based detectors were 3.2 and 2.7, respectively. The first five nodes of the detector had on an average 6.2 and 3.4 HFs per weak classifier.

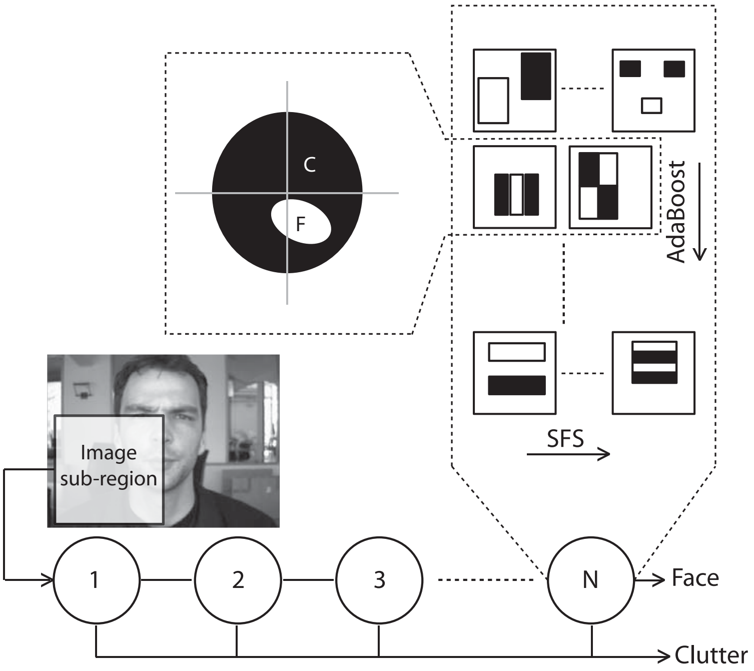


Figure 2.14: Architecture of the object detector. A cascaded classifier structure consists of multiple nodes arranged in series. The output of the node is the weighted majority of decisions of the individual classifiers present at that node. If an image sub-region is classified as clutter by a node, then it is not processed in the successive nodes.

The first five nodes are critical for computational speed of the detector during the testing phase, as a vast majority of clutter images are filtered in them. The first 9 HFs selected by the SFS algorithm are shown in Fig. 2.15.

The face detectors were constructed based on a feature pool containing 207,807 HFs. From this feature pool, 1,405 weak-classifiers were selected and arranged into 12 nodes. Readers are referred to VJ's paper [36] for details on how to select weak classifiers using AdaBoost and how they are arranged as nodes of the cascade classifier. The first five nodes of the face detectors were assigned one weak classifier each, and the rest were assigned $(nn-5)*50$ weak classifiers. Here, $nn$ stands for node number. Every node of the rejection cascade was trained so that their false rejection rate on a database of face class validation set is at most 0.01.

The accuracy and speed of the face detectors were compared on the MIT+CMU face database [29]. This database contains 130 images with 511 faces acquired in real-life

**Algorithm 7**: Selection of co-occurring HFs with Sequential Forward Selection procedure

```
Input: Feature pool that has all possible HFs:
       $\mathbb{F} = [\mathscr{F}_1, \ldots, \mathscr{F}_q]$
Input: Initialize the selected feature pool to null set:
       $\bar{\mathbb{F}} = \{\}$
Input: Stopping condition:
       Maximum number of features to be fused has been reached
       or
       Until the true positive rate on validation set does not
       decrease.
begin
    while Stopping condition is not satisfied do
        Select a feature from $\mathbb{F}$ that along with the features in $\bar{\mathbb{F}}$, produces the least
        possible error when a weak classifier is trained according to Algorithm 6:
        $\mathscr{F}_t = \arg\min_{\mathbb{F}} \epsilon$
        Add $\mathscr{F}_t$ to $\bar{\mathbb{F}}$.
    end
end
Output: $\bar{\mathbb{F}}$
```

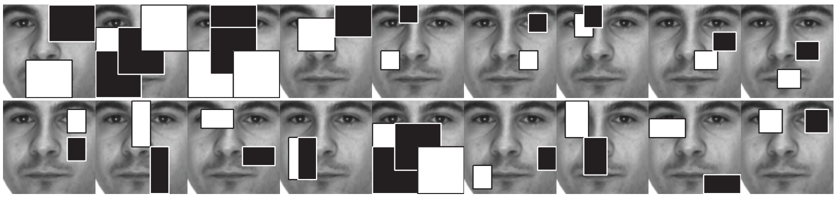


Figure 2.15: Visualization of the first nine features selected by SFS (in the first round of AdaBoost) for MWC-type (top row) and GWC-type (bottom row) face detectors. The first nodes of MWC 9- and GWC 9-based detectors contain the illustrated features. The first nodes of MWC 3- and GWC 3-based detectors contain the first three features illustrated from left to right.

conditions. Some images in this database and the output of GWC- and MWC-based face detectors are shown in Fig. 2.16.

#### 2.2.4.1 Comparison of accuracy

After each node was constructed, the object detector was tested on the MIT+CMU database. Fig. 2.17 shows the true positive rate (`tpr`) and the false positive rate (`fpr`) of the detectors after constructing each node. Three observations can be made from the plots.

1. The `fpr` measurements in the initial nodes of the GWC-based detectors are lower than those of the MWC-based detectors.

2. The `tpr` measurements of the GWC- and the MWC-based detectors are very similar for each node. This was expected because, each node of GWC- and the MWC-based detectors was trained so that its false rejection rate on a validation set of facial images is at most 0.01.

3. The `fpr` measurements in the last nodes of MWC-based detectors converge to that of GWC-based detectors. For MWC 6 and MWC 9, their `fpr` measurement at Node 12 becomes lower than the corresponding values for GWC 6 and GWC 9. Clearly, boosted ensembles of GWC-based detector seem to produce weaker ensembles than the boosted ensembles of MWCs. The reason for this could be that GWCs are less mutually complementary than MWCs. It is well known that complementary classifiers produce better ensembles [14] [19]. The reason why GWCs are less complementary than MWC could be due to the fact that the mean of the Gaussian is forced to lie among the object class measurements (Line 6 of Algorithm 6). The parameters of MWC, on the other hand, do not suffer from such a restriction. The thresholds of the VWC-type classifiers, constituting a MWC, are determined in a brute-force manner, which permits construction of diverse classifiers. A brute-force search of the centers of the GWC is very time-consuming, and therefore, was avoided.

#### 2.2.4.2 Comparison of speed

During the testing phase, the face detectors perform two tasks: 1) computation of integral image and integral image square (See [36]). In our implementation, the computation of integral images was done using Intel$^{®}$Integrated Performance Primitives 6.0 [1]; 2) scan through all possible sub-regions of a test image. The time taken to compute the two integral images constituted less than 1% of the total time required to process a $352 \times 288$ image at 10 resolutions. The rest of the time, $\sim 99\%$, was spent in scanning the image. The time taken to compute integral images is a common overhead for all the detectors. The scanning time, on the other hand, is dependent on the computational efficiency with which a detector can classify clutter images [36], which in turn is dependent on the type of weak classifier used. Assuming that the integral images have been pre-computed, we measured the average time required to label an image sub-region of the MIT+CMU database. The average times are shown in Table 2.3.

The GWC-based face detector outperformed the MWC-based detector in evaluation time. Although GWCs are computationally more expensive to evaluate (See Sec. 2.2.1.1), a GWC-based face detector is able to scan through test images faster than those built with

Table 2.3: The average time taken by GWC and MWC-type face detectors to classify an image sub-region of the MIT+CMU database. The evaluation times were measured on a 2.4 GHz processor. The standard error in measuring the average time, in all the cases, was less than 0.01 $\mu$s.

| Number of fused HFs | Time in $\mu$s | | |
|---|---|---|---|
| | VWC | GWC | MWC |
| 1 | 0.72 | - | - |
| 2 | - | 0.78 | 1.35 |
| 3 | - | 0.86 | 1.39 |
| 6 | - | 1.26 | 2.13 |
| 9 | - | 1.76 | 3.01 |
| *A* | - | 1.39 | 1.41 |

MWCs. This is because the GWC-based detector, on an average, requires fewer classifiers to label a clutter image. One exception is in the comparison of GWC A- and MWC A-based detectors. Although from Fig 2.17, one could conclude that GWC A-based detector requires fewer nodes to classify a clutter image, its speed seems very similar to that to MWC A-based detector. This is because, the first five nodes of the MWC A- and GWC A-based detectors had on an average of 3.4 and 6.2 HFs per node, respectively. The greater computational cost in each of its nodes contributes to reduction in its speed. It was found that the time required by VWC-based detector is very similar to that of GWC 2-based detector, even though GWC 2-based detector has twice the number of HFs in each of its nodes. This, again, is because the initial nodes of GWC 2-based detector classify much more clutter, and therefore, time is saved by not having to process clutter regions in the subsequent nodes.

Fig. 2.18 shows the `tpr`/`fpr`-speed plots measured at node 12 for different configurations of the detector. Three observations can be made.

1. As discussed earlier, the `tpr` of various detectors were preset to a constant value during training. Therefore, we see that all the detectors produce similar `tpr` measurements.

2. It can be noted that the higher the number of co-occurring HFs combined to form a weak classifier, the lower the false positive rate of the detector. However, on the downside, the speed of evaluation of the detector decreases.

3. GWC-based detectors seem to be be better both speed-wise and accuracy-wise when the number of co-occurring HFs are small. For example, GWC 2-based detector is 38% more accurate in `fpr` terms and simultaneously 42% faster when 12-node detectors are compared. Although with more HFs, the `fpr` measurements of MWC-

based detectors become better, the GWC-based detectors maintain a considerable advantage in speed.

### 2.2.5 Conclusions and discussions

We compared the trade-off between speed and generalization ability of the object detectors built with co-occurring Haar-like features (HFs). A key contribution of this paper is an alternative way to fuse information obtained from co-occurring HFs and form Gaussian weak classifiers. In our approach, the information from the co-occurring HF is fused at the feature-level, which permits using Gaussian weak classifiers (GWCs) to make decisions. GWCs compare the feature values of the co-occurring HFs to a $k$-dimensional non-linear decision boundary, which is learnt in a supervised manner during the training process.

The following conclusions can be drawn from our experiments. Firstly, face detectors built using fused co-occurring features are a viable alternative to the Viola & Jones's approach. The proposed approach gives 67% better false positive rate for the same execution time and the same true positive rate. Secondly, feature-level fusion of information from co-occurring Haar-like features produces more accurate detectors. When compared to detectors based on Mita *et al.*'s approach, the proposed detector is 38% more accurate in false positives and simultaneously 42% faster given similar true positive rates. Thirdly, using two or three Haar-like features in a co-occurring feature seems to provide the best speed-accuracy trade-off for the face detection problem.

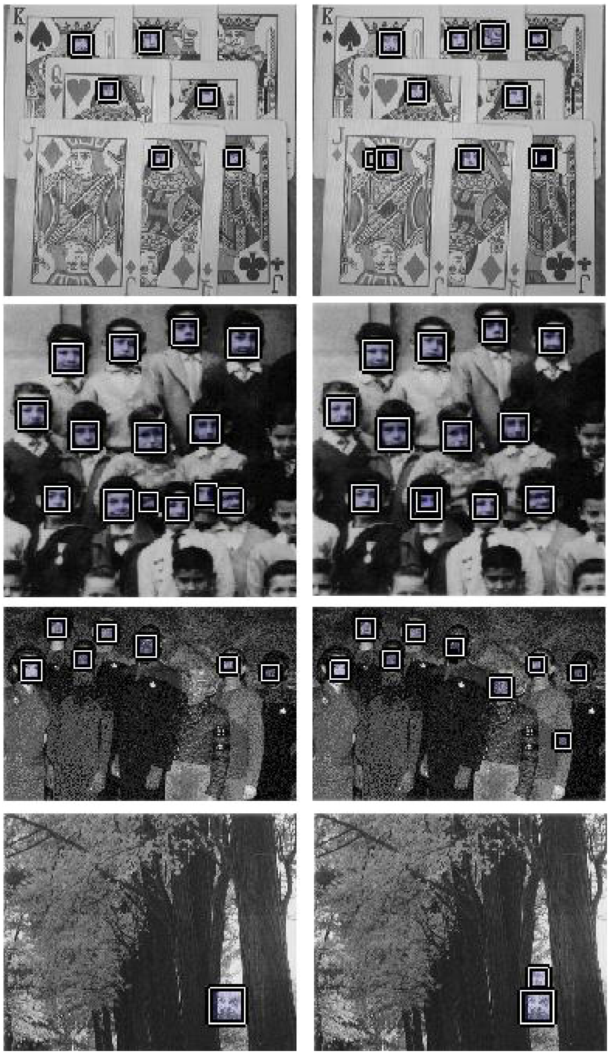


Figure 2.16: Results of the evaluation of detectors based on GWC 3 (left) and MWC 3(right) weak classifiers on images from the MIT+CMU database. In general, it was observed that face detector based on GWC 3 detected fewer faces correctly when compared to MWC 3-based detector. However, GWC 3-based detector was faster by 38% and produced fewer false positives.

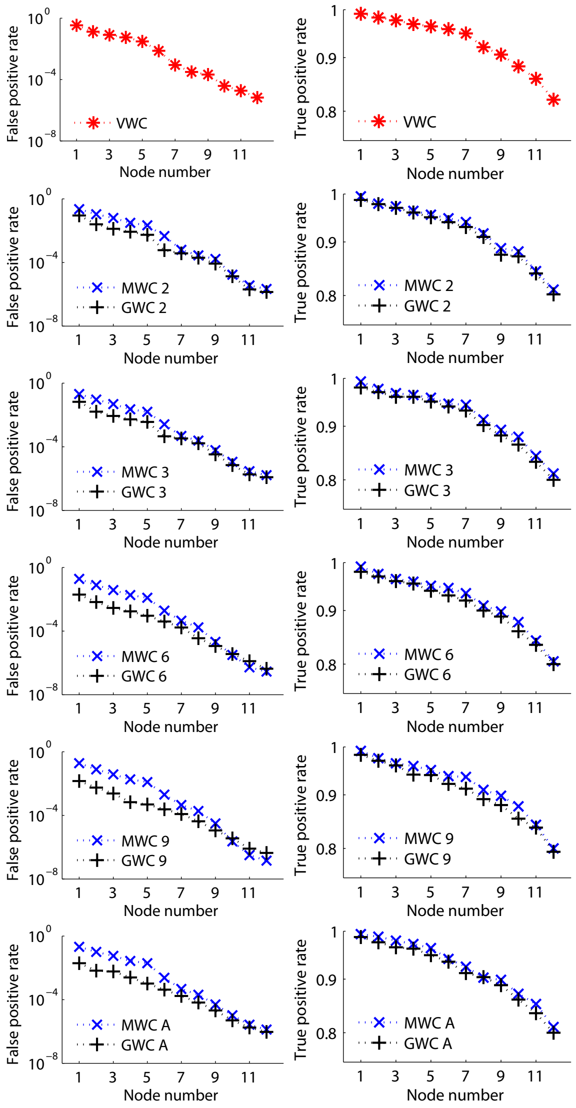


Figure 2.17: A plot of false positive rate (left) and true positive rate (right) at each node for different configurations of face detectors. Tests were performed on the MIT+CMU database.

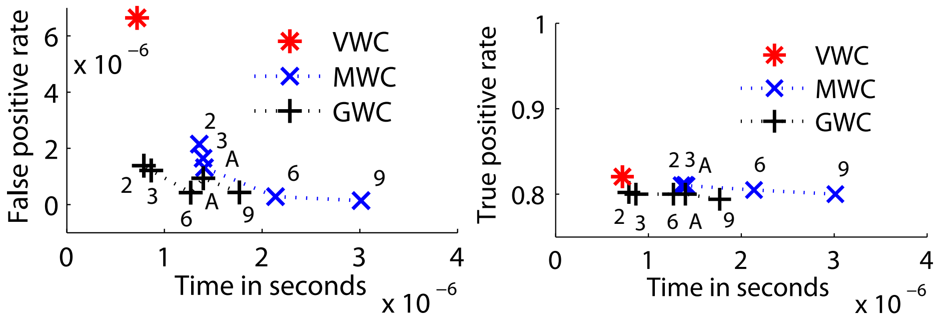


Figure 2.18: A plot of false positive (left) / true positive (right) rate and the average time required for classification for different configurations of face detectors. The numbers besides each measurement point in the plot above indicates the number of co-occurring HFs fused to form a weak classifier. The letter 'A' indicates that the number of HFs in a weak classifier was chosen in an automatic way as detailed in Algorithm 7.

## 2.3 Gaussian weak classifiers based on Haar-like features with four rectangles for real-time face detection

Although, in theory, any number of rectangular regions can be used to form a Haar-like feature (HF), for practical reasons, the number of rectangles used are restricted to two (HF2), three (HF3) or four (HF4). If $n$ is the number of distinct rectangles that can be fit in a template of given size, then the number of HF2s, HF3s, and HF4s that can be constructed is of the order $O(n^2)$, $O(n^3)$, and $O(n^4)$, respectively. As HF4s form the majority of Haar-like features used for training, it is of interest to improve discrimination power of weak classifiers constructed with them. In this section, we investigate GWCs based on HF4s (See Fig. 2.1b).

GWCs classify an image as face or clutter in two steps. Firstly, the response of a HF4 is split into two components, each belonging to a HF2. As discussed in Section 2.3.3, the motivation behind the split is to take advantage of the compact and Gaussian-like distribution of feature values from the face class images. Secondly, the responses of HF2s are used to compute a Mahalanobis distance which is compared to a threshold to make decisions. In our experiments, we compare the speed and the accuracy of the GWC-based face detector with equivalent detectors that use weak classifiers proposed by Viola and Jones [36], Rasolzadeh *et al.* [27] and Mita *et al.* [23]. Our results, presented in Section 2.3.4, show that GWC-based face detectors provide the best trade-off between speed and accuracy.

### 2.3.1 Related work

The seminal paper by Viola and Jones [36] spurred a lot of interest in object detection. Thereafter, several papers were published, mainly focussing on the following three parallel lines of improvements.

1. The geometrical diversity of the HFs was increased to obtain better performance both in terms of accuracy and speed [12][16][18].

2. The AdaBoost [30] procedure used to select weak classifiers in [36] was improved in [16][17][35].

3. The linear weak classifiers used by Viola and Jones were replaced by weak classifiers that provided a better accuracy-speed trade-off. Rasolzadeh *et al.* [27] demonstrated that more accurate pedestrian detectors can be achieved by increasing the discriminating strength of the individual weak classifiers. Their weak classifiers were obtained through response binning [27], which can be thought of as assigning multiple thresholds to the response of HFs. Viola and Jones, in comparison, use a single threshold on the feature value of the HFs to make decisions. Mita *et al.* [23] fuse outputs of multiple linear weak classifiers to form more powerful ones. Their weak classifiers produce lower error rates than the Viola and Jones's linear weak classifiers. The GWCs, that are proposed in this section, fall into the third category of improvements.

### 2.3.2 Weak classifiers based on HFs

Haar-like features [25], shown in Fig. 2.1, consist of two or more rectangular regions enclosed in a template. Such features, when evaluated on an image, produce a feature value as in (2.13).

$$f_t = \sum_{i=1}^{q} w^{(i)} \cdot \mu^{(i)} \tag{2.13}$$

where $i$ is an iterator that iterates through all the $q$ rectangles of the HF. The quantity $\mu^{(i)}$ represents the mean intensity of the pixels in image $\mathbf{x}$ enclosed within the $i^{th}$ rectangle. Every rectangle in the HF is assigned a weight that is represented by $w^{(i)}$. The weights are set to *default* integer numbers such that $\sum_{i=1}^{q} w^{(i)} = 0$ is satisfied. For example, the rectangles of a HF2 as in Fig. 2.1a are assigned default weights 1 and $-1$. The rectangles of a HF3 as in Fig. 2.1c are assigned default weights 1, $-2$ and 1.

Viola and Jones's weak classifiers ($h_{vj}(\mathbf{x})$) compare the feature value $f_t$ to a threshold $\theta$ according to (2.14).

$$h_{vj}(\mathbf{x}) = \begin{cases} 1, \text{face}, & f_t \cdot p \leq \theta \cdot p \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.14}$$

Here, $p \in \{1, -1\}$ is a polarity term, which can be used to invert the inequality relationship between $f_t$ and $\theta$.

Rasolzadeh *et al.*'s weak classifier [27] ($h_r(\mathbf{x})$) compares $f_t$ to two threshold values ($\theta_1$ and $\theta_2$) as shown in (2.15).

$$h_r(\mathbf{x}) = \begin{cases} 1, \text{face}, & \theta_1 \leq f_t \leq \theta_2 \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.15}$$

Mita *et al.*'s weak classifier [23] ($h_m(\mathbf{x})$) fuses $k$ Viola and Jones's weak classifiers to make decisions as shown in (2.16).

$$h_m(\mathbf{x}) = \begin{cases} 1, \text{face}, & \sum_{i=1}^{k} 2^{k-i} h_{vj}^{(i)}(\mathbf{x}) \geq \theta \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.16}$$

#### 2.3.2.1 Gaussian weak classifiers

As stated earlier, we define GWCs using HF4s. A HF4 (Fig. 2.1b) can be considered to be a combination of two HF2s (Fig. 2.1a). Therefore, the feature value $f_t$ of a HF4 can be split into two components, $f_1 = \sum_{i=1}^{2} w^{(i)} \cdot \mu^{(i)}$ and $f_2 = \sum_{i=3}^{4} w^{(i)} \cdot \mu^{(i)}$, each belonging to a HF2. Classification by GWC is performed in two steps. Firstly, a Mahalanobis distance $d$ is computed using $\boldsymbol{f} = [f_1 \ f_2]$ as shown in (2.17). Secondly, the computed distance is

**Algorithm 8**: Steps involved in training a weak classifier

```
Input: Face class training images: $\mathbf{x}^{(i)}, i \in \{1,\dots,m\}$
Input: Weights assigned to images of the face class: $z^{(i)}, i \in \{1,\dots,m\}$, such that $\sum_{i=1}^{m} z^{(i)} = 1$
Input: Weak classifier: $h$
begin
    Evaluate the HF associated with $h$ on all face class images to get $\boldsymbol{f}^{(i)} = [f_1^{(i)}\ f_2^{(i)}], i \in \{1,\dots,m\}$.
    Compute the weighted mean of feature values: $\bar{\boldsymbol{f}} = \sum_{i=1}^{m} z^{(i)} \cdot \boldsymbol{f}^{(i)}$
    Compute the weighted covariance matrix ($\boldsymbol{\Sigma}$): $\boldsymbol{\Sigma}^{(a,b)} = \sum_{i=i}^{m} z^{(i)}(f_a^{(i)} - \bar{f}_a)(f_b^{(i)} - \bar{f}_b)$
    where $a, b \in \{1,2\}$.
end
Output: $\bar{\boldsymbol{f}}, \boldsymbol{\Sigma}$
```

```
Input: Training images (both face and clutter): $\mathbf{x}^{(i)}, i \in \{1,\dots,m\}$
Input: Training labels: $y^{(i)} \in \{1,-1\}, i \in \{1,\dots,m\}$
Input: Weights for each training image: $z^{(i)} \in \Re, i \in \{1,\dots,m\}$
Input: Weak classifier to be trained: $h$
Input: $\bar{\boldsymbol{f}}$ and $\boldsymbol{\Sigma}$
begin
    Evaluate the HF associated with $h$ on all face class images to get $\boldsymbol{f}^{(i)} = [f_1^{(i)}\ f_2^{(i)}], i \in \{1,\dots,m\}$.
    Compute distance $d^{(i)} \in \mathbf{d}$: $d^{(i)} = \sqrt[2]{\left(\boldsymbol{f}^{(i)} - \bar{\boldsymbol{f}}\right)^T \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{f}^{(i)} - \bar{\boldsymbol{f}}\right)}$
    Find $d_t$ and $p$ that minimize the training error: $[d_t, p] = \underset{[d_t \in \mathbf{d}, p \in \{-1,1\}]}{\arg\min}\ \epsilon$
    where, $\epsilon = \sum_{i=1}^{m} z^{(i)} |h(\mathbf{x}^{(i)}) - y^{(i)}|$
end
Output: $d_t$, $p$
```

compared to a threshold to make decision on whether the test image belongs to face or clutter as in (2.18).

The quantities $\bar{\boldsymbol{f}}$ and $\boldsymbol{\Sigma}$ in (2.17) are the mean and the covariance matrix obtained from $\boldsymbol{f}$ computed on a database of face class training images. The computation of these quantities are shown in Algorithm 8 (left). Conceptually, the distance $d$ measures how different a test image is from a mean instance of the images from the face class. In (2.18), $d$ is compared to a threshold value $d_t$ to decide whether the image belongs to the face or to the clutter class. The quantities $d_t$ and $p$ are determined as shown in Algorithm 8 (right).

$$d = \sqrt[2]{\left(\boldsymbol{f} - \bar{\boldsymbol{f}}\right)^T \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{f} - \bar{\boldsymbol{f}}\right)} \tag{2.17}$$

$$h(\mathbf{x}) = \begin{cases} 1, \text{face}, & d \cdot p \le d_t \cdot p \\ -1, \text{clutter}, & \text{otherwise} \end{cases} \tag{2.18}$$

Comparing (2.14), (2.15), (2.16) to (2.17) and (2.18), it can be noted that GWCs are computationally more expensive owing to additional subtraction, matrix multiplication and the square root operation. The question of whether its accuracy is adequate to provide a better accuracy-speed trade-off still remains to be seen.

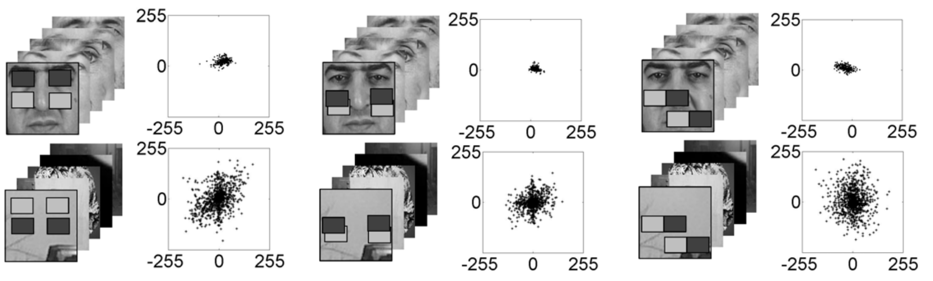


Figure 2.19: Joint distribution of feature values, $f_1$ and $f_2$, obtained from two HF2s when evaluated on face (top row) and clutter images (bottom row). The HF2s used to generate the plot have been super-imposed on the face and clutter class images. As face class images are correlated with each other, we observe that the distribution of feature values from the face class are more compact than those obtained from the clutter class.

### 2.3.3 Motivation for using Gaussian weak classifiers

Rasolzadeh *et al.* [27] experimentally observed that the distribution of the feature values of HFs when evaluated on a object and clutter class images resemble a Gaussian distribution. Two-dimensional joint feature spaces spanned by the feature values, $f_1$ and $f_2$, from three arbitrarily chosen pairs of HF2s are shown in Fig. 2.19.

Consider a hypothetical feature space as in Fig. 2.20a where the measurements from face and clutter class images are overlapped. Geometrically, the computation of feature value $f_t$ of a HF4, can be understood as a projection of the 2D feature space on to a 1D space as shown in Fig. 2.20c. To train a Viola and Jones's type weak classifier, a scalar value $\theta$ is found such that face and clutter distributions are best separated. As shown in the Fig. 2.20(d), $\theta$ partitions the feature space into two regions; the face region, coded white, and the clutter region, coded black. The decision space of the GWC and weak classifiers proposed by Rasolzadeh *et al.* and Mita *et al.* are shown in Figs. 2.20b, 2.20f and 2.20h, respectively. Among the existing weak classifiers, GWC has the potential to extract the maximum discrimination ability from a HF4. As HF4s form the majority of HFs in the feature pool that is used to train the face detector, the classification power of a majority of the weak classifiers can be potentially increased by using GWCs.

### 2.3.4 Results

In the following experiments, we compare the speed and the accuracy of the face detectors constructed with GWCs and the weak classifiers proposed by Viola and Jones [36], Rasolzadeh *et al.* [27] and Mita *et al.* [23].

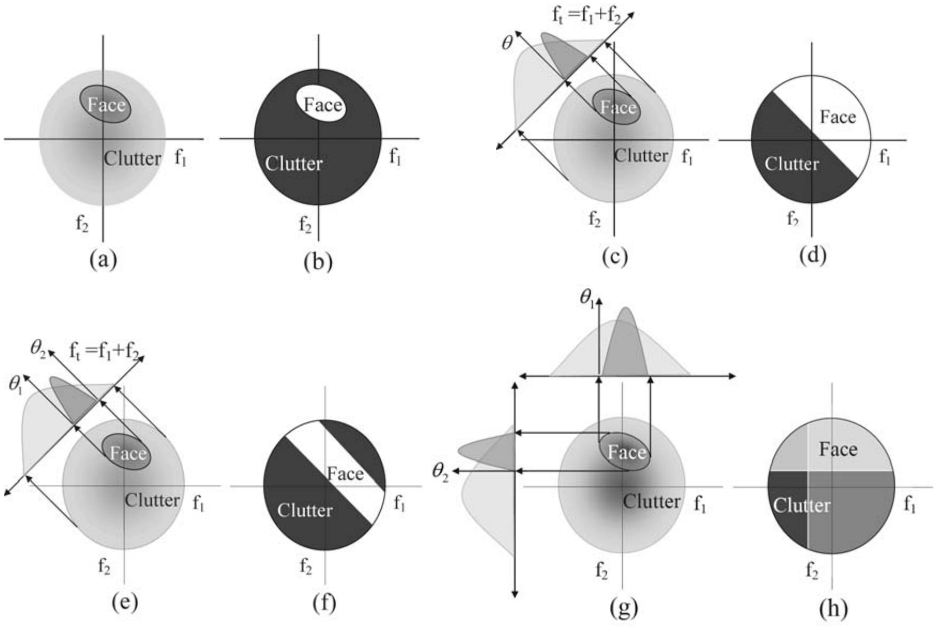


Figure 2.20: A geometrical view of the performance of GWC and the weak classifiers used by Viola and Jones [36], Rasolzadeh *et al.* [27] and Mita *et al.* [23]. (a) shows a hypothetical joint feature space formed by the feature values $f_1$ and $f_2$. (b) shows the partitioned feature space using GWC. (c) shows the projection of 2D space formed by $f_1$ and $f_2$ onto a 1D space, which represents the feature value of a HF4 computed in the traditional way. To train a weak classifier used by Viola and Jones, a threshold $\theta$ is found to separate face from clutter. (d) shows the corresponding partitioned feature space. (e) shows a geometrical view of the Rasolzadeh *et al.*'s weak classifiers which use two thresholds to separate face from clutter; the corresponding partitioned feature space is shown in (f). Mita *et al.*'s weak classifier that fuses two HF2s is illustrated in (g) and (h). The feature values $f_1$ and $f_2$ from the two HF2s are used to construct independent weak classifiers whose results are fused to generate the decision space shown in (h).

To train the face detector, two databases, face and clutter, were collected. The frontal face database was composed of 5,000 images. The facial regions were cropped manually and resized to images of size $20 \times 20$ pixels. For the clutter image database, a total of 27,000 images were downloaded from the internet. These images did not contain faces.

Four face detectors were built. They differed only in the type of weak classifier used to construct them. The weak classifiers used were:

1. GWC: Proposed weak classifiers as in (2.18).
2. VJ: Viola and Jones's [36] weak classifiers as in (2.14).
3. RPP: Rasolzadeh *et al.*'s [27] weak classifiers as in (2.15).
4. MKSH: Mita *et al.*'s [23] weak classifiers as in (2.16).

The face detectors were constructed based on a feature pool containing $175,429$ HF4s. From this feature pool, $2,255$ weak-classifiers were selected and arranged into 14 nodes. The GWC and MKSH-type weak classifiers split the response of a HF4 into two HF2s, and perform classification as defined by (2.18) and (2.16). Readers are referred to Viola and Jones's paper [36] for details on how the features are selected and arranged into nodes. The first five nodes of the face detectors were assigned one weak classifier each, and the rest were assigned $(nn-5)*50$ weak classifiers. Here, $nn$ stands for node number. Every node of the rejection cascade was trained so that their false rejection rate on a database of face class validation set is at most 0.01.

The accuracy and speed of the face detectors were compared on the MIT+CMU face database [29]. Fig. 2.21 shows the Receiver Operating Characteristics (ROC) curves obtained by testing the face detectors on this database. Each point on the ROC curve was generated by varying the number of nodes in the face detector. As the false rejection rate of each node was pre-set to a constant value during training, we observe that the face detectors, at each operating point, have similar true positive rates. A rectangular bounding box has been used to group ROC points generated using face detectors working at similar operating points, *i.e.,* , with the same number of nodes and weak classifiers.

During the testing phase, the face detectors perform two tasks: 1) computation of integral image and integral image square (See [36]). In our implementation, the computation of integral images was done using Intel® Integrated Performance Primitives 6.0 [1]. 2) scan through all possible sub-regions of a test image. The time taken to compute the two integral images constituted less than 1% of the total time required to process a $352\times288$ image at 10 resolutions. The rest of the time, $\sim 99\%$, was spent in scanning the image. The time taken to compute integral images is a common overhead for all four detectors. The scanning time, on the other hand, is dependent on the computational efficiency with which a detector can process clutter images [36], which is dependent on the type of weak classifier used. Assuming that the integral images have been pre-computed, we measured the average time required to label an image sub-region of the MIT+CMU database. The average times are shown along with the legend in Fig. 2.21.

The GWC-based face detector outperformed the rest significantly both in accuracy and evaluation time. Although GWCs are computationally more expensive to evaluate (See Sec. 2.3.2.1), a GWC-based face detector is able to scan through test images faster than those built with the traditional weak classifiers. This is because GWCs, on an average, require fewer classifiers to label a clutter image.

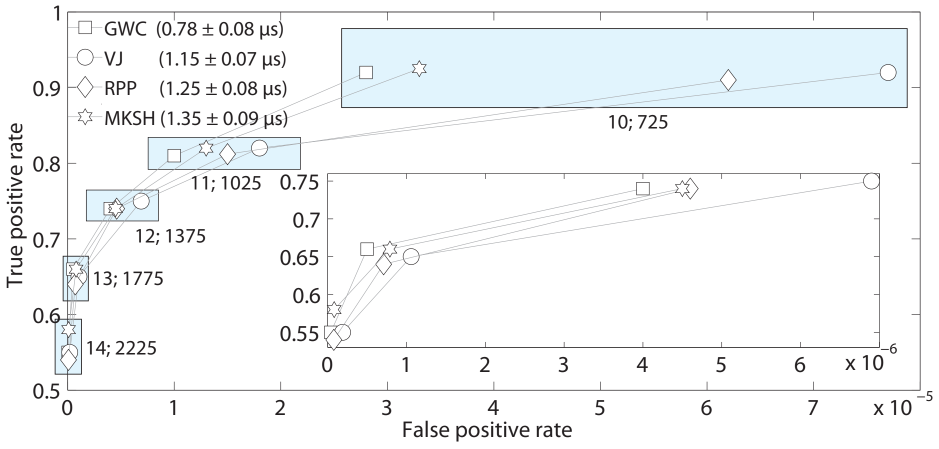


Figure 2.21: Comparison of ROC curves of face detectors constructed with GWC, VJ, RPP, and MKSH-type weak classifiers. The detectors were tested at different operating points which were defined by the number of nodes in them. The ROC points generated at equivalent operating points, *i.e.,* , with same number of nodes and weak classifiers, have been bounded by a rectangular box. The numbers (separated by a semicolon) beside the bounding box indicate the number of nodes and the number of weak classifiers used to build them. The inset shows a zoomed version of the ROC curves for the operating points defined by 12, 13 and 14 nodes. The average time, in microseconds, required to process an image sub-region of the MIT+CMU database (excluding the time required to compute integral images) is listed along with the legend.

### 2.3.5 Conclusions

This section proposes Gaussian weak classifiers (GWCs) as an alternative to the traditional ones proposed by Viola and Jones, Rasolzadeh *et al.* and Mita *et al.* GWCs are formulated based on Haar-like features with four rectangles (HF4s). To make a decision using GWC, the feature values of the two HF2s in a HF4 are compared to a 2D non-linear decision boundary, which is learnt in a supervised manner using images from face and clutter class. Our results on the MIT+CMU face database show that GWC-based face detectors produce at least 40% lesser false positives and require 32% lesser time for the scanning process when compared to Viola and Jones's face detector. In comparison to face detectors based on Rasolzadeh *et al.*'s and Mita *et al.*'s weak classifiers, the decrease in false positives was at least 11% and 10% respectively. Simultaneously, the GWC-based detector was faster by 37% and 42% to make decisions.

## 2.4 References

[1] Intel®integrated performance primitives 6.0, http://software.intel.com/en-us/intel-ipp/, 2009.

[2] CBCL Face Database 1. MIT Center for Biological and Computation Learning. http://www.ai.mit.edu/projects/cbcl, 1996.

[3] S Baker. *Design and evaluation of feature detectors*. PhD thesis, Columbia University, 1998.

[4] S Baker and S K Nayar. Pattern rejection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 544–549, San Francisco, CA, USA, 1996.

[5] T Cooke. Two variations on Fisher's linear discriminant for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):268–273, 2002.

[6] D Delgado-Gomez, L H Clemmensen, B Ersbøll, and J M Carstensen. Precise acquisition and unsupervised segmentation of multi-spectral images. *Computer Vision and Image Understanding*, 106(2):183–193, 2007.

[7] R O Duda, P E Hart, and D G Stork. *Pattern Classification, 2nd Ed.* Wiley, New York, 2000.

[8] D E Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[9] J H Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA, 1992.

[10] A Jain and D Zongker. Feature selection: evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

[11] O Jesorsky, K J Kirchberg, and R Frischholz. Robust face detection using the Hausdorff distance. In *Proceedings of the International Conference on Audio- and Video-Based Biometric Person Authentication, Lecture Notes in Computer Science 2091*, pages 90–95, London, UK, 2001.

[12] M Jones and P Viola. Fast multi-view face detection. Technical report, Mitsubishi Electric Research Laboratories, 2003.

[13] K A De Jong. *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, The University of Michigan, 1975.

[14] L I Kuncheva, C J Whitaker, C A Shipp, and R P W Duin. Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis and Applications*, 6:22–31, 2003.

[15] K Levi and Y Weiss. Learning object detection from a small number of examples: the importance of good features. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 53–60, Washington, DC, USA, 2004.

[16] S Z Li, L Zhu, Z Zhang, A Blake, H Zhang, and H Shum. Statistical learning of multi-view face detection. In *Proceedings of the European Conference on Computer Vision, Lecture Notes in Computer Science 2353*, pages 67–81, London, UK, 2002.

[17] R Lienhart, E Kuranov, and V Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *Proceedings of the DAGM 25th Pattern Recognition Symposium*, pages 297–304, Magdeburg, Germany, 2003.

[18] R Lienhart and J Maydt. An extended set of Haar-like features for rapid object detection. In *Proceedings of the International Conference on Image Processing*, pages 900–903, Rochester, New York, USA., 2002.

[19] W Liu, Z Wu, and G Pan. An entropy-based diversity measure for classifier combining and its application to face classifier ensemble thinning. In *Proceedings of the Fifth Chinese Conference on Biometric Recognition, Lecture Notes in Computer Science 3338*, pages 118–124, Guangzhou, China, 2004.

[20] A M Martinez and R Benavente. The ar face database. Technical Report 24, Computer Vision Center, Universitat Autonoma de Barcelona, 2000.

[21] K Messer, J Matas, J Kittler, J Luettin, and G Maitre. XM2VTSDB: The extended M2VTS database. In *Proceedings of the International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, Washington, DC, USA, 1999.

[22] T Mita, T Kaneko, and O Hori. Joint Haar-like features for face detection. In *Proceedings of the International Conference on Computer Vision*, pages 1619–1626, Washington, DC, USA, 2005.

[23] T Mita, T Kaneko, B Stenger, and O Hori. Discriminative feature co-occurrence selection for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1257–1269, 2008.

[24] D Mumford and B Gidas. Stochastic models for generic images. *Quarterly of Applied Mathematics*, LIV:85–111, 2001.

[25] C P Papageorgiou, M Oren, and T Poggio. A general framework for object detection. In *ICCV '98: Proceedings of the International Conference on Computer Vision*, pages 555–562, Washington, DC, USA, 1998.

[26] S-K Pavani, D Delgado, and A F Frangi. Gaussian weak classifiers based on Haar-like features with four rectangles for real-time face detection. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns, Lecture Notes in Computer Science 5702*, pages 91–98, Münster, Germany, 2009.

[27] B Rasolzadeh, L Petersson, and N Pettersson. Response binning: Improved weak classifiers for boosting. In *Proceedings of the Intelligent Vehicles Symposium*, pages 344–349, 2006.

[28] D Roth, M Yang, and N Ahuja. A SNoW-based face detector. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, pages 855–861, Denver, CO, USA, 2000.

[29] H A Rowley, S Baluja, and T Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

[30] R E Schapire. A brief introduction to boosting. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1401–1406, San Francisco, CA, USA, 1999.

[31] R E Schapire. Theoretical views of boosting and applications. In *Proceedings of the International Conference on Algorithmic Learning Theory*, pages 13–25, London, UK, 1999.

[32] H Schneiderman. *A statistical approach to 3D object detection applied to faces and cars.* PhD thesis, Carnegie Mellon University, 2000.

[33] A Selinger and D Socolinsky. Appearance-based facial recognition using visible and thermal imagery: a comparative study, 1999.

[34] Z Sun, G Bebis, and R Miller. On-road vehicle detection: A review. *IEEE Transactions on Pattern Anaysis and Machine Intelligence*, 28(5):694–711, 2006.

[35] P Viola and M J Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *Proceedings of the Sixteenth Annual Conference on Advances in Neural Information Processing Systems*, pages 1311–1318, Vancouver, BC, Canada, 2002.

[36] P Viola and M J Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[37] P Viola, M J Jones, and D Snow. Detecting pedestrians using patterns of motion and appearance. In *Proceedings of the International Conference on Computer Vision*, pages 734–741, Washington, DC, USA, 2003.

[38] J Šochman and J Matas. Waldboost - learning for time constrained sequential detection. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, volume 2, pages 150–157, Los Alamitos, USA, 2005.

[39] J Wu, J M Rehg, and M D Mullin. Learning a rare event detection cascade by direct feature selection. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, pages 855–861, Vancouver, BC, Canada, 2004.

[40] D Zhang, S Z Li, and D Gatica-Perez. Real-time face detection using boosting in hierarchical feature spaces. In *Proceedings of the International Conference on Pattern Recognition*, pages 411–414, Washington, DC, USA, 2004.

# 3

# Accelerating the training phase of facial detectors

Be a politician; no training necessary.

*Will Rogers*

The Viola and Jones's (VJ's) object detection system, popular for its high accuracy at real-time testing speeds, has a drawback that it is slow to train. A face detector, for example, can take days to train. In applications such as Content-Based Image Retrieval (CBIR), such a long training is not affordable. In this chapter, we propose two training procedures to achieve fast training speed.

In the first approach, we reduce the training time to the order of minutes through the following three modifications to the VJ's approach. Firstly, clutter (non-object) models are used instead of using images. Thus, time is saved in not having to read and evaluate them on thousands of Haar-like features. Secondly, we use a smaller non-redundant set of Haar-like feature than the feature pool used by Viola and Jones. Thirdly, weak classifiers, with fewer parameters to be optimized, are used for training.

In the second approach, we reduce the training time to the order of seconds by introducing further changes. Firstly, Laplacian clutter models that better fit the clutter distributions are used. These models serve to better predict the error of the weak classifier. Secondly, we simplify the training procedure by removing the time-consuming AdaBoost-based feature selection procedure.

Our results show that the accuracy of the detector, built with the proposed approaches, is inferior to that of VJ's for difficult object class such as frontal faces. However, for objects with lesser degree of intra-class variations such as hearts, state-of-the-art accuracy can be obtained. Importantly, for CBIR applications, the fast testing speed of the VJ-type object detector is maintained.

Adapted from:

S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, *A Rapidly Trainable and Global Illumination Invariant Object Detection System.* In Proc. Iberoamerican Congress on Pattern Recognition, Guadalajara, Jalisco, México. Lecture Notes in Computer Science vol. 5856, Pages 877-884, 2009.

S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, *Fast training of Viola-Jones type Object Detectors using Laplacian Clutter Models.* Submitted, 2010.

## 3.1 A Rapidly Trainable and Global Illumination Invariant Object Detection System

Although object detectors based on Haar-like features (HFs)[11] achieve high accuracy rates in real-time [20], training them is a time-consuming task. This is because thousands of weak classifiers based on HFs need to be trained using a database of object and clutter (non-object) images. VJ reported training time in the order of weeks using $180,000$ features on a 466 MHz AlphaStation XP900 [20]. Reduced training time of about 2 days using approximately $20,000$ features can be achieved using the implementation in the OpenCV library [1] on a 3 GHz processor. Though at first glance, it may seem that two days of training time is affordable, the total algorithmic development time generally exceeds this time frame. Many trials may be required to optimize the performance of the detector, which could prolong the effective development time to months. As McCane and Novins [9] point out, long training times make testing new algorithms or verifying past results extremely difficult.

Several possible approaches have been proposed to reduce the high training time. For instance, Wu *et al.* [21] who achieved a reduction in training time of approximately two orders of magnitude by pre-training weak classifiers before the iterative classifier selection procedure. Stojmenovic [19] proposed to reduce the training time by pre-eliminating HFs from the original training set. They eliminate HFs which produce error greater than a pre-determined threshold value. On a database of images containing back-view of Honda Accord cars, they could eliminate 97% of the original features, thereby achieving a potential speed increase of up to two orders of magnitude. However, it is not clear what percentage of HFs can be removed on more challenging images like those of human frontal faces. Pham *et al.* [14] proposed decreasing the training time by pre-computing the global statistics of face and non-face images. They reported a training time of 5 hours and 30 minutes while achieving high accuracy.

In this work, we propose a novel algorithm that reduces the training time to the order of seconds in a conventional desktop computer with a 3 GHz processor. The high training speed is due to the following three reasons. Firstly, a clutter model is used instead of using clutter class images. This results in a substantial reduction of training time because approximately $10^7$ clutter image regions are used for training by traditional training methods. The weak classifiers used in the prosed approach, as will be seen in Section 3.1.1.1, implicitly incorporate the clutter model and therefore, the model need not be trained. Secondly, we heuristically pre-eliminate HFs in the feature pool to obtain a set of features that make independent measurements on clutter. Using lesser HFs during training also contributes to the faster training speed. Further, the weak classifiers used in our procedure have fewer parameters to be optimized and therefore, are faster to train.

### 3.1.1 Haar-like features and weak classifiers

Haar-like features (HFs), shown in Fig. 2.1, are an over-complete set of two-dimensional Haar functions, which can be used to encode local appearance of objects [11]. The feature value $f$ of a Haar-like feature which has $k$ rectangles is obtained as in (3.1). The quantity $\mu^{(i)}$ is the mean intensity of the pixels in image $\mathbf{x}$ enclosed by the $i^{th}$ rectangle and $w^{(i)}$ is

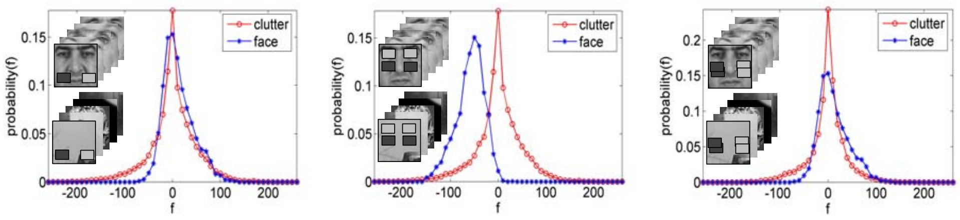


Figure 3.1: Three histograms of feature values obtained by evaluating face and clutter class images on HFs are shown. To the left of the histograms, the HFs that were used for evaluation have been super-imposed on a typical training image. As Huang and Mumford [6] observed, the distribution of feature values from clutter images tends to a Laplacian distribution centered at zero.

the weight assigned to the $i^{th}$ rectangle. The weights assigned to the rectangles of a HF are set to default numbers satisfying (3.2). Weak classifiers that label an image $\mathbf{x}$ as object (+1) or clutter (-1) can be expressed as in (3.3). The quantity $\theta \in \Re$ is a threshold value, and $p \in \{1, -1\}$ can be used to invert the inequality relationship. Training such a weak classifier involves setting appropriate values to its threshold and polarity coefficients ($\theta^*$, $p^*$) such that the overall error is minimized. Formally, $[\theta^*, p^*] = \underset{[\theta,p]}{\arg\min} \sum_{i=0}^{n_o+n_c} \epsilon^{(i)}$. If a training image is correctly classified, then its error is $z^{(i)}$, else it is 0. The term $z^{(i)}$ is the weight assigned to the training image $\mathbf{x}^{(i)}$. The quantities $n_o$ and $n_c$ are the number of object and clutter class training images, respectively. Training the weak classifiers as in (3.3) can be intuitively understood from Fig. 3.1. For each HF shown in Fig. 3.1, histograms of the feature value, $f$, have been obtained from object (human frontal face) and clutter training images. During training, $\theta$ is set to the value of $f$ that best separates object and clutter examples.

$$f = \sum_{i=1}^{k} w^{(i)} \cdot \mu^{(i)} \tag{3.1}$$

$$\sum_{i=1}^{k} w^{(i)} = 0 \tag{3.2}$$

$$h(\mathbf{x}) = \begin{cases} +1, f_{(\theta,p)} > 0 \\ -1, \text{otherwise} \end{cases} \tag{3.3}$$

$$f_{(\theta,p)} = (f - \theta) \cdot p \tag{3.4}$$

#### 3.1.1.1 A clutter model

When a HF is evaluated on a clutter image, the expectation value of the output can be expressed as in (3.5).

$$E(f) = E\left(\sum_{i=1}^{k} w^{(i)}\mu^{(i)}\right) = \sum_{i=1}^{k} w^{(i)} E\left(\mu^{(i)}\right) \tag{3.5}$$

The clutter class, being generic, may contain any image with any appearance pattern. Effectively, every pixel of a generic clutter image is a random variable which can take any value between the minimum and the maximum permitted pixel values in an image representation ($N_{min}$ and $N_{max}$) with equal probability. For example, in gray-level images, $N_{min} = 0$ and $N_{max} = 255$. Therefore, the expected value of mean of pixel values within any rectangular region, $E(\mu) = 0.5(N_{max} + N_{min})$. Rewriting (3.5) using (3.2), we get (3.6).

$$E(f) = 0.5(N_{max} + N_{min}) \sum_{i=1}^{k} w^{(i)} = 0 \tag{3.6}$$

Therefore, the probability that the feature value of a HF on a clutter image to be greater than (or lesser than) 0 is 0.5. Mathematically, $\mathbb{P}(f \cdot p > 0 | \mathbf{x}^{(i)} \in \texttt{Clutter}) = 0.5$. Using the terminology introduced in (3.4),

$$\mathbb{P}(f_{(0,p)} > 0 | \mathbf{x}^{(i)} \in \texttt{Clutter}) = 0.5 \tag{3.7}$$

The clutter model in (3.7) can be observed from the clutter histograms shown in Fig. 3.1. Note that the clutter histograms are all symmetric and centered at $f = 0$.

#### 3.1.1.2 Proposed weak classifier

The proposed weak classifier utilizes the clutter model in (3.7) by setting its threshold $\theta = 0$ so that it labels 50% of the clutter correctly. Since $\theta$ is already set, training the proposed weak classifier only involves setting an appropriate value to the polarity term ($p^*$) such that the training error is minimized as shown in (3.9). As $\theta$ need not be optimized, the training speed of the weak classifiers is much higher than the traditional ones as in (3.3).

$$h(\mathbf{x}) = \begin{cases} +1, & f_{(0,p)} > 0 \\ -1, & \text{otherwise} \end{cases} \tag{3.8}$$

$$p^* = \arg \min_{p \in \{1,-1\}} \sum_{i=0}^{n_o} \epsilon^{(i)} \tag{3.9}$$

The object detectors are built by arranging weak classifiers as in (3.8) according to the rejection cascade architecture [3]. This architecture has been preferred for building object detectors as it is conducive for fast scanning of an image [20]. A rejection cascade, as illustrated in Fig. 3.2, consists of multiple nodes connected in series. Each node is a binary classifier that classifies an input sub-region as object or clutter. Each node consists of multiple weak classifiers which are selected iteratively using the AdaBoost procedure [17]. The weighted decision of all the weak classifiers in a node is output as the decision of the node.

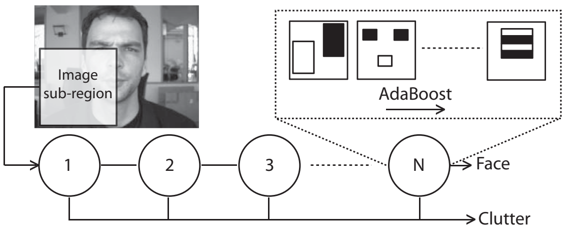


Figure 3.2: A cascaded classifier consists of multiple nodes arranged in a degenerated decision tree fashion. An input image is scanned at different scales and positions for the presence of a face. If an image sub-region is classified as a face by all the sub-regions of the face, then it is labeled a face.

#### 3.1.1.3 Pre-eliminating redundant HFs

As mentioned before, HFs are an over-complete set of features, therefore, they are redundant. Conventional object detectors avoid selecting redundant features in different nodes by training each node with bootstrapped set of clutter images [20]. In other words, features selected for different nodes are suitable for classifying different subsets of clutter images. In our case, since clutter images are not used, the over-complete set of HFs need to be pruned heuristically after each node is built so that neither the previously selected features nor similar ones are selected again. Similarity between two HFs is measured by the amount of overlap between its rectangles. For example, the HFs illustrated in Fig. 3.1(left) and Fig. 3.1(middle) do not overlap at all, therefore, they are considered to make independent measurements on a clutter image. On the contrary, the HFs illustrated in Fig. 3.1(middle) and Fig. 3.1(right) have more than 50% overlap, and therefore they are considered to make redundant measurements. To build the proposed object detector, we generated a feature pool with 7,200 HFs in which no HF in the feature pool has more than 50% overlap with the rest of the features.

### 3.1.2 Experimental setup and results

The proposed weak classifiers described above were trained for two very different object detection problems: detection of human frontal faces in photographs and detection of the human heart in short-axis cardiac Magnetic Resonance Images (MRI). For this purpose, two object databases (`face` and `heart`) were used. The `face` database was composed of 5000 images. The faces in this database exhibit an out-of-plane rotation of up to $\pm 10^\circ$ and various expressions. The `heart` database consisted of 493 short-axis MR heart images. In comparison to the images in the `face` database, the images in the `heart` database exhibit less intra-class appearance variation. The face detectors were tested on MIT+CMU frontal face database [16]. The heart detectors were tested on a set of 293 images. The speed of training of the face and heart detectors, in comparison to other methods, is tabulated in

Table 3.1.

We tested the accuracy of the object detectors by transforming the test images artificially to simulate global illumination changes. On each of the transformed database, the accuracy of the VJ-type detector and the proposed method were measured and the results are tabulated in Table 3.2. We observed that, in contrast to VJ detector, the proposed detector performed consistently to all the monotonic image transformations applied to the test images. This is because, the detector uses weak classifiers that make decision based on the sign of the feature value of a HF, and not based on the magnitude of the feature value of the HF. In theory, the accuracy of the proposed detector should not change if any monotonic transformations are applied to images. However, we see that the accuracy decreases in DS3 and DS8. This is because, two image patches (with different original intensities) might end up have the same average intensities after image transformation, and therefore, not satisfy (3.3) because of saturation of intensity values (as in the case of DS8) or because of rounding errors in the division process (as in the case of DS3). The results of the VJ-type detector with and without variance normalization are also tabulated in Table 3.2. The proposed detector does not require variance normalization procedure as the sign of the feature value of any HF is not affected by the variance normalization process. Thus, the computation of the integral image square and the computation of standard deviation of each image sub-region can be avoided during the detection process, which adds to the speed of detection. The time required to process all the images in the test set by the face and the heart detectors were 41s and 12s. This includes the time to read the image, computation of integral image(s), the scanning, and the clustering process to merge multiple detections. Our implementation of VJ procedure (with variance normalization) took 62s and 14s, respectively. The testing times were measured on a 3 GHz CPU.

The number of false detection by the the face and the heart detectors, along with the state-of-the-art methods, is listed in Table 3.3. The face detector achieved a false positive rate of $9.2 \times 10^{-5}$ (912 false detections), which is approximately 10 times worse than the state-of-the-art detectors. However, the number of false detections by the heart detector was only 2, which represented a false positive rate of $3.3 \times 10^{-6}$.

### 3.1.3 Conclusions

In this chapter, we have presented a novel training procedure for object detection systems and compared its performance, both during training and testing phases, with the state-of-the-art techniques. The advantages of adopting proposed technique include fast training in the order of seconds, global illumination invariance and real-time detection speed. The disadvantage of this method is that it produces more false positives with respect to the state-of-the-art.

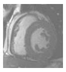
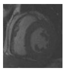
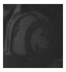
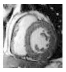
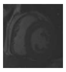
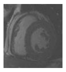
The quick training and testing speed of the proposed technique makes it ideal for content based image retrieval systems - where a user makes a query (an image patch), and asks the system to automatically find similar patches in a huge database of images. The existing methods, by the virtue of being slow to train, cannot be used in such scenarios.

Table 3.1: Comparison of training times of different object detectors

| Method | Number of features in the feature pool | Number of classifiers trained | Number of object images used | CPU speed (GHz) | Training time |
|---|---|---|---|---|---|
| Proposed (Face)[*] | 7,800 | 3,200 | 5,000 | 3.0 | 96s |
| VJ [20] | 40,000 | 4,297 | 9,500 | 0.4 | weeks |
| LZZBZS [7] | n/a | 6,000 | 2,546 | 0.7 | weeks |
| WBMR [21] | 40,000 | 3,870 | 5,000 | 2.8 | 13h20m |
| PC [14] | 295,920 | 3,502 | 5,000 | 2.8 | 5h30m |
| Proposed (Heart)[*] | 7,800 | 1,000 | 493 | 3.0 | 30s |
| VJ (Heart)[*] | 180,000 | 300 | 493 | 3.0 | 22h |

[*] Results from our implementation.

Table 3.2: True positive rate in simulated test datasets

| Method | DS1[a] | DS2[b] | DS3[c] | DS4[d] | DS5[e] | DS6[f] | DS7[g] | DS8[h] |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Proposed (Face)[*†] | 88.0 | 87.4 | 86.5 | 88.0 | 88.0 | 88.0 | 88.0 | 84.7 |
| VJ (Face)[*] | 90.3 | 90.3 | 90.3 | 87.4 | 86.2 | 87.0 | 83.9 | 80.2 |
| VJ (Face)[*†] | 90.3 | 85.1 | 72.5 | 87.4 | 78.7 | 84.0 | 81.2 | 60.5 |
| |  |  |  |  |  |  | | |
| Proposed (Heart)[*†] | 97.3 | 97.3 | 94.6 | 97.3 | 97.3 | 97.3 | 96.7 | 93.8 |
| VJ (Heart)[*] | 98.7 | 98.7 | 98.7 | 90.3 | 85.2 | 96.8 | 93.0 | 76.3 |
| VJ (Heart)[*†] | 98.7 | 81.2 | 63.2 | 90.3 | 20.3 | 69.6 | 35.2 | 0.0 |

[*] Results from our implementation. [†] Results without variance normalization.
[a] DS1: Original test images. [b] DS2: Intensity values are globally divided by 2.
[c] DS3: Intensity values are globally divided by 3. [d] DS4: Histogram equalized images.
[e] DS5: Gamma corrected image ($\gamma = 0.8$). [f] DS6: Gamma corrected image ($\gamma = 0.9$).
[g] DS7: Gamma corrected image ($\gamma = 1.1$). [h] DS8: Gamma corrected image ($\gamma = 1.2$).

Table 3.3: Comparison of accuracy of the face and heart detectors

| Method | Face | | Heart | |
|---|---|---|---|---|
| | FD [a] | TPR [b] | FD [a] | TPR [b] |
| Proposed | 912[*] | 88.0[*] | 2[*] | 97.3[*] |
| VJ [20] | 95 | 90.8 | 2[*] | 98.7[*] |
| LZZBZS [7] | 90 | 92.5 | n/a | n/a |
| WBMR [21] | 85 | 92.5 | n/a | n/a |
| PC [14] | 100 | 90.0 | n/a | n/a |
| RBK [16][✠] | 95 | 89.2 | n/a | n/a |
| SK [18][✠] | 65 | 94.5 | n/a | n/a |
| RYA [15][✠] | 78 | 94.8 | n/a | n/a |

[a] FD: Number of false detections. [b] TPR: True positive rate.
[*] Results from our implementation. [✠] Methods not based on HFs.

## 3.2 Fast training of Viola-Jones type object detectors using Laplacian clutter models

Research in the field of appearance based object detection, over the last few years, has been on finding better trade-off between accuracy and testing speed of the object detectors. Viola and Jones's (VJ) detector [20], though it was less accurate than the previously available detectors [16][18][15], gained popularity because it runs in real-time. Following their seminal paper, several approaches [12] [7] [8] [10] were proposed to improve the trade-off between speed and the accuracy of detection.

Although the previously mentioned approaches work accurately in real-time, a major problem in adopting them is that they are computationally very expensive to train. VJ report training times in the order of weeks using a 466 MHz computer. Using modern computers with 3 Ghz processor and 6 GB of memory, the same process can be completed in approximately 2 days. Nevertheless, such long training times make their detector unsuitable for Content Based Image Retrieval (CBIR) applications, where training is expected to occur in real-time. Many techniques have been proposed to reduce the training time of VJ type detectors [21][19][14][13]. The fastest of these techniques, proposed by Pavani *et al.* [13] (PDF), trains object detectors in the order of minutes. In this chapter, we propose two modifications to PDF's approach, which makes the detector to be trained even faster. Firstly, more accurate clutter (non-object) models are used, which are estimated by fitting Laplacian distributions to histograms of feature values on clutter images. Secondly, the time-consuming AdaBoost-based classifier selection process [20] is replaced with a simpler procedure.

We evaluate object detectors in terms of four factors: training speed, testing speed, accuracy and sensitivity to illumination changes. Our results show that the proposed approach is the fastest available; the training time is reduced up to 10-times compared to the PDF's approach. The training time for an object detector with 500 object class training images is approximately 2 seconds, which makes it very attractive real-time searches. A drawback of the proposed approach is that its accuracy is approximately 10 times lower than that of VJ's training approach in terms of false positive rate for complex set of objects such as human frontal faces. For easier objects such as human heart, the accuracy is comparable to that of the state-of-the-art. This makes the proposed approach ideal for use in localization of industrial objects, traffic signs *etc.*, which exhibit low intra-class variations.

### 3.2.1 Viola-Jones object detector

Viola and Jones's object detector [20] uses *weak* classifiers to classify if an image region belongs to object or to clutter class. The weak classifiers are arranged in a series of nodes according to the cascade classifier architecture [2]. Each node may contain more than one weak classifier, in which case, the output of a node is the weighted average of the decisions of all the weak classifiers in the node. Each weak classifier is based on a Haar-like feature (HF) [11] (See Fig. 2.1), whose feature value is compared to a threshold to make a decision.

Training complexity: Training an object detector involves setting appropriate thresholds to weak classifiers such that the cascade classifier has high true positive rate and simultaneously low false positive rate. This is done in a supervised fashion using three large databases: The object and clutter image databases, and a third database, called bootstrapping clutter database. To train a weak classifier, the HF involved is evaluated on the object and clutter class images, and a threshold is selected such that the weighted misclassification error on the training images is minimum. See [20] for details. The bootstrapping database consists of $\sim 10^6$ images which is used to replenish clutter image database after building a node. Bootstrapping process ensures that each node is trained to focus on a different "sub"class of clutter images.

If $M$ ($=\sim 180,000$) is the total number of available HFs, then the complexity of training a VJ-type object detector is $O(KNlogNMT)$. Here $T$ stands for the number of iterations of the AdaBoost procedure [4], which is used repeatedly to select the best possible weak classifiers and arrange them in nodes. The quantities $N$ and $K$ are the number of training images (object + clutter) and the total number of nodes in the cascade architecture, respectively. The values of $K$ and $T$ are set according to the precision needed in the detector; they generally vary between $20-40$ and $1-200$, respectively. Further, after constructing each node, bootstrapping procedure needs to be performed to replenish the clutter image database used to build the previous node. This process has the complexity $O(PQ)$, where $P$ is the total number of images in the bootstrapping clutter image database, and $Q$ is the sum of all the weak classifiers selected in the previous nodes.

### 3.2.2 Related approaches

The following approaches have been proposed to reduce the training time of VJ-type object detectors. Wu *et al.* [21] achieved a reduction in training time of approximately two orders of magnitude by decoupling the training process of the weak classifiers (by precomputing them) from the classifier selection process. Thus, they avoid training weak classifiers at each round of the feature selection process, which is time-consuming. The accuracy of their detector is even better than that of VJ's, however, the training time is still in the order of hours, which makes it unsuitable for real-time searches. Stojmenovic [19] proposed to reduce the training time by pre-eliminating HFs from the original training set. They eliminate HFs which produce error greater than a pre-determined threshold value. On a database of images containing back-view of Honda Accord cars, they could eliminate 97% of the original features, thereby achieving a potential speed increase of up to two orders of magnitude. However, it is not clear what percentage of HFs can be removed on more challenging images like those of human frontal faces. Pham *et al.* [14] proposed decreasing the training time by pre-computing the global statistics of face and non-face images. They reported a training time of 5 hours and 30 minutes while achieving high accuracy. Although, the above-mentioned approaches reduce the training time, the required time still is in the order of hours, and therefore, these methods are prohibitively expensive for use in near real-time searches.

PDF [13] proposed three modifications to the VJ's training procedure which trades-off accuracy of the detector to achieve fast training speed. Firstly, they used clutter models instead of using clutter images to train the object detectors. Thus, time is saved in not having to read and evaluate thousands of clutter images during training. Secondly, they used a non-redundant set of HFs to train the object detectors. The non-redundant set

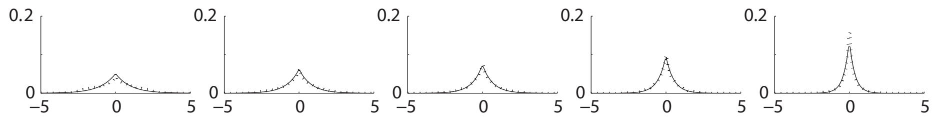


Figure 3.3: Histogram of feature values, represented by dotted lines, obtained when five randomly picked Haar-like features were evaluated on a database of clutter images. The solid lines illustrate the best fit with a Laplacian distribution.

of HFs is much smaller than the redundant set of features which VJ use. Thirdly, simpler weak classifiers, which are faster to train, are used. The complexity of their training approach is $O(NM) + O(KT)$. Note that, since clutter images are not used, $N$ is smaller. Similarly, as a smaller set of non-redundant HFs are used, the value of $M$ is smaller as well ($\sim 7,200$).

### 3.2.3 Methodology

In the following, we propose Laplacian clutter model as an alternative to the simple clutter models[1] used by PDF [13]. Following this, we describe a simplified cascade building procedure, which also contributes to accelerating the training process.

#### 3.2.3.1 Clutter model

As proposed by Huang and Mumford [6], we model feature values obtained on clutter images using a Laplacian distribution. The Laplacian distribution has two parameters. The location parameter, $\mu$, and the scale parameter, $b$. Given $n$ feature values $\boldsymbol{f} = [f_1, f_2, ..., f_n]$ obtained by evaluating a HF on a database of clutter images, the maximum likelihood estimators of $\mu$ and $b$ are given by (3.10) and (3.11), respectively. Fig. 3.3 shows histograms of feature values (dotted lines) obtained by evaluating five HFs on a database of clutter images. The solid lines represent the best fit to the histograms using a Laplacian distribution.

$$\mu = \text{median}\left(\boldsymbol{f}\right) \tag{3.10}$$

$$b = \frac{1}{n}\sum_{i=1}^{n} |f_i - \mu| \tag{3.11}$$

As pointed out in [13], the clutter distributions are symmetrical with their median at $f = 0$. As the median of feature values is known a priori for all HFs, the Laplacian distribution can specified only with the scale parameter, $b$. Further, note that the clutter

[1]PDF [13] used a simple clutter model, which assumes that the probability of a feature value of a HF, on a clutter image, to be greater/lesser than 0 is 0.5.

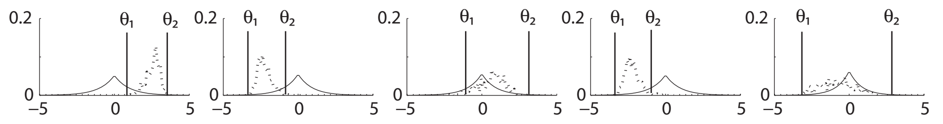


Figure 3.4: The dotted lines represent histograms of feature values of Haar-like Features when evaluated on a database of object images. The solid lines show the Laplacian clutter model estimated for that feature. The thresholds, $\theta_1$ and $\theta_2$, are set to minimum and maximum feature value recorded for object class images, respectively.

models are not specific to any object detection problem. Once computed, they can be used for various object detection problems, and therefore, their computation does not add to the training time.

#### 3.2.3.2 Training procedure

This section describes how Laplacian clutter models are used to train the object detector.

Training a weak classifier:

Weak classifiers that label an image $\mathbf{x}$ as object or clutter can be expressed as in (3.13). Training such a weak classifier involves setting appropriate values to its thresholds ($\theta_1$, $\theta_2$). The quantity $f$ refers to feature value of a HF which is obtained as in (3.12). In (3.12), $k$ refers to the number of rectangles in a HF. The quantity $\mu^{(i)}$ is the mean intensity of the pixels in image $\mathbf{x}$ enclosed by the $i^{th}$ rectangle and $w^{(i)}$ is the weight assigned to the $i^{th}$ rectangle.

$$f = \sum_{i=1}^{k} w^{(i)} \cdot \mu^{(i)} \tag{3.12}$$

$$h(\mathbf{x}) = \begin{cases} \text{Object}, \theta_2 > f > \theta_1 \\ \text{Clutter, otherwise} \end{cases} \tag{3.13}$$

Training the weak classifiers as in (3.13) can be intuitively understood from Fig. 3.4. Histograms of the feature value, $f$, from five different HFs, when evaluated on object class images are shown with dotted lines. As illustrated in Fig. 3.4, $\theta_1$ and $\theta_2$ are set minimum and maximum feature value recorded for object class images, respectively. This ensures that all the object class training images are classified correctly.

The error of each weak classifier, $\epsilon$, is intuitively the proportion of clutter images that produce a feature value between $\theta_1$ and $\theta_2$. The Laplacian clutter model can be used to estimate the error a weak classifier will make on clutter images. The error, as shown in (3.14), is the difference of cumulative sum of the distribution at $\theta_2$ and $\theta_1$.

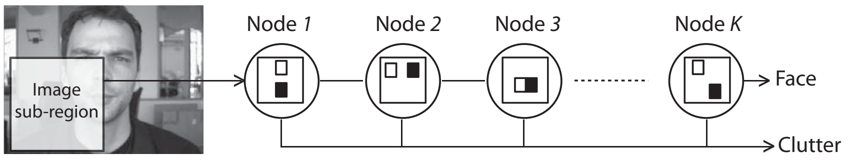


Figure 3.5: A cascaded classifier consists of multiple nodes arranged in a degenerated decision tree fashion. An input image is scanned at different scales and positions for the presence of a face. If an image sub-region is classified as a face by all the nodes of the cascade, then it is labeled a face.

$$\epsilon = 0.5 \left\{ \operatorname{sign}(\theta_2) \left( 1 - \mathrm{e}^{\frac{-|\theta_2|}{b}} \right) - \operatorname{sign}(\theta_1) \left( 1 - \mathrm{e}^{\frac{-|\theta_1|}{b}} \right) \right\} \tag{3.14}$$

As $\epsilon$ can be estimated without having to evaluate clutter images on thousands of HFs, savings in training speed is achieved.

Training the cascade:

Fig. 3.5 illustrates the cascade classifier architecture. The cascade consists of a series of nodes, each of which is a binary classifier that labels an image sub-region as belonging either to object class or to clutter class.

Generally, each node is built by selecting complementary weak classifiers using machine learning algorithms such as AdaBoost [20][13][7][8]. This procedure makes sure that that each node has a very high true positive rate (TPR). In the current set up, as each weak classifier is trained to have a high TPR, the time-consuming AdaBoost procedure can be avoided.

A cascaded classifier, as shown in Fig. 3.5, is trained in two steps.

1. All the available weak classifiers are trained as described in Section 3.2.3.2. This procedure has a complexity of $O(NM)$.

2. The weak classifiers are sorted according to their estimated error, as computed using (3.14). The weak classifier with the least error is assigned to the first node, the one with the second least error to the second node, and so on. This process can be completed in $O(MlogM)$ time.

The complexity of the procedure is $O(NM) + O(MlogM)$. This turns out to be significantly lower than $O(NM) + O(KT)$, which is required to train PDF's detector [12]. This is because, the proposed approach does not depend on the parameter $T$, which accounts for the time-consuming AdaBoost procedure.

### 3.2.4 Results

In this section, object detectors built according to the proposed training approach, are compared with the state-of-the-art approaches in terms of training speed, testing speed, accuracy and sensitivity to ambient illumination. Two object classes are considered for this purpose: human frontal faces and human heart images from magnetic resonance images. Human frontal faces are an ideal object class for study as many researchers have already published their results on this class, and thus, the object detectors can be compared in a fair manner. We chose human heart image regions as an object class mainly because for two reasons. Firstly, PDF's [13] approach, on which the proposed training approach is based on, publishes results on human hearts as well, and therefore, provides a reference to compare the two algorithms. Secondly, the human heart class presents lesser degree of intra-class appearance variations than the human frontal faces (which can vary due to race, sex, genetics and ambient lighting among other factors), and therefore, one can assess how much intra-class variation a training strategy can handle.

**Training speed:** Table 3.4 lists the parameters used for training face and heart detectors. The time required for training the detectors is shown in the sixth column. Note that, as different computers with different CPU speeds, were used to train the detectors, the training times are not directly comparable. In column seven of Table 3.4, we attempt to "normalize" the training time by dividing the total required time by the training parameters. The proposed training approach was the fastest; the face detector took 30 seconds while the heart detector took 2 seconds. As previously mentioned, Laplacian clutter models are common for all object classes, and therefore they can be pre-computed. The training time reported for the proposed approach does not include the time to compute Laplacian clutter models, which normally takes 90 seconds for $7,800$ HFs using a database of clutter images with $10,000$ images.

**Accuracy:** The accuracy of the object detectors built using the proposed approach is compared with other VJ type [13][20][7][21][14] and non-VJ type [16][18][15] object detectors in Table 3.5. The accuracy of the object detectors are measured in terms of TPR and Number of false detections (FD). Face detectors were tested on the standard MIT+CMU face database [16]. The heart detectors were published on a set of 293 MR images of the heart. The accuracy of the object detectors built with the proposed approach can be compared with the state-of-the-art results in Table 3.5. The face detector produced 979 false detections, which is approximately 10 times worse than the VJ face detector. However, the number of false detections by the heart detector was only 2, which is similar to that of VJ heart detector.

**Sensitivity to illumination:** As the perceived appearance of any object depends on the illumination conditions, all appearance-based object detectors, like the proposed one, are sensitive to ambient lighting conditions. To test the sensitivity of the object detector to illumination conditions, we generated 7 synthetic test databases (DS2, DS3 ... DS8) by applying image transformations to the original test database (DS1). The image transformations, detailed in the footnote of Table 3.6, simulate changes in illumination. The test images obtained after the transformations are shown in Table 3.6.

As pointed out in [13], PDF's detector is mostly invariant to image transformations. This is because, their weak classifiers are based on the sign of the feature values, and not

Table 3.4: Comparison of training time of different object detectors

| Method | Number of features in the feature pool (A) | Number of classifiers trained (B) | Number of object images used (C) | CPU speed in GHz (D) | Training time✠ | Normalized training time in seconds† |
|---|---|---|---|---|---|---|
| Proposed (Face)* | 7,800 | 3,200 | 5,000 | 3.0 | 30s | $8.0 \times 10^{-11}$ |
| PDF [13](Face)* | 7,800 | 3,200 | 5,000 | 3.0 | 96s | $2.5 \times 10^{-10}$ |
| VJ [20] | 40,000 | 4,297 | 9,500 | 0.4 | weeks | n/a |
| LZZBZS [7] | n/a | 6,000 | 2,546 | 0.7 | weeks | n/a |
| WBMR [21] | 40,000 | 3,870 | 5,000 | 2.8 | 13h20m | $2.2 \times 10^{-6}$ |
| PC [14] | 295,920 | 3,502 | 5,000 | 2.8 | 5h30m | $1.3 \times 10^{-9}$ |
| Proposed (Heart)* | 7,800 | 1,000 | 493 | 3.0 | 2s | $1.7 \times 10^{-10}$ |
| PDF [13](Heart)* | 7,800 | 1,000 | 493 | 3.0 | 30s | $2.6 \times 10^{-9}$ |
| VJ [20] (Heart)* | 180,000 | 300 | 493 | 3.0 | 22h | $9.9 \times 10^{-7}$ |

* Results from our implementation. ✠ h: hours; m: minutes; s: seconds
† = Training time / (A*B*C*D)

Table 3.5: Comparison of accuracy of the face and heart detectors

| Method | Face | | Heart | |
|---|---|---|---|---|
| | FD [a] | TPR [b] | FD [a] | TPR [b] |
| Proposed | 979* | 86.5* | 2* | 95.4* |
| PDF [13] | 912* | 88.0* | 2* | 97.3* |
| VJ [20] | 95 | 90.8 | 2* | 98.7* |
| LZZBZS [7] | 90 | 92.5 | n/a | n/a |
| WBMR [21] | 85 | 92.5 | n/a | n/a |
| PC [14] | 100 | 90.0 | n/a | n/a |
| RBK [16]✠ | 95 | 89.2 | n/a | n/a |
| SK [18]✠ | 65 | 94.5 | n/a | n/a |
| RYA [15]✠ | 78 | 94.8 | n/a | n/a |

[a] FD: Number of false detections. [b] TPR: True positive rate.
* Results from our implementation. ✠ Methods not based on HFs.

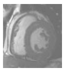
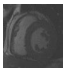
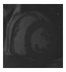
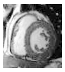
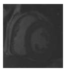
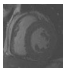
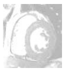
on their magnitude. Monotonic images transformations do not change the sign of the feature value, however, its magnitude is affected, and therefore, the accuracy of the VJ's and the proposed detector varies if the test images are obtained through strong image transformations. Nevertheless, note that the detectors built using VJ's and the proposed approaches were invariant to image transformations DS2 and DS3. This is because of the variance normalization procedure [20] performed during testing. Variance normalization nullifies linear transformations used to obtain DS2 and DS3. However, variance normalization is not robust against non-linear transformations used to obtain DS4, DS5, DS6, DS7 and DS8. Variance normalization, nevertheless, is important to maintain the generalization power of object detector built using VJ's approach [13].

**Testing speed:** The time required to process all the images in the test set by the face and the heart detectors using the proposed approach was 68 and 15 seconds. VJ takes 62 and 14 seconds, meanwhile, PDF's object detector takes 41 and 12 seconds. This includes the time to read the image, computation of integral image(s), the scanning the image at multiple scales on a 3 GHz CPU. The speed of the proposed approach is comparable to the VJ approach, while PDF's detectors were the fastest as their approach does not need variance normalization.

### 3.2.5 Conclusions and discussions

In this chapter, one of the main difficulties in adopting Viola and Jones type object detectors is addressed: their training time. One may need to wait for hours, if not days, to train a Viola and Jones type object detector, which makes the approach difficult for use in real-time searches for applications such as content-based image retrieval (CBIR). Given that a user's satisfaction of a web-based application reduces drastically after approximately 9

Table 3.6: True positive rate measurements in simulated test datasets

| Method | DS1[a] | DS2[b] | DS3[c] | DS4[d] | DS5[e] | DS6[f] | DS7[g] | DS8[h] |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Proposed (Face)* | 86.5 | 86.5 | 86.5 | 84.3 | 82.7 | 82.4 | 79.2 | 76.7 |
| PDF [13] (Face)* | 88.0 | 87.4 | 86.5 | 88.0 | 88.0 | 88.0 | 88.0 | 84.7 |
| VJ [20] (Face)* | 90.3 | 90.3 | 90.3 | 87.4 | 86.2 | 87.0 | 83.9 | 80.2 |
| |  |  |  |  |  |  |  | |
| Proposed (Heart)* | 95.4 | 95.4 | 95.4 | 97.3 | 83.2 | 93.3 | 89.7 | 61.9 |
| PDF [13] (Heart)* | 97.3 | 97.3 | 94.6 | 97.3 | 97.3 | 97.3 | 96.7 | 93.8 |
| VJ [20] (Heart)* | 98.7 | 98.7 | 98.7 | 90.3 | 85.2 | 96.8 | 93.0 | 76.3 |

* Results from our implementation.
[a] DS1: Original test images. [b] DS2: Intensity values are globally divided by 2.
[c] DS3: Intensity values are globally divided by 3. [d] DS4: Histogram equalized images.
[e] DS5: Gamma corrected image ($\gamma = 0.8$). [f] DS6: Gamma corrected image ($\gamma = 0.9$).
[g] DS7: Gamma corrected image ($\gamma = 1.1$). [h] DS8: Gamma corrected image ($\gamma = 1.2$).

seconds [5], it is important to reduce the training time to the order of seconds, to be of use in real-time image searches.

#### 3.2.5.1 How is fast training achieved?

The reduction in training time, with respect to Viola and Jones procedure, is primarily because of three reasons. Firstly, Laplacian clutter models are used instead of clutter images. Thus, lots of time is saved in reading, and evaluating thousands of clutter images on Haar-like features (HFs). The Laplacian clutter models, do not depend on the object class images, and therefore, can be pre-computed. Secondly, like Pavani *et al.* [13], we prune the highly redundant (or over-complete) set of HFs [20] to obtain a smaller set of HFs which are less redundant. This also accounts for savings in training time, as lesser number of features need to be evaluated during training. Thirdly, the time-consuming AdaBoost-based classifier selection procedure is omitted. A simplified cascade building procedure is used, where only one classifier is assigned to a node, thus eliminating the need to use AdaBoost.

#### 3.2.5.2 Pros and cons:

The proposed training strategy was used to learn two object classes, human frontal faces and human heart regions in magnetic resonance images. Apart from training time, the results were compared to the state-of-the-art detectors in terms of three other factors:

accuracy, sensitivity to illumination, and testing speed. The accuracy of the face detectors were approximately 10 time worse than the Viola and Jones's face detector. However, the accuracy of the heart detector was comparable to that of Viola and Jones's detector. The face detector built using the proposed approach was unable to learn all the intra-class variations present in the training set. The training set for human hearts contained lesser intra-class variations, which was learnt accurately by the proposed training approach. The testing speed and the tolerance to illumination changes of the proposed detector is similar to that of Viola and Jones's detector. The main advantage in using the proposed architecture is its training speed. The fastest training time reported has been approximately 30 seconds with a training database of 500 images [13]. The proposed architecture reduces the training time to approximately 2 seconds for a training database of 500 images, which is a 93% decrease in training time with respect to the fastest available method.

#### 3.2.5.3 Which objects can be learnt?

The proposed training approach was tested on objects such as stop-sign, bicycle wheel *etc.* as shown in Fig. 3.6. Note that the detector works poorly on bicycle wheels. This is because the HFs encode the background rather than the object itself. The wheel on the first column of Fig. 3.6 was detected because it was used as the training image. This problem was not observed on other opaque objects shown in Fig. 3.6. The stop-sign, IEEE-logo, chess board and do-not-enter-sign detectors worked without many false positives, which showed that the proposed training scheme is ideal for detecting objects with small intra-class variations.

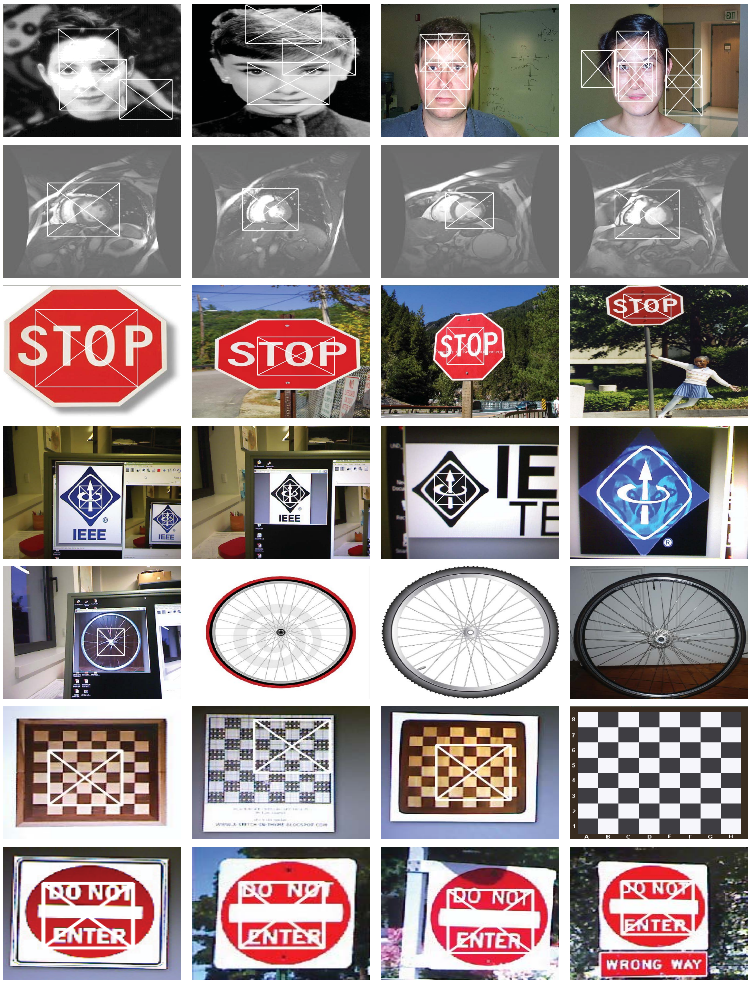


Figure 3.6: The outputs of face, heart, stop-sign, IEEE-logo, bicycle wheel, chess board and do-not-enter-sign detectors are shown in each row. The face detector produced lots of false negatives, which says that the proposed method is not good detecting human faces. However, for objects with lesser intra-class variations such as stop-sign, IEEE-logo, human hearts, do-not-enter-sign and chess board the number of false detections reduced drastically. The detector performs poorly on transparent objects or objects with holes such as bicycle wheels. In this case, the background is learnt instead of the object itself.

## 3.3 References

[1] OpenCV library, http://sourceforge.net/projects/opencvlibrary/, 2009.

[2] S Baker. *Design and evaluation of feature detectors*. PhD thesis, Columbia University, 1998.

[3] S Baker and S K Nayar. Pattern rejection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 544–549, San Francisco, CA, USA, 1996.

[4] Y Freund and R E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the European Conference on Computational Learning Theory*, pages 23–37, Jerusalem, Israel, 1995.

[5] J A Hoxmeier and C Dicesare. System response time and user satisfaction: An experimental study of browser-based applications. In *Proceedings of the Proceesings of the Sixth Americas Conference on Information Systems*, pages 10–13, Long Beach, CA, 2000.

[6] J Huang and D Mumford. Statistics of natural images and models. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 1, page 547, Ft. Collins, CO, USA, 1999.

[7] S Z Li, L Zhu, Z Zhang, A Blake, H Zhang, and H Shum. Statistical learning of multi-view face detection. In *Proceedings of the European Conference on Computer Vision, Lecture Notes in Computer Science 2353*, pages 67–81, London, UK, 2002.

[8] R Lienhart and J Maydt. An extended set of Haar-like features for rapid object detection. In *Proceedings of the International Conference on Image Processing*, pages 900–903, Rochester, New York, USA., 2002.

[9] B McCane and K Novins. On training cascade face detectors. In *Proceedings of Image and Vision Computing New Zealand*, pages 239–244, Auckland, New Zealand, 2003.

[10] T Mita, T Kaneko, and O Hori. Joint Haar-like features for face detection. In *Proceedings of the International Conference on Computer Vision*, pages 1619–1626, Washington, DC, USA, 2005.

[11] C P Papageorgiou, M Oren, and T Poggio. A general framework for object detection. In *ICCV '98: Proceedings of the International Conference on Computer Vision*, pages 555–562, Washington, DC, USA, 1998.

[12] S-K Pavani, D Delgado, and A F Frangi. Gaussian weak classifiers based on Haar-like features with four rectangles for real-time face detection. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns, Lecture Notes in Computer Science 5702*, pages 91–98, Münster, Germany, 2009.

[13] S-K Pavani, D Delgado, and A F Frangi. A rapidly trainable and global illumination invariant object detection system. In *Proceedings of the Fourteenth Iberoamerican Congress on Pattern Recognition*, pages 877–884, Guadalajara, Mexico, 2009.

[14] M-T Pham and T-J Cham. Fast training and selection of Haar features using statistics in boosting-based face detection. In *Proceedings of International Conference on Computer Vision*, pages 1–7, Rio de Janeiro, Brazil, 2007.

[15] D Roth, M Yang, and N Ahuja. A SNoW-based face detector. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, pages 855–861, Denver, CO, USA, 2000.

[16] H A Rowley, S Baluja, and T Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

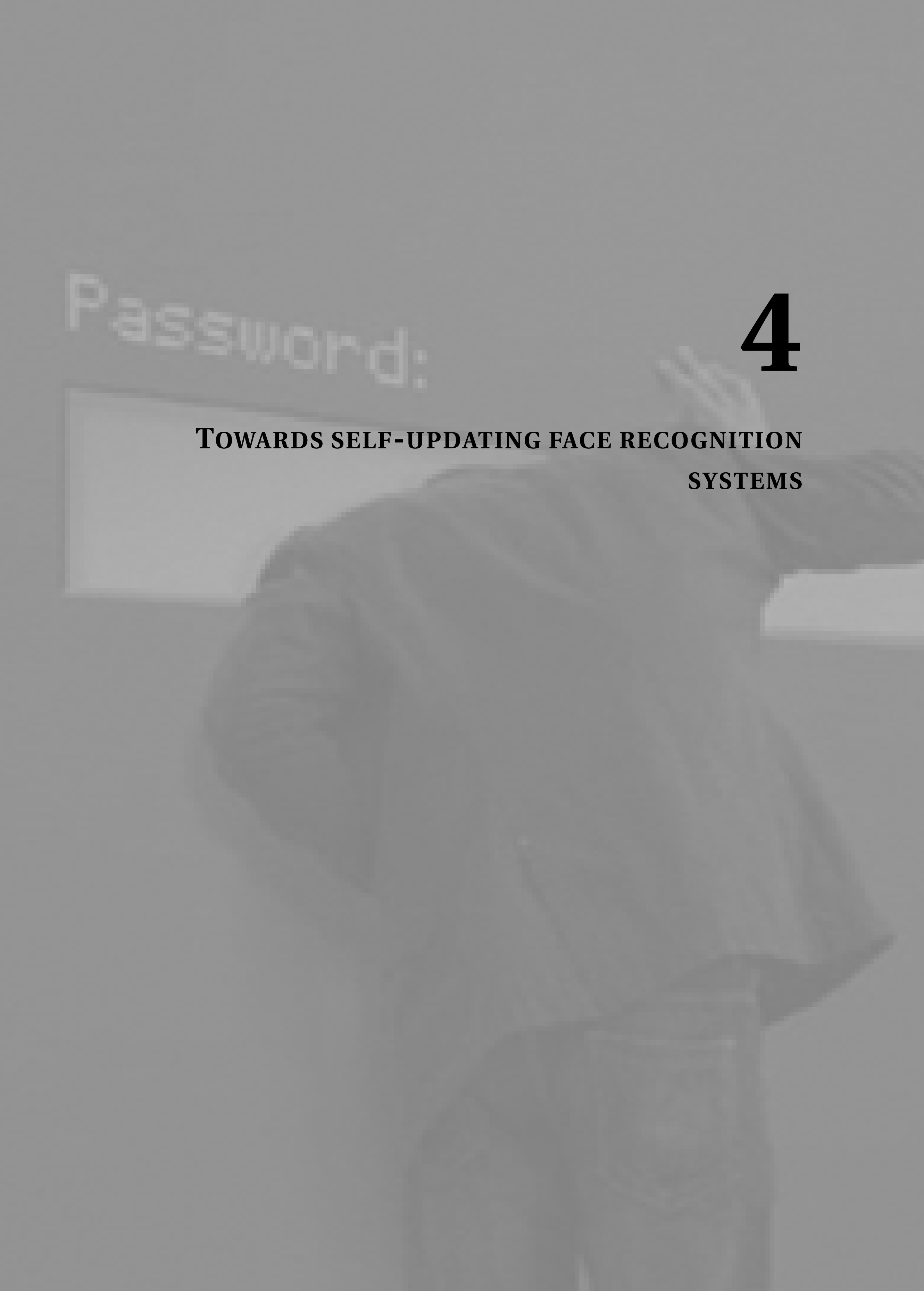
[17] R E Schapire. A brief introduction to boosting. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1401–1406, San Francisco, CA, USA, 1999.

[18] H Schneiderman. *A statistical approach to 3D object detection applied to faces and cars*. PhD thesis, Carnegie Mellon University, 2000.

[19] M Stojmenovic. Pre-eliminating features for fast training in real time object detection in images with a novel variant of AdaBoost. In *Proceedings of the International Conference on Computational Intelligence and Security*, pages 1–6, Guangzhou, China, 2006.

[20] P Viola and M J Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[21] J Wu, S C Brubaker, M D Mullin, and J M Rehg. Fast asymmetric learning for cascade face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):369–382, 2008.



# 4

# TOWARDS SELF-UPDATING FACE RECOGNITION SYSTEMS

Unhappily, I can scarcely choose what not to learn unless I learn enough about it to be sure I do not need to learn it all

*Samuel Sandmel*

The changes in facial appearance, which are certain to occur, can be gradual or abrupt due to factors such as ageing, hair-growth, surgery, weight changes, sun-exposure, ancestry, sex, health, disease, drug use, diet and sleep deprivation. When changes occur, the accuracy of the face recognition systems decrease as a consequence.

To maintain the accuracy of the face recognition systems, the images belonging to the training set need to be updated periodically. A simple way to update the training set is through manual supervision. Manual supervision is time consuming and costly, which underlines the need to have automatic methods of learning facial appearance changes. A popular method used in the literature is called the Self-update procedure, where the system learns the biometric characteristics of the user every time he/she interacts with it. Typically, the most confident test images are added to the training set according to the predicted label. Then, the system is re-trained with the newly added data.

A commonly acknowledged problem with the self-update methods is the corruption of biometric traits due to misclassification. In this chapter, we propose the use of four confidence measures that can reduce the number of misclassifications. Further, we investigate the optimal design of the Self-updating face recognition systems (SUFRS), by performing an extensive study using two challenging databases. our results show novel insights into the relationship between the complexity of the face recognition algorithm, its learning capacity of the FRSs and their tolerance to the corruption of training data.

Adapted from:

S-K. Pavani, F.M. Sukno, C. Butakoff, X. Planes and A.F. Frangi, *A Confidence-Based Update Rule for Self-updating Human Face Recognition Systems.* In Proc. International Conference on biometrics, Alghero, Italy. Lecture Notes in Computer Science vol. 5558, pages 91-98, 2009.

S-K. Pavani, F.M. Sukno, D. Delgado-Gomez, C. Butakoff, X. Planes and A.F. Frangi, *Self-Updating Frontal Face Recognition Systems: Design and Experimental Evaluation.* Submitted, 2010.

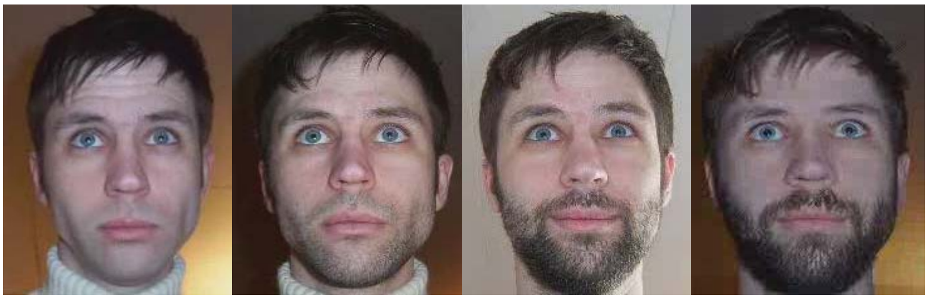


Figure 4.1: An example of facial appearance changes occurring in a person because of beard growth.

## 4.1 Design and Experimental Evaluation of Self-Updating Frontal Face Recognition Systems

The accuracy of Face Recognition Systems (FRSs) on a user has been shown to decrease with time [32] [12] [53] [21]. The decrease in the accuracy is the result of changes in facial appearance because of factors such as ageing, hair-growth, surgery, weight changes, sun-exposure, ancestry, sex, health, disease, drug use, diet, sleep deprivation, and bio-mechanical factors [35] [25] [48]. Fig. 4.1 illustrates the appearance changes in a person with an example.

Evidently, to maintain the accuracy of the FRS over time, the FRS needs to be re-trained periodically with new images of users. One way to maintain a FRS up-to-date is through the self-update procedure [56][36]. In this procedure, the system is re-trained with the unlabeled data, which is acquired every time a user interacts with the system. Typically, the most confident unlabeled facial images are added to the training set according to the predicted label. In this way, the FRS can utilize the unlabeled data, which are available free-of-cost [6], to train itself. The main advantage is that the self-update procedure avoids the costs of supervised enrollment every time there is a performance degradation on a user due to appearance changes.

However, as noted in [7][36][22], a major risk in automatically incorporating new training data is the potential for a gradual corruption of the training images. Since images are added in a non-supervised manner, images on which one of the system components (*e.g.,* face detection, segmentation or classification) failed could also be added. This leads to the corruption of the biometric data belonging to the users. For example, the image shown in Fig. 4.2a, should not be added to the training set because the face was not accurately separated from the background and the facial features were not identified correctly. In this particular example, the information learnt from this image will be incorrect, which in turn, might corrupt the biometric data of the user even if the face was classified correctly.

In this chapter, we study how the choice of the face recognition algorithm used affects the long-term accuracy of the self-updating FRSs. Such a study is important because the classification strategy used influences the degree of corruption that occurs and also, how the classifier is affected by the corrupted training data. To this end, we built FRSs with three different face recognition algorithms: Eigenfaces [49] (`EF`), Fisherfaces [4] (`FF`), and Similarity-Based Fisherfaces [10] (`SFF`). These three algorithms were chosen because of the differing levels of inductive reasoning[1] in their training process. Our implementation of `EF` uses purely transductive reasoning[2] [51] (*i.e.,* it does not use inductive reasoning). `FF` was mildly inductive, as the global statistics of the user data was used to find the best discriminating sub-space. `SFF`, in comparison to `FF`, is more inductive as user-specific models are built, which are later used for recognition.

The experiments reported in this study were performed on two large image databases where users were monitored for a long periods of time on a day-to-day basis. The first database, `GEFA` [30], was collected automatically using a FRS. This database has 14,279 images of 129 individuals acquired during five months. Secondly, we used images complied from YouTube [2] videos, which document facial appearance changes over multiple years. Videos, in which people were artificially aged, were also used. In total, we extracted 31,951 images from 25 videos documenting extreme facial appearance changes. Both these databases are available to the scientific community free-of-cost (See Appendix). The images in both the databases contain mainly frontal face images, and few with small in-plane and out-of-plane rotations.

The rest of the chapter is organized into the following sections. In Section 4.1.1, we overview some of the current work performed to tackle the problem of gradually changing appearance of faces. In Section 4.1.2, we overview basic building blocks of the face recognition system that is used in our experiments. Section 4.1.3 describes four confidence measures that can be used to determine the reliability of extracting information automatically from an unlabeled image. Sections 4.1.4 and 4.1.5 present the experimental setup and the results, respectively. This chapter is concluded in Section 4.1.6.

### 4.1.1 Comparison to previous work

Over the years, two main approaches were developed to handle the ever-changing appearance of human faces. In [42], [47], [43], and [19], the FRS is made invariant to ageing by simulating the effect of ageing on the appearance of a face. Such approaches are generally failure-prone, as the future appearance of a face is not only dependent on ageing, but also on other unpredictable factors such as skin tanning, health and beard growth. These approaches are primarily intended for applications where recent images of a person are not available. For example, they are used in predicting the current appearance of a person who has been missing for a long time.

There are applications where simulation of ageing becomes unnecessary. For example, a FRS installed at home interacts with the users almost on a daily basis. In such scenarios,

[1]Induction uses training data to find approximations of functions that describe the data [51]. Inference on the test data are obtained by evaluating the test data on the derived functions, a process commonly referred to as deduction.

[2]Transductive inference is based on direct generalization from the training set to the test set, thus avoiding the intermediate problem of estimating a function.

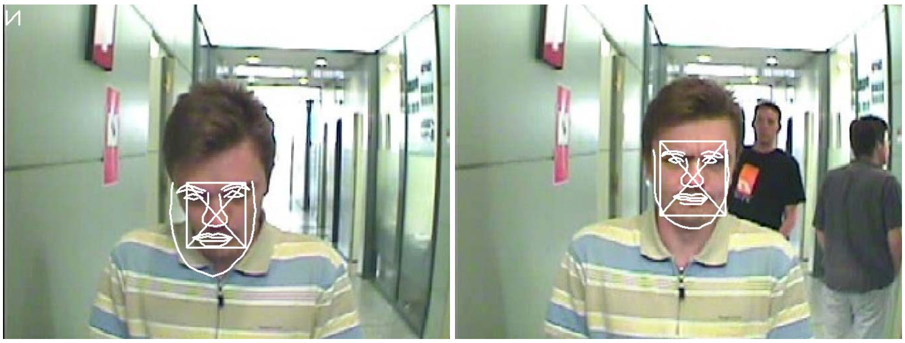


Figure 4.2: Examples of bad (left) and good (right) facial segmentation. Including inaccurately segmented images corrupts the training set.

the appearance of the users can be learnt every time the system sees a user, and therefore, the system can remain up-to-date.

Aryananda [3] and Mou *et al.* [26] presented FRSs that automatically collect training images as users interact with it. Their experiments were preformed on relatively small databases, which explains why they report the learning occurred without any corruption of data. Further, all the images belonging to a user were acquired in a short span of time and therefore, none of the users exhibit changes in appearance with time.

In [37], Elastic Bunch Graph Matching (EBGM) [54] along with graph mincut-based approach [5] is used to automatically collect training data from a database of unlabeled facial images. The tests were made on Equinox database which had 56 users with 129 facial images each. Although, an independent impostor dataset was not used to update the system, the authors acknowledge the presence of misclassified users. The question of whether the biometric system will breakdown if the EBGM verification algorithm is re-trained with newly acquired facial images still remains to be explored.

Recently, Franco *et al.* [13] presented a study of self-updating face recognition systems on large datasets. The main drawback of their testing methodology was the absence of impostors who could potentially corrupt the training data. They conclude that the percentage of corrupted data after the update, was negligible, and that a noticeable degradation of accuracy was not observed.

Our work differs from the previous works in the following ways. Unlike the approaches presented in [17] [37] [38] [22] where verification problem is considered, we consider the more generalized problem of identification, where one-to-many matching is performed. As proposed in [30], we use confidence values generated from other components of the FRS such as face detection and segmentation blocks, which provide complementary information to the classifier confidence. In all of the previous approaches, only the classifier confidence is used for the update process. In contrast to update strategy used by Franco *et al.* [13], we perform worst-case analysis of the FRSs, wherein the system is updated with

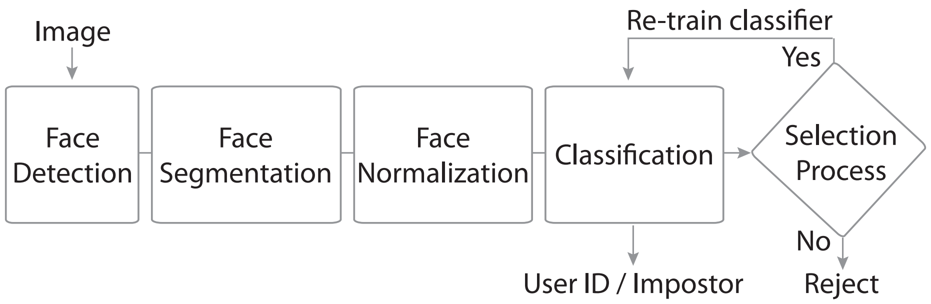


Figure 4.3: Flow chart of the FRS. The input image is checked for the presence of face(s). If any, the face(s) are segmented and identified. The selection process determines whether the information extracted from each face is reliable enough so that it can be used to re-train the classifier.

an independent impostor data first, thereby, increasing the probability of corrupting the training set [40]. Further, this is the first study on the self-updating capacity of the FRS, based on different recognition algorithms that use varying levels of inductive reasoning, performed on large and meaningful databases.

### 4.1.2 Components of the face recognition system

The system used in this chapter, whose flowchart is shown in Fig. 4.3, has five main building blocks: face detection, segmentation, normalization, classification and selection block for automatic learning. In the following, the algorithms used in each block are described.

#### 4.1.2.1 Face detection

The face detection block determines whether there are face(s) in an image or not. If there are face(s), the detector outputs their position and size. The main requirements of the face detection block are that it works at real-time speed while being highly accurate. The real-time speed is necessary in order to reduce the waiting time of the user to be identified. High accuracy is needed to reduce the inconvenience caused to a user if his/her face is not located by the system. Research in face detection is vast [16], there are many detectors that provide high degree of accuracy and real-time speed. In our system, we used the detector proposed in [29], which uses weak classifiers based on Haar-like features [28] with optimally weighted rectangles. This detector, when tested on the MIT+CMU dataset [39], produced a false acceptance rate of $5 \times 10^{-6}$ while correctly classifying 93% of the faces in the database at real-time speeds.

#### 4.1.2.2 Face segmentation

Separation of a face from the background and accurate localization of facial features is important for four reasons. Firstly, it is possible to extract facial features based on either shape (directly using the resulting parametrization in model space) or image intensities, taking into account region correspondences. It is worth pointing out that one of the better performing algorithms in FRVT 2006 [33], FaceIt® from L-1 Identity Solutions [1], uses both shape and texture for classification. Secondly, as will be detailed in the following section, segmenting a face enables to determine the pose and the expression of a face. Using the result of segmentation, the face can be warped to an expression/pose-normalized representation which facilitates identification by reducing intra-class variations. Thirdly, segmentation of a face removes the background regions, which have been shown to have an adverse effect on the identification process [4]. Finally, it has been shown [24] that if the eye position is accurately determined, the total error rate scores of all users decrease substantially. The segmentation process aims to precisely delineate facial components, and therefore, including it in the framework will potentially lead to more accurate classification.

For the segmentation of the prominent facial features, our system employs Active Shape Models with Invariant Optimal Features (IOF-ASM) [46]. This algorithm combines local image search with global shape constraints based on a Point Distribution Model (PDM) [9]. IOF-ASM has demonstrated a significant improvement in segmentation accuracy as compared to the linear ASM [9] and Optimal Features ASM [50] (a nonlinear extension of the linear ASM) in the tests performed on AR [23], XM2VTS [24] and EQUINOX [41] databases. The segmentation process is quite fast; running it on a $100 \times 100$ face approximately takes one third of a second on a 3 GHz processor with an average error of approximately 2% of the inter-eye distance.

#### 4.1.2.3 Face normalization

It has been pointed out by Poh *et al.* [34] that intra-class variations of a user's face can be as diverse as inter-class variations between different users. The aim of the face normalization block is to facilitate identification by reducing the intra-class appearance variations due to pose, expression and illumination.

To reduce the effect of changes in pose and expression in the appearance of a face, the texture obtained from the test image is represented in a shape independent form. In other words, all facial images are warped onto some common shape. After the face is segmented, its texture is warped by a piece-wise affine transform (as in Active Appearance Models [8, 44]) onto the mean shape $\bar{\boldsymbol{x}}$ of the Point Distribution Model (PDM). Specifically, given source shape (segmentation result) and target shape (mean shape of the PDM), a Delaunay triangulation is applied to the landmarks to obtain a triangular mesh for both of them. Afterwards, knowing the correspondence between triangles in the source and target shapes, each pixel in every triangle of the source shape is mapped into the corresponding triangle of the target shape using barycentric coordinates. See Fig. 4.4 for an illustration. Further, histogram equalization on the extracted texture to normalize the effect of ambient lighting on the appearance of a face.

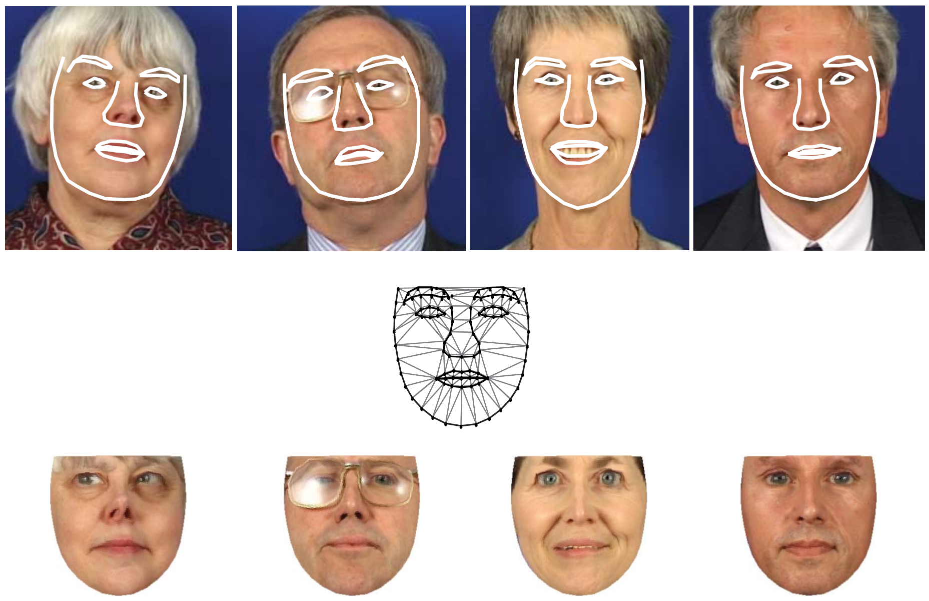


Figure 4.4: Illustration of texture warping. Images with known shapes on the top row are warped onto the mean shape of the PDM in the middle. The resulting images are on the bottom row.

#### 4.1.2.4 Classification

To remove redundancy and to obtain compact texture and shape representation, the shape and the warped texture data are projected to the PCA subspace, spanned by an independent dataset of frontal face images which does not include user information, and its coordinates in this subspace are the parameters used for our experiments. Formally:

$$\boldsymbol{b}_g = \Phi_g^T(\boldsymbol{g} - \bar{\boldsymbol{g}}) \tag{4.1}$$

where $\boldsymbol{b}_g$ are the parametric representation of a face $\boldsymbol{g}$, $\bar{\boldsymbol{g}}$ is the average facial appearance or shape across the training set and $\Phi_g$ are the axes of the PCA subspace. The PCA parameters are whitened [27] to normalize their magnitude.

In this study, we experimented with three recognition algorithms: Eigenface (EF) [49], Fisherface (FF) [4], and Similarity-based Fisherfaces (SFF) [10] which uses personalized face models for each user.

As illustrated in Fig. 4.5, the shape and the texture parameters extracted from a test image are classified independently and the decisions of the classifiers are fused. The fusion of decisions is covered in Sec. 4.1.3.5. In the following, the three classification strategies (EF, FF and SFF) are described.

1. In our implementation of the EF strategy, the shape and texture data are projected to a low dimensional subspace, generated by Principal Component Analysis (PCA) [15]. The PCA subspace was computed from an independent face dataset that does not contain user data, and therefore, the axes of the subspace remain the same even if the training data change during the update process. During testing, the shape and texture PCA parameters are input to an *Open set Transductive Confidence Machine $k$-Nearest Neighbor classifier* (OSTCM-$k$NN) [20]. The OSTCM-$k$NN classifier is suitable for open set multiclass classification problems. It provides confidence measures that intuitively provide a rejection option, and thereby, it permits impostor identification. Note that this implementation of EF strategy does not use any inductive reasoning during its training.

2. The FF strategy uses Fisher's Linear Discriminant (FLD) analysis [15] to determine a low dimensional space, which maximizes the ratio of between-class scatter and within-class scatter of the PCA parameters, computed from the training data. The transformed texture and shape parameters are input to a OSTCM-$k$NN classifier. Note that, to make a decision on a test face, the FF strategy uses a low dimensional subspace that depends on the current training set. Thus, inductive inference is used during its training.

3. In the SFF strategy, the PCA vectors obtained from shape and texture data are transformed to Similarity-Based face Representation (SBR) proposed by Delgado-Gomez *et al.* [10]. Similarity-based face representation is obtained by linearly projecting a user's data (shape or texture) to several linear sub-spaces. Each of these sub-spaces is associated to a different user and the value of projection (or score) on a test image on a sub-space measures the similarity of a test person to the user. The shape and texture scores are input to a OSTCM-$k$NN classifier. The SFF strategy uses stronger induction rules than FF as personalized models are created using the training data.

#### 4.1.2.5 Selection block

Every time a face has been identified, it goes through an automatic selection process that determines whether it is suitable to be added to the training set of the appropriate user. The selection process deems two categories of images to be unsuitable for automatic learning: 1) images with outdated facial appearance, 2) images, where the algorithms, used in our system, produce unreliable results. The selection process is detailed in the next section.

### 4.1.3 Selection process for automatic updates

The selection process is based on the following four confidence measures:

#### 4.1.3.1 Temporal confidence ($\mathscr{C}_t$)

The accuracy of FRSs has been shown to decrease linearly with the time elapsed between enrollment and testing [32]. In our experiments, we assume that an image is valid to

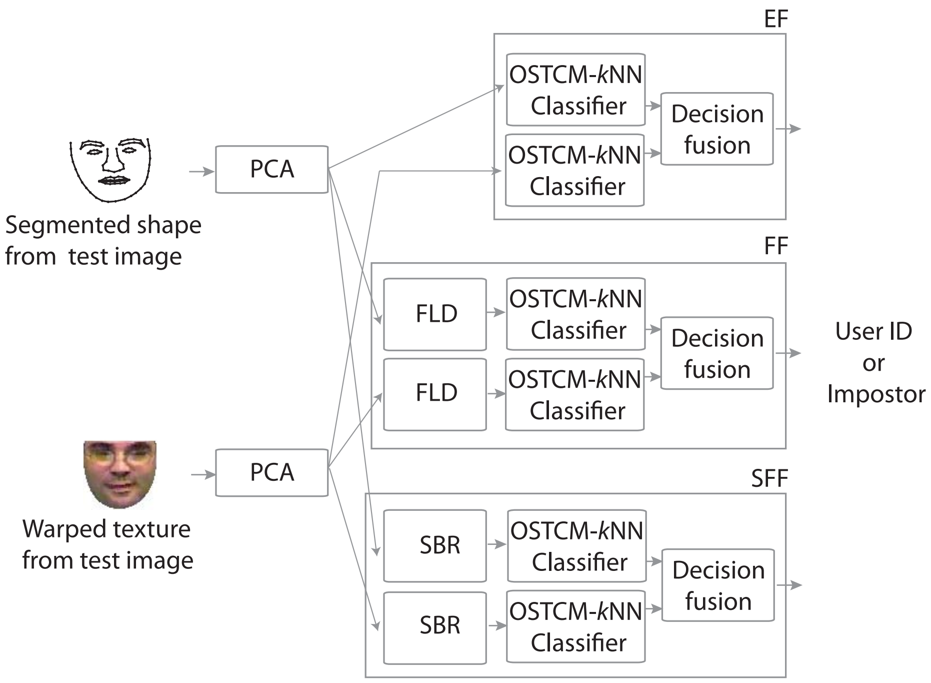


Figure 4.5: Flow chart of the steps involved in the classification process. For all three face recognition algorithms tested, the shape and texture data extracted from a test image are classified separately and the decision is fused to obtain a final decision.

represent a user if it is no more than 1,825 days (~ 5 years) old, after which it is deemed unsuitable to represent the user's appearance. The value of $\mathscr{C}_t$ is computed as in (4.2). The quantity $t_e$ stands for the time elapsed (in days) between the time of acquisition of the data and the current time.

$$\mathscr{C}_t = \max\left(0, 1 - \frac{t_e}{1825}\right) \tag{4.2}$$

### 4.1.3.2 Confidence of the face detector ($\mathscr{C}_d$)

When the frontal face detectors (as the one described in Section 4.1.2.1) are used to detect faces, two main observations can be made. Firstly, multiple detections are generally found over a face if it is frontal. Faces with in-plane and out-of-plane rotations get fewer detections. Secondly, the likelihood $p$ of a face (computed as the weighted ratio of the number of weak classifiers that label it as face to the total sum of weights of all the weak classifiers) is high for a frontal face and *vice versa*. We use both these criteria to compute

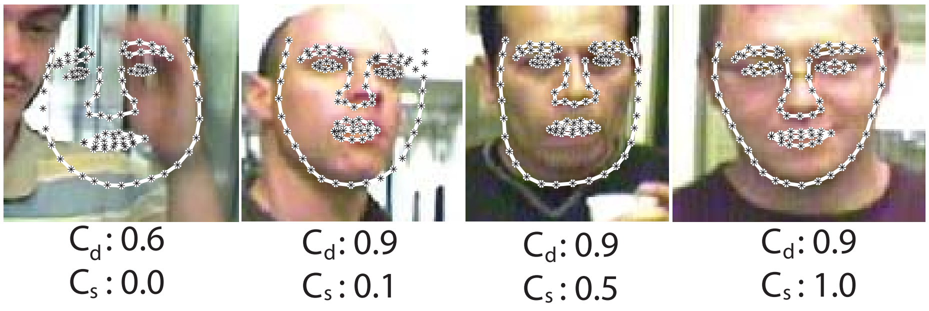


Figure 4.6: Example images arranged according to their $C_d$ and $C_s$ value.

the confidence of face detection as in (4.3).

$$\mathscr{C}_d = \sqrt[n]{\bar{p}} \tag{4.3}$$

where $\bar{p}$ is the mean of likelihood values, $p$, computed for all detection windows that are fused to form a single detection region ($\bar{p}$ is always between 0 and 1). The quantity $n$ denotes the number of detections around a face. It can be observed that as $n$ increases, $\mathscr{C}_d$ also increases. The difficult faces, for example, the non-frontal or insufficiently illuminated, generally have fewer detection windows, and, therefore, tend to have lower confidence of detection. High values of $\bar{p}$ and $n$ were observed for uniformly illuminated frontal faces.

#### 4.1.3.3 Confidence of the segmentation algorithm ($\mathscr{C}_s$)

One of the drawbacks of the segmentation algorithm described in Sec. 4.1.2.2 is its occasional failure to converge to the real contours of the face. This situation is illustrated in Fig. 4.2a. Although, visually, this fact is very clear, the segmentation algorithm itself does not provide an automatic way to detect bad segmentation results. Sukno and Frangi [45] propose a *reliability score*, which is computed using shape and the texture information, extracted by the segmentation algorithm during the matching process. The reliability score indicates if the fitting of the statistical face model to a test face is accurate or not. For details on its implementation, readers are referred to [45]. Fig. 4.6 shows the confidence of segmentation, $C_s$, computed on several test images.

#### 4.1.3.4 Confidence of the classification algorithm ($\mathscr{C}_c$)

The confidence of OSTCM-$k$NN classifier is computed using $\boldsymbol{p}$-values [52]. The $\boldsymbol{p}$-values are computed in two steps:

Firstly, the strangeness ($\alpha$) of a test exemplar $i$ with a label $y$ is computed as shown in (4.4).

$$\alpha_i = \left(\sum_{j=1}^{k} d_{ij}^{y}\right)\left(\sum_{j=1}^{k} d_{ij}^{-y}\right)^{-1} \tag{4.4}$$

The quantity, $\sum_{j=1}^{k} d_{ij}^{y}$ refers to the sum of all distances of the test exemplar $i$ to the $k$ nearest exemplars in the training set that belong to a class with a label $y$. The quantity, $\sum_{j=1}^{k} d_{ij}^{-y}$ refers to the sum of all distances of the test exemplar $i$ to the $k$ nearest exemplars in the training set that do not belong to a class with a label $y$. The distance between two vectors is defined as the smallest angle between them. The angle based distance metric was shown to be a good choice for PCA-based algorithms [31]. All the experiments in this chapter were performed using the five nearest neighbors ($k = 5$). Note that the lower the strangeness index, the higher is the probability of an exemplar $i$ to belong to class $y$.

Secondly, the $\boldsymbol{p}$-values are computed according to (4.5).

$$p_y(e) = \frac{\alpha_1 + \alpha_2 + \ldots + \alpha_l + \alpha_{new}^{y}}{(l+1)\alpha_{new}^{y}} \tag{4.5}$$

Here, $[\alpha_1, \alpha_2, \ldots, \alpha_l]$ are the strangeness values computed from the $l$ training vectors, and $\alpha_{new}^{y}$ is the strangeness value computed for the test exemplar with a putative label $y$. If there are $c$ classes of training data, then there are $c$ $p$-values for each test exemplar. The label that yields the largest $p$-value is chosen as the label of the test exemplar. The $p$-values can be thought of as a vector of probabilities of a test face belonging to one of the users enrolled in the FRS. The $p$-values are min-max normalized to restrict their values between 0 and 1. Min-max normalization was chosen over the more frequently used z-normalization [20] as it provides a bounded range of values.

#### 4.1.3.5 Fusion of shape and texture $p$-values

As depicted in Fig. 4.5, the shape and the texture information obtained from a test face are classified independently. The decisions of the shape and the texture classifiers are fused in the following manner. If $\boldsymbol{p}_s$ and $\boldsymbol{p}_t$ represent the shape and the texture $p$-values, the result of fusion, $\boldsymbol{p}_f$, is obtained according to (4.6).

$$\boldsymbol{p}_f = \boldsymbol{w}_s \cdot \boldsymbol{p}_s + \boldsymbol{w}_t \cdot \boldsymbol{p}_t \tag{4.6}$$

where $\boldsymbol{w}_s = \{w_s^1 w_s^2 \ldots w_s^c\}$ and $\boldsymbol{w}_t = \{w_t^1 w_t^2 \ldots w_t^c\}$ represent person-specific weights for fusing shape and texture $p$-values. Computing personalized weights is helpful because some users may be better discriminated by shape than by texture and *vice versa.*

The weights vectors, $\boldsymbol{w}_s$ and $\boldsymbol{w}_t$, are determined by examining the relative discriminating ability of the shape and texture data enrolled for a user. Every face in the training set, belonging to a user, is used as test data. Following classification using shape and texture data, $\boldsymbol{p}_s$ and $\boldsymbol{p}_t$ are obtained. The $p$-value noted for the genuine user label (illustrated with 'x' in Fig. 4.7), and the $p$-values recorded for other users (illustrated with '.' in Fig. 4.7) are noted. The optimal line of projection, determined using Fisher's linear discriminant

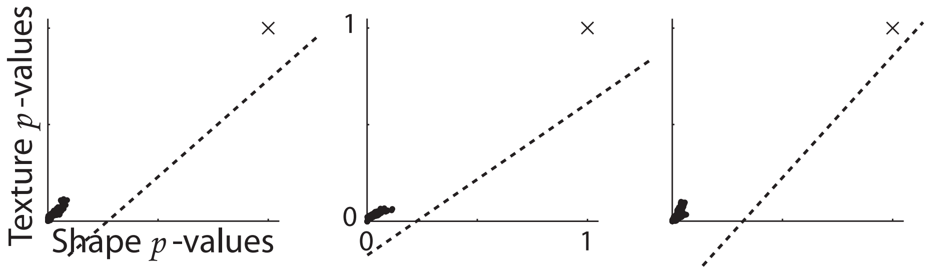


Figure 4.7: Illustration of the optimal line of projection (represented as a dotted line), whose slope specifies the weights for fusing shape and texture $p$-values. The markers 'x' and '.' represent $p$-values obtained for genuine user labels and the rest of the users, respectively. Note that, as a result of min-max normalization, the $p$-values recorded for the genuine user labels are 1.

analysis [15], that best separates the genuine and impostor $p$-values specifies the weights for optimal fusion for a particular user.

Note that the optimal weights for fusion changes every time the training set changes. In our study, we allow only the `SFF` strategy to use personalized models. Therefore, we update the optimal weights only for the `SFF` strategy. For the other two recognition algorithms, the weights are determined once, during manual enrollment of training images, and are unmodified during the update process.

#### 4.1.3.6 Computation of confidence value

The confidence of the classifier is derived as the difference of the largest $\boldsymbol{p}_f$-value with the second largest one [20]. Here, $\boldsymbol{p}_f$-value refers to the fused shape and texture $\boldsymbol{p}$-values. The confidence value indicates how improbable the classifications other than the predicted labels are.

#### 4.1.3.7 Image selection using confidence measures

The selection process proceeds as follows. A test face, with a label $j$, is added to the training set of the $j^{\text{th}}$ user, only if the confidence values computed on it ($\mathscr{C}_d$, $\mathscr{C}_s$ and $\mathscr{C}_c$) are greater than preset thresholds. The minimum required $\mathscr{C}_d$ and $\mathscr{C}_s$ values were determined using the interquartile distance in the distribution of $\mathscr{C}_d$ and $\mathscr{C}_s$ values of faces in the training set. In the following experiments, it was assumed that when $\mathscr{C}_d$ and $\mathscr{C}_s$ values are greater than $Q_1 - 3 \times (Q_3 - Q_1)$, then the performance of detection and segmentation algorithms are satisfactory. Here, $Q_1$ and $Q_3$ represent the first and the third quartiles, respectively. Otherwise, the image is deemed unfit for learning. The threshold for $\mathscr{C}_c$ was varied during the experiments and is detailed in Sec. 4.1.4.6.

It is important to note that as images are continuously added to the training set during the update process, it is possible to overwhelm the FRS simply by accumulating a huge amount of training data. In order to limit the number of images per user, every time a new image is added to an already full training set, the image with the lowest $\mathscr{C}_t$, is eliminated such that the total number of images per person is at most $\mathscr{N}$.

### 4.1.4 Experimental setup

In the following, the image datasets used for testing the self-updating FRS is described. Then, the update process is detailed followed by the procedure to set the impostor detection thresholds.

#### 4.1.4.1 Image databases

Two image databases were collected to perform the experiments: Gradually Evolving Facial Appearance (`GEFA`) and YouTube (`YT`). Both these databases contain frontal face images of people acquired over long periods of time, almost on a daily basis, so that changes in facial appearance is recorded.

#### 4.1.4.2 `GEFA`

This database contains images of 129 individuals (belonging to 15 different nationalities) acquired over a period of five months on a daily basis. A total of 14,192 images were collected. These images were acquired in a typical access control scenario. The lighting was more or less uniform and most of the faces in the database are frontal. There are also lots of cases where the individuals try to impersonate others by changing the pose of the face and the expression. Some of the typical images of this database are shown in Fig. 4.8. Of 129 individuals, 35 were selected for the *users group*, and the remaining 74 were chosen for the *impostors group*. Whether an individual belonged to the *user*- or *impostor group* was decided based on the frequency with the individual interacted with the system. Individuals who were regular throughout the acquisition period were chosen for the *users group*, and the rest for the *impostors group*. Among the images belonging to *users group*, the first 12 images, with high face detector and segmentation confidence, were selected manually for the initial training set. It was made sure that the initial training set, belonging to each user, does not contain corrupted data due to possible errors in face detection and segmentation. The rest of the images were used for the update procedure. Fig. 4.10 (left) shows the distribution of images in the `GEFA` database according $C_d$ and $C_s$ values. The presence of images with low $C_d$ and $C_s$ values in the update set can be noted, which potentially can cause corruption of training data.

#### 4.1.4.3 `YT`

This database contains 30,157 images of 25 individuals extracted from videos downloaded from YouTube website [2]. Some of the typical images in this database are shown in Fig. 4.9. The faces in this database are frontal with neutral expression. They exhibit high degree of appearance variation due to ageing and beard growth. As in `GEFA` database, the first

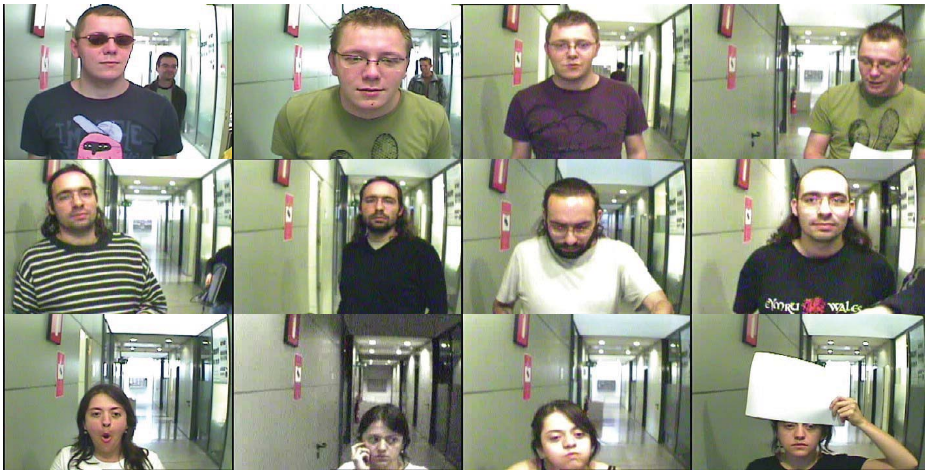


Figure 4.8: Typical images in the GEFA database.

12 images with high face detection and segmentation confidence values were chosen for the training set, and the rest were used for the update procedure. Fig. 4.10 (right) shows the distribution of images in the YT database according to $C_d$ and $C_s$ values. As in GEFA database, the presence of images that can potentially cause corruption can be noted.

#### 4.1.4.4 Limitations

GEFA and YT do not contain images of users who exhibit abrupt appearance changes. All the appearance changes occur gradually over time. We assume that it is in the interest of higher security that individuals who exhibit abrupt appearance changes should be enrolled again after operator-assisted verification.

#### 4.1.4.5 Update procedure

As self-learning approaches incrementally change the training data, the next identification result relies on the previously updated users. Even if the previously updated data are the same but the order in which the images are fed into the system are different, then the matching score of the next matching may be different. Ryu *et al.* [40] test three different update procedures on fingerprint update problem: 1) *Impostor first*, where impostor data are first presented to the system followed by data from genuine users; 2) *Genuine first*, where data from genuine users are passed through the update procedure followed by impostor data; 3) *Randomized*, where data from genuine users and impostors are updated randomly. Understandably, the *Impostor first* strategy produced the worst performing

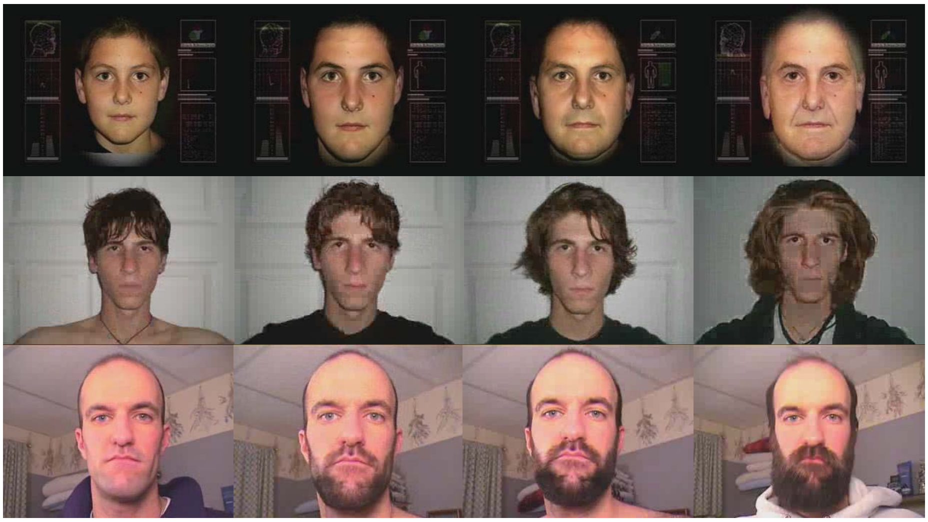


Figure 4.9: Typical images in the YT database.

ROC curves, as impostors have a higher probability of corrupting the training data. In our experiments, we follow the *Impostor first* update strategy in order to test the worst-case scenario.

Note the absence of a separate test database for GEFA and YT databases. We follow the continuous update+test strategy proposed by Poh *et al.* [34] in order to maximize the number of images available for the update process. In this strategy, the true labels of all the update data are known a priori, which permits to monitor the correct classification rates of a user's data during the update process.

To reduce variability in the measurements, a 10-fold cross-validation is performed during the update process. In each fold, the individuals in the training and the update set were randomized. From the 35 individuals labeled *user group* in the GEFA database, 30 were randomly selected as users, and the remaining were added to the 74 individuals in the *impostor group* to form impostors. For YT database, for each fold, 20 individuals were selected randomly as users and the the remaining 5 as impostors.

#### 4.1.4.6 Impostor detection

A face is labeled an impostor when the confidence of classification is low. The following two different strategies were used to set the threshold on classifier confidence.

1. Global impostor thresholds (GIT): In this strategy, the acceptance threshold for a genuine user was set globally, and was varied from 0 to 1.0 in increments of 0.1. Note

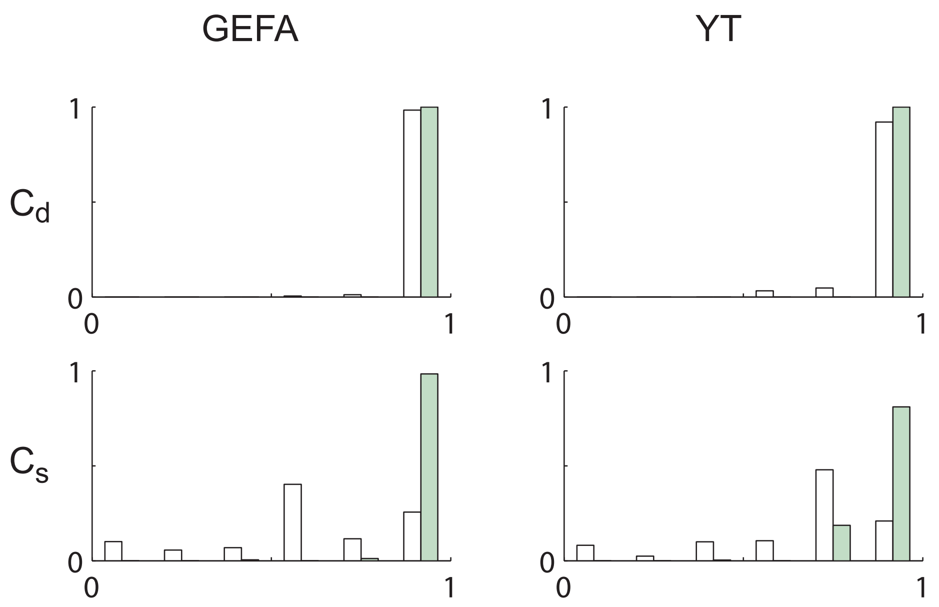


Figure 4.10: Normalized histograms of $C_d$ values (top row) and $C_s$ values (bottom row) for `GEFA` (left) and `YT` (right) databases. The shaded bars represent histogram values from the training set, and the white bars represent histogram from the update set.

that if the impostor threshold is set to 0, then, none of the faces will be labeled as impostor. Similarly, if the impostor threshold is set to 1.0, then all the faces classified by the system will be labeled as impostors.

2. User-specific impostor thresholds (UIT): As argued by Yager *et al.* [55] and Marcialis *et al.* [22], different users of FRS might need different impostor thresholds. The group of users who are vulnerable to impersonation (*Lambs* [11]), might need higher impostor threshold than the rest of the users. In order to determine the impostor threshold for each user, we followed the following strategy. The users enrolled in the FRS are randomly split into non-overlapping sets of *temporary users* and *temporary impostors*. The FRS is trained with the data belonging to *temporary users*, and the data from *temporary impostors* is used to test the system. Each data sample in the *temporary impostors* will be assigned a label belonging to *temporary users* and a confidence value. This process is repeated 50 times by selecting random *temporary users* and *temporary impostors*. At the end of 50 rounds, almost every user will have impostor confidence values assigned. Using these confidence values, the user-specific thresholds are set following the outlier criterion described in Frigge *et*

*al.* [14]. Using the first and the third quartiles, $Q1$ and $Q3$, the impostor threshold ($\theta$) is set in the following manner.

$$\theta = Q3 + k(Q3 - Q1) \tag{4.7}$$

In our experiments, the quantity $k$ was varied from 0 to 6. The higher the value of $k$, the higher will the impostor threshold. In case a user does not have any confidence values assigned, then $\theta$ is set to a default value of 0.2.

### 4.1.5 Results

The update process was performed for `GEFA` and `YT` separately. It was repeated for each impostor threshold on the FRSs based on `EF`, `FF` and `SFF`.

For every impostor detection threshold, the number of faces that were correctly- and incorrectly learnt, during the update process was noted. Normalizing the values by the total number of images learnt, at a threshold setting, the proportion of correctly- and incorrectly learnt faces can be computed, which is visualized in Fig. 4.11. Once the update process is completed, the extent of corruption in the training set was computed as the ratio of wrongly assigned faces in the training set to the total number of faces. The amount of corruption at the end of update process is also shown in in Fig. 4.11. The following observations can be made from the figure.

1. The stricter the impostor thresholds, both for `GEFA` and `YT`, the lower the proportion of incorrectly learnt faces. This correlates with the corruption of the training set observed after the update process.

2. Unlike the incorrectly learnt faces curve, the proportion of correctly learnt images need not always peak at the lowest impostor threshold settings. This is the result of the *Impostor first* update strategy. By the time FRS is updated with the user data, impostors corrupt the training set, which makes it impossible to learn the user data correctly.

3. The FRS based on `SFF`, when updated with `YT`, learns very few faces that were correctly classified. However, when updated with `GEFA`, this trend was not observed. This confirms that the learning capacity of an algorithm depends on the intrinsic properties of the dataset, such as the number of *wolves* (users who are good in impersonating others [11]) and *lambs* in it. As can be observed in Fig. 4.11, greater proportions of the training data are corrupted when tested with `YT` than when tested with `GEFA`.

4. In general, it was observed that using user-specific impostor detection thresholds leads to lower corruption and an increase in the proportion of correctly learnt facial data.

Figs. 4.12 and 4.13 show plots of Correct Classification Rate (CCR) and the Classifier confidence ($C_c$) as a function of time for different threshold settings. To generate the plots, we time-normalize the images in the dataset. That is, the first image in the update set belonging to a user has a time of 0, and the last one has a time of 1. The following observations can be made from this plot.

1. At high values of impostor thresholds, when very few correctly classified faces are learnt, it can be noted that the $C_c$ constantly decreases with time. Subtle decreases in CCR can be noted as well. This can be explained from the fact that, as people change with time, if the appearance and shape information is not updated, a decrease in $C_c$ and CCR will continue until the users are not recognized correctly anymore. The decrease in the $C_c$ measurements can be noted better in `YT` database than in `GEFA`. This is due the fact that `GEFA` was acquired within a relatively short time period of 5 months, and therefore, the changes in facial appearance was less in comparison to those in `YT` database. `YT`, as shown in Fig. 4.9, documents relatively higher degree of appearance changes, and therefore the original training data will be very different from the test data if correct learning does not take place.

2. When tested with `GEFA`, `SFF` performs better than `FF`, which in turn performs better that `EF` in terms of CCR for most threshold settings. This trend is reversed when FRS is updated with `YT`. It is clear that the recognition algorithms that make decisions based on inductive rules, `SFF` and `FF`, are affected more by corruption in the training set than the purely transductive implementation of the `EF` algorithm.

3. `EF`, in general, performs better in `YT` than in `GEFA`. The possible reason for this could be that the images in `GEFA` were acquired using the same camera and in similar lighting conditions. This might make users appear similar to each other. People in `YT` used different cameras in different ambient lighting conditions, which facilitates discrimination between users.

4. When corruption is minimal and, simultaneously, when there is a high percentage of correctly learnt faces (*e.g.,* when FRS based on EF is tested with `YT` with a global impostor threshold = 0.5), we notice that the $C_c$ measurements actually increase with time. However, the interval of impostor thresholds where this occurs is very narrow. Lowering the threshold increases corruption, resulting in decline in CCR over time. Increasing the threshold also has a similar effect, this, however, is because no learning takes place as the impostor thresholds are very high.

5. If optimal impostor threshold can be defined as the threshold setting where at which no corruption and simultaneously maximum learning occurs, then, one can observe that the optimal threshold varies with both database and the recognition algorithm.

Fig. 4.14 shows an example of how rapidly a database gets corrupted when impostors are added to the training set. The plot was obtained by updating FRSs with `YT` with GIT = 0.2. We can observe that FRS based on `EF` starts accepting impostors first, followed by the ones based on `FF` and then `SFF`. Once impostor data is introduced into the system, FRS based on `SFF` corrupts itself at a faster rate than the ones based on `FF` and `EF`. This

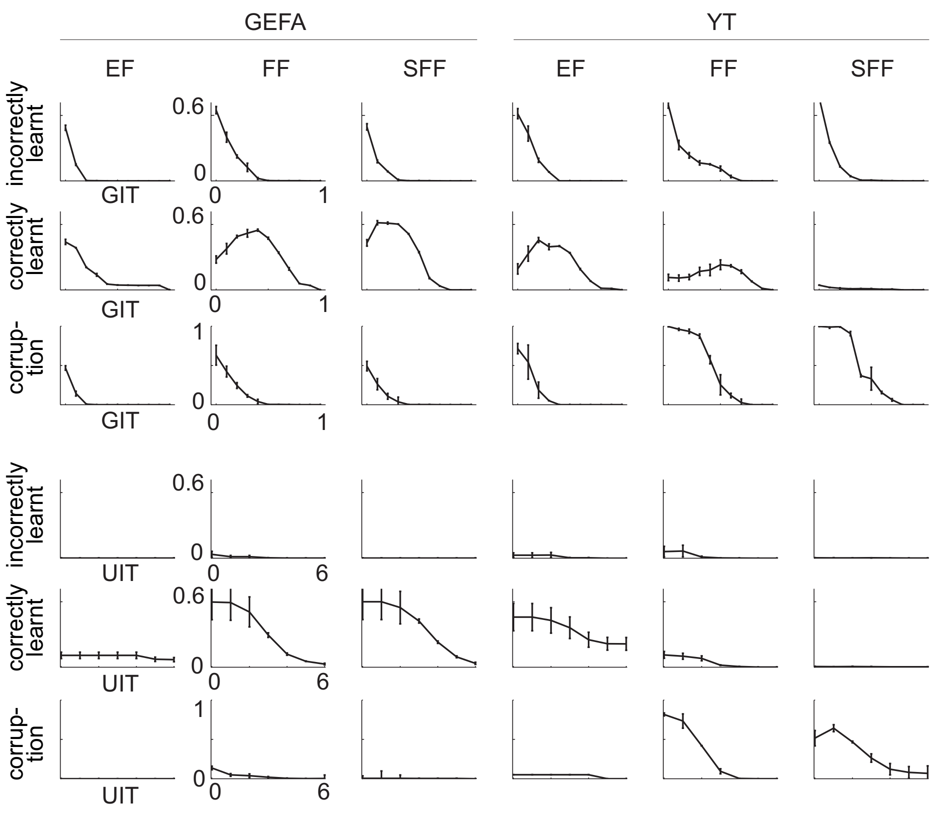


Figure 4.11: The effect of varying the impostor thresholds on the learning capacity of the FRSs. The figure is arranged in two blocks: the first three rows plot the learning capacity by varying GIT, and the second three rows by varying UIT. The first two rows of each block show plots of correctly learnt- and incorrectly learnt images with during the update process for different impostor threshold settings. The third row shows the corruption in the training set after the update process is completed. Error bars indicate the standard error of the mean. The dynamic range of the axes are the same along each row, which is mentioned in the second column.

example clearly shows that the higher the level of induction in the recognition algorithm, the earlier the FRS will fail.

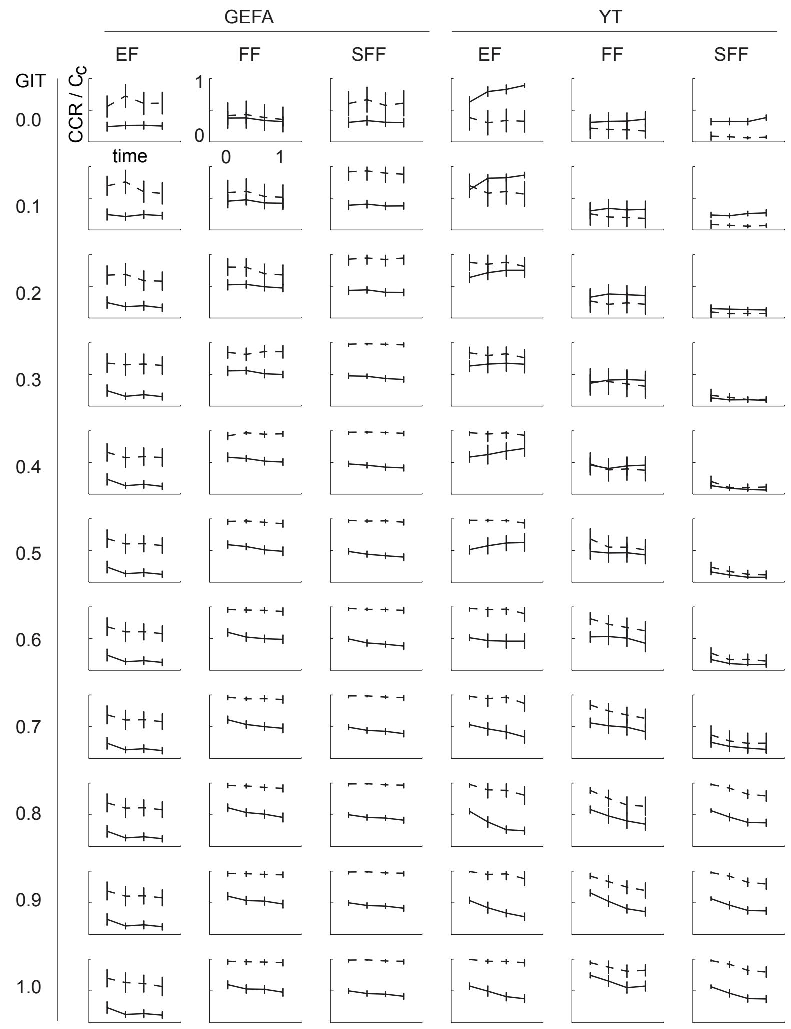


Figure 4.12: Plot of correct classification rate (CCR) and the confidence of classification ($C_c$) as a function of time. CCR and $C_c$ are represented with a dashed line and a solid line, respectively. Error bars indicate the standard error of the mean. The dynamic range of the axes in this figure are the same, whose limits are 0 and 1, both in horizontal and vertical directions.

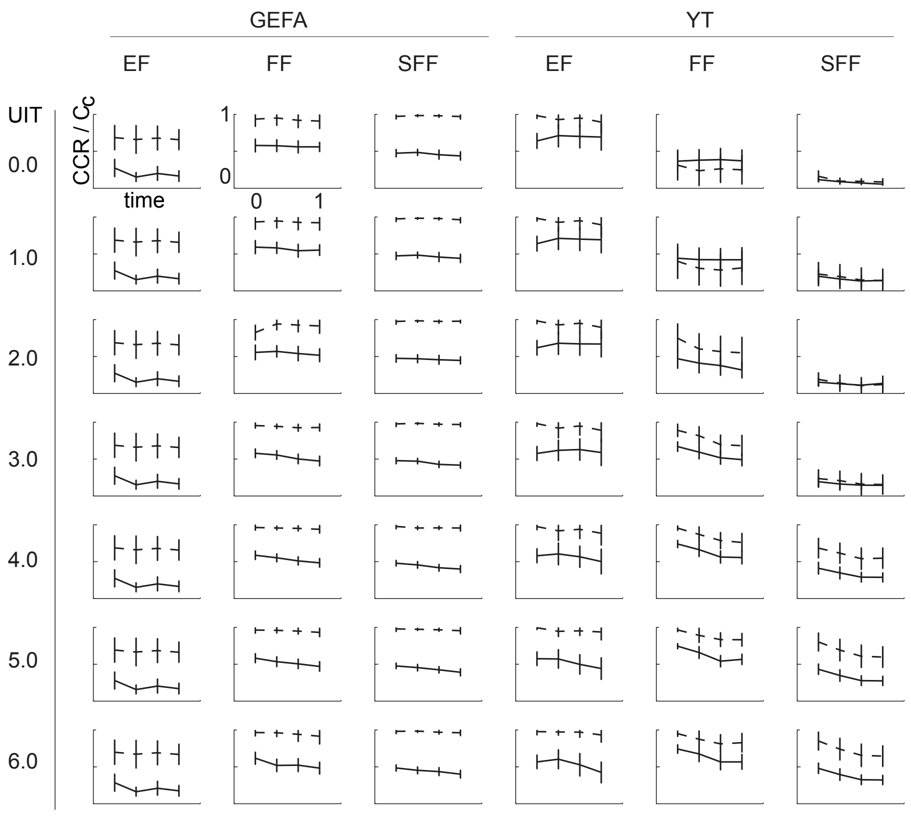


Figure 4.13: Plot of correct classification rate (CCR) and the confidence of classification ($C_c$) as a function of time. CCR and $C_c$ are represented with a dashed line and a solid line, respectively. Error bars indicate the standard error of the mean. The dynamic range of the axes in this figure are the same, whose limits are 0 and 1, both in horizontal and vertical directions.

### 4.1.6 Summary and discussion

This chapter studies the learning capacity and corruption in self-updating Face Recognition Systems (FRSs) built with recognition algorithms that use three differing levels of induction: Eigenfaces (`EF`), Fisherfaces (`FF`) and Similarity-based Fisherfaces (`SFF`). The FRSs were updated using two large and meaningful facial image databases: `GEFA` and `YT` databases with 14,279 and 31,951 images, respectively. Worst case analysis of different FRSs over different databases shows two clear conclusions.

Firstly, the higher the level of induction in the classification algorithm, the faster the

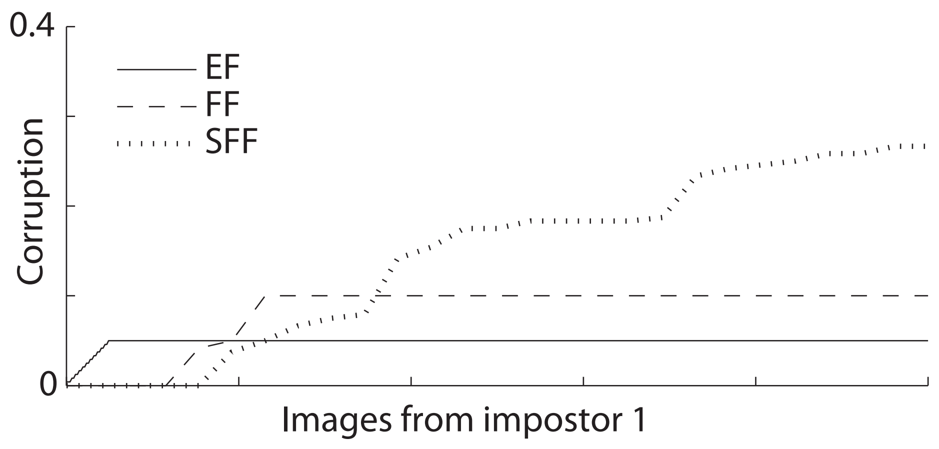


Figure 4.14: Plot of corruption in training set as the images from the first impostor are fed to the system. Corruption was measured as the ratio of the number of wrongly assigned faces in the training set to the total number of faces.

training set of the FRS will be corrupted. It is interesting to note that, over the years, researchers have proposed using higher levels of induction in face recognition algorithms (*i.e.,* using personalized models) to obtain better classification rates. Our results show that, the state-of-the-art algorithms need not necessarily be the best choice for use in a self-updating FRS.

Secondly, our experiments confirm previous studies that the self-updating FRSs can indeed be maintained error-free during the update process. However, the optimal impostor threshold (global or user-specific), at which no corruption and simultaneously maximum learning occurs, for a given classification algorithm seems to be database-dependent. As noted in [22], the existence of errors is a function of the number of *Lambs* and *Wolves* [11] in a database. Currently, it appears that the only way to keep the self-updating biometric system error-free is through periodic manual checking of the training images.

It should be noted that we perform a worst-case analysis of a self-updating FRS, where the system is updated with the impostor data first. By the time genuine users are tested, the training set might be corrupted entirely. A self-updating FRS installed in a real-life scenario, will most likely corrupt at a slower speed, as one can assume that the probability of a user interacting with the system is greater than that of an impostor. Nevertheless, if the system is allowed to run for sufficient time, a decrease in performance will be noted due to the corrupted data.

Two different measures can be implemented in the current setup to reduce the corruption of training data in self-updating FRSs. Firstly, robust methods [18] may be used to estimate the axis of projection in FF and SFF classification techniques. A disadvantage of using robust methods is that the learning capacity of the FRS will reduce as well. This is because, a change in appearance of a user will most probably be an outlier amongst the previously acquired training data. Robust methods will tend to ignore the outlier, and therefore, the newly included data will not be learnt. Secondly, increasing the amount of training data stored per user should decrease the ill-effects of corruption. This is because the addition of one corrupted data sample will corrupt the database relatively less if the training set were bigger. Recently, we studied the effect of increasing the number of data samples stored per user, and found that as the number of data samples per person increases, the slower the FRS gets corrupted [30]. Note that the above-mentioned measures may merely help to prolong the stability of the FRS. They cannot maintain the FRS without corruption.

The study presented in this chapter opens up few areas for future research. We use four metrics to prevent corruption of training data: time confidence, face detector confidence, segmentation confidence, and classifier confidence. Are there better ways of estimating these values? Better estimations of these confidence values can potentially lead to reduction in corruption. Perhaps, additional confidence measures can be derived from the image data to better predict if a data sample belongs to the impostor class or not. An interesting architecture to consider is the use of ensemble of face recognition algorithms working in parallel, to predict the label of a test face. The probability of corruption should intuitively reduce, and therefore, increase the stability of self-updating FRSs.

In closing, our recommendation for the design of a self-updating FRS is the following. Firstly, minimize inductive reasoning in the classification strategy, especially if the number of training samples for a user is very small. Secondly, store as much training data as possible. Perhaps, inductive rules can be derived for a user once the number of images in his/her training database reaches a certain number. Finally, consider batch-updates instead of instantaneous online updates under manual supervision. Perhaps, the operator might be presented with a yes/no tool, where he/she could say whether the automatically gathered images belongs to a particular user or not.

### 4.1.7 Databases download information

Information to obtain GEFA database can be found here: http://cilab2.upf.edu/gefa. The YT database was composed of frames extracted from the following videos:

- http://www.youtube.com/watch?v=henv2naXuGo
- http://www.youtube.com/watch?v=xbKKsx2xC64
- http://www.youtube.com/watch?v=lRHPWZpRNX4
- http://www.youtube.com/watch?v=QY8gja0_CCA
- http://www.youtube.com/watch?v=LCIOF3JFTpc
- http://www.youtube.com/watch?v=8lGlC9e8MdM
- http://www.youtube.com/watch?v=QjAku6fnjEM
- http://www.youtube.com/watch?v=jHvBSW0iw3U

- http://www.youtube.com/watch?v=h0fcMXKKMLA
- http://www.youtube.com/watch?v=y6YUjyPksyE
- http://www.youtube.com/watch?v=TVQkJZEfzkQ
- http://www.youtube.com/watch?v=KCbs__ew5wc
- http://www.youtube.com/watch?v=fvGCVdxu15Y
- http://www.youtube.com/watch?v=AlZTiEHeTvE
- http://www.youtube.com/watch?v=Vc_PU3D3QNE
- http://www.youtube.com/watch?v=1a7082PYcSk
- http://www.youtube.com/watch?v=6B26asyGKDo
- http://www.youtube.com/watch?v=55YYaJIrmzo
- http://www.youtube.com/watch?v=KvP0fEKCanI
- http://www.youtube.com/watch?v=sPxMbVMJDY4
- http://www.youtube.com/watch?v=lKzt8970r9I
- http://www.youtube.com/watch?v=fa5rzZroNyU
  (4 videos)

## 4.2 References

[1] FaceIt SDK ®, L-1 Identity Solutions, Inc. http://www.identix.com/pages/101-faceit-sdk, 2009.

[2] Youtube, LLC, http://www.youtube.com, 2009.

[3] L Aryananda. Recognizing and remembering individuals: online and unsupervised face recognition for humanoid robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1202–1207, Lausanne, Switzerland, 2002.

[4] P N Belhumeur, J P Hespanha, and D J Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[5] A Blum and S Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 19–26, Williamstown, MA, USA, 2001.

[6] A Blum and T Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100, Madison, Wisconsin, USA, 1998.

[7] I Cohen, F G Cozman, N Sebe, M C Cirelo, and T S Huang. Semisupervised learning of classifiers: Theory, algorithms, and their application to human-computer interaction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(12):1553–1567, 2004.

[8] T F Cootes, G J Edwards, and C J Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[9] T F Cootes, C J Taylor, D H Cooper, and J Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[10] D Delgado-Gomez, J Fagertun, B Ersbøll, F M Sukno, and A F Frangi. Similarity-based Fisherfaces. *Pattern Recognition Letters*, 30(12):1110–1116, 2009.

[11] G Doddington, W Liggett, A Martin, M Przybocki, and D Reynolds. Sheep, goats, lambs and wolves: a statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1351–1354, Sydney, Australia, 1998.

[12] P J Flynn, K W Bowyer, and P J Phillips. Assessment of time dependency in face recognition: An initial study. In *Proceedings of the International Conference on Audio- and Video-Based Biometric Person Authentication,Lecture Notes in Computer Science 2688*, pages 44–51, 2003.

[13] A Franco, D Maio, and D Maltoni. Incremental template updating for face recognition in home environment. *Pattern Recognition, In Press*, 2010.

[14] M Frigge, D C Hoaglin, and B Iglewicz. Some implementations of the boxplot. *The American Statistician*, 43(1):50–54, 1989.

[15] K Fukunaga. *Introduction to statistical pattern recognition.* Academic Press Professional, Inc., 2 edition, 1990.

[16] E Hjelmas and B K Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, 2001.

[17] X Jiang and W Ser. Online fingerprint template improvement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1121–1126, 2002.

[18] S-J Kim, A Magnani, and S P Boyd. Robust fisher discriminant analysis. In *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems*, pages 659–666, Vancouver, B.C., Canada, 2005.

[19] A Lanitis, C J Taylor, and T F Cootes. Toward automatic simulation of aging effects on face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):442–455, 2002.

[20] F Li and H Wechsler. Open set face recognition using transduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1686–1697, 2005.

[21] H Ling, S Soatto, N Ramanathan, and D W Jacobs. A study of face recognition as people age. In *Proceedings of International Conference on Computer Vision*, pages 1–8. Rio de Janeiro, Brazil, 2007.

[22] G L Marcialis, A Rattani, and F Roli. Biometric template update: an experimental investigation on the relationship between update errors and performance degradation in face verification. In *Procedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 847–856, Orlando, FL, USA, 2008.

[23] A M Martinez and R Benavente. The ar face database. Technical Report 24, Computer Vision Center, Universitat Autonoma de Barcelona, 2000.

[24] K Messer, J Kittler, M Sadeghi, S Marcel, C Marcel, S Bengio, F Cardinaux, C Sanderson, J Czyz, L Vandendorpe, S Srisuk, M Petrou, W Kurutach, A Kadyrov, R Paredes, E Kadyrov, B Kepenekci, F B Tek, G B Akar, N Mavity, and F Deravi. Face verification competition on the XM2VTS database. In *Proceedings of the International Conference on Audio- and Video-Based Biometric Person Authentication, Lecture Notes in Computer Science 2688*, pages 964–974, Guildford, UK, 2003.

[25] L Millsopp, L Brandom, G Humphris, D Lowe, C Stat, and S Rogers. Facial appearance after operations for oral and oropharyngeal cancer: A comparison of casenotes and patient-completed questionnaire. *British Journal of Oral and Maxillofacial Surgery*, 44:358–363, 2006.

[26] D Mou, R Schweer, and A Rothermel. Automatic databases for unsupervised face recognition. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition Workshop*, pages 90–98, Washington, DC, USA, 2004.

[27] P Navarrete and J Ruiz-del Solar. Comparative study between different eigenspace-based approaches for face recognition. In *Proceedings of the International Conference on Fuzzy Systems, Lecture Notes in Artificial Intelligence 2275*, pages 178–184, 2002.

[28] C P Papageorgiou, M Oren, and T Poggio. A general framework for object detection. In *ICCV '98: Proceedings of the International Conference on Computer Vision*, pages 555–562, Washington, DC, USA, 1998.

[29] S-K Pavani, D Delgado, and A F Frangi. Gaussian weak classifiers based on Haar-like features with four rectangles for real-time face detection. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns, Lecture Notes in Computer Science 5702*, pages 91–98, Münster, Germany, 2009.

[30] S-K Pavani, D Delgado, and A F Frangi. A rapidly trainable and global illumination invariant object detection system. In *Proceedings of the Fourteenth Iberoamerican Congress on Pattern Recognition*, pages 877–884, Guadalajara, Mexico, 2009.

[31] V Perlibakas. Distance measures for PCA-based face recognition. *Pattern Recognition Letters*, 25(6):711–724, 2004.

[32] P J Phillips, P Grother, R Micheals, D M Blackburn, E Tabassi, and M Bone. Face recognition vendor test 2002: Evaluation report. Technical report, National Institute of Standards and Technology - 6264, 2003.

[33] P J Phillips, W T Scruggs, A J O'Toole, P J Flynn, K W Bowyer, C L Schott, and M Sharpe. FRVT 2006 and ICE 2006 large-scale experimental results. *IEEE Transactions on Pattern Analysis and Machine Intelligence, In press*, 2009.

[34] N Poh, R Wong, J Kittler, and F Roli. Challenges and research directions for adaptive biometric recognition systems. In *Proceedings of International Conference on Biometrics, Lecture Notes in Computer Science 5558*, pages 753–764, Alghero, Italy, 2009.

[35] N Ramanathan, R Chellappa, and S Biswas. Computational methods for modeling facial aging: A survey. *Journal of Visual Languagesand Computing*, 20(3):131–144, 2009.

[36] A Rattani, B Freni, G L Marcialis, and F Roli. Template update methods in adaptive biometric systems: A critical review. In *Proceedings of International Conference on Biometrics, Lecture Notes in Computer Science 5558*, pages 847–856, Alghero, Italy, 2009.

[37] A Rattani, G L Marcialis, and F Roli. Biometric template update using the graph mincut algorithm: A case study in face verification. In *Procedings of the Sixth Biometrics Symposium*, pages 23–28, Tampa, FL, USA, 2008.

[38] A Rattani, G L Marcialis, and F Roli. An experimental analysis of the relationship between biometric template update and the doddington's zoo: A case study in face verification. pages 434–442, 2009.

[39] H A Rowley, S Baluja, and T Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

[40] C Ryu, H Kim, and A K Jain. Template adaptation based fingerprint verification. In *Proceedings of the Eighteenth International Conference on Pattern Recognition*, volume 4, pages 582–585, Hong Kong, China, 2006.

[41] A Selinger and D Socolinsky. Appearance-based facial recognition using visible and thermal imagery: a comparative study, 1999.

[42] A Sethuram, E Patterson, K Ricanek, and A Rawls. Improvements and performance evaluation concerning synthetic age progression and face recognition affected by adult aging. In *Proceedings of International Conference on Biometrics, Lecture Notes in Computer Science 5558*, pages 62–71, Alghero, Italy, 2009.

[43] R Singh, M Vatsa, A Noore, and S K Singh. Age transformation for improving face recognition performance. In *Proceedings of the Second International Conference on Pattern Recognition and Machine Intelligence*, pages 576–583, 2007.

[44] M B Stegmann. Analysis and segmentation of face images using point annotations and linear subspace techniques. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, 2002.

[45] F M Sukno and A F Frangi. Reliability estimation for statistical shape models. *IEEE Transactions on Image Processing*, 17(12):2442–2455, 2008.

[46] F M Sukno, S Ordas, C Butakoff, S Cruz, and A F Frangi. Active shape models with invariant optimal features: Application to facial analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1105–1117, 2007.

[47] J Suo, S-C Zhu, S Shan, and X Chen. A compositional and dynamic model for face aging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):385–401, 2010.

[48] M A Taister, S D Holliday, and H I M Borrman. Comments on facial aging in law enforcement investigation. *Forensic Science communications*, 2(1), 2000.

[49] M Turk and A Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[50] B Van Ginneken, A F Frangi, J J Staal, B M Ter Haar Romeny, and M A Viergever. Active shape model segmentation with optimal features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[51] V N Vapnik. *Statistical Learning Theory.* Wiley, 1998.

[52] V Vovk, A Gammerman, and C Saunders. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 444–453, Bled, Slovenia, 1999.

[53] H Wang and P J Flynn. Experimental evaluation of eye location accuracies and time-lapse effects on face recognition systems. In *Proceedings of the Fifth International Conference on Audio- and Video-Based Biometric Person Authentication, Lecture Notes in Computer Science 3546*, pages 627–636, Hilton Rye Town, NY, USA, 2005.

[54] L Wiskott, J-M Fellous, N Kruger, and C Von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:775–779, 1997.

[55] N Yager and T Dunstone. The biometric menagerie. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):220–230, 2010.

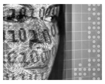
[56] X Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences, University of Wisconsin-Madison, 2005.

# 5 Conclusions

Positive match rates of the best performing recognition algorithm after facial cosmetic surgery:

| | |
|---|---|
| Rhytidectomy | 2.0% |
| Rhinoplasty | 37.3% |
| Liposhaving | 19.1% |
| Brow lift | 37.0% |
| Mentoplasty | 38.5% |

*IEEE Spectrum, Sept 2009*



The general objective of the work performed in this thesis was to advance the state-of-the-art in facial biometrics. Facial biometrics, as emphasized previously, is still not foolproof, and it requires significant improvements. During the last four and a half years, we have focused on three interesting sub-problems, and the main conclusions derived from our studies are detailed in the following paragraphs.

There is an ever-present need to improve the testing speed and the accuracy of the face detectors. The faster the detector, the more the number of images that can be processed which would translate into higher level of vigilance in large premises like airports, universities, or big public open spaces. The need for greater accuracy is obvious: to decrease false positives and to increase the probability of correctly detecting a face. To this end, we proposed two modifications to the state-of-the-art in face detection: the Viola and Jones-type detectors. Firstly, we propose assigning optimal weights to the rectangles of the Haar-like features as an alternative to the default weights used traditionally. We prove experimentally and give intuitive insights into why assigning optimal weights to the rectangles is beneficial. The resulting weak classifier is computationally more expensive than the traditional formulation; however, the increase in accuracy it provides improves the overall speed-accuracy trade-off of the detector. The increase in speed and accuracy for different training methods can be noted in Fig. 2.8 and Table 2.2. Secondly, we propose Gaussian weak classifiers based on co-occurring Haar-like features for improving the speed-accuracy trade-off. Our experiments showed that the proposed approach gives 67% better false positive rate for the same execution time and the same true positive rate when compared with Viola and Jones's approach. With respect to detectors based on Mita *et al.*'s approach, the proposed detector is 38% more accurate in false positives and simultaneously 42% faster given similar true positive rates.

As per the *No free Lunch* theorem[1], the better speed-accuracy trade-off of the methods proposed in Chapter 2 come with a price: their training time. The increase in the training time of the proposed detectors with respect to the Viola and Jones's training procedure was approximately 600%! It is true that for applications such as face detection, where all the user is interested is in getting a good "face" model, one would not mind if the training process takes months. However, there are applications in CBIR where instant training is required. Imagine a situation where a security guard wants to locate a suspicious luggage in all the images obtained in an airport as soon as possible. In such a scenario, Viola and Jones's detector cannot be used mainly because of its slow training process. It is indeed a pity as its testing speed and its generalization power would have come in handy for this search. To this end, we posed ourself a question, *can we identify the bottle-necks of the training process of the object detector and identify the means to make it faster?* Our solution to this problem

[1]The *No free lunch* theorem broadly states that any elevated performance over one class of problems is exactly paid for in performance over another class; Wolpert, D.H., and Macready, W.G., "No free lunch theorems for optimization", IEEE Transactions on Evolutionary Computation, 1(1), 67–82, 1997

mainly revolved around answering another question *do the object detectors need to know what is not an object to learn what is an object?* We propose two techniques to improve the training speed in Chapter 3. What is common in these two techniques is the absence of clutter images in the training process. One needs millions of clutter images to train the detector, and by not using them, one could save lots of time in reading and evaluating them. In Sec. 3.1, we reduce the training time to the order of minutes by using a simple clutter model that basically says that the probability of feature value of a Haar-like feature on a clutter image being greater/lesser than 0 is 0.5. Using this model also enables the weak classifiers to be invariant to global illumination changes. Please check Sec. 3.1.2 for more details. In Sec. 3.2, we propose Laplacian distributions to model histograms of feature values obtained from clutter images. These more powerful models help decrease the training time to the order of few seconds, and thereby truly enabling real-time image search capability for Viola and Jones-type object detectors. We finish this chapter by emphasizing the *No free Lunch* theorem again: the decrease in training time comes with a reduction in accuracy. The face detectors from the proposed methods produce 10 times more false positives than the Viola and Jones's approach. However, for simpler objects, such as human heart images, comparable accuracy is achieved. We are aware that most computer vision scientists are generally averse to new methods that favor training speed over accuracy. However, our work draws motivation from the Google search engine results - where we, as users, quite easily tolerate errant search results, and we probably would stop using Google if its response time would be in the order of hours.

Face detection systems are, in general, trained with huge databases with faces of people from different ages / expressions / facial hair growth / scars, *etc.* To this effect, the face detector will not have problems in detecting the face of a male user if he, say, decides to grow a beard. However, facial appearance changes become a big issue in case of face recognition systems. This is because, the systems are trained with relatively few images of a user - generally acquired over a short period of time. The user may be asked to change his expression and pose during the acquisition process, but, there is no way the system will know how the user will appear in the future. Of course, if the user's appearance changes, the system will not properly recognize the user anymore. One solution is to re-enroll the user with new images of the user and remove the old ones. But, can this process be done automatically? Assuming that the user interacts with the face recognition system frequently, can the system learn efficiently without mistakes? These are the questions we try to answer in Chapter 4. This chapter studies the learning capacity and corruption in self-updating Face Recognition Systems (FRSs) built with recognition algorithms that use three differing levels of induction: Eigenfaces (`EF`), Fisherfaces (`FF`) and Similarity-based Fisherfaces (`SFF`). The FRSs were updated using two large and meaningful facial image databases: `GEFA` and `YT` databases with 14,279 and 31,951 images, respectively. Worst case analysis of different FRSs over different databases shows two clear conclusions. Firstly, the higher the level of induction in the classification algorithm, the faster the training set of the FRS will be corrupted. It is interesting to note that, over the years, researchers have proposed using higher levels of induction in face recognition algorithms (*i.e.,* using personalized models) to obtain better classification rates. Our results show that, the state-of-the-art algorithms need not necessarily be the best choice for use in a self-updating FRS. Secondly, our experiments confirm previous studies that the self-updating FRSs can indeed be maintained error-free during the update process. However, the optimal impostor threshold (global or user-specific), at which no corruption and simultaneously maximum learning occurs, for a given classification algorithm seems to be database-dependent.

As it is evident, the research performed in this thesis covers a small area of facial biometrics - which needs considerable amount of work to be able to be useful in critical scenarios. We hope that our effort has sparked further interest in this area leading to more usable and robust systems.

# PUBLICATIONS

## Publications in journals

1. S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, Haar-like Features with Optimally Weighted Rectangles for Rapid Object Detection, Pattern Recognition, 43(1): 160-172, 2010

2. J. Ortega-Garcia, J. Fierrez, F. Alonso-Fernandez, J. Galbally, M.R. Freire, J. Gonzalez-Rodriguez, C. Garcia-Mateo, J-L. Alba-Castro, E. Gonzalez-Agulla, E. Otero-Muras, S. Garcia-Salicetti, L. Allano, B. Ly-van, B. Dorizzi, J. Kittler, T. Bourlai, N. Poh, F. Deravi, M.W.R. Ng, M.C. Fairhurst, J. Hennebert, A. Humm, M. Tistarelli, L. Brodo, J. Richiardi, A. Drygajlo, H. Ganster, F.M. Sukno, S-K. Pavani, A.F. Frangi, L. Akaruna and A. Savran, The Multi-Scenario Multi-Environment BioSecure Multi-modal Database (BMDB). IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(6):1097-1111,2010

3. S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, Gaussian Weak Classifiers based on Co-occurring Haar-like Features for Face Detection. Submitted, 2010.

4. S-K. Pavani, F.M. Sukno, D. Delgado-Gomez, C. Butakoff, X. Planes and A.F. Frangi, Self-Updating Frontal Face Recognition Systems: Design and Experimental Evaluation. Submitted, 2010.

## Publications in peer-reviewed conferences

1. S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, A Rapidly Trainable and Global Illumination Invariant Object Detection System. In Proc. Iberoamerican Congress on Pattern Recognition, Guadalajara, Jalisco, México. Lecture Notes in Computer Science vol. 5856, Pages 877-884, 2009.

2. S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, Gaussian Weak Classifiers Based on Haar-Like Features with Four Rectangles for Real-time Face Detection. In Proc. Computer Analysis of Images and Patterns, Münster, Germany. Lecture Notes in Computer Science vol. 5702, pages 91-98, 2009.

3. S-K. Pavani, F.M. Sukno, C. Butakoff, X. Planes and A.F. Frangi, A Confidence-Based Update Rule for Self-updating Human Face Recognition Systems. In Proc. International Conference on biometrics, Alghero, Italy. Lecture Notes in Computer Science vol. 5558, pages 91-98, 2009.

4. F.M. Sukno, S-K. Pavani, C. Butakoff and A.F. Frangi. Automatic Assessment of Eye Blinking Patterns through Statistical Shape Models. In Proc. International Conference on Computer Vision Systems, Liege, Belgium. Lecture Notes in Computer Science vol. 5815, pages 33-42, 2009.

5. S-K. Pavani, D. Delgado-Gomez and A.F. Frangi, Fast training of Viola-Jones type Object Detectors using Laplacian Clutter Models. Submitted, 2010.

## Media coverage

Some of the work presented in the thesis have been covered in the following news articles:

- *Desarrollan un sistema de biometría facial que crea un DNI del rostro*
  Radio Televisión Española (02/11/2009)
- *Científicos españoles trabajan en un DNI facial para cada persona*
  Diario ABC (03/11/2009)
- *Researchers develop a facial biometrics system capable of creating a facial DNI*
  Alpha Galileo (04/11/2009)
- El DNI facial deja de ser ciencia ficción
  Publico (04/11/2009)
- *Científicos desarrollan un sistema para crear un DNI*
  ADN (04/11/2009)
- *Un grupo de científicos desarrolla un sistema para crear el 'DNI'*
  20 minutos (04/11/2009)
- *Los ordenadores revolucionan el reconocimiento de personas*
  El periodico de Catalunya (09/11/2009)

# BIOGRAPHY

Sri-Kaushik Pavani is a PhD candidate in Computer Science and Visual Communication in Universitat Pompeu Fabra, Barcelona. He received the M.S. degree in Electrical Engineering from the Texas Tech University in 2003, and the B.E. degree in Electronics and Communication Engineering from the Shanmugha College of Engineering, India in 2001. His current work is mainly focused on the development and implementation of object detection systems for access control and monitoring in intelligent environments.

# INDEX

Notes:

Notes:

Notes:

Notes:

Notes:

Notes: