FACE DETECTION USING LEARNING NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF

THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

FAİK BORAY TEK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

JUNE 2002

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Mübeccel Demirekler
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Gözde Bozdağı Akar
Co-Supervisor

Assist. Prof. Dr. A. Aydın Alatan
Supervisor

Examining Committee Members

Prof. Dr. Uğur Halıcı

Assoc. Prof. Dr. İsmet Erkmen

Assoc. Prof. Dr. Gözde Bozdağı Akar

Assoc. Prof. Dr. Volkan Atalay

Assist. Prof. Dr. A. Aydın Alatan

ABSTRACT

FACE DETECTION USING LEARNING NETWORKS

Tek, Faik Boray

M.Sc, Department of Electrical and Electronics Engineering

Supervisor: Assist. Prof. Dr. A. Aydın Alatan

Co-Supervisor: Assoc. Prof. Dr. Gözde Bozdağı Akar

JUNE 2002, 72 pages

Face detection is a challenging computer vision problem. Given a still image or an image sequence, the goal of face detection is to locate all regions that contain a face regardless of any three dimensional transformation and lighting condition. There are two main categories that may serve as a solution for this problem: feature-based and image-based approaches. In this thesis, two different image-based and learning oriented solutions are compared, to observe the learning dynamics and face detection performances. In the first

approach, named Sparse Network of Winnows (SNoW) based face detector, the problem space is assumed to be linearly separable and a linear threshold function is offered for the solution which is supported by a sparse feature mapping architecture. For the second approach, discarding the linear separability assumption, a neural network in the form of a multilayer perceptron solution is used which assumes to represent any function using arbitrary decision surfaces by utilizing nonlinear activation functions. Observations in the comparative experiments show that the methods show closer performances for the classification in the face and non-face space. Furthermore, simple architecture of the SNoW learning method enables faster training and evaluation with respect to the neural network counterpart.

Keywords: Face Detection, Learning, Neural Networks, Sparse Network Of Winnows.

ÖZ

ÖĞRENEN AĞLAR KULLANARAK YÜZ BULMA

Tek, Faik Boray

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Assist. Prof. Dr. A. Aydın Alatan

Ortak Tez Yöneticisi: Assoc. Prof. Dr. Gözde Bozdağı Akar

HAZİRAN 2002, 72 sayfa

Yüz bulma, yapay görünün zorlu problemlerinden biridir. Yüz bulma probleminin amacı, bir tek veya ardışık görüntü verildiğinde, yüz içeren bölgeleri 3 boyutlu transformasyona ve ışıklandırma koşullarına bağımlı olmadan belirlemektir. Bir tek veya ardışık görüntüde bulunan yüzü bulmaya yönelik yaklaşımlar niteliklere dayanan ve imgeye dayanan metodlar olarak iki ana kategoriye ayrılabilir. Bu çalışmada iki farklı öğrenme temelli metot, öğrenme dinamikleri ve yüz bulma performansları ile karşılaştırılmaktadır. Metot-

lardan biri problem uzayının doğrusal olarak ayrılabilir olduğu öngörüsünü kullanan seyrek harman ağı yaklaşımıdır. Diğer öğrenme metodu ise problemin doğrusal olarak ayrılabilir olduğu öngörüsünden bağımsız olarak dağınık ayrışımlar sağlayan geleneksel yapay sinir ağları yaklaşımıdır. Karşılaştırmalı deney sonuçları, yüz ve yüz-olmayan uzayında, her iki metodun, yakın sınıflama performanslarına sahip olduğunu göstermektedir. Buna ek olarak, seyrek harman ağı temelli metodun, yapay sinir ağı temelli metoda göre daha hızlı eğitme ve sınama yapabildiği gözlenmiştir.

Anahtar Kelimeler: Yüz bulma, Öğrenme, Seyrek harman ağı, Yapay sinir ağları.

ACKNOWLEDGMENTS

Firstly, I would like to express my gratitude to my advisor Assist. Prof. Dr. A. Aydın Alatan for his powerful guidance, support and attention from the beginning till end. Also thanks to Assoc. Prof. Dr. Gözde Bozdağı Akar for her guidance and motivation in several phase of master education and thesis, even before the beginning. I also would like to acknowledge and show my regards to my undergraduate advisor in Başkent University, Prof. Dr. Mustafa Karaman, who let me notice the speciality of being electronics engineer and taste of research. I thank to friends in Signal Processing and Remote Sensing Laboratory especially to Uğur Murat Leloğlu, Onur Çilingir, Burcu Kepenekci, Murat Deviren for sharing their experience and knowledge. Special thanks to Ahmet Piroğlu, and Uğur Turan for their high interest and useful help during implementations. Finally, I would like to thank to Ebru Piroğlu and my family for their love and encouragement which has enabled me to finish this thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

LIST OF ABBREVIATIONS

CCT : Correlated Color Temperature

CMU-130 : Carnegie Mellon University face detection test image set

DFFS : Distance From Face Space

FERET : The Facial Recognition Technology Database

MLP : MultiLayer Perceptron

NN : Neural Network

MIT-23 : Massachusetts Institute of Technology face detection test image set

NRGB : Normalized Red Green Blue

LTU : Linear Threshold Units

PCA : Principal Components Analysis

PDBNN : Probabilistic Decision-Based Neural Networks

PDM : Point Distribution Model

SVM : Support Vector Machines

SNoW : Sparse Network of Winnows

CHAPTER 1

INTRODUCTION

Computer vision, in general, aims to duplicate (or in some cases compensate) human vision, and traditionally, have been used in performing routine, repetitive tasks, such as classification in massive assembly lines. Today, research on computer vision is spreading enormously so that it is almost impossible to itemize all of its subtopics. Despite of this fact, one can list relevant several applications, such as face processing (i.e. face, expression, and gesture recognition), computer human interaction, crowd surveillance, and content-based image retrieval. All of these applications, stated above, require face detection, which can be simply viewed as a preprocessing step, for obtaining the "object". In other words, many of the techniques that are proposed for these applications assume that the location of the face is preidentified and available for the next step.

Face detection is one of the tasks which human vision can do effortlessly. However, for computer vision, this task is not that easy. A general definition of the problem can be stated as follows: Identify all of the regions that contain a

face, in a still image or image sequence, independent of any three dimensional transformation of the face and lighting condition of the scene. There are several methods issued for this problem and they can be broadly classified in two main classes, which are *feature-based*, and *image-based* approaches. Previous research has shown that both feature-based, and image-based approaches perform effectively while detecting upright frontal faces, whereas feature-based approaches show a better performance for the detection scenarios especially in simple scenes.

In this thesis, two different image-based and learning oriented solutions are compared, in order to observe their learning dynamics and detection performances. Both of these solutions are adapted from machine learning theories and utilizing some common image processing concepts. One of the solutions named SNoW (Sparse Network of Winnows) based face detector uses a sparse feature mapping architecture, and its problem space is assumed to be linearly separable. On the other side, there exists a multilayer neural network solution, which assumes to represent any function using nonlinear feature mapping. The aim of this thesis is to examine these two supervised learning techniques with their adaptation to face detection problem. The aim of this thesis is to examine these two supervised learning techniques with their adaptation to face detection problem.

1.1 Challenges In Face Detection

Face detection is the problem of determining whether a sub-window of an image contains a face. Looking from the point of view of learning, any variation which increase the complexity of decision boundary between face and non-face classes, will also increase the difficulty of the problem. For example, adding tilted faces into the training set increases the variability of the set, and may increase the complexity of the decision boundary. Such complexity may cause the classification to be harder. There are many sources introducing variability when dealing with the face. They can be summarized as follows:

- *Image plane variations* is the first simple variation type one may encounter. Image transformations, such as rotation, translation, scaling and mirroring may introduce such kind of variations. Utilization of image pyramids with a sliding detector window is one common way to deal with such transformations for the input image. Variations in the global brightness, contrast level can also be expressed in the same category. Typical examples for such variations can be seen in Figure 1.1.

- *Pose variations* can also be listed under image plane variations aspects. However, changes in the orientation of the face itself on the image can have larger impacts on its appearance. Rotation in depth and perspective transformation may also cause distortion. The common way to deal pose variation is to isolate pose types (i.e. frontal, profile, rotated). Some

examples for such pose variations are shown in Figure 1.1.

- *Lighting variations* may dramatically change face appearance in the image. Such variations are the most difficult type to cope with due to fact that pixel intensities are directly affected in a nonlinear way by changing illumination intensity or direction. For example, when using skin color as a feature for face detection, varying color temperature [44] of the light source may cause skin color filtering to fail. Some examples for lighting variations shown in Figure 1.1

- *Background variations* is another challenging factor for face detection in cluttered scenes. Discriminating windows including a face from non-face is more difficult when no constraints exist on background. Most of the examples shown in Figure 1.1 have complex backgrounds which makes the face detection problem harder.

1.2 Scope of the Thesis

Reviewing the literature, it is quite difficult to state that there exists a complete system which solves face detection problem with all variations included. Using learning-based face detectors may handle most of the listed variation types by utilizing some of the image processing tools. In this thesis study, for decreasing variation in the training examples, pose variations are constrained

Figure 1.1: Examples of several variations.

by excluding profile and excessive rotated faces. In order to improve scale and rotation invariancy, resampled and rotated images are used during the training of both learning methods. In the evaluation, multiscale input pyramid is used to cope with large scale variations, which are not handled during training. Additionally, histogram equalization is used to handle lighting variations and to improve dynamic range in the input images.

One of the main arguments for learning networks is that the training must provide adequate *generalization*. Generalization is the expected outcome of training. However, a system, which provides enough classification during the learning stage for target classes, may result in poor generalization for the face detection in a real environment. Two different learning approaches, based on Sparse Network of Winnows [34] and feed-forward neural network [36], are examined in comparison for obtaining a better insight for the generalization problem. Each of the learning networks are trained using similar approaches to compare classification performances. Each of the resulting trained networks are used as the predictor in same multi-scale face searching system to compare generalization performances for face detection.

1.3 Outline of Thesis

Following pages of this thesis is organized as follows:

Chapter 2 introduces the literature for the face detection research, including

the reported true detection rates. In addition, some performance evaluation methodologies are introduced to clarify how the performances are quantified among different approaches.

Chapter 3 introduces how the systems learn to discriminate face and non-face examples form each other. In detail, the preparation of training data before classification stage, theoretical formation behind the examined face detectors, and finally evaluation methods including heuristics are introduced in this chapter.

Chapter 4 describes the simulation results for comparing the two learning methods for face detection.

Chapter 5 summarizes the conclusions of this thesis and gives some future directions.

CHAPTER 2

BACKGROUND ON FACE DETECTION

Over the last ten years, there has been a great deal of research concerning important aspects of face detection. Using generalized face shape rules, motion, and color information many segmentation schemes have been presented [54, 52, 33]. The use of probabilistic [41] and neural network methods [37] has made face detection possible in cluttered scenes and variable scales.

Face detection research can be heuristically classified in two main categories: feature-based approaches and image-based approaches.

According to the taxonomy in Figure 2.1, feature-based methods make explicit use of face knowledge and follow the classical detection methodology, in which low level features that are used prior to analysis mostly rely on heuristics or advance templates. The apparent properties of the face, such as skin color and face geometry, are used at different levels of the system. Since features are the main ingredients, these techniques are named as the feature-based approach. These approaches [11] have embodied the majority of interest in face detection research starting as early as the 1970s.

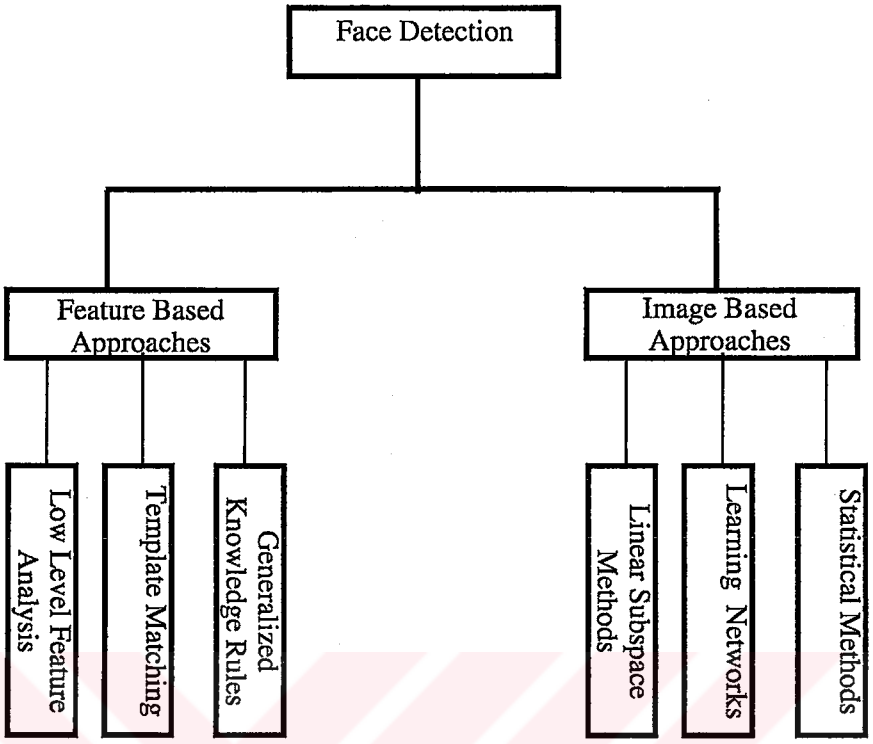


Figure 2.1: Face detection methods divided into main and sub categories.

Taking the advantage of the current advances in pattern recognition theory, image-based approaches address face detection as a general pattern recognition problem. Partly due to well known work by [45], these approaches have attracted much attention in recent years, and have demonstrated remarkable results. According to the image-based methods, face detection is a two class (face, non-face) object recognition problem which uses pure image (intensity) representations instead of abstract feature representations.

2.1 Feature-Based Approaches

Most feature-based approaches share similar consecutive steps. Usually, the first step is to make pixel level eliminations by utilizing low level feature(s)e.g. skin color filtering, edge detection. Due to the low level properties, the result that is generated in the first step is ambiguous. In the second step, visual features which are not eliminated in the first step are organized within a global face knowledge or geometry. Using this feature analysis, feature ambiguities are reduced and the locations of face and facial features are determined. The final step may involve the use of templates or active shape models.

2.1.1 Low Level Feature Analysis

2.1.1.1 Edges

As a useful primitive feature in computer vision, edge representation was applied to early face detection system by Sakai et al. [38]. Later, based on this work, a hierarchial frame work was proposed by Craw et al. [7] to trace the human head line. This approach included a line follower which is implemented with a curvature constraint. Some more recent examples of edge-based techniques can be found in [10, 13, 51, 56].

Edge detection is the important step in edge-based techniques. For detecting edges, various types of edge detector operators are used. The Sobel operator is the most common filter among others for detecting edges [13, 5].

Also, a variety of 1st and 2nd derivatives (Laplacian) of Gaussians have also been used in some approaches [38, 12]. While a large scale Laplacian was used to obtain lines [38], and steerable and multi-scale-orientation filters are preferred in [12].

In a general face detector, which uses edge representation, labeling of the edges are needed. Then the labelled edges are tried to be matched against to a face model. Govindaraju [10] accomplishes this goal by labelling edges as the left side, hairline, or right side of a front view face and then tries to match these edges against a face model by using predetermined ratio of an ideal face.

2.1.1.2 Skin Color

Human skin color has been used and proven to be effective feature for face detection, and related applications. Although skin color differs among individuals, several studies have shown that the major difference exists in the intensity rather than the chrominance. Several color spaces have been used to label skin pixels including RGB [14, 39], NRGB (normalized RGB) [8, 28, 17], HSV (or HSI) [18, 43], YCrCb [50], CIE-XYZ [4], CIE-LUV [54]. Although, the effectiveness of the different color spaces is arguable, common point of all above works is the removal of intensity component. Terrilon et al. [46] recently presented a comparative study of several widely used color spaces for face detection. In this study, the authors compare normalized TSL (tint-saturation-luminance), NRGB and CIE-xy chrominance spaces, and CIE-DSH,

HSV, YIQ, YES, CIE-L* u* v* , and CIE L* a* b* chrominance spaces by modelling skin color distributions with either a single Gaussian or a Gaussian mixture density model in each space. In their face detection test, the normalized TSL space provides the best results, however, their general conclusion is about the most important criterion for skin color filtering, which is the degree of overlap between skin and nonskin distributions in a given space (and this is highly dependent on the number of skin and nonskin samples, available).

Color segmentation can basically be performed using appropriate skin color thresholds where skin color is modeled through histograms or charts [3, 16, 43]. More complex methods make use of statistical measures that model face variation within a wide user spectrum [1, 8, 28, 53]. For instance, Oliver et al. [28] and Yang et. al. [53] employ a Gaussian distribution to represent a skin color cluster, consisting of thousands of skin color samples, taken from the different human races. The Gaussian distribution is simply characterized by its mean and covariance matrix. Any pixel color of an input image is compared with the skin color model by computing the Mahalanobis distance [9]. This distance measure gives an idea of how close the pixel color resembles the skin color of the model.

Even though color information seems to be an efficient tool for identifying facial areas, the skin color models may fail when the spectrum (correlated color temperature) of light source varies significantly. In addition, characteristics of

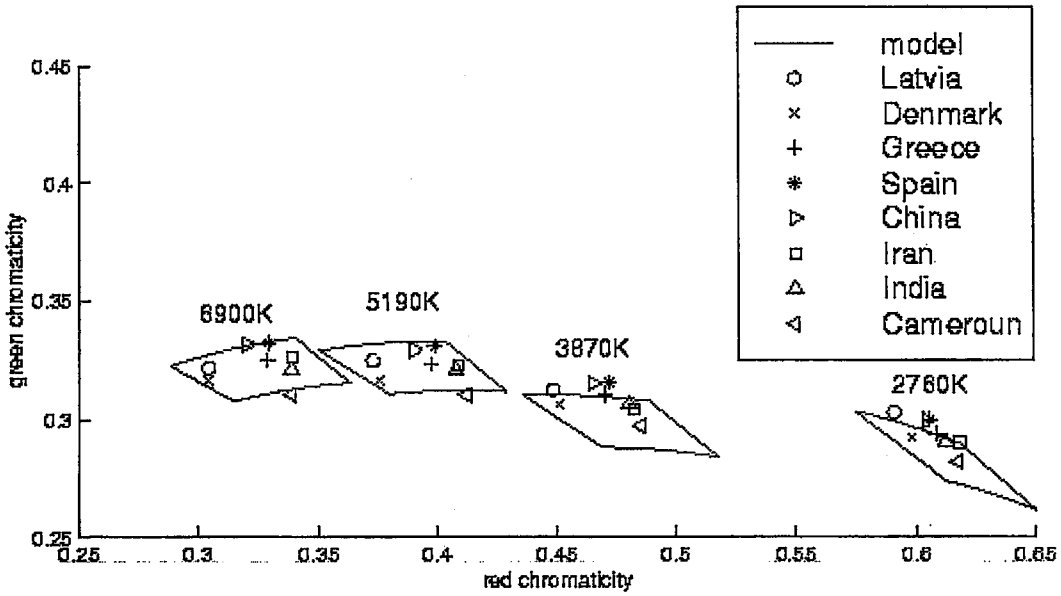


Figure 2.2: Skin color locus in NRGB space w.r.t to light source CCT [44].

acquisition device (specifically white balance) will also effect color transformation between the environment and the image. To address this problem, Störring et al. [44] modeled skin color based on the reflectance model of the skin, the camera parameters, and the spectrum of the light source. In particular, these researchers have estimated and verified skin color area in the chromaticity plane for different light sources, while the camera characteristics are given (Figure 2.2). An important conclusion of their work was the dependency of the skin color model on the spectrum of the light source and camera characteristics [44].

We have also studied the skin color information to utilize a skin color filter in the preprocessing step in face detection. However, in general, the skin color filters are constructed by using fixed boundaries (thresholds) for sample pixel

(a) (b) (c)

Figure 2.3: Responses of skin color filters for an image taken under fluorescent light.(a) Input color image, (b)HSI skin color filter response with fixed bounding thresholds, (c) NRGB skin color filter response with varying bounding thresholds.

distributions in color space. Illumination and camera parameters are omitted. Hence, the exhaustiveness in the variations for sample pixel set may bottleneck performance of the resulting skin color filter. Response of two skin color filters for same color image can be seen in Figure 2.3. Note that the HSI skin color filter with fixed thresholds is unsuccessful in determining skin color pixels. On the other side, NRGB skin color filter that is using adjustable thresholds is successful in determining skin color pixels by adding false alarms. Although, it may be more deeply experimented, we may state that a varying threshold skin color filter which includes self adaptation to image illumination properties (e.g. CCT) may result more effective skin color filtering results.

2.1.1.3 Motion

Motion information is a convenient way of locating moving objects when a video sequence is provided. It is possible to narrow face searching area uti-

lizing this information. The simplest way to achieve motion information is frame difference analysis. Accumulated frame difference is improved frame difference analysis which is used by many reported face detection research [33, 47]. Besides face region, Luthon et. al. [24], also employ frame difference to locate facial features, such as eyes. Another way of measuring visual motion is through the estimation of moving image contours. Compared to frame difference, results generated from moving contours are always more reliable, especially when the motion is insignificant [25].

2.1.2 Template Matching

Given an input image, the correlation values in predetermined standard regions, such as face contour, eyes, nose and mouth are calculated independently. Although, this approach has the simplicity, it has been insufficient for face detection since it can not handle variations in scale, rotations pose and shape. Multiresolution, multiscale, subtemplates and deformable templates have been proposed to achieve scale and shape invariance template matching [26, 20].

In [26], Miao et al. proposed a hierarchical template matching method for face detection. Initially, the input image is rotated from -20^o to 20^o degrees to handle rotation. Then, each rotated image form a mosaic at different scales in which edges are extracted using Laplacian operator. The face template consists of six facial components of two eyebrows, two eyes, nose, and mouth.

Face candidates are located by matching templates of face models represented in edges. In the final step, some heuristics are used to determine existence of a face. Experiments show better detection performance for images containing single face, rather than multiple.

Kwon et al. [20] proposed a detection method based on *snakes* and templates. In this approach, an image is first convolved with a blurring filter then with morphological operator to enhance edges. A modified *n-pixel* snake is used to find and eliminate small curve segments. Each candidate is approximated using an ellipse and for each of these candidates, a deformable template method is used to find detailed features. If a sufficient number of facial features are found, and their ratio satisfy the ratio tests based on the template, a face is considered to be detected.

Lanitis et al. [21] established a detection method utilizing both shape and intensity information. In this approach, training images are formed in which contours are manually labeled with sampled points, and vector sample points are used as shape feature vectors to be detected. They use a point distribution model (PDM) together with the principal components analysis (PCA) to characterize the shape vectors over an ensemble of individuals. A face shape PDM can be used to detect face in test images using active shape model search to estimate face location and shape parameters. The shape patch is then deformed to the average shape, and intensity parameters are

extracted. Then the shape and intensity parameters are used together for measuring euclidian distance from the faceness.

2.1.3 Generalized Knowledge Rules

In generalized knowledge-based approaches, the algorithms are developed based on heuristics about face appearance. Although, it is simple to create heuristics for describing the face,the major difficulty is in translating these heuristics into classification rules in an efficient way. If these rules are over detailed, they may come up with missed detections; on the other hand, if they are more general they may introduce many false detections. In spite of this, some heuristics can be used at an acceptable rate in frontal faces existed on uncluttered backgrounds. Yang and Huang [52] used a hierarchical knowledge-based method to detect faces. Their system consists of three level rules going from general to detail. This method does not report a high detection rate, their ideas for mosaicing (multi-resolution), and multiple level rules have been used by more recent methods.

2.2 Image-Based Approaches

In contrast to feature-based approaches, image-based approaches utilize example image representations, instead of abstract representations consisting of features. In general, image-based approaches rely on machine learning and

statistical analysis. Face detection is a two class (face, non-face) classification problem which rely on learned characteristics generally in the form of distributions. The specific need for a face knowledge is avoided by formulating the problem as a learning paradigm to discriminate a face pattern from a non-face pattern.

Image-based approaches can be better understood by considering statistical supervised pattern recognition. A raw image can be taken as random variable x, and this random variable is characterized by class-conditional density functions $p(x/face)$ and $p(x/non-face)$. If the dimensionality of x was not so high, a Bayesian or maximum likelihood classification would be possible. Hence, image-based approaches utilize more complex techniques such as subspace representations and learning networks to overcome the high dimensionality of the problem space.

Most of the image-based approaches apply a window scanning technique for detecting faces. The window scanning algorithm employs an exhaustive search of the input image for possible face locations at all scales, but there are variations in the implementation of this algorithm for almost all the image-based systems. Typically, the size of the scanning window, the subsampling rate, the step size, and the number of iterations vary depending on the method proposed and the need for a computationally efficient system.

2.2.1 Linear Subspace Methods

In the late 1980s, Sirovich and Kirby [42] developed a technique using PCA to efficiently represent human faces. Given a set of face images, the proposed technique obtains the principal components of the distribution of faces, expressed in terms of eigenvectors (of the covariance matrix of the distribution). Then, each individual face in the set can be approximated by a linear combination of the largest eigenvectors (*eigenfaces*) corresponding to largest eigen values, using appropriate weights.

Later, Turk and Pentland [47] improved this technique for face recognition. Their method takes the advantage of the distinct nature of the weights of eigenfaces for individual face representation. Since, face reconstruction, by using its principal components, is an approximation, a residual error is defined in the algorithm as a preliminary measure of "faceness".This residual error which they termed "distance-from-face-space" (DFFS), gives an indication of face existence through the observation of global minima in the distance map.

Pentland et al. [31] later proposed a facial feature detector, using DFFS generated from eigenfeatures (*eigeneyes, eigennose, eigenmouth*), which are obtained from various facial feature templates in a training set. The feature detector is better while accounting for features under different viewing angles, since features of different discrete views were used during the training. The performance of the eye locations was reported to be 94% with 6% false positive

rate in a database of 7562 frontal face images in front of a plain background.

More recently, Moghaddam and Pentland have further developed this technique within a probabilistic framework [27]. Unlike the usual PCA framework, they did not discard the orthogonal complement of face space. This leads to uniform density assumption of the face space. Hence, so they developed a maximum likelihood detector which takes into account both face space and its orthogonal complement to handle arbitrary densities. They report a detection rate of 95% on a set of 7000 face images for detecting the left eye. Compared to the DFFS detector, the results were significantly better. On a task of multi-scale head detection of 2000 face images from the FERET [32] database which includes mug shot faces in front of a uniform background, the detection rate was 97%.

PCA is an intuitive and appropriate way of constructing a subspace for representing an object class in many cases. However, for modelling the variety in face images, PCA is not necessarily optimal. Face space might be better represented by dividing it into subclasses. Several methods have been proposed which are mostly based on some mixture of multidimensional Gaussians. This approach was first applied to face detection by Sung and Poggio [45]. They modelled the empirical distribution of face and non-face patterns using six multi dimensional Gaussian clusters, as it is shown in Figure 2.4. Their system includes two main components, distribution models for face/non-face

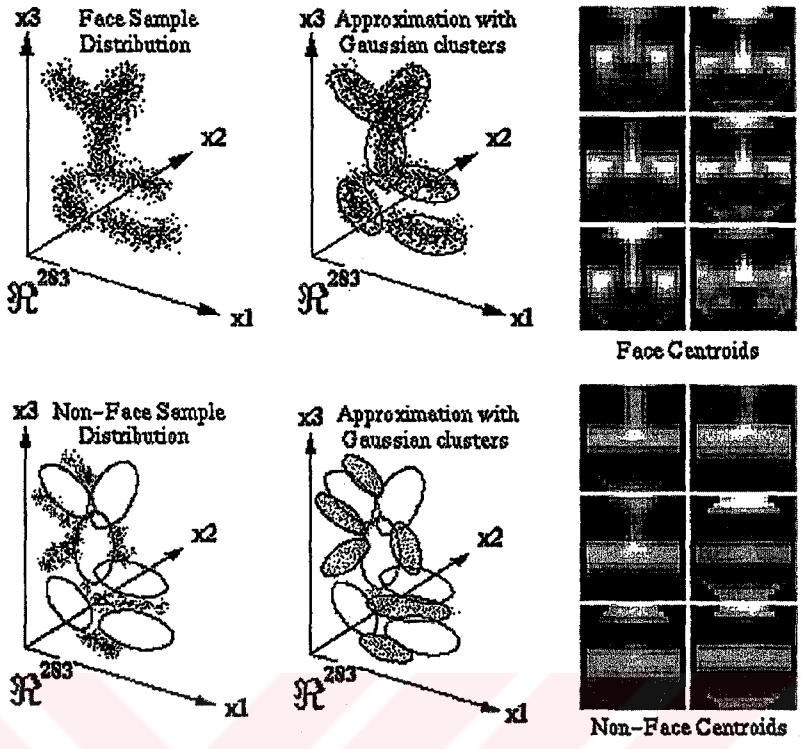


Figure 2.4: Distribution-based canonical face model. Top Row: Empirical distribution of face patterns using six multi-dimensional Gaussian clusters, whose centers are as shown on the right. Bottom Row: Sample of non-face patterns using six multidimensional Gaussian clusters to help localize the boundaries of the face distribution. The final model consists of six Gaussian face clusters and six non-face clusters. Sung and Poggio [45] ©1998 IEEE.

patterns and a multilayer perceptron (MLP) classifier. Training of MLP was established using 47316 examples. In order to collect non-face patterns, they have introduced a method called as *bootstrap*. By using bootstrap method, they train network using a small set of non-face examples. Then run the face detector and add false detected windows into set of non-face examples. They have reported 81.9% true detection rate in a database of 23 cluttered scene images including 149 faces with 13 false detections.

2.2.2 Learning Networks

Since face detection can be understood as a two class pattern recognition problem, several neural network-based approaches have been introduced for solution. A review of the neural network-based face detection methods can be found in Viennet et al. [49].

Other than basic multilayer perceptron approaches (MLP) [2, 15], the first advanced neural approach which reported significant results on a large, complex dataset was by Rowley et al. [37]. The system incorporates face knowledge in a retinally connected neural network as shown in Figure 2.5. The neural network is designed to look at windows of 20 x 20 pixels. There is one hidden layer with 26 units, where 4 hidden units connected to 10 x 10 pixel subregions, 16 units connected to 5 x 5 subregions, and 6 units connected to 20 x 5 pixels via input units. The input window is pre-processed through lighting correction (a best fit linear function is subtracted) and histogram equalization. This pre-processing method was adopted from Sung and Poggio's system mentioned earlier and it is illustrated in Figure 2.6. A major problem which arises with window scanning techniques, is overlapping detections. Rowley et al. [36] deals with this problem through two heuristics:

1. **Thresholding:** the number of detections in a small neighborhood surrounding the current location is counted, in output pyramid which is including both location and scale and if it is turn out to be above a

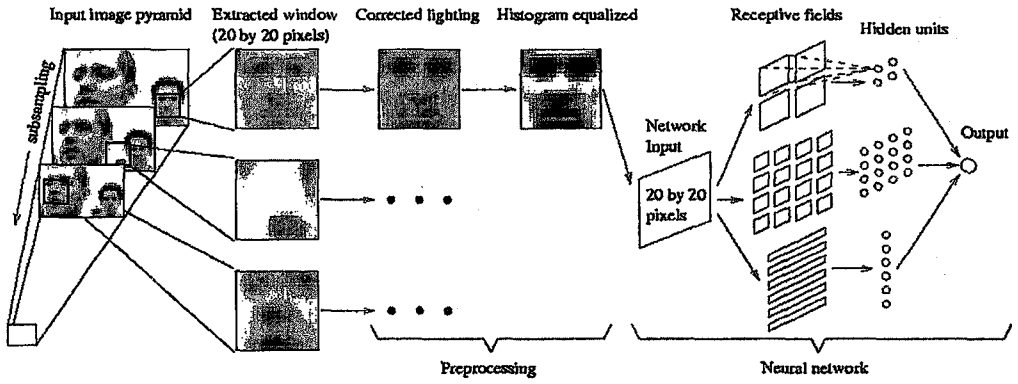


Figure 2.5: The system by Rowley et al. [37] (©1998 IEEE).

certain threshold, a face is assumed to be present at this location.

2. **Overlap elimination:** when a region is classified as a face by using heuristic thresholding, then overlapping detections are likely to be false positives and removed.

Moreover, they also train multiple neural networks and combine the output with an arbitration strategy (ANDing, ORing, voting, or a separate arbitration neural network) [36].

In Lin et al. [22], a fully automatic face recognition system is proposed based on probabilistic decision-based neural networks (PDBNN).A PDBNN is a classification neural network with a hierarchical modular structure. Instead of converting input image to a raw vector, they preferred to use features based on intensity and edge information.

Sparse Network of Winnows (SNoW), which is a new learning architecture in visual domain, is applied to face detection by Roth et al. [34]. The SNoW

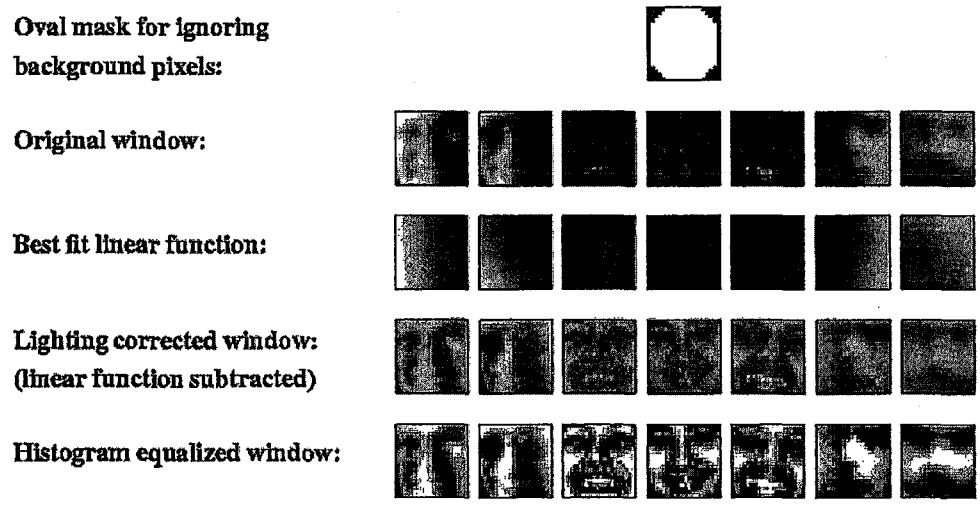


Figure 2.6: The preprocessing method applied by Rowley et al. [37] (©1998/2001 IEEE). A linear function is fit to the intensity values in the window and then subtracted from the image. Finally, histogram equalization is applied to improve contrast.

system, applied to face detection is a learning network consisting of two linear threshold units (LTU) (representing the subnetworks for face and non-face). The two target subnetworks operate on an input space of Boolean features. The best performing system derives features from 20 x 20 subwindows in the following way: for 1 x 1, 2 x 2, 4 x 4, and 10 x 10 subwindows, compute (position x intensity mean x intensity variance). This gives Boolean features in a 135424-dimensional feature space, since the mean and variance have been quantized into a predefined number of classes. The LTUs are separate from each other and are sparsely connected over the feature space. During training, weights linked to face and non-face subnetworks are promoted or demoted utilizing Winnow update algorithm [23] according to mistakes made in classification. Similar to the previously mentioned methods, Roth et al. also use the

bootstrap method of Sung and Poggio for generating training samples and pre-process all images with histogram equalization. Moreover, window scanning technique is used in multiscales during evaluation stage as similar to previous mentioned methods. This method is one of the underlying methods used in this thesis, hence will be examined in detail in the next chapter.

2.2.3 Statistical Approaches

There are several statistical approaches for face detection. Some of the proposed systems are based on information theory [6], a support vector machine [30] and Bayes [41] decision rule.

Colmenarez and Huang [6] proposed a system based on Kullback relative information (Kullback divergence). This divergence is a nonnegative measure between two probability density functions for a random process X^n. During training, for each pair of pixels in the training set, a joint-histogram is used to create probability functions for the classes of faces and non-face. Since pixel values are highly dependent on neighboring pixel values, X^n is treated as a first order Markov process and the pixel values in the gray-level images are re-quantized to four levels. The authors use a large set of 11 x 11 images of faces and non-face for training, and the training procedure results in a set of look-up tables with likelihood ratios. In order to further improve performance, pairs of pixels which contribute poorly to the overall divergency are dropped from the

look-up tables and not used in the face detection system. This technique is further improved by including error bootstrapping which is described earlier, and later the technique was also incorporated in a real-time face tracking system [6].

Another major approach is, Support Vector Machines (SVM) can be considered as a new paradigm to train polynomial function, or neural network classifiers. While most methods training classifier (e.g. Bayesian, neural networks) based on minimizing of training error *empirical risk*, SVMs exist on another principle called *structural risk minimization*, which aims to minimize an upper bound on the expected generalization error. A SVM classifier is a linear classifier. And, the optimal hyperplane is defined by a weighted combination of a set of training (support) vectors, and is chosen to minimize expected classification error of the preciously unseen test patterns. Osuna et al. [29] develop an efficient method to train a SVM for large scale problems, and applied it to face detection. SVMs also applied to the problem in wavelet domain to detect pedestrians and faces [30]. Kumar and Poggio [19] recently incorporated Osuna et al.'s SVM algorithm in a system for real-time tracking and analysis of faces. They apply SVM algorithm on segmented skin regions of the input images to avoid exhaustive scanning.

As another approach, Schneiderman and Kanade [40, 41] describe two face

detectors based on Bayes decision rule (presented as a likelihood ratio test as

$$\frac{P(image|object)}{P(image|non-object)} > \frac{P(non-object)}{P(object)}$$

If the likelihood ratio (left side) of above equation is greater than the right side, then it is decided that an object (a face) is present at the current location. The advantage of this approach is the optimality of the Bayes decision rule [9], if the representations are accurate. In the proposed face detection system [40], the posterior probability function is derived based on a set of modifications and simplifications. The resolution of a face image is first normalized to 64 x 64 pixels, and the face images are decomposed into 16 x 16 subregions while there is no modelling of statistical dependency among the subregions. Afterwards, the subregions are projected onto a 12-dimensional subspace by using local eigenvector coefficients constructed by PCA, and the entire face region is normalized to have zero mean and unit variance. In addition, the authors also used wavelets to obtain image visual attributes instead of eigenvectors in a consecutive work [41]. By the help of this approach, a view-based detector is developed with a frontal view detector and a right profile detector (to detect left profile images, the right profile detector is applied to mirror reversed images). Some examples of outputs which are processed using wavelets can be seen in Figure 2.7.

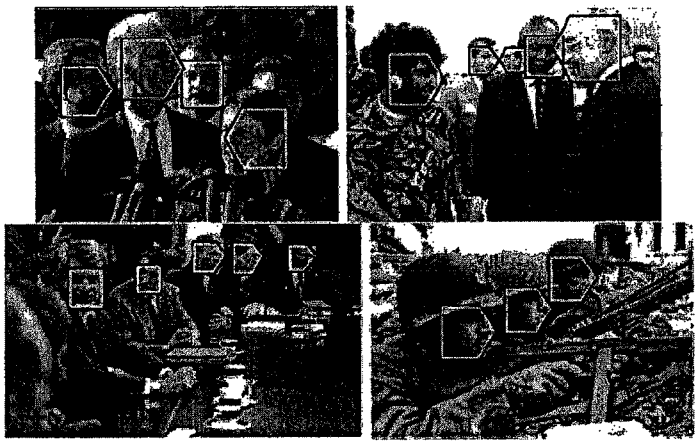


Figure 2.7: Face detection examples from Schneiderman and Kanade [41] (©2000/2001 IEEE).

2.3 Performance Evaluation: Benchmark Sets and Counting Criteria

In order to make a fair comparison among different methods, it is important to count true and false detections rates for the same benchmark test image sets. A performance comparison is given in Table 2.1 for some of the major methods that are experimented on the same test sets. These test sets are in gray scale and have been collected at CMU by Rowley et al. [37] and at MIT by Sung and Poggio [45]. The CMU test set also includes the MIT set. Since some systems report results while excluding some of the images from these sets, it will be appropriate to consider each set in two separate columns of Table 2.1. Thus, both of these two test sets are divided into two, forming the following four sets:

CMU-130: The entire CMU set with 130 images with 507 labelled faces (which includes the MIT set). The images are of varying size and quality including faces of varying size, illumination, and cluttered backgrounds.

CMU-125: The CMU set excluding five images which contain hand-drawn and cartoon faces; a total of 125 images with 483 labelled faces. There are additional faces in this dataset (for a total of over 495 faces), but the ground-truth for only 483 of the images has been established based on excluding some of the occluded faces and non-human faces. However, the ground-truth for the 483 images established by Rowley et al. does not necessarily refer to the same 483 faces in all the reported results since at least one of the papers [34] indicate that they have labelled the set of faces themselves.

MIT-23: The entire MIT dataset (also known as set B of the CMU set) with 23 images. The number of labelled faces was originally 149 by Sung and Poggio (which is the number used in the reported results in [45], but it was changed to 155 (which is the number used in the reported results in [37, 29, 22]) when included in the CMU set.

MIT-20: The MIT set excluding 3 images with hand-drawn and cartoon faces giving a total of 20 images with 136 labelled faces.

Table 2.1: Results Reported in Terms of Percentage True Detection with Number of False Positives on the CMU and MIT Test Image Sets

Method	CMU130	CMU-125	MIT-23	MIT-20
Schneiderman et al. (*PCA*) [40]		94.4%, 65		
Schneiderman et al. (*Wavelet*) [41]		90.2%, 110		
Yang et al. [55]		92.3%, 82		89.4%, 3
Roth et al. [34]		94.8%, 78		94.1%, 3
Rowley et al. [37]	86.2%, 23		84.5%, 8	
Colmenarez & Huang [6]	93.9%, 8122			
Osuna et al. [29]			74.2%, 20	
Lin et al. [22]			72.3%, 6	
Sung & Poggio [45]			79.9%, 5	

It is quite difficult to come up with comparative conclusions from Table 2.1, because there exist several problems due to different assumptions about some experimental concepts. There are 4 major experimental details, which differ between these methods and make the comparison difficult:

Description of Face: Since the CMU dataset contain a large number of faces, because there seems to be some disagreement on the number of faces the dataset actually contains. This is due to the fact there are human faces, animal faces, cartoon faces, and line-drawn faces present in the dataset.

Counting detections: Since a detection is a placement of a window at a location in the input image, there is a need of decision on how accurate this placement needs to be. Yang et al. [6] make this decision based on the rule "...a detected face is a successful detect, if the subimage contains eyes and mouth". For all the systems using the window scan-

ning technique, a face might be detected at several scales and at several positions close to each other. Rowley et al. [37] address this problem by using two merging heuristics, while few others seem to care about this.

ROC curve: It is well known that some systems have high detection rates while others have a low number of false positives. Most systems can adjust their thresholds depending on the constraints in their application. This can be reported in terms of a Receiver Operator Characteristics (ROC) curve to which shows the correct detections against false positives.

Description of Training: Some systems use a large training set and generate additional training samples by rotating, mirroring, and adding noise, while others have a smaller training set. Moreover, the proposed bootstrap training algorithm by Sung and Poggio [45] is implemented in some of the systems.

In our experiments, we will try to make these point clear, so that a realistic comparison can be made with provided in the future.

CHAPTER 3

FACE DETECTION USING LEARNING NETWORKS

In general, learning in face detection rely on machine learning and statistical analysis. Here, face detection is a two class (face, non-face) classification problem which stems from the learned characteristics generally in the form of distributions. However, this does not mean all of learning-based approaches oblique to estimate distributions of target classes.

Following the statistical pattern recognition aspects, a raw image can be taken as random variable x and this random variable is tried to be characterized by class-conditional density functions $p(x|face)$ and $p(x|non-face)$. If the dimensionality of x were not so high, a Bayesian or maximum likelihood classification would be possible. Although, Kanade et al. developed [40] a Bayes decision rule in combination with PCA and wavelet features, our main focus is to have a framework which approaches face detection as a problem of dividing face and non-face with an appropriate hypersurface. This surface

division may be constructed by a linear surface using a linear function to obtain a hyperplane boundary, or by arbitrary surface divisions using non-linear functions of a neural classification system.

The most of efficient learning methods in the literature, either make use of a linear decision surface over the feature space [27, 34, 29], or arbitrary nonlinear surfaces [37, 22, 45]. In both of these cases, a priori information about the distributions of data to be classified may not be essential, although they are implicitly used during the training stage.

The first learning method composing this thesis work is the *Sparse Network of Winnows* (SNoW), which is one of the linear classifiers that utilize linear threshold functions in a sparse feature space to provide a linear division surface for the target classes. Method assumes linear separability of the problem by the help of the special sparse architecture of the feature space. The method uses linear threshold functions of degree one, (which can also be in higher order polynomial functions resulting higher order surfaces) to separate face and non-face examples by a linear hyperplane. The second learning method is based on the well known *Neural Networks*, utilizes a multilayer perceptron (MLP) solution, for the problems which linear discriminant functions are insufficient in converging to an efficient linear decision surface. In MLP, a complex assembly of perceptron units are used to increase representative capabilities. It is assumed that the nonlinear mapping of the input representation to hidden

layers will enable arbitrary separability of the face and non-face examples, if each hidden perceptron unit achieves an individual separation.

One of the expected outcome at the end of learning is the *generalization.* Learning is achieved using a set of training data, whereas the performance of the system is evaluated on previously unused data. Thus, a method provides successful classification in training data does not require to be successful in typical real world data. Thus, classification performance or convergence time during training, may simply result from *memorization* capability of the system towards the input training data. Hence, training performance may not provide any hypothesis for the test data performance. Hence, for face detection, or for any classification, generalization will be the measure for the performance on the real world examples, provided unknown distribution of training data sampled from the same unknown distribution of test data. More discussion on generalization will be provided in next sections and one can find more comprehensive analysis in [9].

The following sections provide detailed information about Sparse Network of Winnows-based and Neural Network-based face detectors especially, the fundamental details and their architectures. Moreover, for constructing an efficient target training set, active learning, which is shared among various learning-based face detection methods, is also explained. Multiscale face searching and complementary heuristics that might be used (which is also

shared among learning-based face detection methods) during evaluation is explained to finalize face detection system implementation.

3.1 Sparse Network of Winnows-Based Face Detection

The SNoW (Sparse Network of Winnows) is an example for a single layer network which consists of linear threshold functions. The SNoW architecture (Figure 3.1) is a sparse network of linear units over a common predefined or incrementally learned feature space [34]. Nodes in the input layer represent binary relations for input and these relations are used as input features. An input image is mapped into binary features which are active in it. This representation of the input is feed into the input layer and propagates through the network. Target nodes (face, non-face) are connected via weighted links to the input nodes (features). According to the stated information up to this point, the system may resemble a multi layer neural network. However, it should emphasized that, SNoW does not include hidden layers with nonlinear transfer functions, although, a linear unit may be implemented in conjunction with other units.

Let $V_t = \{i_1, \ldots, i_n\}$ be set of features that are active (exist) in an example and are linked to the target node t. Then the linear unit corresponding to t is

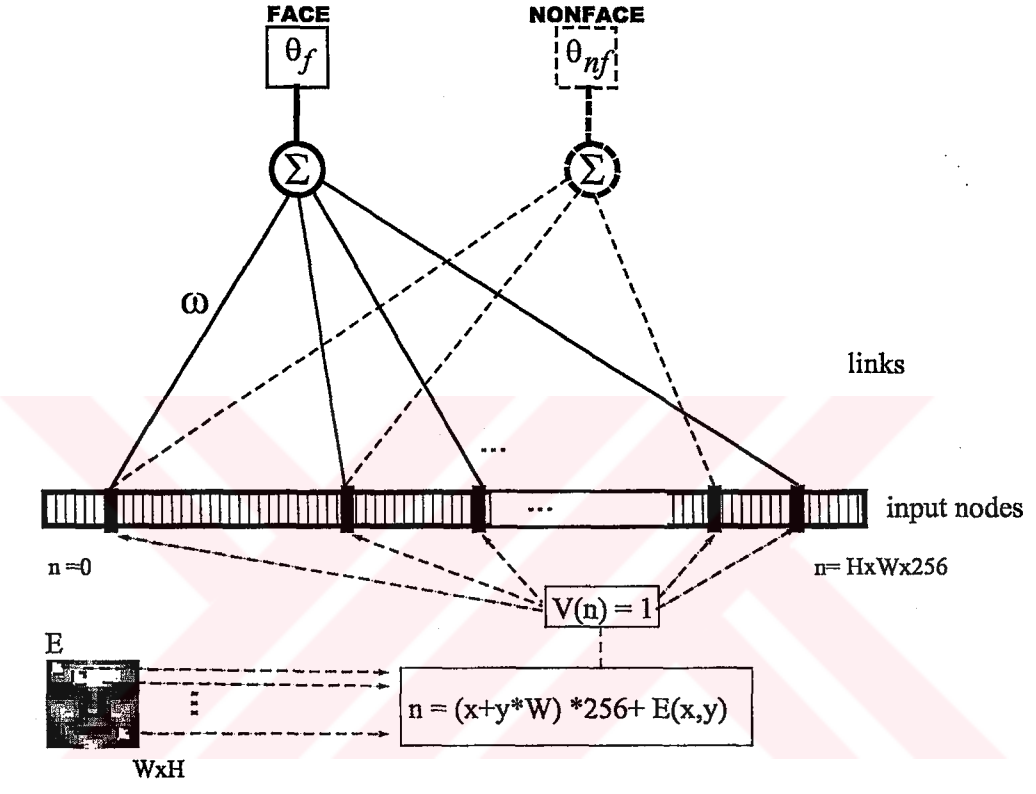


Figure 3.1: SNoW learning architecture for face detection. Given an example E with dimensions $W * H$, a pixel intensity value at locations (x, y) have a intensity value $(0 < E(x, y) < 255)$ which is encoded as $n = (x + y * W) * 256 + E(x, y)$. This representation ensures different points in *Position* x *Intensity* space are mapped to different features indexes in range $[0, W * H * 256]$. For example, in an example with dims $20 * 20$ will only have 400 active features $(V(n) = 1)$ out of 102400 potential features.

active if and only if Equ. 3.1 is true

$$\sum_{i \in V_t} \omega_i^t > \theta^t \tag{3.1}$$

where ω_i^t is the weight of the link connecting the ith feature to the target node t, and θ^t is the threshold for the target node t.

In the SNoW-based face detector, a single SNoW unit which includes two subnetworks, one for each of targets (face, non-face) is used. A training example is feed through each of these subnetworks: an example, labeled (L) as face is a positive example for the face subnetwork and a negative example for the non-face subnetwork, and vice-versa. Learning is on-line and mistake-driven which is supplied through Littlestone's Winnow update rule [23]. Winnow update is a multiplicative rule which is suitable in problems when the set of features are not known a priori. This mechanism is implemented via the sparse architecture of SNoW. That is, (1) input features are allocated in a data driven way such that: an input node for the feature i is initialized only if the feature i is active in the input image and (2) a link (i.e., a non-zero weight) exists between a target node t and a feature i if and only if i has been active in an image labelled as $L = t$.

In addition to θ_t at target t, the Winnow multiplicative update rule has two update parameters to support mistake-driven learning [23]. A promotion parameter $\alpha > 1$ and a demotion parameter $0 < \beta < 1$ are being used to update current representation of the target t (set of active weights linked to

target t) when a mistake in prediction is made. Thus, if the result of Equ. 3.1 is false (prediction is 0), and the received label is true ($L = t$), then active weights for features in the current example V_t are promoted in a multiplicative way:

$$\forall i \in V_t \quad \omega_i^t \leftarrow \alpha.\omega_i^t.$$

If the result of Equ. 3.1 is true (prediction is 1), and the received label is false ($L \neq t$), then active weights for features in the current example V_t are demoted in a multiplicative way:

$$\forall i \in V_t \quad \omega_i^t \leftarrow \beta.\omega_i^t.$$

All other weights (weights which are irrelevant) remain unchanged.

The key feature of SNoW is that the number of examples it requires to learn a linear function grows linearly with the number of relevant features and only logarithmically with the total number of features [35]. This is an important property in face detection domain in which number of potential features is vast, while a relatively small portion is active (relevant) with co-occurrence. Winnow is known to learn efficiently any linear threshold function and to be robust in the presence of various kinds of noise and in cases where no linear threshold function can make perfect classification [23].

Once target subnetworks (face and non-face) have been learned and the network is being evaluated, a decision mechanism is employed, which selects the dominant active target node in the SNoW unit via a winner-take-all mechanism

to produce a final prediction. Thus, for a given previously unseen example, let $V_{face} = \{i_1, \ldots, i_n\}$ and $V_{non-face} = \{i_1, \ldots, i_n\}$ be respective sets of features that are active (exist) for $face$ and $non - face$ targets, the result of Equ. 3.2 determines prediction such that: 1 for a face and 0 otherwise.

$$\sum_{i \in V_{face}} \omega_i^{face} > \sum_{i \in V_{non-face}} \omega_i^{non-face} \tag{3.2}$$

The SNoW-based face detector uses Boolean features that encode the positions and intensity values of pixels. Let the pixel at (x,y) location of an image with width w and height h have intensity value $I(x, y)$ $(0 < I(x, y) < 255)$. This information is encoded as a feature indexed as $256 * (y * w + x) + I(x, y)$. This representation ensures that different points in the *position* $*$ *intensity* space are mapped to different features. Furthermore, the feature indexed $256 * (y * w + x) + I(x, y)$ is active if and only if the intensity in position (x,y) is $I(x, y)$. In our experiments, the values for w and h are 20 and each face or non-face example has been normalized to an image of 20 x 20 pixels. Note that although the number of potential features in our representation is 102400 (20*20*256), only 400 of those are active (present) in each example, and it is plausible that many features will never be active. Since the algorithm's complexity depends on the number of active features in an example, rather than the total number of features, the sparseness also ensures efficiency.

3.2 Neural Network-Based Face Detection

When data is not linearly separable, a linear discriminant function may not be able to converge to an efficient decision surface. However, there still exist a nonlinear solution to obtain arbitrary decision region, using a suitable nonlinear function. Unfortunately, finding such nonlinear function can be problematic. Thus, multilayer neural networks or multilayer perceptron basically implement linear discriminants in a space where the inputs have been mapped in a nonlinear way.

Network architecture or topology plays an important role in neural network classification. In other words, problem specifications will determine the optimal topology. Knowledge of the problem domain stemming from heuristics can be adapted to neural architectures. Thus, neural network-based face detector should have a special topology tailored for face detection. Our implementation of neural network topology, which can be seen in Figure 3.2, is a simulated version of neural network-based face detection system developed by Rowley et. al. [37]. In this system, an input image is first separated into overlapping horizontal, vertical and square slices and then passed through the network. This special input slicing was chosen to allow the hidden units to represent features that might be important for face detection. Slicing is pre-arranged to create overlapping horizontal slices that include face hair connection, eyebrows, eyes, nose, mouth and chin regions. Every pixel in horizontal, vertical, and square

slices are presented to the input neurons in different blocks, and this input neurons are linked to the corresponding hidden neurons. Each of the hidden neurons in different blocks have three copies in order to improve training and evaluation performance.

As in general feedforward multilayer neural networks, the network that is implemented has two primary modes of operation:

Feedforward Operation: An input example is sliced and each slice is feed into associated input units (neurons) to propagate through hidden units. Each hidden unit computes weighted sum of its net inputs

$$net_j = \sum_{i=0}^{d} \omega_i . x$$

to form its scalar net input function. Each hidden unit emits an output that is a nonlinear function of its net activation.

$$y_j = f(netj)$$

We have used a tanh function which provides an output in the range $[-1.0, 1.0]$. Same above calculations are made for block output units and final output unit. When feedforward operation is complete, this means a prediction is made: 1.0 is the desired output for a face and -1.0 is for a non-face example.

Learning (Backpropagation) Operation: The difference between the output and desired output for the label t is corresponding to training

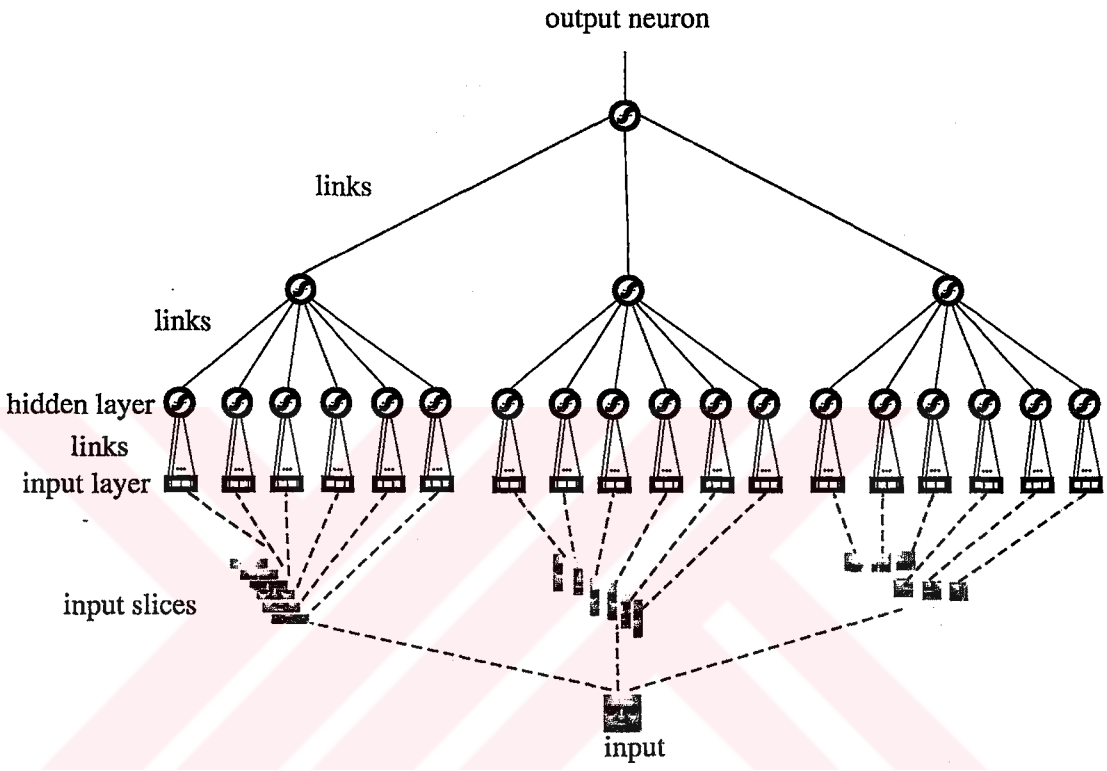


Figure 3.2: Neural network learning architecture for face detection. The topology is a simulated version of neural network-based face detection system developed by Rowley et. al. [37]. An input example is first separated into overlapping horizontal, vertical and square slices than passed through the network. Slicing is pre-arranged to create overlapping slices that include face hair connection, eyebrows, eyes, nose, mouth and chin regions. Each of the hidden neurons in different blocks have three copies (not illustrated above) in order to improve training and evaluation performance.

error, which is a scalar function of the weights and minimized when the neural network outputs match the desired outputs. Thus, learning is done by adjusting the weights to reduce the error. The implemented learning algorithm for the system is standard backpropagation algorithm with *momentum* [9]. Backpropagation is one of the most popular methods in supervised learning for multilayer neural networks. The addition of momentum is for enriching the backpropagation to add a fraction α of the previous weight update and it is based on the notion from physics. Let $\Delta\omega(m) = \omega(m) - \omega(m-1)$ and let $\omega_{bp}(m)$ be the calculated update according to the gradient descent. Then

$$\omega(m+1) = \omega(m) + (1-\alpha)\omega_{bp}(m) + \alpha\Delta\omega(m-1)$$

represents learning using backpropagation with momentum [9].

3.3 Active Learning: Bootstrap Method

Training both of these learning-based face detection systems is a complicated task, due to the difficulty in characterizing non-face examples. It may be easy to collect a representative set for face patterns, but unfortunately the same is not true for non-face patterns. Since the main goal is classification and not an accurate representation, one should not necessarily need a representative set of non-face images. In order to achieve accurate classification, it is more important to use non-face samples that are most likely to be misclassifications

for the face. In other words, to collect examples which do lie in close to the boundary covering face samples will be sufficient for the classification. The common solution for this problem is named *"bootstrap"* and developed by Sung and Poggio [45]. The *bootstrap* method reduces the need for a huge sized non-face training data using selectively adding images to training set as training progresses. Thus, instead of trying to collect non-face examples before the simulations, they may be collected during training stage. The bootstrap algorithm can be summarized as follows:

1. Create an initial set of non-face examples by collecting or randomly generating N non-face images.

2. Train[1] the learning network by using this non-face examples and the face examples.

3. Run the system with any image[2] which do not contain a face. Collect the subimages which are misclassified as faces and add them in non-face training example set.

4. Go to step 2 and repeat until the system do not give false alarms on the bootstrap set of non-face examples.

[1]Selection of training in this step may vary among implementation. In consecutive bootstraps, an iterative or non-iterative training may be chosen according to learning speed in method.

[2]It might be useful to construct a large set of scene images which do not contain any face.

If the learning system is stable (converging to decision surface), a non-face training set with bootstrap additions will be formed, after a sufficient number of bootstrap iterations. The aim is to make those gathered non-face examples to encase face sample boundary, enabling an efficient classification. In order to ensure this, a generalization curve may be extracted by evaluating the system on a small test set (*validation set* [9]) to reach a desired maximum for the ratio true detections/false detections. Also, searching for a minimum in total training error in a validation set may be used for this purpose as well.

It is valuable to note that the quality of face training set will effect the generalization and efficiency of the learning. For example, utilization of the unconstrained variations may increase the complexity of the sample boundary unless boundary becomes smoother. However, some constrained variations (i.e, rotations, scale, lighting) must be included to avoid specific fitting (learning may lack of generalization) of learning function to the sample space. Therefore, a preprocessing applied to both face and non-face example may increase the quality of learning. This preprocessing is generally employed by using manually cropped faces, histogram equalization, and background masks.

3.4 Multiscale Face Searching

Multiscale face searching by using up/down sampled input pyramids is a technique for providing scale invariant face detection. Since learning networks

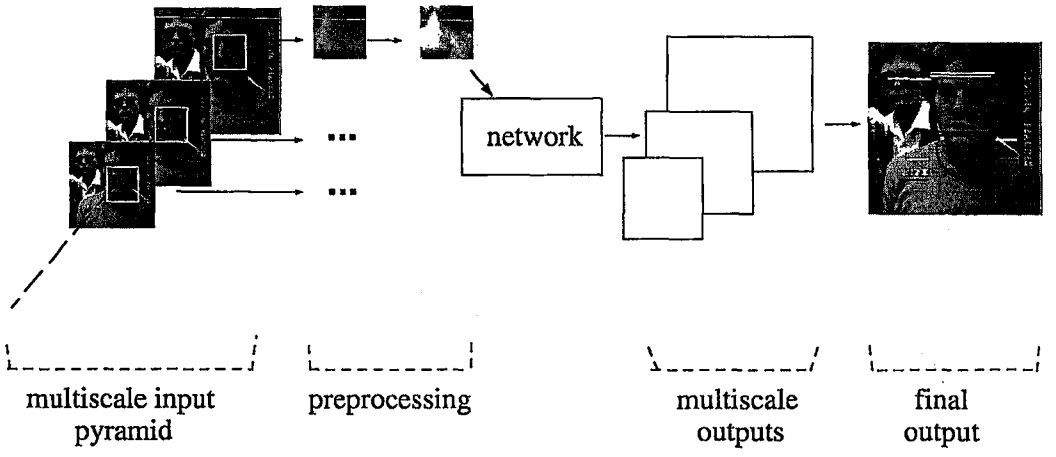


Figure 3.3: The detection system in the evaluation phase. An input image pyramid formed by subsampling image with reduced ratio width and height. A window with dimensions WxH is extracted for each scale and location and histogram equalization is performed before passing it to the network. Predictions for each scale and locations is again forming an output pyramid. This output pyramid is combined to finalize the detection.

architecturally receive an example in a fixed size window, scale variations included in the face training set is not sufficient to provide scale invariancy in real world evaluation scenarios. Thus, in order to detect a face at any location in the input, the network evaluation is applied to every pixel location in the image using a sliding window. Moreover for detecting images larger than the network window size, the input image is repeatedly resampled to form an input pyramid to apply network evaluation at each scale. An example input image pyramid can be seen in Figure 3.3.

After a window of size WxH (same as in the training) is extracted from a particular location and at one of the scales of the input pyramid, it is preprocessed and passed to the network. In this thesis, the preprocessing is histogram

equalization for both of the learning algorithms. Then, learning algorithm makes its own feature mapping (SNoW), or region separation (Neural network) to evaluate input and make a prediction. This procedure is shown in Figure 3.3 using a module showing evaluation of the learning algorithms: accepting an input of fixed sized window extracted from the input image, and producing a prediction output as a binary decision (face,non-face) or in the values, showing the strength of prediction, which can be used for further processing.

3.4.1 Complementary Heuristics

The examples in Figure 3.3 show the output of the learning networks with a number of false positives. Hence, there are strategies developed [36]to improve the reliability of the detection:

Elimination of Weak Detections: An unwanted false detection may occur when a face detected at multiple positions and scales or a non-face window is detected as a face. In order to overcome this problem, a local thresholding can be applied. Counting the number of neighboring detections (binary outputs) or summing neighbor detection strengths (output strengths) may give an output for network confidence in the location that is considered. A local thresholding applied to detection confidence in particular detection location may clean up all the other weak detections. However, when two neighboring detections have the same strength

there still exists a problem which is not solved in this thesis. Note that, the expectation of confidence is creating a trade off between number of true and false detections.

Elimination of Overlapping Detections: If a particular location is identified as a face, then all other detections overlapping it are likely to be false extra detections, and therefore be eliminated.

3.5 Summary

In this chapter, a detailed information about learning and evaluation mechanisms in face detection is provided. From learning aspects, face detection is a two class classification problem. In order to give an insight to this problem, two learning algorithms, which have different assumptions about the nature of the problem are examined. In SNoW-based face detector, problem space is assumed to be linearly separable. For the solution, a linear threshold function for both face and non-face classes is offered, and this is supported by a sparse feature mapping architecture. On the other side, regardless of linear separability assumption, there exists a neural networks solution which assumes to represent any function using arbitrary decision surfaces utilizing nonlinear activation functions. Learning procedures and architectural mechanisms are explained in this section. Apart from these further theoretical limits, mistake bounds, generalization dynamics can be further found in appropriate

references. Empirical observations and experiments to find out the learning dynamics for these learning algorithms in comparison are presented in the following chapter.

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, experimental results concerning training and evaluation will be introduced. In training experiments, each of the learning methods provide several features enabling a control for learning efficiency. For testing, besides using some different constants, both of the methods share the same testing setup.

4.1 Training Experiments

In following subsections, experimental training setups for both learning methods, will be explained in order to provide complete picture about the experiments. Setup for training data is exactly equal for both of the learning methods, and will be explained in subsection: training examples.

4.1.1 SNoW Experimental Setup

General SNoW learning paradigm has the following features which may effect learning efficiency during training:

Eligibility Threshold is used to increase input's confidence. Since, feature allocation is achieved in a data driven way, a threshold is necessary to determine the number of times a feature must occur before it may be enabled for allocation. In our experimental setup this value is set as 2.

Prediction Threshold is used to constrain prediction in evaluation. Since, winner take all mechanism is used for predictions, this value may be used to filter for ensuring winner node's confidence. In experimental setup this value is set as 0.

Training Epochs specifies the number of passes through training examples. Multiple passes through training data may improve the learning in network. In experimental setup this value is set as 2.

Target Threshold and Initial Weights must be properly chosen to enable classification. Thus, initial (nonzero) weights must be set according to the number of active features and target threshold values. In these experiments, 20x20 examples are used, which results in a number of 400 active features. Target threshold is set as 1.0 for both (face and non-face) nodes, which lead us to randomize initial weights around 1/400.

4.1.2 Neural Network Experimental Setup

It is not easy to train a neural network without fine tuning training constants. Since feedforward operation is not ensuring convergence to a separation plane

(gradient descent may be unstable), several constants are experimented and finalized to achieve learning. Hence, a special training method proposed by Rowley et. al. [36] is used to overcome neural network training complications. Since the neural network tries to learn classification from large number of examples, for each iteration (batch) only a fixed number of examples from whole face and non-face example sets are used to reduce time consumption. This, value is set as 100 for both sets. The architecture of the neural net, which is illustrated in Figure 3.3, is implemented which has 3 different blocks corresponding to three types of input slices. In each of these three blocks there exist 100 input, 18 (6x3) hidden, and a single output neuron. Network has total 100x18x3+1 internal connections.

Initial Weights choosing initial (nonzero) weights is a complicated task, can effect network in an unexpected way; even an instability may occur. Hence, the initial weights for input neurons and hidden neurons chosen according to number of their connections [9]. In this setup, input layer weights are randomized around 1/40, while hidden neurons randomized around 1/0.099.

Learning Rate and Momentum Constant are critical constants, and used in weight update formula. A large learning rate constant may result to instability, and a small one may increase the convergence time, whereas momentum constant enables a control on the convergence path

by defining rate of momentum preservation. In our experiments learning rate η and momentum constant α were chosen as 0.1 and 0.97, respectively.

Outputs Constants, The network is trained to produce a (1.0) for a face and a (-1.0) (*tanh* activation function) for a non-face example. During training a prediction error that is greater than 0.02 is counted as a misclassification.

4.1.3 Training Examples

In order to construct face example training set, 1600 images collected from Olivetti [48], FERET [32] and TUBITAK-BILTEN face image databases which have variations in pose (but only frontal), facial expression and illumination conditions, are utilized. Each face image is manually cropped, scaled and aligned to fill 20x20 window including eyes, nose, and mouth. For increasing the data size and variations, random subsamplings and rotations (up to ± 15 degrees) are applied to each face image which finally result in a total of 13157 examples. Finally, for all the examples in the set, histogram equalization is applied to provide wide dynamic range. Some typical examples from the face example set is seen in Figure 4.1.

Bootstrap method is used to collect non-face examples, due to previously explained reasons (see Chapter 3:Active Learning). Training started with 2500

Figure 4.1: Random chosen face examples from face training set.

randomly collected non-face images, and then totally 255 landscape scene images which do not include any face examples are used in bootstrap operation. In bootstrap process, all candidate non-face examples are equalized in order to improve dynamic range. In the finalized non-face example set there are 14206 non-face examples. In the bootstrap process, for each scene image 150 random addition are allowed for the non-face example set. Averaged non-face example images are shown in Figure 4.2. Note that, averaged non-face examples are close to face examples, which is implying power of the bootstrap process that we are collecting non-face examples which are too close to face example set in the whole space.

4.1.4 Training

During training, two experiments are conducted in order to understand dynamics of learning process for both methods.

Figure 4.2: Averaged non face examples from non face training set.

The first phase of experiment is achieved by collecting state data that is obtained as number of misclassifications for SNoW and epoch errors for neural network during consecutive iterations. Figures 4.4 and 4.3 show number of misclassifications and epoch error vs. training iterations, for SNoW and neural network, respectively. These plots also include data coming from iterations during bootstrap process which can not be considered apart from the training phase. During bootstrap process, misclassifications for SNoW monotonically increase, because non-face examples become larger and closer to face space. After bootstrap process is complete, learning becomes more effective, resulting a sharp decrease for number of misclassifications in the training set, as it finally converges to a separation plane. Note that, convergence to a perfect separation plane takes below 30 training iterations in SNoW, after non-face example set is completed by several bootstrap additions. For the neural network, epoch error vs. training iteration graph is plotted in Figure 4.3, because a limited

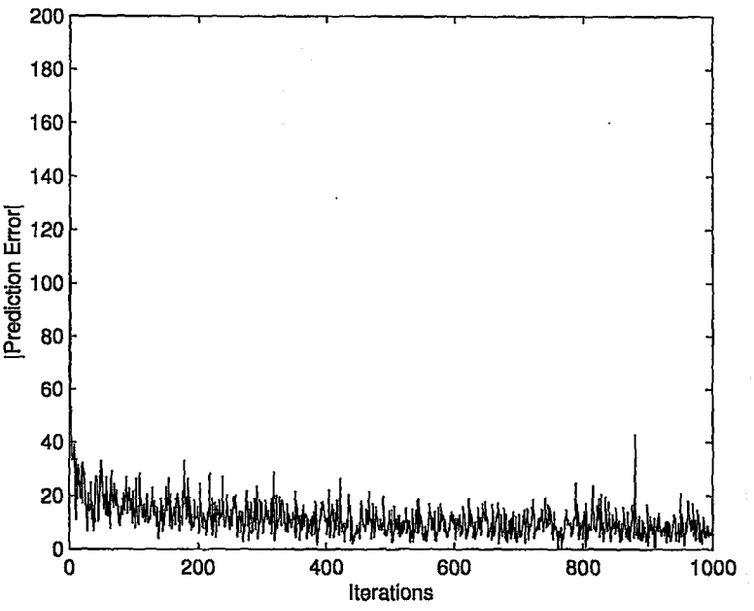


Figure 4.3: Training record for neural network-based face detector showing training error vs. iterations.

100 face and 100 non-face examples are randomly selected and trained from the whole set in each epoch. This has a consequence of enormously large iterations for enabling separation. It is nearly impossible to obtain a perfect classification (no misclassifications case). In epoch error vs. iteration graph, in each epoch (100 by 100 training) neural network starts with a peak error which is decreasing with an expected trace along consecutive iterations.

In the second phase of experiments, another result is obtained during the training of both learning methods, for examining their generalization abilities. During training iterations, networks may overfit the training examples and may loose generalization capability, as the iterations grow. Thus, one should not, simply use number of misclassifications as a learning criteria. For ensur-

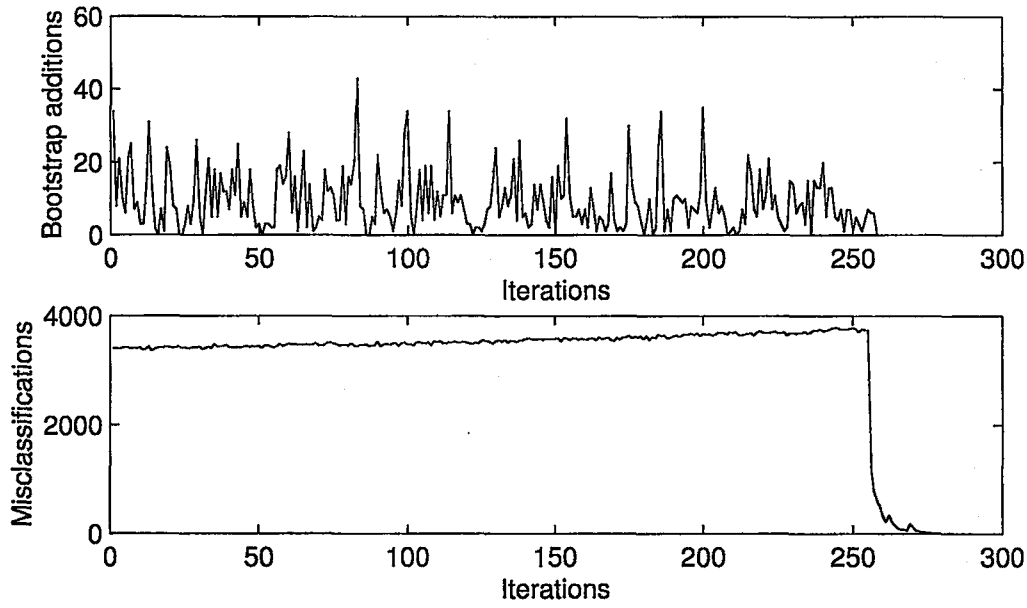


Figure 4.4: Training record for SNoW-based face detector showing number of misclassifications vs. iterations.

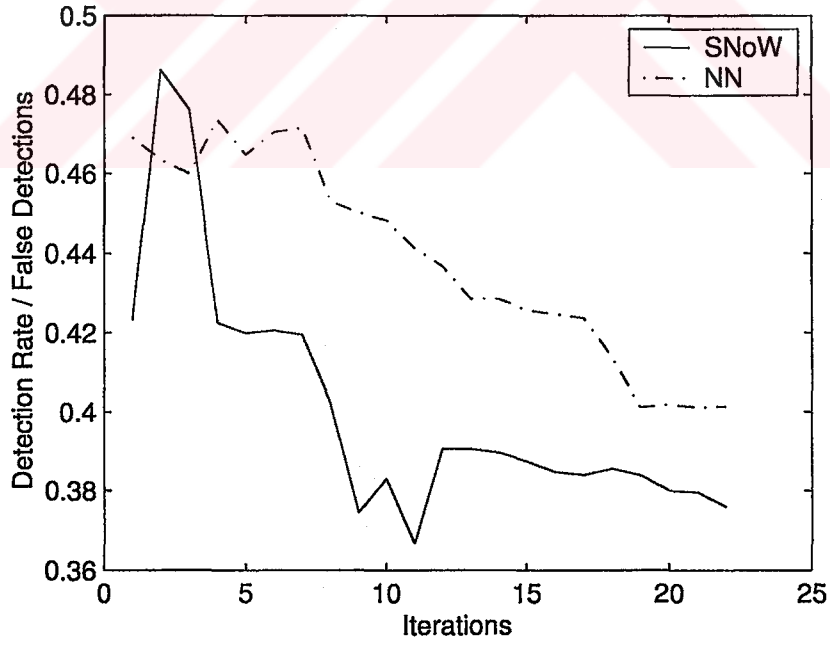


Figure 4.5: Detection Rate/False Detections vs iteration for SNoW and neural networks.

ing maximum generalization, a validation set (small test set including 3 images including 30 labelled faces) is tested at each iteration, and detection rate over number of false detections is plotted against iteration number. Figure 4.5 is showing true detection rate over number of false detections vs. iteration number for SNoW and neural network. The iteration number (an its corresponding network state), giving the maximum generalization value, is used in further experiments to evaluate the detection performances. It can be seen that peak generalization value for SNoW is slightly greater than NN. On the other side, NN has an greater average generalization among the consecutive iterations.

4.2 Evaluation Experiments

In the evaluation experiments, the maximum generalization training states (previously determined) are used for each network. The same experimental setup is used for both learning methods. Although, some internal thresholds differ, each of the networks use the same multiscale search mechanism. Each of the algorithms is tested using 8 step multiscale pyramid (by subsampling ranging from 1.2 to 0.4). In order to benchmark evaluation performances of the networks, MIT-23 dataset is used [45]. Table 4.1 shows evaluation performance results (detection rate with number of false positives) on MIT-23 which contains 23 images with all frontal 157 faces. An additional test set is created which aims to test scale, lighting and rotation invariancy. This test set

has 3 images which a total of 39 faces (including scale, lighting and rotation variations) which is overlayed in front of a background. In order to fully understand the quantitative meaning of detection rates and false detections, one must remember that networks have to make predictions on total 10000 windows for a small image of 100 by 100 pixels. This value is above 10 million for MIT-23 and near a million for our small test set. Additionally, a detected window is counted as a true detection if eyes, nose, and mouth of the person are all included in this window.

Table 4.1 shows detection rates, when heuristic processing is applied with different threshold values. A heuristic threshold $th = (2, 2)$ means that, one use heuristics applied within 2 pixel neighborhood with threshold value of 2.It is observable that an increasing heuristic threshold decreases the number of false positives as well as the true detection rate. According to Table 4.1, NN true detection rates are higher than SNoW, while introducing more false detections as well. Hence, SNoW generalization performance is slightly greater than NN (true detection rate over false detections), as it is expected from training experiments. It is obvious that, one can tune several internal evaluation parameters to produce greater detection rates for both of the networks, while considering trade off between true detection rate and number false detections. Some examples of the detection results for SNoW and neural network can be seen in Figures 4.6 through 4.8. In order to give an idea about response speeds,

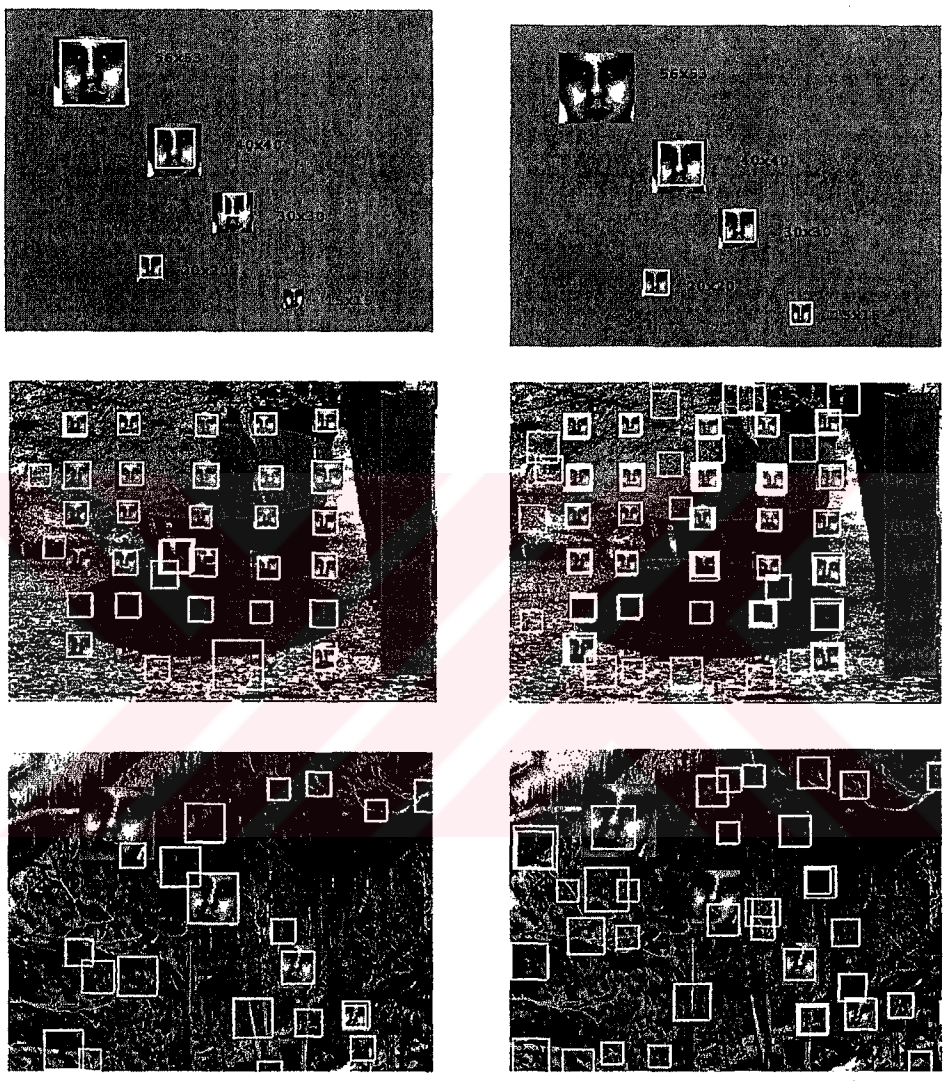
Table 4.1: Results in Terms of Percentage True Detection with Number of False Positives on the CMU set

Method	Scale&Rotation&Lighting Test	MIT-23
SNOW	87.1% f.d = 29	65.9% f.d = 330
SNOW + $Heuristics_{th=(2,2)}$	61.5% f.d = 0	34.1% f.d = 151
NN	89.1% f.d = 55	80.2% f.d = 765
NN+ $Heuristics_{th=(2,2)}$	69.2% f.d = 2	75.1% f.d = 515

Table 4.2: Evaluation Response Timings of SNoW and NN for a 20x20 Input

Method	Evaluation Time (ms)
SNoW	0.19 ms
NN	2.05 ms

Table 4.2 shows evaluation timings for both methods to make a prediction for a 20x20 window on a pentium III 866MHz PC. Having the advantage of being simple and sparse, SNoW is faster than NN, which makes it preferable for online applications.



(a) (b)

Figure 4.6: a)SNoW and b)NN detection examples for scale lighting and rotation variances test.

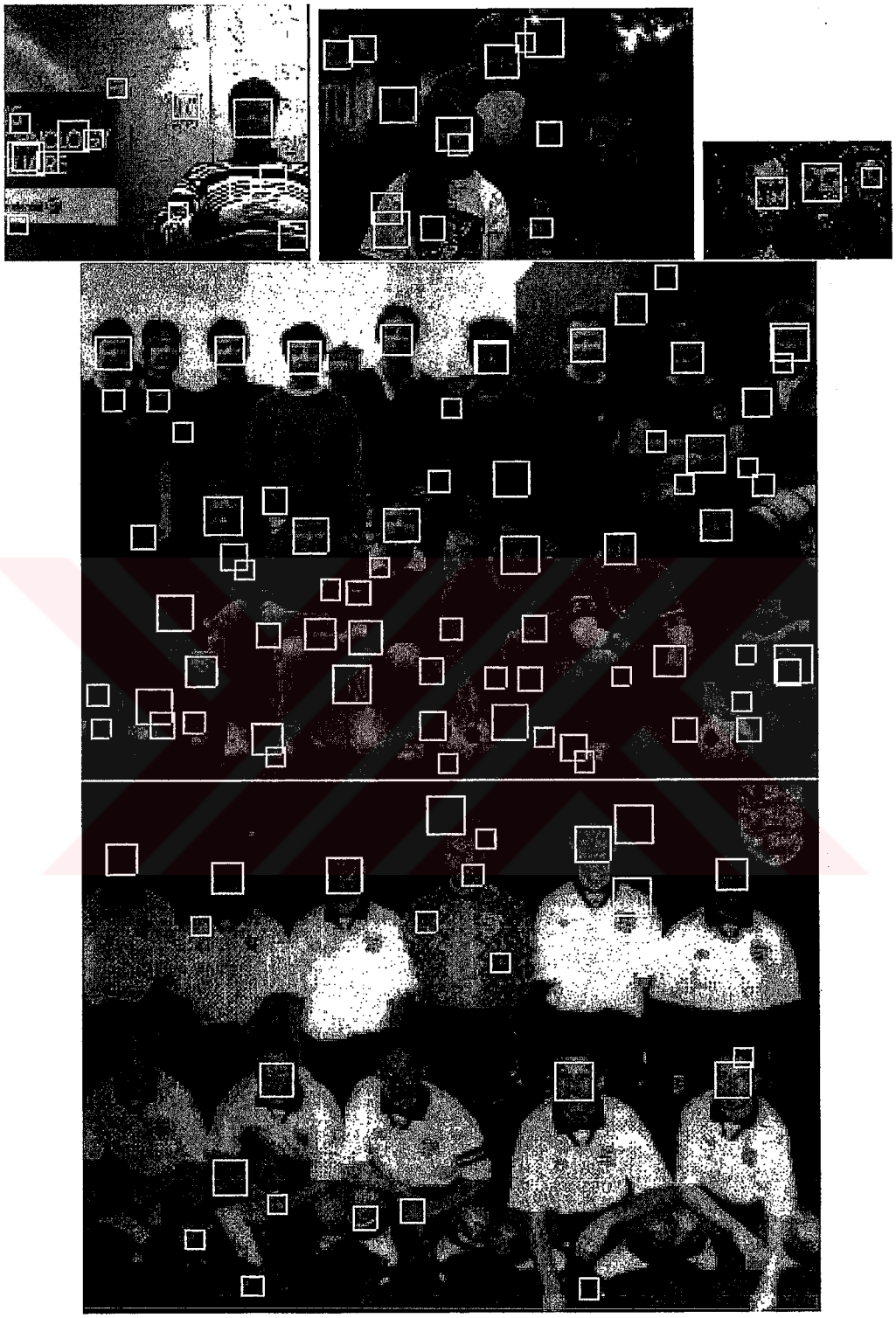


Figure 4.7: SNoW detection examples from MIT-23, $Heuristics_{th=(0,0)}$

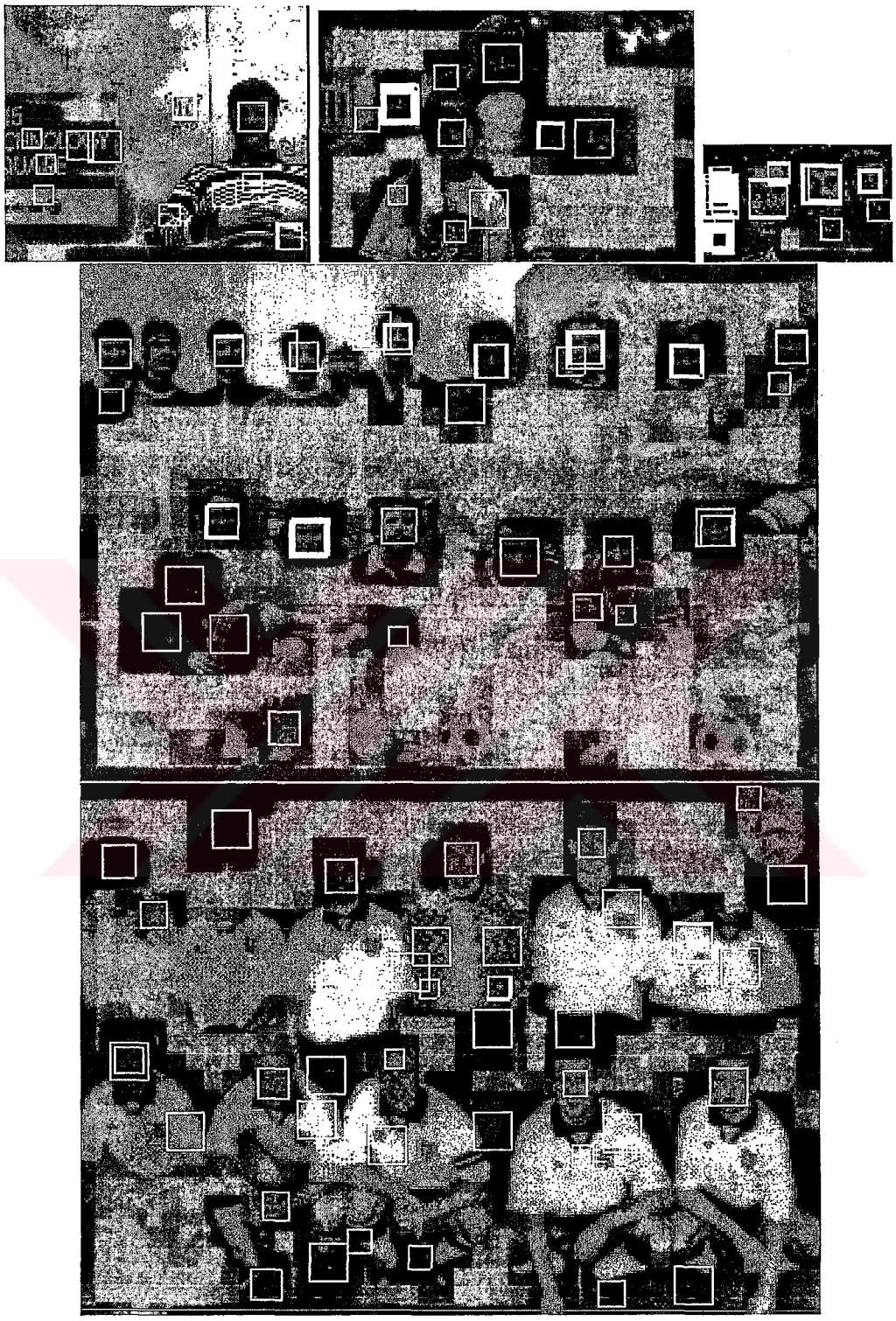


Figure 4.8: NN detection examples from MIT23, $Heuristics_{th=(0,0)}$

CHAPTER 5

CONCLUSIONS

5.1 Summary

This thesis is mainly focused on face detection using learning networks which simply take face detection as a supervised pattern recognition problem. Prior face knowledge and any other knowledge-based heuristics are ignored and a prediction is only made according to the learned characteristics from training images. Additionally, theoretic and common practical adaptations of learning networks for face detection task are studied and main steps for a face detection learning scheme are outlined ranging from training through testing.

In order to make a comprehensive study, SNoW and neural network-based face detection systems are implemented, and experimented in training and evaluation phases with a comparative insight. The first network, sparse network of winnows, is an example for linear classifiers which assumes problem space as linearly separable. Second network, feed-forward neural network, is an example for arbitrary nonlinear classifiers which have no assumptions on

the linearity of the space.

5.2 Conclusions

Experimental results in the training phase showed that both SNoW and NN face classifiers are able separate examples included in face and non-face training data from each other. Thus, both networks supplied an implicit surface division with the help of adjustments for internal weights. Experiments concerning efficiency of these divisions, showed trained networks have similar generalization (true detection rate over total number of false detections) abilities. While peak generalization value of SNoW classifier is slightly higher than its neural network counterpart. As the iterations grow, both networks yield a remarkable decrease in generalization abilities. Hence, it is possible to state networks become to overfit training data and this yields decrease in generalization abilities as the iterations grow.

In the evaluation experiments, NN face classifier hits higher true detection rates with respect to the SNoW, while introducing more false detections as well. True detection rates are 80.2% with 765 false detections and 65.9% with 330 false detections for NN and SNoW respectively. NN face classifier reaches 15% higher detection rate with the trade of for more than twice false detections. In order to decrease number of false detections, heuristics are applied to raw prediction outputs of both networks. This process is slightly

lowered number of false detections, while decreasing true detection rate as well. Hence, NN face classifier conserves higher true detection rates and higher false detections count with respect to its SNoW counterpart. Additionally, having the advantage simplicity and sparseness, SNoW is quite faster than NN, which makes it preferable for real-time applications.

5.3 Future Work

There exist several directions for researches in the future. Further improvements can be made in terms of training examples for achieving a better generalization. Additionally, more informative features can be used instead of intensity values. For example, a simple quantization for gray levels may also achieve better evaluation results. Moreover, utilization of color values instead of gray levels may also improve learning mechanism, but in that case lighting variations must be sufficient large in the training sets.

The internal parameters for both networks can be further optimized to improve generalization abilities. The training and evaluation complexities for neural network also needs improvement and may be reduced by optimizing network topology.

Sparse network of winnows is applied to several natural language processing tasks which are reported in literature, and it is recently introduced to visual domain with SNoW-based face detector. Internal dynamics may be further

studied in order to achieve improvements, and to design different adaptations for other visual tasks.

REFERENCES

[1] A. Albiol, C. A. Bouman, and E. J. Delp. Face detection for pseudosemantic labeling in video databases. *Proceedings International Conference on Image Processing*, 1999.

[2] G. Burel and D. Carel. Detection and localization of faces on digital images. *Pattern Recognition Letters*, 15:963–967, 1994.

[3] J. Cai and A. Goshtasby. Detecting human faces in color images. *Image and Vision Comput.*, 18:63–75, 1999.

[4] Q. Chen, H. Wu, and M. Yachida. Face detection by fuzzy matching. *Proc. of th 5th IEEE Conf. on Computer Vision*, 1995.

[5] J. Choi, S. Kim, and P. Rhee. Facial components segmentation for extracting facial feature. *Proc. 2.nd Int. Conf. on Audio- and Video-based Biometric Person Authentication*, 1999.

[6] A. J. Colmenarez and T. S. Huang. Face detection with information-based maximum discrimination. *IEEE Proc. of Int. Conf. on Computer Vision and Pattern Recognition*, 1997, 6.

[7] I. Craw, H. Ellis, and J. R. Lishman. Automatic extraction of face-feature. *Pattern Recognition Lett.*, pages 183–187, Feb 1987.

[8] J. L. Crowley and F. Berard. Multi-model tracking of faces for video communications. *IEEE Proc. of Int. Conf. on Computer Vision and Pattern Recognition*, Jun 1997.

[9] R. O. Duda, P. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience, 2 edition, 2001.

[10] V. Govindaraju. Locating human faces in photographs. *Int Jour. Computer Vision*, 19, 1996.

[11] E. Helomas and B. K. Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83:236–274, 2001.

[12] R. Herpers, K.-H. Lichtenauer, and G. Sommer. Edge and keypoint detection in facial regions. *IEEE Proc. of 2nd Int. Conf. on Automatic Face and Gesture Recognition*, pages 212–217, 1996.

[13] A. Jacquin and A. Eleftheriadis. Automatic location tracking of faces and facial features in video sequences. *IEEE Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition*, Jun 1995.

[14] T. S. Jebara and A. Pentland. Parameterized structure form motion for 3d adaptive feedback tracking of faces. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 144–150, 1997.

[15] P. Juell and R. Marsh. A hierarchical neural network for human face detection. *Pattern Recognition Letters*, 29:781–787, 1996.

[16] S. Kawato and J. Ohya. Real-time detection of nodding and head-shaking by directly detecting and tracking the between-eyes. *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 2000.

[17] S. Kim, N. Kim, S. C. Ahn, and H. Kim. Object oriented face detection using range and color information. *Proc. of the 3rd Int. Conf. on Automatic Face and Gesture Recognition*, pages 76–81, 1998.

[18] R. Kjedsen and J. Kender. Finding skin in color images. *Proc. of the 2nd Int. Conf. on Automatic Face and Gesture Recognition*, pages 312–317, 1996.

[19] V. Kumar and T. Poggio. Learning-based approach to real time tracking and analysis of faces. *Proc. 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, 2000.

[20] Y. H. Kwon and N. da Vitoria Lobo. Face detection using templates. *Int Conf. on Pattern Recognition*, pages 764–767, 1994.

[21] A. Lanitis, C. J. Taylor, and T.F Cootes. An automatic face identification system using flexible appearance models. *Image and Vision Computing*, 13:393–401, 1995.

[22] S.-H. Lin, S.-Y. Kung, and L.-J. Lin. Face recognition/detection by probabilistic decision-based neural network. *IEEE Trans. Neural Networks*, 8:114–132, 1997.

[23] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear threshold learning using winnow. *Machine Learning*, 2:285–318, 1988.

[24] F. Luthon and M. Lievin. Lip motion automatic detection. *Scandinavian Conference on Image Analysis*, 1997.

[25] S. McKenna, S. Gong, and H. Liddell. Real-time tracking for an integrated face recognition system. *Workshop on Parallel Modelling of Neural Operators*, Nov 1995.

[26] J. Miao, B. Yin, K. Wang, L. Shen, and X. Chen. A hierarchical multiscale and multiangle system for human face detection in a complex background using gravity-center template. *Pattern Recognition*, 32:1237–1248, 1999.

[27] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(1), 1997.

[28] N. Oliver, A. Pentland, and F. Berard. A real-time face and lips tracker with facial expression recognition. *IEEE Trans. on Pattern Recognition*, 33:1369–1382, 2000.

[29] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. *IEEE Proc. of Int. Conf. on Computer Vision and Pattern Recognition*, 6, 1997.

[30] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *Proc. of 6th Int. Conf. on Computer Vision*, pages 555–562, 1998.

[31] A Pentland, B. Moghaddam, and T. Strarner. View-based and modular eigenspaces for face recognition. *IEEE Proc. of Int. Conf. on Computer Vision and Pattern Recognition*, pages 84–91, 1994.

[32] J. P. Phillips, H. Wechsler, J. Huang, and P. Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image Vision Computing*, 16(5), 1998.

[33] M. J. T. Reinders, P. J. L. van Beek, B. Sankur, and J. C. A. van der Lubbe. Facial feature localization and adaptation of a generic face model for model-based coding. *Signal Processing: Image Communication*, pages 57–74, 1995.

[34] D. Roth, M.-H. Yang, and N. Ahuja. A snow-based face detector. *Advances in Neural Information Processing Systems*, 12, 2000.

[35] D. Roth, M-H. Yang, and N. Ahuja. Learning to recognize 3d objects. *To Appear, http://l2r.cs.uiuc.edu/ danr/publications.html*, 2001.

[36] H. A. Rowley. *Neural network-based face detection.* PhD thesis, Carnegie Mellon University Computer Science Dep., May 199.

[37] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. Pattern Analysis Machine Intelligence*, 20:23–38, 1998.

[38] T. Sakai, M. Nagao, and T. Kanade. Computer analysis and classification of photographs of human faces. *Proc. First USA-Japan Computer Conference*, 1:2–7, 1972.

[39] S. Satoh, Y. Nakamura, and T. Kanade. Name-it:naming and detecting faces in news videos. *IEEE Multimedia*, 6:22–35, 1999.

[40] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 6, 1998.

[41] H. Schneiderman and T. Kanade. A statistical model for 3d object detection applied to faces and cars. *IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[42] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journ. Optical Society America*, 4:519–524, 1987.

[43] K. Sobottka and I. Pitas. Extraction of facial regions and features using color and shape information. *Proc. of Int. Conf. on Pattern Recognition*, 1996.

[44] M. Störring, H. J. Andersen, and E. Granurn. Skin color detection under changing lighting conditions. *7th Symposium on Intelligent Robotic Systems*, pages 20–23, 1999.

[45] K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions On Pattern An. and Ma. In.*, 20(1):39–51, Jan 1998.

[46] J.-C. Terrillon, M. Shirazi, H. Fukamachi, and S. Akamatsu. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. *in Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 2000.

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

[48] Oli vetti & Oracle Research Laboratory. The olivetti & oracle research laboratory face database of faces. http://www.camorl.co.uk./facedatabase.html.

[49] E. Viennet and F. Fogelman Soulié. *Connectionist methods for human face processing, in Face Recognition: From Theory to Application.* Springer-Verlag, Berlin/New York, 1998.

[50] H. Wang and S.-F Chang. A highly efficient system for automatic face region detection in mpeg video. *IEEE Trans. on Circuits and Systems for Video Technology*, pages 615–628, 1994.

[51] J. Wang and T. Tan. A new face detection method based on shape information. *Pattern Recog. Lett*, 21:463–471, 2000.

[52] G. Yang and T. S. Huang. Human face detection in complex background. *Pattern Recognition*, 27:53–63, 1994.

[53] J. Yang and A. Waibel. A real-time face tracker. *IEEE Proc. of the 3rd Workshop on Applications of Computer Vision*, 1996.

[54] M.-H. Yang and N. Ahuja. Detecting human faces in color images. *Proc. of the 1th IEEE Conf. on Image Processing*, pages 127–130, 1998.

[55] M.-H. Yang, N. Ahuja, and D. Kriegman. Face detection using mixtures of linear subspaces, in proceedings. *4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, 2000.

[56] A. L. Yuille, P. W. Hallinan, and D. S. Cohen. Feature extraction from faces using deformable templates. *Int. J. Computer Vision*, 8:99–111, 1992.