Human Face Processing Techniques With Application To Large Scale Video Indexing

LE DINH DUY

DOCTOR OF
PHILOSOPHY

Department of Informatics,

School of Multidisciplinary Sciences,

The Graduate University for Advanced Studies (SOKENDAI)

2006 (School Year)

September 2006

HUMAN FACE PROCESSING TECHNIQUES WITH APPLICATION TO LARGE SCALE VIDEO INDEXING

BY

LE DINH DUY

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Department of Informatics of
The Graduate University for Advanced Studies (SOKENDAI), 2006

Tokyo, Japan

Abstract

Human faces play an important role in efficiently indexing and accessing video contents, especially in large scale broadcasting news video databases. It is due to faces are associated to people who are related to key events and key activities happening from all over the world. There are many applications using face information as the key ingredient, for example, video mining, video indexing and retrieval, person identification and so on. However, face appearance in real environments exhibits many variations such as pose changes, facial expressions, aging, illumination changes, low resolution and occlusion, making it difficult for current state of the art face processing techniques to obtain reasonable retrieval results.

This thesis studies human face processing techniques whose target is to efficiently apply to a general framework for large scale video mining and indexing. In this framework, faces firstly are extracted, filtered and normalized from video sequences by using a fast and robust face detector. Next, similar faces are grouped into clusters. Then, these face clusters are labeled by the person names extracted from the video transcripts.

To extract faces from video, we propose a multi-stage approach that uses cascades of classifiers to yield a coarse-to-fine strategy to reduce significantly detection time while maintaining a high detection rate. This approach is distinguished from previous work by two features. First, we use a cascade of AdaBoost classifiers that is trained to be invariant to translation up to 25% of the original window size to detect quickly face candidate regions. Second, we use SVM classifiers which reuse the features selected by AdaBoost in the previous stage for robust classification and simple training. Reusing these features brings to two advantages: (i) These features do not need to be re-evaluated because they have already

been evaluated. (ii) By using SVM classifiers with powerful generalization, using too many features in the cascade is avoided, with the important results of saving training time and avoiding over-fitting.

Furthermore, to help to reduce the training time, we propose two feature selection methods that quickly select a small and optimal subset of features by using mutual information and feature variance. In the feature selection method using mutual information, we propose using a more efficient discretization method that uses minimum description length principle (MDLP) to estimate probability densities of continuous random variables. This approach can be considered as a generalization of previous ones that mainly use a single threshold for discretization. In the other feature selection method, features are selected based on their distances to principle components computed by PCA (principle component analysis) from the data distribution. Using this approach, the final classifier is able to run faster than that using the traditional PCA-based feature extraction method since it avoids computation cost of the subspace projection. These proposed feature selection methods are integrated seamlessly and efficiently into the multi-stage based framework for face detection described above.

The organization of the extracted faces is usually done automatically by using a clustering method. In many video indexing applications, k-means clustering is very common. However, it suffers from a number of serious drawbacks. For example, it can not be applied to general similarity measures; the number of clusters must be provided in advance; it generates many bad clusters when the input data is noisy; and it is not scalable to handle large datasets. Instead, we propose using the relevant set correlation (RSC) clustering model from which the GreedyRSC clustering heuristic derived. This clustering model can help to avoid all the problems of k-means clustering. Furthermore, it is very efficient in finding high quality clusters in such noisy datasets as face datasets extracted from video. These high quality clusters along with person names extracted from video transcripts are useful to identify

important people appearing frequently in video databases that can be done by an association method based on the statistical machine translation.

The proposed techniques are integrated in developing a video indexing and retrieval system that can help users to access and navigate contents in news video databases easily and quickly. The system can show representative names and faces appearing in videos ranked by their occurrence frequency, and access to related news stories by using these faces or names. Furthermore, it can show possible associations between names and faces. Our approach is generic and has the potential to handle very large scale video datasets effectively and efficiently.

To My Parents and Wife.

Acknowledgments

First of all, I wish to express my gratitude to my supervisor, Professor Shin'ichi Satoh for his valuable guidance and advice on research during the past three years. It is my luck and my pleasure to be his student, and I really appreciate all what he has done to me.

Second, thanks go to my committee members, Prof. Haruki Ueno, Prof. Atsushi Imiya, Prof. Norio Katayama, Prof. Akihiro Sugimoto and Prof. Asanobu Kitamoto, for their advice and encouragement to my thesis.

I am grateful to Professor Michael Edward Houle for sharing with me his wide knowledge on clustering techniques. His knowledge is very useful for me to complete my thesis.

I also would like to thank to Prof. Duong Anh Duc, Prof. Dong Thi Bich Thuy and Prof. Henri Angelino for encouraging me to apply to the PhD program of National Institute of Informatics (NII); and thank to NII for the financial support on my research.

I have a pleasant stay at NII and I am indebted to my friends, Ved Kafle, Vo Duc Khanh, Dao Minh Son, Nguyen Phuoc Tat Dat, Nguyen Luu Thuy Ngan and Vu Quang Minh. I really enjoyed my discussions and collaborations with them. Especially I owe much to Fuminori Yamagishi, who is always willing to help me a lot since I first came to Japan, I learned a lot about Japanese culture and life from him.

I would like to take this special occasion to thank my parents for their long support and encouragement.

Finally, thanks to my lovely wife, Nguyen Huong, for her endless love, tireless support and unwavering encouragement.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivations and Objectives

The advancement of digital technology in recent years has made large scale multimedia data more available to users. Therefore, effective and scalable tools for indexing, manipulating and retrieving video contents are strongly needed [84]. In video databases, human face processing plays a very important role, since people's activities are highly related to important events, especially in broadcast news videos. There are many potential applications in content-based indexing and retrieval of large scale video databases using face information as the key component:

- Video annotation: by labeling faces by corresponding names, video databases can be organized by presence of individuals. As a result, large and realistic face databases can be built from many semi-supervised datasets available on the Internet. These databases are very useful in developing robust face detection and identification systems [9, 8, 76]. Figure 1.1 shows an example of video annotation.

- Video retrieval: by giving a face of the target individual, the system can retrieve ranked video segments, e.g. shots or stories, related to that individual. Consequently, video databases can be accessed easier and bring many benefits to television editors, multimedia designers, journalists and commercial companies [4, 83].

- Video summarization: by organizing the extracted faces, the system can list principal casts in movies, find people who appear frequently on video, and name significant events appearing in video databases [21].

- People identification: by extracting faces from video and check against gallery databases, the system can identify persons appearing at different times. It is useful for security and authentication systems.

Motivated by these applications, the main objective of this thesis is to develop techniques that can automatically extract and organize a huge number of human faces for video indexing and retrieval. We will show how the proposed techniques can be integrated in a video indexing and retrieval system that can help users to access and navigate contents of large scale news video databases easily and quickly.

1.2 Challenges

Extracting and organizing human faces from large scale video databases are challenging problems due to the following reasons:

- Face appearance varies largely due to intrinsic factors such as aging, facial expressions and make-up styles, and extrinsic factors such as pose changes, lighting conditions and partial occlusion (see Figure 1.2). These factors make it difficult to construct good face models. Many efforts have been made in the fields of computer vision and pattern recognition [101, 107], but good results have limited to restricted settings.

- Cluttered background, low resolution and quality make faces hard to be determined.

- Detailed visual information often must be represented as high dimensional feature vectors. Evidence suggests that when the representational dimension of feature vectors is high, an effect known as "the curse of dimensionality" causes exact similarity search

to access an unacceptably-high proportion of the data elements [10, 93]. This makes constructing learning methods that can be efficient and scalable more complicated.

In addition, following problems are specific when handling large scale broadcast news video databases:

- There are many different approaches which have been proposed to build fast and robust face detection systems, but it is difficult to implement such systems. State of the art systems such as [36, 81, 92] were only tested on small-size sets of still images. e.g. CMU+MIT test set [75]. Applying them to face detection in video needs to develop post-processing methods to improve the accuracy [38].

- Due to restricted conditions in acquiring news, meaningful faces usually have small size which makes much information to be lost and similarity measures used in face matching are unreliable. Figure 1.3 shows an example of a meaningful face (*Clinton* in this case) appearing in a news video frame that is smaller than that appearing in an Internet news document.

- Speed constraints: the face detection needs to be fast to handle huge and dynamic data in video streams broadcast on different channels and time periods.

- Open dataset: Usually, the number of *people of interest* appearing in video is unknown and grows overtime. This makes popular clustering techniques such as k-means become unsuitable. Figure 1.4 shows rare faces extracted from video.

- Names extracted from video captions and transcripts can be used to find shots containing faces of the person of interest. However, in practice, faces and names may not necessarily appear together [98].

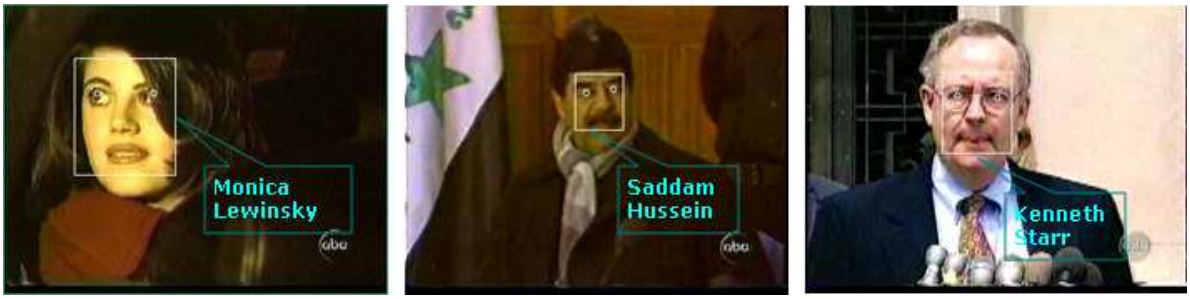

Figure 1.1: An example of video annotation.

Figure 1.2: Variations of face appearance due to (a) illumination, (b) image quality and (c) orientation.

1.3 Problem Statement

A general framework for a retrieval system using face information usually consists of following components as shown in Figure 1.5:

- The first component, **"Face Extraction"**, is used to extract faces from video frames. A tracking system is needed to group a large number of faces in a video sequence belonging to one individual into one representative face. Extracted faces then are normalized to eliminate effects in illumination and poses before passing to the second component.

- The second component, **"Face Grouping"**, is used to organize the faces into clusters. This task is usually done by using a clustering method that can handle large and high

Figure 1.3: An example of meaningful faces in (left) a news video frame (32x37 pixels) and in (right) an Internet news document (75*86 pixels). Meaningful faces in news video are smaller than that in Internet news documents.

Figure 1.4: An example of rare faces extracted from video.

dimensional datasets. The resulting clusters are further post-processed to be able to be used in video indexing and retrieval systems.

- The third component, **"Name Extraction"**, is used to extract person's names from video transcripts. These names are filtered and then are used for improving performance of the retrieval process.

- The fourth component, **"Name-Face Association"**, is used to make the correspondence between important names and faces. Consequently, the result can be used to

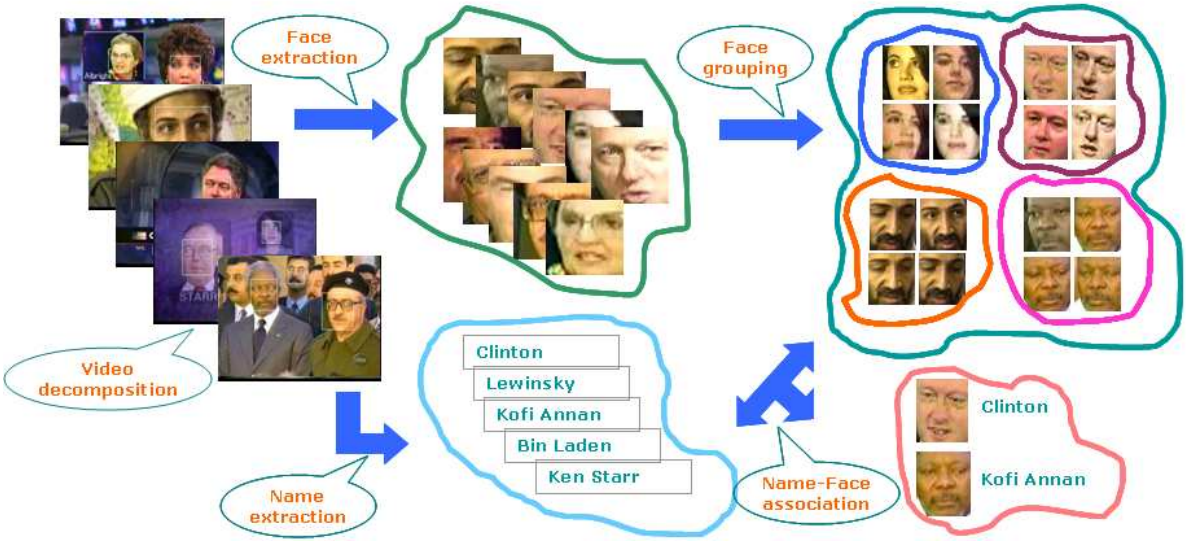

Figure 1.5: A general framework for a video retrieval system using human face information.

generate a summarization based on important people or to form the entry page (page zero) for browsing and navigating the database.

From the above framework, the following research questions are raised:

- How to extract features that are not only compact but also efficient for face representation in classification tasks such as detection and clustering? The selected features should not be redundant, should well characterize both inter-class variability and intra-class variability.

- How to design a fast and robust face extraction system applicable for large scale video databases? Such a system requires a flexible structure for combining advantages of existing classifiers.

- How to organize a large number of the extracted faces in meaningful groups for easily browsing, searching and navigating? Such a method should automatically determine the number of groups, provide intuitively ways to control the quality of clusters and should not be restricted to specific distance measures.

- How to extract the faces of important people and identify them by using multi-modal analysis ?

1.4 Contributions

By answering those research questions, our contributions include:

- Propose two efficient feature selection methods to quickly select a small subset of features by using mutual information and feature variance. These feature selection methods can help not only to reduce the training time dramatically but also to maintain the high detection accuracy. In feature selection methods using conditional mutual information based approach, binarization using a threshold is often used to estimate probability densities of continuous random variables. To generalize it, we propose using a more efficient discretization method using minimum description length principle (MDLP) and prove that this approach outperforms previous ones. On the other hand, to reduce the computation cost of PCA (principle component analysis)-based feature selection, we propose a method to select features based on their distance to principle components computed by PCA from the data distribution. These two feature selection methods are integrated seamlessly and efficiently into the multi-stage based framework for face detection described below.

- Propose a multi-stage approach which is fast, robust and easy to train - for a face detection system. Motivated by the work of Viola and Jones [91], this approach uses cascades of classifiers to yield a coarse-to-fine strategy to reduce significantly detection time while maintaining a high detection accuracy. However, it is distinguished from previous work by two features. First, a new stage, rejection stage, has been added to detect face candidate regions more quickly by using a larger window size and a larger moving step size. Second, support vector machine (SVM) classifiers are used instead of AdaBoost classifiers in the last stage, classification stage; and Haar wavelet

features selected by the previous stage are reused for the SVM classifiers for robust and efficient classification. We integrate the conditional mutual information-based feature selection method described above to quickly select informative features before using AdaBoost and the feature variance based feature selection to reduce further the number of features returned by AdaBoost before using SVM. By combining AdaBoost and SVM classifiers, the final system can achieve both fast and robust detection because most non-face patterns are rejected quickly in first stages, while only a small number of promising face patterns are classified robustly in later stages. The proposed multi-stage-based system has been shown to run faster than the original AdaBoost-based system while maintaining comparable accuracy.

- Identify successfully the most suitable face representation and cluster technique and integrate these ones into a unified framework that help to organize large scale video databases and make them to be easily accessed by end users. This is one of the retrieval systems that works on such large scale video datasets as TRECVID 2003 and NHK News 7; and uses face information for accessing video contents.

1.5 Thesis Overview

This thesis is organized as follows:

- Chapter 2: First, we review state of the art feature extraction methods for object detection in general and face detection in specific. Then, two proposed feature selection methods are presented and evaluated.

- Chapter 3: We introduce our multi-stage approach to building a face detection that is fast, robust and easy to train. Extensive experiments on different benchmark datasets are shown to prove advantages of our proposed method.

- Chapter 4: We introduce a general framework for large scale video indexing and retrieval using face information extracted by the techniques described in the previous chapters. Prototypes, interfaces and demonstrations are presented to illustrate effectiveness of the news video indexing and retrieval system.

- Chapter 5: We summarize our contributions and discuss future work.

Chapter 2

Feature Extraction and Selection

2.1 Introduction

Feature extraction and feature selection are crucial issues in building any fast and robust classification system. Feature extraction involves the study of finding appropriate representations of the object class. Meanwhile, feature selection involves the study of finding a small subset out of a given large set of features. They are significant due to the following reasons:

- Object appearance is of high variations due to many intrinsic and extrinsic factors. For example, face appearance varies largely due to pose changes, facial expressions, make-up styles, occlusions, scales and so on. An appropriate object representation that satisfies the following conditions will make the discrimination task much easier and faster [28]:
  - well characterize both inter-class variations and intra-class variations for robust classification.
  - can be easily extracted from raw images for rapid processing.
  - have a small number of features for computational cost reduction.

- Extracting features that meet the above criteria usually leads to a huge feature set. For example, the number of Haar wavelet features used in [91] for face detection is hundreds of thousands. However, only small and incomplete training sets are available. As a result, these systems easily suffer from the curse of dimensionality and over-fitting if no any feature selection process is used.

- Huge feature sets usually include many irrelevant and redundant features that can degrade the generalization performance of classifiers, waste storage space and increase running and training time [11].

- Selecting an optimal feature subset from a huge feature set can improve the accuracy and speed of classifiers. Furthermore, less complex model is easier to understand and verify. In face detection, the success of systems such as those in [49, 91] comes mainly from efficient feature selection methods.

In this chapter, we firstly review state of the art techniques in feature extraction and feature selection. Next, we propose two novel feature selection methods that can be applied in object detection systems in general as well as face detection in specific. Then, the proposed methods are evaluated in a face detection framework to show the advantages and effectiveness.

2.2 Feature Extraction

One of the simplest way to feature extraction is to use the pixel intensities of the raw input image. However, the pixels have very high degree of variability, normalization steps, such as histogram equalization, linear brightness subtraction [75], are usually used. To provide more informative and descriptive model of object classes, state of the art feature extraction methods usually study how to efficiently encode local, oriented, multi-scale, intensity differences of the pixels.

2.2.1 Wavelet-based Features

Wavelet features are very informative and compact for object representation because they provide a multi-resolution representation of the target object in which the features at different scales capture different levels of detail [70]. The coarse scale wavelets encode large regions

while the fine scale wavelets describe smaller, local regions. According to the study of C. Papageorgiou [70], the wavelet coefficients preserve all the information in the original image, but the coding of the visual information differs from the pixel-based representation in two significant ways, that make the intra-class variations minimized and the extra-class ones maximized simultaneously.

First, the difference in average intensity between local regions along different orientations is encoded in a multi-scale framework. Constraints on the values of the wavelets can express visual features of the object class; strong response from a particular wavelet indicates the presence of an intensity difference, or boundary, at that location in the image while weak response from a wavelet indicates a uniform area.

Second, the use of an over-complete basis, for example, Haar basis, allows us to propagate constraints between neighboring regions and describe complex patterns. The quadruple density wavelet transform provides high spatial resolution and results in a rich, over complete dictionary of features.

This section introduces two types of wavelets, Haar wavelet and Gabor wavelet, that have been used widely in many face detection and face recognition systems. Other types of wavelets used in similar contexts can be found in [55, 80].

2.2.1.1 Haar Wavelet

Haar wavelet features were firstly used in [70] for face and people detection and then have been widely used in many face detection systems [50, 49, 56, 64, 91]. Normally, four kinds of features modeled from adjacent basic rectangles with the same size and shape are used (Figure 2.1). The feature value is defined as the difference of sum of the pixels within rectangles. Each feature is parameterized by four factors: the position within the window (x, y), the width (Dx) and the height (Dy) (Figure 2.2).

Besides having good connection with human visual system modeling, the Haar wavelets are popular since they can be computed extremely quickly by using the integral image

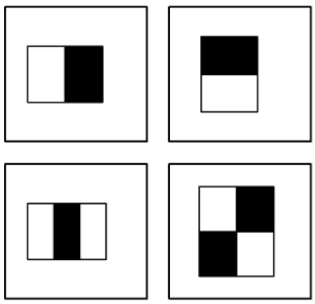

Figure 2.1: Haar wavelet features.

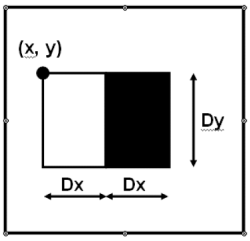

Figure 2.2: Parameters of a Haar wavelet feature.

definition [91]. The integral image at location (x, y) is defined as:

$$ii\,(x,y) = \sum_{x'<=x,y'<=y} i\,(x',y')\,,$$

where $ii\,(x,y)$ is the integral image and $i\,(x,y)$ is the original image.

In practice, $ii\,(x,y)$ can be computed simply by using the following recursive function:

$$ii\,(x,y) = ii\,(x,y-1) + ii\,(x-1,y) + i\,(x,y) - ii\,(x-1,y-1)\,,$$

and the sum of the pixels within a rectangle can be computed from four integral image values of its vertices, for example, $Sum(D) = 1 + 4 - (2 + 3)$ (Figure 2.3).

Haar wavelet features are recently extended by adding an efficient set of 45° rotated features [73] as shown in Figure 2.4 to capture nature of the target object more efficient.

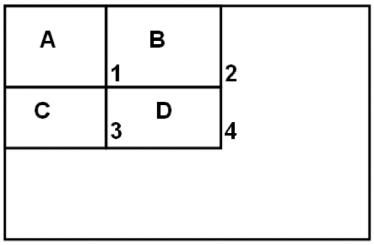

Figure 2.3: Feature evaluation based on an integral image.

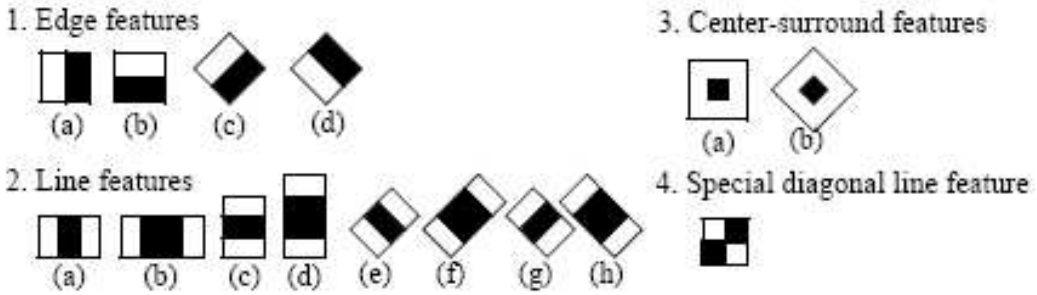

Figure 2.4: Extended Haar Wavelet Features proposed by Lienhart et al. [73].

2.2.1.2 Gabor Wavelet

Gabor wavelet features are defined as:

$$\psi_{\mu,\nu}(z) = \frac{k_{\mu,\nu}^2}{\sigma^2} exp\left(-\frac{k_{\mu,\nu}^2 z^2}{2\sigma^2}\right)\left[exp(ik_{\mu,\nu}z) - exp\left(-\frac{\sigma^2}{2}\right)\right],$$

where μ and ν define the orientation and scale of the Gabor kernels respectively, $z = (x; y)$, and the wave vector $k_{\mu,\nu}$ is defined as:

$$k_{\mu,\nu} = k_\nu e^{i\phi_\mu},$$

where

$$k_\nu = \frac{k_{max}}{f^\nu},\ k_{max} = \frac{\pi}{2},\ f = \sqrt{2},\ \phi_\mu = \sigma\frac{\mu}{8},\ \sigma = 2\pi.$$

The Gabor representation of a face image is computed by convolving the face image with the Gabor filters. Let $f(x, y)$ be the face image, its convolution with a Gabor filter $\psi_{\mu,\nu}(z)$ is defined as:

$$G_{\psi,f}(\mu, \nu, x, y) = f(x, y) * \psi_{\mu,\nu}(z),$$

where $*$ denotes the convolution operator.

In most face recognition systems [53, 54, 94, 106], Gabor kernels at five scales $\nu \in \{0, 1, 2, 3, 4\}$ and eight orientations $\mu \in \{0, 1, 2, 3, 4, 5, 6, 7\}$ are usually used. At each pixel position, 40 Gabor features are computed by convolving the input image with the real part of Gabor filters. As a result, there are 40×M×N Gabor features for one M×N image.

Figure 2.5 shows the real part of the Gabor kernels at five scales and eight orientations and their magnitudes. The kernels exhibit desirable characteristics of spatial frequency, spatial locality, and orientation selectivity. Figure 2.6 shows the Gabor wavelet representation (the real part and the magnitude) of a face sample image. These representation results display scale, locality, and orientation properties corresponding to those displayed by the Gabor wavelets in Figure 2.5. These pictures were presented in [53].

2.2.2 Local Binary Patterns

The LBP operator proposed by Ojala et al. [67, 68] is a powerful method for texture description. It is invariant with respect to monotonic grey-scale changes, hence no grey-scale normalization needs to be done prior to applying the LBP operator. This operator labels the pixels of an image by thresholding the neighborhoods of each pixel with the center value and considering the result as a binary number. Figure 2.7 shows an example of LBP calculation.

The 256-bin histogram of the labels computed over a region can be used as a texture descriptor. Each bin (LBP code) can be regarded as a micro-texton. Local primitives which are encoded by these bins include different types of curved edges, spots, flat areas etc. Figure 2.8 shows some examples.

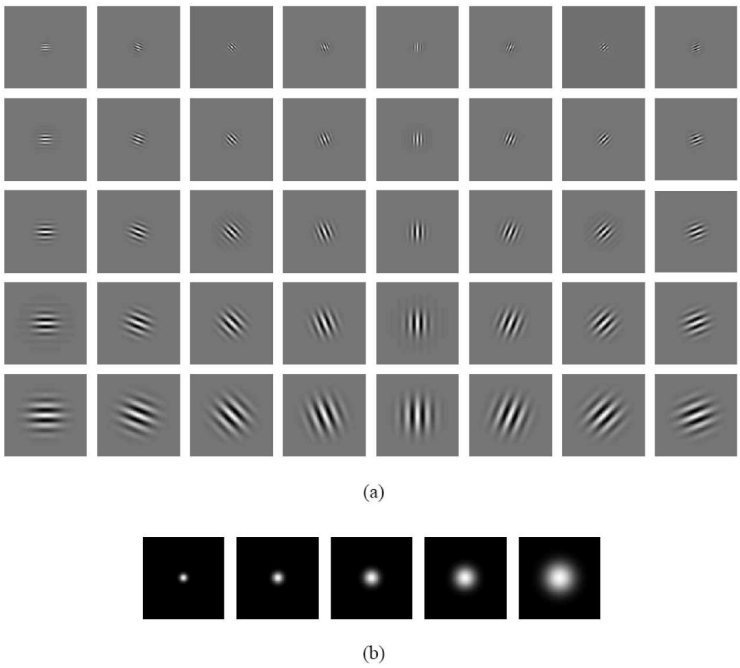

Figure 2.5: Gabor wavelets. (a) The real part of the Gabor kernels at five scales and eight orientations. (b) The magnitude of the Gabor kernels at five different scales [53].

Recently, the LPB operator has been extended to consider different neighborhood sizes [68]. For example, the operator $LBP_{4,1}$ uses only the 4 neighbors while $LBP_{16,2}$ considers the 16 neighbors on a circle of radius 2. In general, the operator $LBP_{P,R}$ refers to a neighborhood size of P equally spaced pixels on a circle of radius R that form a circularly symmetric neighbor set. $LBP_{P,R}$ produces 2^P different output values, corresponding to the 2^P different binary patterns that can be formed by the P pixels in the neighbor set. It has been shown that certain bins contain more information than others. Therefore, it is possible to use only a subset of the 2^P local binary patterns to describe the textured images. Ojala et al. [68] defined these fundamental patterns (called also "uniform" patterns) as those with a small number of bitwise transitions from 0 to 1 and vice versa. For example, 00000000 and 11111111 contain 0 transition while 00000110 and 01111000 contain 2 transitions and so on. Accumulating the patterns which have more than 2 transitions into a single bin yields an LPB descriptor, denoted $LBP^{u2}_{P,R}$, with less than 2^P bins.

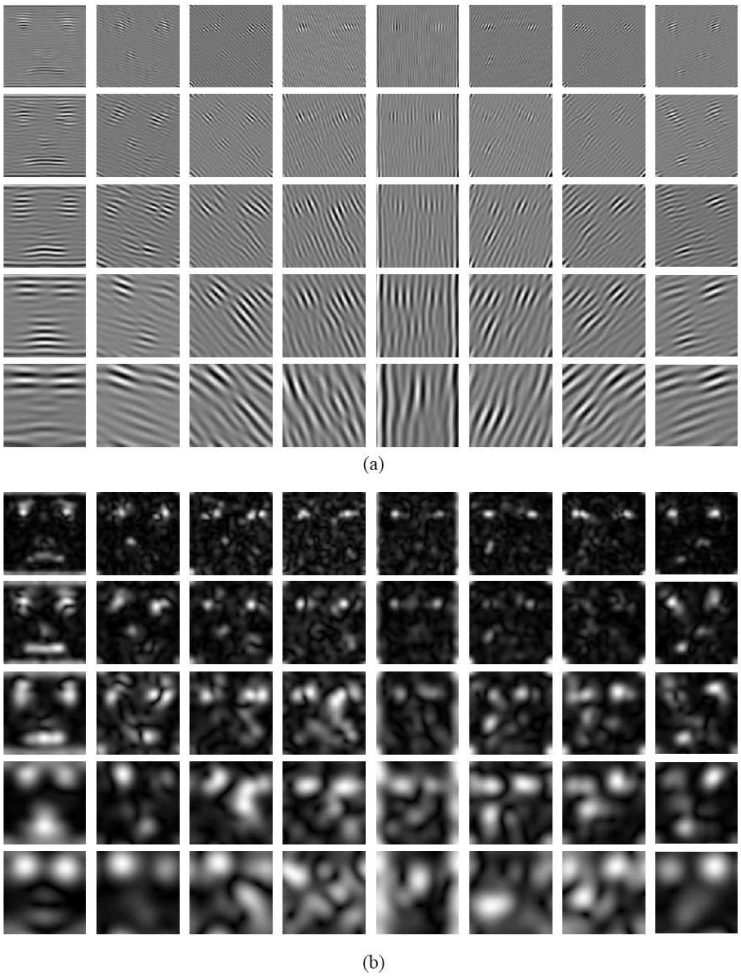

Figure 2.6: Gabor wavelet representation (the real part and the magnitude) of a sample face image. (a) The real part of the representation. (b) The magnitude of the the representation [53].

An LBP description computed over the whole face image encodes only the occurrences of the micro-patterns without any indication about their locations. To overcome this effect, Ahonen et al. [3] divided the input face image into several (e.g. 7×7=49) non-overlapping blocks from which the local binary pattern histograms are computed and concatenate these histograms into a single histogram (Figure 2.10). In such a representation, the texture of facial regions is encoded by the LBP while the shape of the face is recovered by the concatenation of different local histograms.

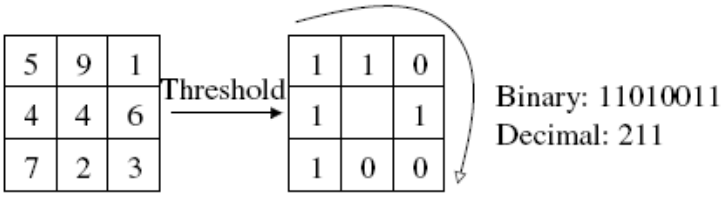

Figure 2.7: The basic LBP operator.

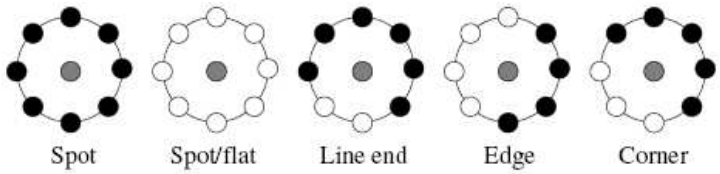

Figure 2.8: Different texture primitives detected by the LBP [68].

However, this representation is more adequate for larger sized images (such as the FERET images [72]) and leads to a relatively long feature vector typically containing thousands of elements. Therefore, Hadid et al. [28] proposed a new facial representation which is efficient for low-resolution images used for face detection. This representation uses overlapping regions and a 4-neighborhood LBP operator ($LBP_{4,1}$) to avoid statistical unreliability due to long histograms computed over small regions. Additionally, the holistic description of a face is enhanced by including the global LBP histogram computed over the whole face image.

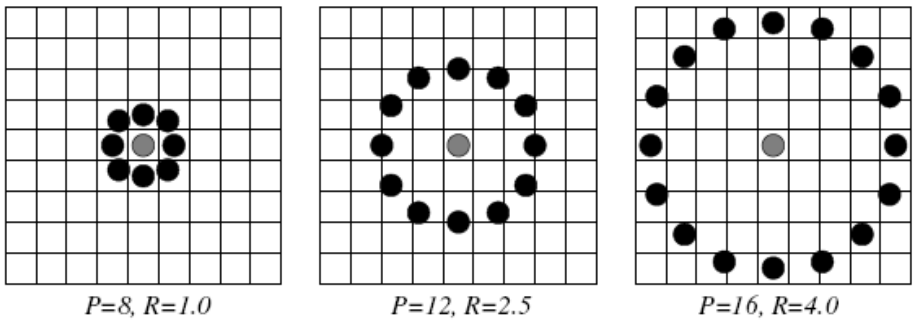

Figure 2.9: Circularly symmetric neighbor sets. Samples that do not exactly match the pixel grid are obtained via interpolation [68].

Figure 2.10: LBP representation for high resolution face image in face recognition systems proposed by Ahonen et al. [3].

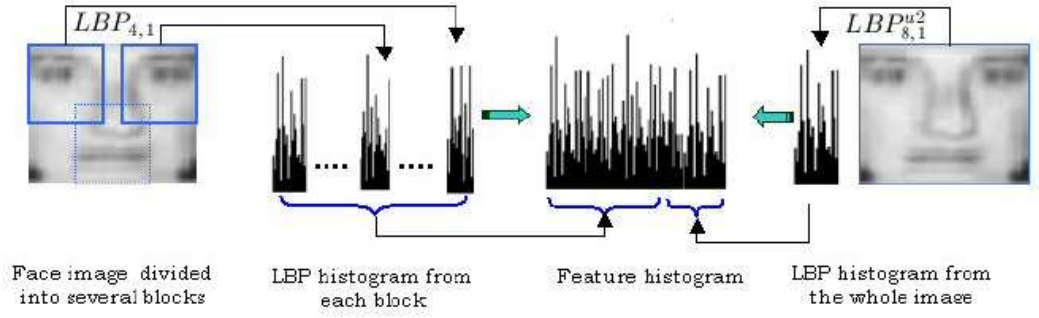

Figure 2.11: LBP representation for low resolution face image in face detection systems proposed by Hadid et al. [28].

For example, a 19×19 face image is divided into 9 overlapping regions of 10x10 pixels (overlapping size=4 pixels). From each region, a 16-bin histogram using the $LBP_{4,1}$ operator is computed and the results are concatenated into a single 144-bin histogram. Additionally, $LBP^{u2}_{8,1}$ is applied to the whole 19×19 face image to obtain a 59-bin histogram which is added to the 144 bins previously computed. As a result, a (59+144=203)-bin histogram is used as a face representation (Figure 2.11).

2.2.2.1 Local Gabor Binary Pattern Histogram Sequence (LGBPHS)

Recently, Zhang et al. [106] have proposed Local Gabor Binary Pattern Histogram Sequence (LGBPHS) , which is not only robust to the variations of imaging condition but also with

much discriminating power. Briefly speaking, LGBPHS is actually a representation approach based on multi-resolution spatial histogram combining local intensity distribution with the spatial information, therefore, it is robust to noise and local image transformations due to variations of lighting, occlusion and pose. Additionally, instead of directly using the intensity to compute the spatial histogram, multi-scale and multi-orientation Gabor filters are used for the decomposition of a face image, followed by the local binary patterns (LBP) operator. The combination of Gabor and LBP further enhances the representation power of the spatial histogram greatly.

In this approach, a face image is modeled as a "histogram sequence" by the following procedure as depicted in Figure 2.12:

- An input face image is normalized and transformed to obtain multiple Gabor Magnitude Pictures (GMPs) in frequency domain by applying multi-scale and multiorientation Gabor filters;
- Each GMP is converted to Local Gabor Binary Pattern (LGBP) map;
- Each LGBP Map is further divided into non-overlapping rectangle regions with specific size, and histogram is computed for each region;
- The LGBP histograms of all the LGBP Maps are concatenated to form the final histogram sequence as the model of the face.

Experimental evaluations on different face datasets have proved the effectiveness and robustness of this feature to the general variations of lighting, expression, and occlusion.

2.2.3 Edge Orientation Histogram

2.2.3.1 Edge Detection

Local edge orientation histogram (EOH) proposed by Levi and Weiss [48] can be used for object representation from which robust classifiers can be learned from a small number

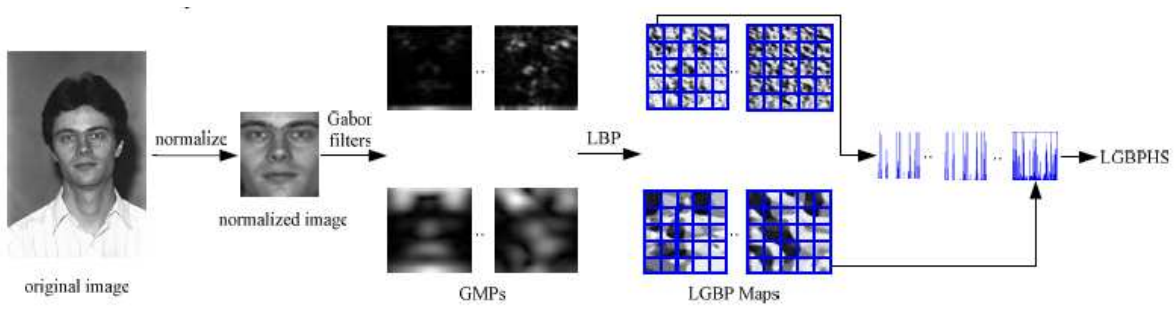

Figure 2.12: Face representation using LGBPHS proposed by Zhang et al. [106].

of training samples. In the following paragraphs, we take the descriptions of this feature extraction method from [48].

In this approach, edges firstly are detected by using Sobel masks due to their simplicity and efficiency. The gradients at the point (x, y) in the image I can be found by convolving Sobel masks with the image.

$$G_x(x, y) = Sobel_x * I(x, y)$$

and

$$G_y(x, y) = Sobel_y * I(x, y)$$

where $Sobel_x$ and $Sobel_y$ are the x and y Sobel masks respectively. The strength of the edge at the point (x, y)

$$G(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2}$$

In order to ignore noise, $G(x, y)$ is thresholded such that

$$G'(x, y) = \begin{cases} G(x, y) & if\ G(x, y) < T \\ 0 & otherwise \end{cases}$$

A major drawback of Sobel masks is that the value of the threshold T has to be set manually. In their experiments the value of T was set to be between 80 and 110. The orientation of

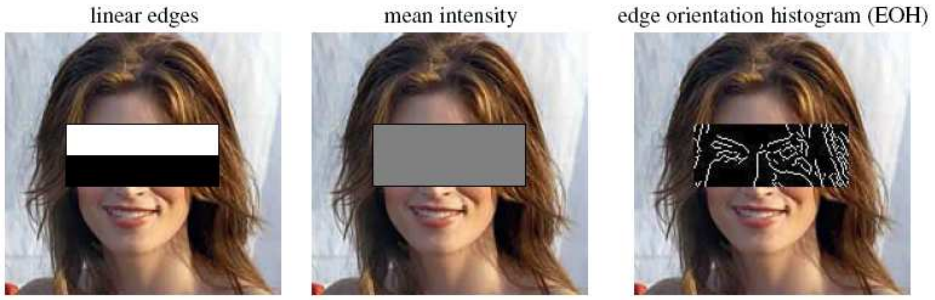

Figure 2.13: An illustration of local orientation histograms proposed by Levi and Weiss [48] over linear edges and mean intensity features.

the edge is

$$\theta(x,y) = arctan\left(\frac{G_y(x,y)}{G_x(x,y)}\right)$$

The edges are then divided into K bins. The value of the k_{th} bin is denoted as:

$$\psi_k(x,y) = \begin{cases} G'(x,y) & if\ \theta(x,y) \in bin_k \\ 0 & otherwise \end{cases}$$

After edges are computed, the edge orientation histogram is defined as:

$$E_k(R) = \sum_{(x,y)\in R} \psi_k(x,y),$$

where R is some sub-window in the image.

Then a set of features, A, is defined such that:

$$A_{k_1,k_2}(R) = \frac{E_{k_1}(R)+\epsilon}{E_{k_2}(R)+\epsilon}.$$

Since $A_{k_1,k_2}(R) \in \Re$, each feature yields two potential weak hypothesis $A_{k_1,k_2}(R) \geq T$ and $A_{k_1,k_2}(R) \leq T$ for some threshold $T \in \Re$. For the first weak hypothesis ($A_{k_1,k_2}(R) \geq T$) these features capture R's where k_1's orientation is dominant in respect to k_2's orientation relation.

In addition, a slightly different set of features, which measures the ratio between a single orientation and the others, is defined to find the dominant edge orientation in a specific area rather than the ratio between two different orientations:

$$B_k(R) = \frac{E_k(R) + \epsilon}{\sum_i E_i(R) + \epsilon}.$$

A third set of features which captures symmetry in the image was also proposed based on recommendations from [85] in which the authors showed that symmetry played an important role in object recognition.

$$Symm(R_1, R_2) = \frac{\sum_{k \in K} |E_k(R_1) - E_k(R_2)|}{sizeof(R_1)},$$

where R_1 and R_2 are rectangles of the same size and are positioned at opposite sides of the symmetry axes. The L_1 norm between the two histograms is divided by the size of R_1 such as to preserve the scale invariance property. As for the previous types of features, the symmetry features can be used not only to find symmetry but also to find places where symmetry is absent. For example, the lower and the upper part of the face are not symmetric to each other.

2.2.4 Fragment-Based Features

Unlike other methods that use local 2-D features, Ullman et al. [89] used object fragments to represent the object class. These fragments are taken directly from example views of objects in the same class. The shape fragments used to represent faces, for instance, be different from shape fragments used to represent cars, or letters in the alphabet. The fragments are selected and divided into equivalence sets that contain views of the same general region in the objects under different transformations and viewing conditions. The use of fragment views achieves superior generalization capability with a smaller number of example views

compared with more global methods. In the following paragraphs, the descriptions of this feature extraction method from [89, 90] are presented.

The use of the combination of image fragments to deal with intra-class variability is based on the notion that images of different objects within a class have a particular structural similarity – they can be expressed as combinations of common substructures. Roughly speaking, the idea is to approximate a new image of a face, say, by a combination of images of partial regions, such as eyes, hairline etc. of previously seen faces. Examples of fragments for the class of human faces (roughly frontal) and the class of cars (sedans, roughly side views) are illustrated in Figure 2.14. The fragments used as a basis for the representation were selected by the principle of maximizing mutual information $I(C, F)$ between a class C and a fragment F. This is a natural measure to employ, because it measures how much information is added about the class once we know whether the fragment F is present or absent in the image. In the ensemble of natural images in general, prior to the detection of any fragment, there is an a-priori probability $p(C)$ for the appearance of an image of a given class C. The detection of a fragment F adds information and reduces the uncertainty (measured by the entropy) of the image. Selected fragments are those that will increase the information regarding the presence of an image from the class C by as much as possible, or, equivalently, reduce the uncertainty by as much as possible. This depends on $p(F|C)$, the probabilities of detecting the fragment F in images that come from the class C, and on $p(F|NC)$ where NC is the complement of C.

A fragment F is highly representative of the class of faces if it is likely to be found in the class of faces, but not in images of non-faces. This can be measured by the likelihood ratio $p(F|C)/p(F|NC)$. Fragments with a high likelihood ratio are highly distinctive for the presence of a face. However, highly distinctive features are not necessarily useful fragments for face representation. The reason is that a fragment can be highly distinctive, but very rare. For example, a template depicting an individual face is highly distinctive: its presence in the image means that a face is virtually certain to be present in the image. However,

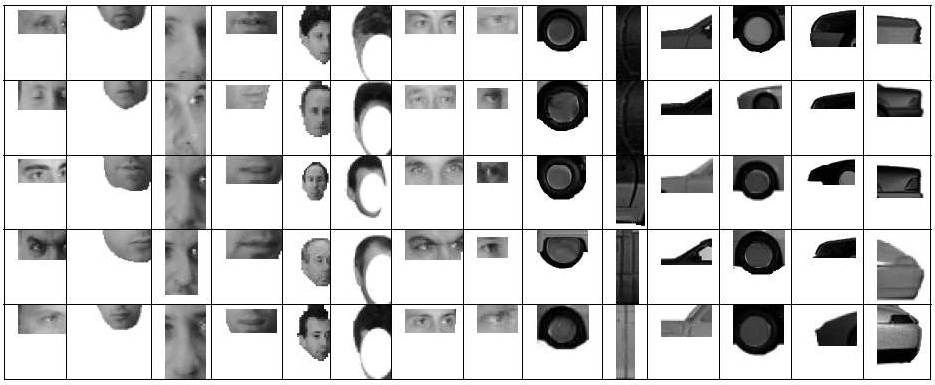

Figure 2.14: Examples of informative fragments used to represent faces and cars as shown in [89].

the probability of finding this particular fragment in an image and using it for making classification is low. On the other hand, a simple local feature, such as a single eyebrow, will appear in many more face images, but it will appear in non-face images as well. The most informative features are therefore fragments of intermediate size.

2.2.5 Feature Extraction Using Principal Component Analysis

The main steps to extract features using Principal Component Analysis (PCA) are summarized in the following. The details are given in [88].

Each face image $I(x, y)$ is represented as an $N \times N$ vector Γ_i.

The average face Ψ is computed as:

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

where M is the number of face images in the training set.

The difference between each face and the average face is given as:

$$\Phi_i = \Gamma_i - \Psi$$

A covariance matrix is then estimated as:

$$C = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T = AA^T,$$

where

$$A = [\Phi_1\Phi_2...\Phi_M].$$

Eigenvectors u_i and corresponding eigenvalues λ_i of the covariance matrix C can be evaluated by using a Singular Value Decomposition (SVD) method [88]:

$$Cu_i = \lambda_i u_i.$$

Because matrix C is usually very large ($N^2 \times N^2$), evaluating eigenvectors and eigenvalues is very expensive. Instead, eigenvectors v_i and corresponding eigen values μ_i of matrix A^TA ($M \times M$) can be computed. After that, u_i can be computed from v_i as follows:

$$u_i = Av_i,\ j = 1, ..., M.$$

To reduce dimensionality, only a smaller number of eigenvectors $K(K << M)$ corresponding to the largest eigenvalues are kept. A new face image Γ, after subtracting the mean ($\Phi = \Gamma - \Psi$) can then be reconstructed in eigenspace by the formula:

$$\tilde{\Phi} = \sum_{i=1}^{K} w_i u_i,$$

where $w_i = u_i^T\Psi$ are coefficients of the projection and can be considered as a new representation of the original face in this eigenspace.

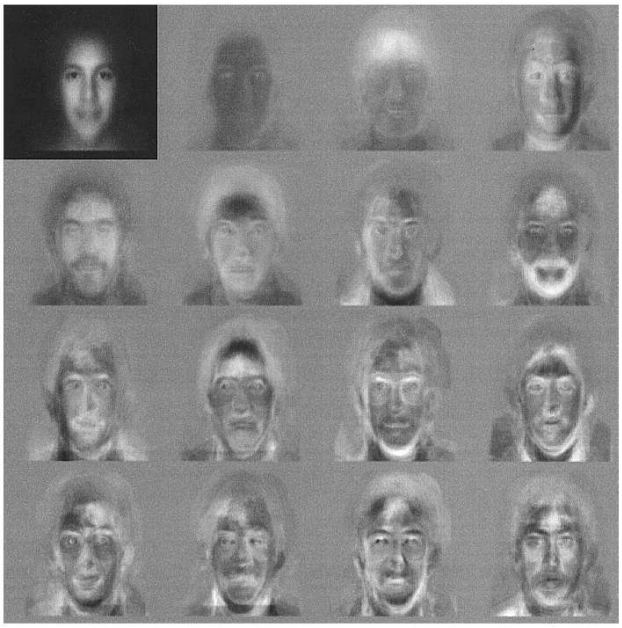

Figure 2.15: The mean face and eigenfaces [88].

2.2.6 Orientation Features

In [63], Mikolajczyk et al. proposed a method for object representation by orientation-based features and local groupings of these features. This approach was motivated by the excellent performance of SIFT descriptors [58, 59] which are local histograms of gradient orientations . SIFT descriptors are robust to small translation and rotation, and this is built into this approach in a similar way. In the following paragraphs, it is the descriptions of this feature extraction method from [63].

These features are the dominant orientation over a neighborhood and are computed at different scales. Here 5 scale levels and a 3-by-3 neighbourhood are used. Orientation is either based on first or second derivatives.

In the case of first derivatives, the gradient orientation is extracted. This orientation is quantized into 4 directions, corresponding to horizontal, vertical and two diagonal orientations. Then the score for each of the orientations is determined using the gradient magnitude. The dominant direction is the one which obtains the best score. If the score is below a thresh-

Figure 2.16: Orientation features. (a) Head image. (b) Gradient image. (c) Dominant gradient orientations. (d) Positive Laplacian responses. (e) Dominant orientations of the second derivatives [63].

old, it is set to zero. Figure 2.16(b) shows the gradient image and Figure 2.16(c) displays the dominant gradient orientations where each of the 5 values is represented by a different gray-level value.

A human face can be represented at a very coarse image resolution as a collection of dark blobs. An excellent blob detector is the Laplacian operator [52]. This filter is used to detect complementary features like blobs and ridges. The Laplacian ($d_{xx} + d_{yy}$) and the orientation of the second derivatives ($arctan(d_{yy}/d_{xx})$) are computed and dark blobs with the positive Laplacian responses are selected. Figure 2.16 shows the positive Laplacian responses. The dominant orientation is selected similarly to the gradient features. Second derivatives are symmetrical therefore their responses on ridges of different diagonal orientations are the same. Consequently there are 3 possible orientations represented by this feature. Figure 2.16(e) displays the dominant second derivative orientations where each orientation is represented by a different gray-level value.

Since a single orientation has a small discriminatory power, neighboring orientations into larger features are grouped. The technique described below was successfully applied to face detection [81]. They use two different combinations of local orientations. The first one combines 3 neighbouring orientations in a horizontal direction and the second one combines 3 orientations in a vertical direction. Figure 2.17(a) shows the triplets of orientations. A single integer value is assigned to each possible combination of 3 orientations. The number

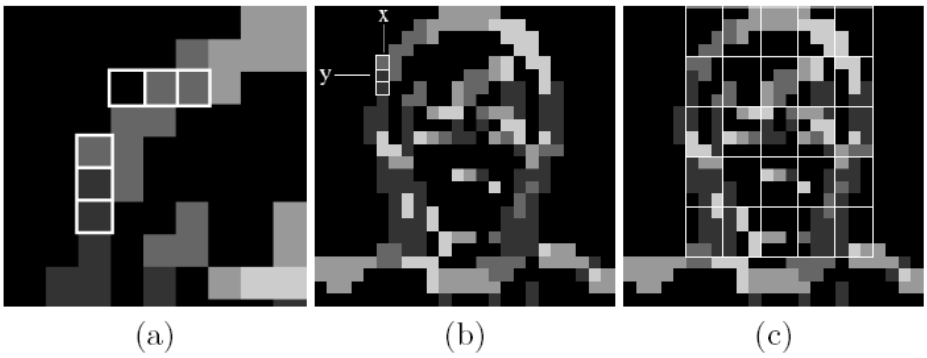

Figure 2.17: Local groups of features. (a) Two groups of local orientations. (b) Location of the feature on the object. (c) Grid of quantized locations [63].

of possible values is therefore $v_{max} = 5^3 = 125$ for the gradient and $v_{max} = 4^3 = 64$ for the Laplacian. More than 3 orientations in a group significantly increase the number of possible combinations and poorly generalize. In summary, at a given scale there are four different feature group types v_t: horizontal and vertical groups for gradient orientations and horizontal and vertical groups for the Laplacian.

2.2.7 Discussion

We have reviewed the state of the art feature extraction methods used in object detection systems. It is believed that the most significant purpose of feature extraction is to find a discriminant feature space for object representation so that it can both minimize intra-class variation and maximize inter-class variations. Such a feature space will make training classifiers much easier and free from using large training sets. Successful approaches described above have highlighted some criterion for designing good features.

First, region based approach is most suitable since the neighborhoods play a very important role in forming features. Local binary patterns (LBP) features [67, 68], scale invariance feature transforms (SIFT) features [59], edge orientation histogram (EOH) [48] and histograms of oriented gradients (HoG) [17] and more [62] are typical examples.

Second, multi-scale, multi-orientation, and multi-resolution spatial histogram should be incorporated smoothly. Approaches described in [53, 70, 92, 94, 106] have proved this truth.

Third, sets of extracted features are often over-complete. These sets contain many irrelevant and redundant features. Therefore, it must use some feature selection method to find the optimal set.

2.3 Feature Selection

Generally, feature selection methods can be categorized into two kinds: the filter-based approach and the wrapper-based approach [27]. The filter-based approach is independent of any induction algorithm, but the wrapper-based approach is associated with a specific induction algorithm to evaluate the appropriateness of the selected feature subset.

In the filter-based approach, features are normally selected based on their individual predictive power. This power is measured by Fisher scores, Pearson correlation [18, 27], or mutual information [23, 40, 71]. The major advantage of these methods is their speed and ability to scale to huge feature sets. However, because the mutual relationships between features are often not taken into account, the selected features might be highly redundant and less informative. For example, two features with high individual predictive power, when combined together, might not bring significant performance improvement. Meanwhile, combining two features of which one has low predictive power but is useful when combined with others would thus be more effective for improving performance.

Since wrapper-based feature selection methods use machine learning algorithms as a black box in the selection process, they can suffer from over-fitting when used with small training sets. Furthermore, in practical object detection systems as in [90, 91], the feature sets usually have hundreds of thousands of features, so using wrapper-based methods is obviously inefficient due to the very high computation costs they incur. For example, in the

state-of-the-art face detection system in [91], choosing a 6,061-feature set out of a 180,000-feature set using AdaBoost took several weeks.

In this section, we propose two feature selection methods using the filter-based approach. The first method [46] uses conditional mutual information as main criteria to judge the relevancy of features while the second method [44] uses the variation degree of features for selection.

2.3.1 Fast Feature Selection from Huge Feature Sets Using Conditional Mutual Information

Conditional mutual information (CMI) based feature selection methods have been proposed [6, 23, 40, 71, 90] to take full advantage of approaches described above for handling large-scale feature sets.

The main idea of CMI-based methods is to select features which maximize their relevance with the target class and simultaneously minimize mutual dependency between selected ones. It does not select a feature similar to already selected ones, even if it is individual powerful, as selecting it might not increase much information about the target class [23].

One of the important tasks in using CMI-based methods is mutual information estimation, which involves computing the probability densities of continuous random variables. In [40], Kwak and Choi used a Parzen window-based density estimation method in which many parameters such as kernel function and window width are complicated to determine. For simplification, the features are often discretized. So far, object detection systems like [23, 90] treat features as binary random variables by choosing appropriate thresholds. However, binarizing features is not a suitable way to handle highly complex data for which finding the best threshold is difficult. Using multiple thresholds to discretize data is better than using a binary approach. Such a simple method is equal-width binning , which divides the range of feature values into m equally sized bins, where m must be known in advance.

Our method is also a CMI-based feature selection method. However, the method's main distinguishing point is that it employs the entropy-based discretization method [20] to discretize features. This discretization method is simpler than the Parzen window-based density estimation method and is more efficient than binary discretization. Furthermore, contrary to equal-width binning, it can automatically determine the optimal number of bins based on data distribution. Experiments show that the proposed method can efficiently handle huge feature sets such as Haar wavelets [91] and Gabor wavelets [94] for face detection, significantly reducing the training time while maintaining high classification performance.

2.3.1.1 Conditional Mutual Information

Huge feature sets usually contain four kinds of features: *(i) irrelevant features, (ii) weakly relevant and redundant features, (iii) weakly relevant but non-redundant features, and (iv) strongly relevant features;* (iii) and (iv) are the objectives of feature selection methods [102]. To measure the relevance of a feature, an entropy-based measure, which quantifies the uncertainty of random variables, is normally used.

The entropy of a discrete random variable X is defined as

$$H(X) = -\sum_i P(x_i)log(P(x_i)),$$

and the conditional entropy of X after another variable Y is known is defined as:

$$H(X|Y) = -\sum_j P(x_j)\sum_i P(x_i|y_j)log(P(x_i|y_j)).$$

The mutual dependence between two random variables is measured by mutual information

$$I(X;Y) = H(X) - H(X|Y) = H(X) + H(Y) - H(X,Y)$$

The conditional mutual information is defined as:

$$I(C;X|Y) = H(C|Y) - H(C|X,Y) = I(C;X,Y) - I(C;Y).$$

In the first step, the most relevant feature F_1^i, which has the largest amount of mutual information, is selected. In the second step, however, the condition to select feature F_2^i is not its mutual information alone, but how much information F_1^i can add with respect to the already existing F_1^i. Therefore, F_2^i is selected so as to maximize the information it can add:

$$I(C;F_i^i|F_1^i) = I(C;F_i^i,F_1) - I(C;F_1^i).$$

Following the same process, we iteratively add the feature that brings the highest increase of the information content contained in the current selected feature set. The next feature F_t^i to be added at iteration t is defined by

$$F_t^i = arg\ max_{i=1..K}\left\{min_{F_j\in\mathcal{F}}\ I(C;F_i|F_j)\right\}.$$

To simply estimate mutual information, the easiest way is to discretize features in binary values by specifying thresholds [23, 90]. However, for complex data, doing this is not efficient; therefore, we use the entropy-based method proposed by Fayyad and Irani [20] for discretization. This method is a supervised method, so it is generic and can adapt very well to any kind of data distribution.

2.3.1.2 Entropy-based Subspace Splitting

This section briefly describes automatic subspace splitting using entropy-based discretization presented in [20]. Discretization is a quantizing process that converts continuous values into discrete values. It typically consists of four steps [57]:

- **Step 1:** Sorting the continuous values of the feature to be discretized.

- **Step 2:** Evaluating candidate *cut-points* and selecting the best cut-point for splitting. A cut-point is a threshold value that divides the range of continuous values into two intervals; one interval is less than or equal to the threshold, and the other interval is greater than the threshold.

- **Step 3:** Splitting the data into two intervals using the cut-point selected in step 2.

- **Step 4:** Continuing discretization with each intervals until a stopping criteria is satisfied. The stopping criteria is usually selected by considering a trade-off between lower arity (the number of intervals or the number of bins) and its effect on the accuracy of classification tasks. A higher arity can complicate the understanding of an attribute, while a very low arity may damage predictive accuracy negatively.

Cut-Point Selection:

Given a set S of sorted continuous values $A_1, A_2, ..., A_N$, candidate cut-points are usually selected as mid-points of every successive pair of A_i and A_{i+1}. On the other hand, candidate cut-points are

$$T_j = \frac{A_i + A_{i+1}}{2}$$

where $i = 1, ..., N-1$ and $j = 1, ..., N-1$.

For each cut-point T that splits set S into two subsets S_1 and S_2, the *class entropy* of a subset S_i is defined as

$$Ent(S_i) = -\sum_{j=1}^{k} P(C_j, S_i) log(P(C_j, S_i)).$$

where k is the number of classes $C_1, C_2, ..., C_k$, and $P(C_j, S_i)$ is the proportion of examples in S_i that have class C_j.

To evaluate the resulting class entropy after set S is partitioned into two sets S_1 and S_2, *the class-information entropy of the partition induced by cut-point* T is defined by taking

the weighted average of their resulting class entropies

$$E(A, T, S) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2).$$

The best cut-point selected in step 2 is the cut-point T_A for which $E(A, T_A, S)$ is minimal amongst all the candidate cut-points.

Stopping Criteria:

Given set S and a potential binary partition π_T, specified on S by the given cut-point T, a stopping criteria is used to decide whether or not this partition should be accepted. If the answer is YES, the discretization will continue with each partition given by π_T; otherwise, the discretization process will stop.

Suppose $Prob\{HT|S\}$ is the probability of a YES answer, and $Prob\{NT|S\}$ is the probability of a NO answer. Partition π_T is only accepted if $Prob\{HT|S\} > Prob\{NT|S\}$. However, in practice, there is no easy way to estimate these probabilities directly. Instead, Fayyad and Irani [20] proposed using MDLP to indirectly estimate them.

The minimum description length (MDL) of an object is defined as the minimum number of bits required to uniquely specify that object out of the universe of all objects. To employ minimum description length principle (MDLP) in choosing the stopping criteria, Fayyad and Irani formulated the above problem as a communication problem between a sender and a receiver. It is assumed that the sender has the entire set of training examples, while the receiver has the examples without their class labels. The sender needs to convey the proper class labeling of the example set to the receiver. It says that the partition induced by a cut-point is accepted if and only if the length of the message required to send before partition is more than the length of the message required to be sent after the partition.

By inferring from coding hypothesis, the stopping criteria is defined as follows:

MDLP Criteria:

A partition induced by cut-point T for a set S of N examples is accepted iff:

$$Gain(A,T,S) > \frac{log_2(N-1)}{N} + \frac{\Delta(A,T,S)}{N},$$

where

$$Gain(A,T,S) = Ent(S) - E(A,T,S) = Ent(S) - \frac{|S_1|}{N}Ent(S_1) - \frac{|S_2|}{N}Ent(S_2),$$

and

$$\Delta(A,T,S) = log_2(3^k - 2) - [kEnt(S) - k_1Ent(S_1) - k_2Ent(S_2)],$$

where k, k_1, k_2 is the number of classes in S, S_1, S_2

Extensive experiments [20, 57] recommended that this method should be the first choice for variable discretization because it gives a small number of cut-points while maintaining consistency.

The outline of the proposed feature selection method is shown in Algorithm 2.1.

2.3.1.3 Experiments

For experiments, a set of face and non-face patterns of size 24x24 was used. A set of 10,000 face patterns were collected from the Internet. Another set of 10,000 complex non-face patterns was false positives collected by running a face detector based on a cascade of 17 AdaBoost classifiers at different locations and scales on 8,440 images that contained no faces; the images included various subjects, such as rocks, trees, buildings, scenery, and flowers. The 10,000 patterns in each set were divided into a training set of 6,000 patterns and a test set of 4,000 patterns.

Two types of features that are Haar wavelet features and Gabor wavelet features were used in our experiments. Haar wavelet features have been widely used in many face detec-

tion systems [91, 50]. They consists of four kinds of features modeled from adjacent basic rectangles with the same size and shape. The feature value is defined as the difference of the sum of the pixels within the rectangles. In total, 134,736 features were used for training classifiers.

To prove the effectiveness of the proposed feature selection method (*CMI-Multi*), we compared it with two other feature selection methods that are forward feature selection (*FFS*) [95] and a CMI-based method using binary features (*CMI-Binary*) [23, 90] on the data set and feature sets described above. All classifiers were trained using AdaBoost similar to [91].

We chose the forward feature selection proposed by Wu et al. [95] because it has very impressive results, not only reducing significantly the training time of the AdaBoost-based face detection system [91] by about 100 times, but also maintaining comparable performance.

Figure 2.18 shows performance of classifiers trained by Haar feature subsets selected by three feature selection methods. The figure indicates that the proposed method, *CMI-Multi*, outperforms the others while the performances of *FFS* and *CMI-Binary* were comparable to one another.

A similar result was also shown when the three feature selection methods were tested on Gabor wavelet features. In this case, CMI-based feature selection methods clearly outperformed *FFS*, and *CMI-Multi* was confirmed to be more efficient than *CMI-Binary*. Because our proposed method uses same principle as *FFS*, which only trains weak classifiers once, it is extremely fast compared with AdaBoost [91]. We built two cascades of AdaBoost classifiers that use *CMI-Multi* and AdaBoost [91] as feature selection methods. Testing on the standard benchmark MIT+CMU test set, they had comparable performance. However, *CMI-Multi* was trained faster than was AdaBoost by approximately 70 times.

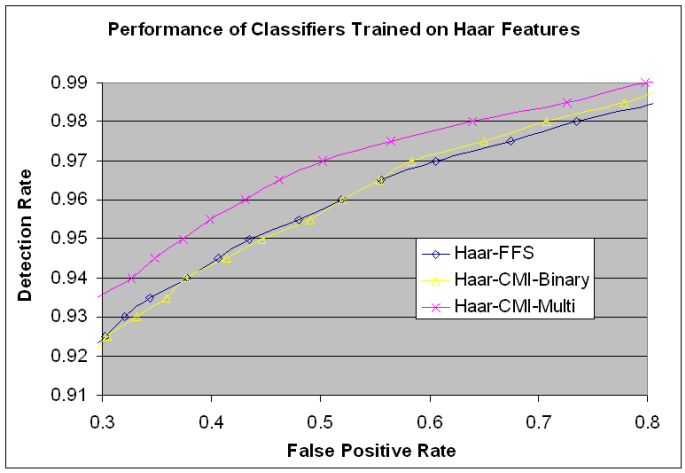

Figure 2.18: Comparison of performance of classifiers trained by subsets selected by different feature selection methods.

2.3.2 Efficient Feature Selection Using Principle Components

In practice, people usually use feature-extraction methods, such as principle-component analysis (PCA), linear discriminant analysis (LDA), and independent-component analysis (ICA) [5, 61, 88], which try to map data from high-dimensional space to lower-dimensional space for feature reduction which is implied as feature selection. However, these methods often suffer from high computation cost when performing projections from the input space to the feature subspace.

To address these problems, we propose a simple yet efficient feature-selection method in which the main idea is to select features whose corresponding axes are closest to principle components computed by PCA from the data distribution. This is a very naive feature-selection method, but experimental results on different kinds of features show that when working with support vector machine (SVM)-based classifiers, our proposed method has comparable performance, but faster speed, compared to a feature-selection method based on PCA directly.

We investigate the principle components computed by PCA in the projection space to select corresponding axes in the original space. Selected axes are those closest to these prin-

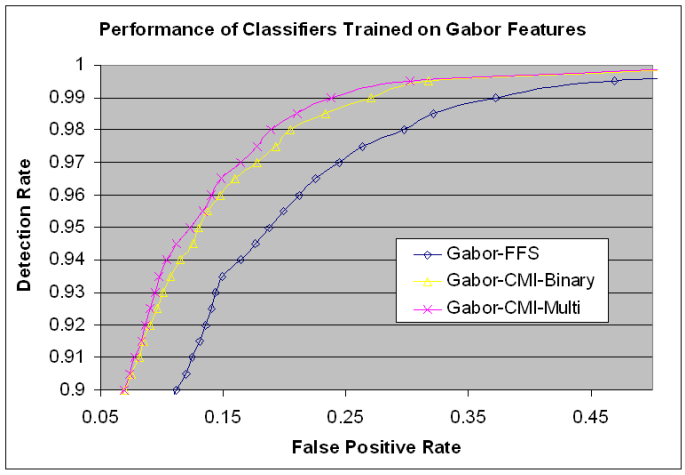

Figure 2.19: Comparison of performance of classifiers trained on Gabor wavelet features.

ciple components. Specifically, starting from each principle component e_i in the projection space, we try to find the principle axis x_j in the original space closest to e_i. As a result, the j^{th} feature will be selected.

The method is illustrated in Figure 2.20. According to the data distribution, e_1 and e_2 are principle components sorted by their corresponding eigen values. By using PCA for feature extraction, we can map data from (x_1, x_2) to (y_1, y_2). And by using the proposed feature-selection method, starting from e_1, x_1, which is the closest to e_1, is found. Hence, the first feature, i.e, x_1, will be selected. The proposed algorithm is summarized as follows:

- **Step 1:** Compute principle components $\{e_1, e_2, ..., e_N\}$ from the data distribution by PCA and sort them in the order of the magnitude of eigen values.
- **Step 2:** For each principle component e_i, find the axis x_j that is closest to e_i.
- **Step 3:** Select feature j^{th}.

2.3.2.1 Experiments

We demonstrated efficiency of our feature-selection method by building a face detector based on SVM. For training, we used face samples and non-face samples mentioned in section 3.6.1.

Algorithm 2.1. CMI-Based Feature Selection Algorithm

Input
(+) the pool of K features F_i, $i = 1..K$
(+) the number of features to be selected T
Output
the set of selected features $\mathcal{F}$

Set $\mathcal{F}$ to be empty
For $k = 1, ..., K$:
(+) Discretize feature F_i using MDLP based method
(+) Compute $I(F_i, C)$, mutual information between feature F_i and binary class variable C
Add F_i into $\mathcal{F}$ where $F_i = arg\ max_{i=1..K}\ I(C; F_i)$
For $t = 1, ..., T$:
(+) Add F_i into $\mathcal{F}$ where
$F_i = arg\ max_{i=1..K} \left\{ min_{F_j \in \mathcal{F}}\ I(C; F_i | F_j) \right\}$

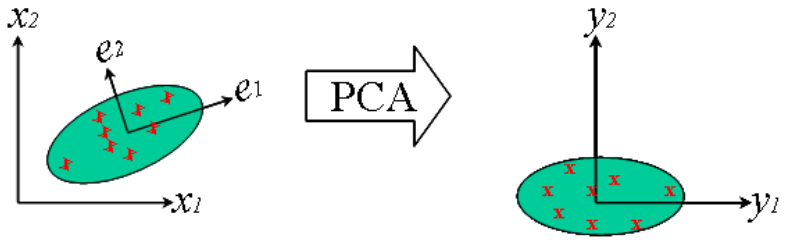

Figure 2.20: Feature extraction by using PCA.

We used the intensity of pixels as features. LibSVM [15] was used to train SVM classifiers with a RBF kernel on selected feature subsets. We compared the performances of SVM classifiers trained on subset features selected by our method and subset features selected from PCA-based feature extraction in which the top-100 and top-200 eigenvectors were used. The results in Figure 2.21 show that the performances of the SVM classifiers are comparable, particularly when the number of features in each subset is large enough, e.g., 200. However, in terms of speed, the SVM classifier trained on a 200-feature set selected by our method can process 86 patterns per second (PPS) while the SVM classifier trained on the top-200 eigenvectors can only process 80 PPS (i.e., approximately 1.08 times slower).

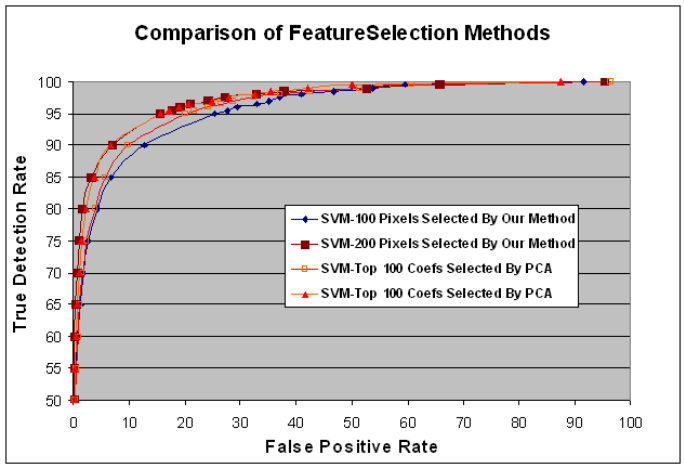

Figure 2.21: Comparable performances of SVM classifiers trained on different feature subsets selected from different selection method when the number of selected features is large enough.

Figure 2.22: Image of 200 pixels (depicted in white) selected by the proposed selection method.

Figure 2.22 shows 200 pixel features selected by our method. It is easy to see that selected pixels belong to major parts of facial features such as eyes, mouth, and nose.

Because the Haar wavelet feature set defined above is over-complete (close to 200,000 features), to use it with SVM [45, 43, 41], first, the maximum 200 features are selected by AdaBoost [24, 91]. Then, from the same feature set, the first-50 features are selected in the order they are added in the training process, and another first-50 features are selected by using our method. The performances of the SVM classifiers trained on these two subsets are shown in Figure 2.23. This figure indicates that, in terms of performance, using our feature-selection method is slightly better than not using it. In terms of speed, the SVM

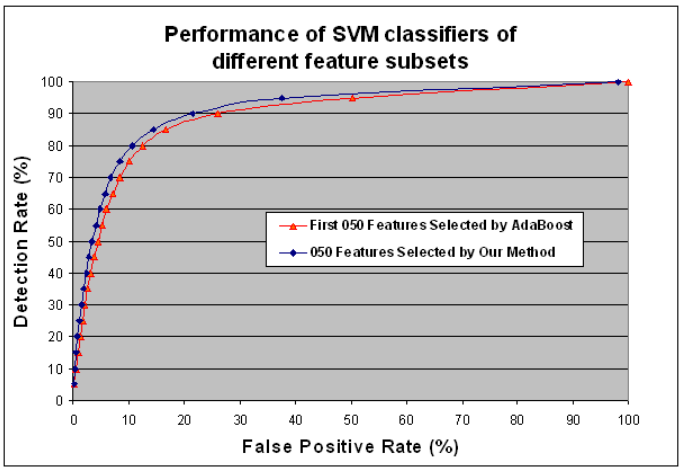

Figure 2.23: Performances of two 50-feature subsets selected by different methods.

classifier trained on the feature subset selected by our method has 3,405 support vectors and runs at a speed of 538 PPS, while that trained on the first-50-feature subset has 4,017 support vectors and runs at a speed of 469 PPS (approximately 1.15 times slower).

2.3.3 Discussion

We have developed two feature selection methods for building fast and robust face detection systems. The first method uses conditional mutual information to filter out quickly irrelevant features from huge feature sets. The estimation of mutual information is simplified by using MDLP-based discretization method. Integrated into AdaBoost-based object detection systems, it can not only reduce the training time significantly but also achieve high classification performance. Experiments on two popular feature sets such as Haar wavelets and Gabor wavelets have demonstrated the effectiveness of the proposed method. A simple yet efficient method for selecting a good feature subset for building object-detection systems. The second method investigates at variance of input data and selects features which are closest to principle components computed by PCA. With this method, by reducing dimensionality of feature vectors, the final classifier runs faster while maintaining high prediction

accuracy. In experiments on different kinds of features used for face detection, the method demonstrated promising results.

Chapter 3

Multi-Stage Approach to Fast Face Detection

3.1 Introduction

Once relevant features of an input pattern are extracted and selected, the feature vector is formed and then passed to classifiers to classify as a face or a non-face. Recently, with advances in machine learning research, neural network [75, 87], support vector machine (SVM) [30, 31, 69, 74], probability density estimation [65, 81] and AdaBoost [36, 51, 55, 49, 91, 92] are typical choices for building robust face detectors.

In a typical face detector that is scale- and location-free, the number of analyzed patterns is usually very large (160,000 patterns for a 320×240 pixel image) because the face classifier has to scan over the input image at every location and every scale (see Figure 3.1). However, the vast majority of the analyzed patterns are non-face. Statistics from [31] have shown that the ratio of non-face to face patterns is about 50,000 to 1. Face detectors based on single classifiers such as SVM [31, 69, 74] and neural network [75, 87] are usually slow because they equally process non-face and face regions in the input image.

To deal with the problem of processing a large number of patterns, a combination of simple-to-complex classifiers has been proposed [31, 36, 74, 79, 91, 97]. In particular, fast and simple classifiers are used as filters at the earliest stages to quickly reject a large number of non-face patterns and slower yet more accurate classifiers are then used for classifying face-like patterns. In this way, the complexity of classifiers can be adapted corresponding to the difficulty in the input patterns. In [74], nonlinear SVM classifiers using pixel-based features were arranged into a sequence with increasing number of support vectors, while

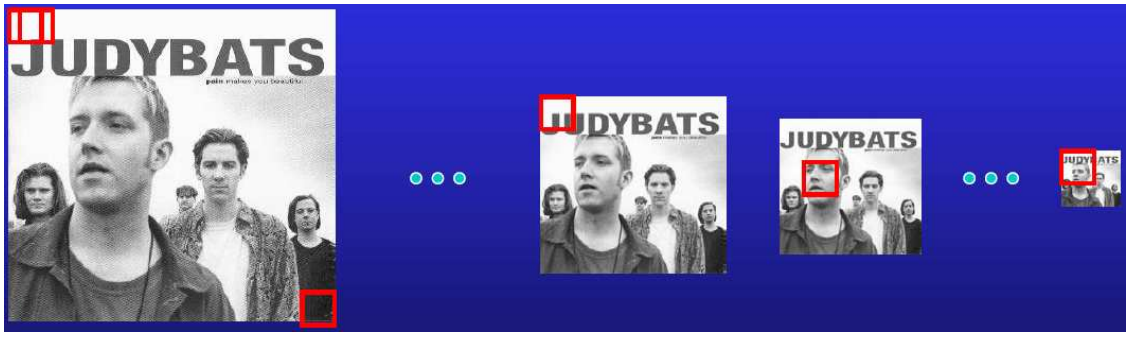

Figure 3.1: A typical face detection process in which the detector scans over the input image at every location and every scale [101].

in [31], linear SVM classifiers trained at different resolutions were used for rejection and a reduced set of principle component analysis (PCA)-based features were used with a nonlinear SVM at the classification stage in order to reduce computation time. In [91], AdaBoost-based classifiers were arranged in a degeneration decision tree or a cascade. Using about 10 features of the first two layers, more than 90% of non-face patterns were rejected. Many researchers believe that the cascade structure of classifiers is the key factor in enhancement of current real-time face detectors. Therefore, a boosting chain [96, 97] and a nested cascade [35, 36] have recently been proposed.

This work is motivated by Viola and Jones [91, 92] who proposed a framework for fast and robust face detection. Their success comes mainly from three contributions:

- The cascaded structure of simple-to-complex classifiers reduces computation time dramatically.

- AdaBoost is used to select discriminative and significant features from a pool of a very large number of features and then construct the classifier. The output classifier built from these selected features is very fast and robust in classification. Compared to SVM-based classifiers or neural network-based classifiers, AdaBoost-based classifiers are hundreds of times faster.

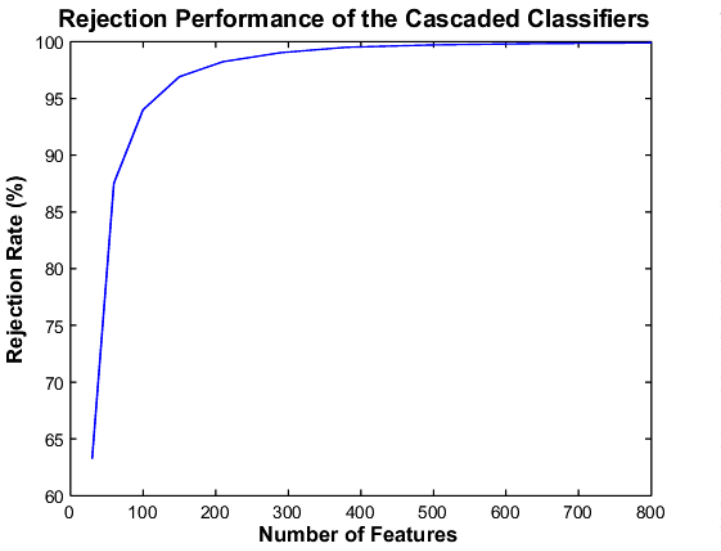

Figure 3.2: Rejection rate versus number of features for cascaded AdaBoost classifiers.

- Haar-wavelet features used for all stages are informative [95] and can be evaluated extremely quickly due to the introduction of the integral image.

However, this framework still has the following problems:

- First, the cascaded classifiers that use AdaBoost and Haar-wavelet features are only efficient in quickly rejecting simple non-face patterns. To robustly classify complex patterns, it is necessary to use a larger number of features and layer classifiers. This need is apparent because when face and non-face patterns become hard to distinguish, weak classifiers are too weak to boost [105]. With the first several layers in our experiment (Figure 3.2), using some 800 weak classifiers, more than 99.9% of non-face patterns were rejected. However, enabling the later layers into robustly classifying a smaller number of remaining patterns, it requires many more, around 5,660, weak classifiers, thus making the training task much more complicated.

- Second, the training process is complicated. It requires a long time because the training time is proportional to the number of features in the input feature set (which is

normally hundreds of thousands) and the number of training samples (which is generally tens of thousands). In our experiment, with 20,000 training samples and 134,736 features, the average training time for choosing one feature associated with the weak classifier was about 30 minutes on a PC (Pentium 4, 2.8 MHz, 512-MB RAM). Therefore, training a cascade of classifiers with around 6,060 features [91] might take on order of several weeks.

Another thing that complicates the training process is that AdaBoost-based classifiers are constructed by adding features after each round of boosting, so several training parameters must be tuned manually while training. In practice, for stopping training a classifier, at least the following three parameters must be determined in advance: minimum detection rate, maximum false positive rate, and maximum number of boosting rounds (or the number of weak classifiers of each layer). Because the complexity of the training sets varies throughout layers in the cascade, a way to choose these parameters automatically and optimally has not been determined. For example, in the first layers, it is quite easy to train a classifier with a minimum detection rate of 99.9% and a maximum false-positive rate of 50%. However, in later layers, choosing the detection rate of 99.9% will give a false positive rate greater than 97% [95]. Adding more features directly increases computation time and might cause over-fitting.

We therefore propose a multi-stage approach to build a face-detection system by adopting the advantages of Viola and Jones' approach and by introducing a method to address the above problems. Specifically, for quick rejection of non-face patterns, we have reused two key ingredients of Viola and Jones' system, that is, the cascaded structure of simple-to-complex classifiers and AdaBoost trained with Haar-wavelet features. Furthermore, for robust classification and simple training, we propose using SVM classifiers for later layers. The contribution of this approach is three fold:

- First, to detect face candidate regions, a new stage (using a larger window size and a larger moving step size) has been added. We use 36×36-pixel window-based classifiers with a moving step size of 12 pixels, to quickly detect the candidate face regions. The idea of using larger windows and moving the step size was proposed in [75], but it severely degraded performance. To improve speed while maintaining high accuracy, our approach takes advantage of the combination of the Haar wavelet features and the AdaBoost learning for fast and robust evaluation

- Second, we have investigated how to efficiently reuse the features selected by AdaBoost in the previous stage, for the SVM classifiers of the last stage. Reusing these features brings to two advantages: *(i) Haar wavelet features are very fast in evaluation and normalization [91]. Furthermore, these features do not need to be re-evaluated because they have already been evaluated. (ii) By using SVM classifiers with powerful generalization, using too many features in the cascade is avoided, with the important results of saving training time and avoiding over-fitting.*

- Third, the training time of AdaBoost classifiers has been shortened by using simple sampling techniques to reduce the number of features in the feature set. Experiments showed that for rejection, the performance gained by using a sampled feature set was comparable to that of a full feature set. Along with using several SVM classifiers instead of many AdaBoost classifiers in later layers, the total training time has been significantly reduced.

3.2 Related Work

Several studies have worked on addressing the drawbacks of Viola and Jones' system. Wu et al. [95] used direct feature selection to reduce training time while maintaining comparable performance. Their idea is to separate the training process into two stages: feature selection

and classifier construction. In Viola and Jones' work, features are selected by the discriminative performance of their associated weak classifiers through the boosting process. This process is very time consuming because all weak classifiers must be trained every time one feature is selected. With the new proposal of Wu et al., weak classifiers are trained only once and features are selected by the direct feature selection method which directly maximizes the learning objective of the output classifier. They claim that their method is 100 times faster than Viola and Jones' method.

Another direction is to optimally build the cascade to improve its overall performance. Sun et al. [86] and Bourdev and Brandt [12] proposed a scheme to optimally tune parameters in layer classifiers. However, their approaches are somewhat complicated and are not easy to implement. Xiao et al. [97] and Huang et al. [35, 36] proposed a boosting chain structure in which subsequent layers utilize the historical information of the previous layers. This significantly reduces the number of features used in each layer. Discrete AdaBoost uses a binary weak classifier that is too weak to boost in the case of a hard distinguished dataset. Studies based on Real AdaBoost [78], such as [36, 37, 50, 55, 49, 60], introduced new kinds of weak classifiers that are stronger than binary weak classifiers. These new real-valued weak classifiers can effectively discriminate face and non-face distributions, so the total number of features used is also reduced dramatically. Face detection systems such as [36, 50] only used around 800 features. However, the main problem with these systems is how to choose the most appropriate number of bins. A small number of bins might not accurately approximate the real distribution while a large number of bins might cause over-fitting, increase computation time and waste storage space. Even that, our system can benefit from this approach when building the rejection stage and can thus reduce the training time even further.

Skin color was also used in face detection [25]. The main advantages of this approach are fast computation and rotation-invariant. However, several issues must be addressed to build a robust face detector. For example, how to choose appropriate color model to

model variations in lighting conditions and races and how to handle multiple adjacent faces efficiently? So far, skin color is only used to speed-up face detection systems by finding face candidate regions.

3.3 System Overview

The proposed face detection system consists of three stages that classify a 24×24-pixel window as either a face or a non-face. To detect faces of different sizes and locations, the detector is applied at every location and scale in the input image with a scale factor of 1.2, which is similar to the other approaches [31, 75, 87]. An outline of this system is given in Figure 3.3 and Figure 3.4.

The first stage is a cascade of classifiers used to detect face candidate regions by evaluating 36×36-pixel input windows, with a moving step of 12 pixels. If a 36×36-pixel window is detected as the existence of a face, 144 (i.e. 12×12) likely face positions are collected and passed to the next stage. The second stage is a cascade of classifiers used to investigate 24×24-pixel windows extracted from face candidate locations returned from the previous stage.

The main purpose of designing these two stages is trying to filter out a large number of non-face patterns as quickly as possible before passing complex patterns to the final stage classifier. This is done by taking advantages of Viola and Jones' approach [91], in which Haar wavelet features and the cascaded AdaBoost classifiers enable extremely fast computation .

Although the cascade of 24×24 AdaBoost classifiers rejects non-face patterns rapidly, it is still influenced by the large number of 24×24 patterns that it must process. For this reason, the first stage, which is a cascade of 36×36 classifiers, is added to decrease the number of analyzed patterns. To this end, this stage is trained specially to make the classifiers invariant to small face translations. These classifiers can detect faces that are off-center by up to six pixels in up-down and/or left-right directions. An illustration of the

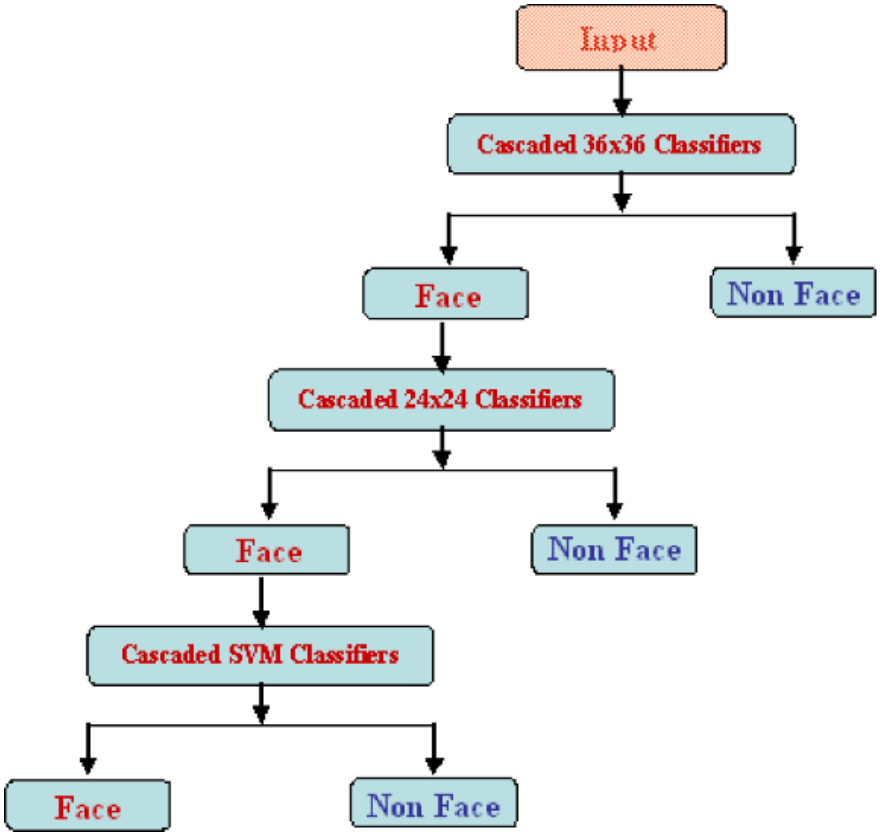

Figure 3.3: Three-stage face detection system.

difference between 24×24 and 36×36 face training samples is depicted in Figure 3.5. The 36×36-pixel window is chosen in accordance with the idea in [75] stated that the classifier can be trained to be invariant to translation by up to 25% of the original window size. With this flexible classifier, the moving step size can be increased by up to 12 pixels to dramatically reduce the number of analyzed patterns. The efficiency of this stage will be discussed further in section 3.6.3.

The last stage is a cascade of nonlinear SVM classifiers that reuses features that have been selected by a AdaBoost classifier in the second stage classifier . These feature values are evaluated and scaled to be between 0 and 1 to form a feature vector. In our experiments, only 100 features were used, making classification faster than it would have been using pixel-based SVM classifiers [31, 74].

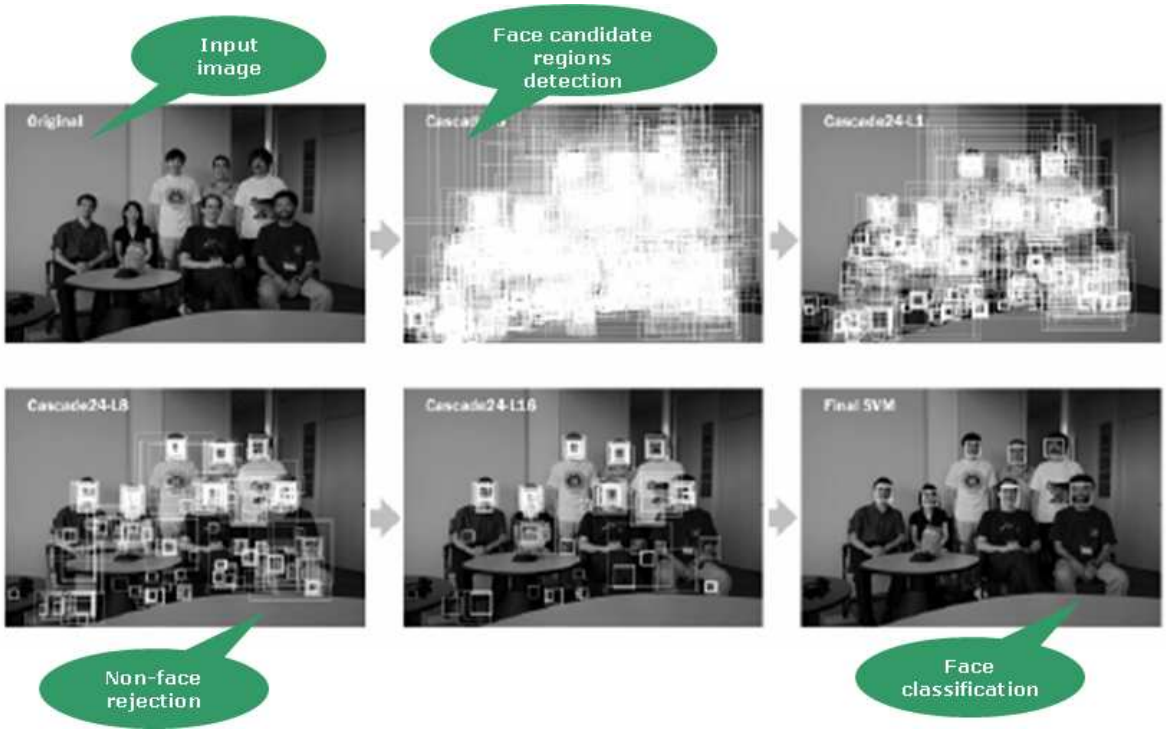

Figure 3.4: Face detection process using the multi-stage approach.

3.4 Training Cascaded Classifiers

3.4.1 AdaBoost Learning

Boosting is used to improve the classification performance of any given simple learning algorithm [24]. Given T weak classifiers $h_t(x)$ learned through T rounds of boosting, the

Figure 3.5: Difference between the cascade of 24×24 AdaBoost classifiers (CAB24) and the cascade of AdaBoost 36×36 classifiers. CAB24 is trained to detect 24×24 face patterns located exactly at the center of the 24×24 input window (left), while CAB36 can detect the presence of a 24×24 face pattern that might be off-center by up to six pixels in up-down and/or left-right directions (right).

strong classifier is formed by a linear combination:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x),$$

where α_i are coefficients found in the boosting process.

Each weak classifier h_j is associated with a feature f_j and a threshold θ_j such that the number of incorrect classified examples corresponding to the weak classifier is minimized:

$$h_j(x) = \begin{cases} 1 & if\ p_j f_j(x) < p_j \theta_j \\ 0 & otherwise \end{cases},$$

where polarity p_j indicates the direction of the inequality sign.

In each round of boosting, the best weak classifier h_t that has the lowest error ϵ_t will be chosen. The error of each weak classifier is measured with respect to the set of weights over each example of the training set:

$$\epsilon_j = \sum_{i=1}^{N} w_i \left| h_j(x_i) - y_i \right|,$$

where w_i and y_i are the weight and the label of the training example x_i, respectively. After each round, these weights are updated such that the weak learner will focus much more on the hard examples in the next round.

3.4.2 Cascade of classifiers

The main idea of building a cascade of classifiers is to reduce the computation time by giving different treatments to different complexities of input windows (Figure 3.6). Only input windows that have passed through all layers of the cascade are classified as faces.

Training cascaded classifiers that can achieve both good detection rate and less computation time is quite complex: a higher detection rate requires more features, but more

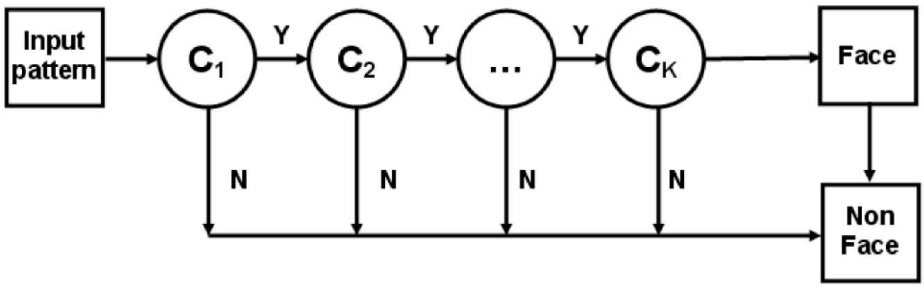

Figure 3.6: A cascade of classifiers for object detection.

features correspond to more time needed for evaluation. To simplify this, the detection rate goal and the false positive rate goal for each layer are usually set beforehand. Viola and Jones [91] stated that, if the layer classifier has not achieved the predefined target goals after 200 features are used, the training process will stop and a new layer will be added.

3.5 SVM classifier

The support vector machine is a statistical learning method based on the structure-risk minimization principle. It has been very efficiently proved in many pattern recognition applications [14, 31, 74]. In the binary classification case, the objective of the SVM is to find the best separating hyperplane with a maximum margin.

The form of SVM classifiers is:

$$y = sign(\sum_{i=1}^{N} y_i \alpha_i K(x, x_i) + b),$$

where x is the d-dimensional vector of an observation example, $y \in \{-1, +1\}$ is a class label, and x_i is the vector of the i^{th} training example. N is the number of training examples and $K(x, x_i)$ is a kernel function. $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_N\}$ is learned by solving the following

quadratic programming problem:

$$minQ(\alpha) = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j),$$

subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C,\ \forall i.$$

C is a predefined parameter that is a trade-off between a wide margin and a small number of margin failures. All the x_i corresponding to non-zero α_i are called support vectors.

It is important to choose the appropriate kernel and parameter C in order to obtain the robust SVM classifier. Although many kernels have been introduced by researchers, the following four kernels are commonly used:

- linear: $K(x_i, x_j) = x_i^T x_j$.
- polynomial: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$.
- radial basis function (RBF): $K(x_i, x_j) = exp(-\gamma \left\| x_i - x_j \right\|^2), \gamma > 0$.
- sigmoid: $K(x_i, x_j) = tanh(\gamma x_i^T x_j + r), \gamma > 0$.

where γ, r and d are kernel parameters.

Compared to AdaBoost classifiers, SVM classifiers run much more slowly because of the large number of support vectors and the heavy kernel computation. To control the trade-off between the number of support vectors and errors, Scholkopf et al. [82] proposed using a new parameter ν ($0 \leq \nu \leq 1$) instead of the parameter C. They proved that the parameter ν is an upper bound of the fraction of margin errors and a lower bound of the fraction of support vectors. The descriptions and implementations of C-SVM and ν-SVM are provided by LibSVM [15, 16].

Figure 3.7: Face patterns used for training the 24×24 window-based classifier.

Figure 3.8: Face patterns used for training the 36×36 window-based classifier.

3.6 Experiments

3.6.1 Experiment Setup

For training, we collected 7,500, 24×24-size face patterns from the Internet. Non-face patterns were generated at different locations and scales from 8,440 images with various subjects, such as rocks, trees, buildings, scenery, and flowers, containing no faces. Some examples of the collected 24×24 face patterns are shown in Figure 3.7.

Face patterns for training the 36×36 classifiers are generated by selecting 36×36 windows containing the 24×24 face window of the input image. Figure 3.8 shows some examples of 36×36 face patterns that include various kinds of floating positions and backgrounds.

To train the cascade of 24×24 AdaBoost classifiers used in the rejection stage, the same 7,500 face patterns were used for all layers. Non-face patterns of the training and the validating sets of the first layer in the cascade were selected randomly. Non-face patterns of the subsequent layer classifiers are false positives collected by the partially trained cascade on the set of non-face images. For each layer classifier, 7,500 non-face patterns were used for training and 7,500 other non-face patterns were used for validating.

The same Haar wavelet feature set as proposed in [91] was used in these experiments. To compare the performance of classifiers, we implemented a full cascade of classifiers trained by AdaBoost, similar to that used by Viola and Jones [91]. The training parameters of each layer were set as follows. The minimum of the detection rate was 99.7%, the maximum of the false positive rate was 50.0% and the maximum of the number of features in each layer was 200. This setting resulted in a face detector that consists of 38 layers with 6,360 features.

All experiments were run on a PC (Pentium 4, 2.8 MHz, 512-MB RAM). The training process was terminated when no more false positives were found in the non-face images of the data set.

3.6.2 Simplification of Training the Rejection Stage

If K is the number of Haar wavelet features and N is the number of training patterns, the learning time of AdaBoost to train M weak classifiers is roughly $O(KNM)$ [91]. Therefore, if the number of training patterns is fixed, the learning time can be shortened when either the number of features in the feature set or the number of weak classifiers in the final cascade is reduced. In our approach, the cascaded classifiers are only used for efficient rejection, so we can reduce both of these numbers in order to keep the training time for the full system reasonable.

As mentioned in section 2.2.1.1, each feature is parameterized by a tuple of four parameters (x, y, Dx, Dy). A set of features is then formed by changing these parameters in corresponding steps $(Stepx, Stepy, StepDx, StepDy)$. A feature set, on the other hand, is pa-

rameterized by $(x+a.Stepx, y+b.Stepy, Dx+c.StepDx, Dy+d.StepDy)$. One of the simplest ways to sub-sample the feature set is to change parameters $(Stepx, Stepy, StepDx, StepDy)$, for example, from a full feature set $(1, 1, 1, 1)$ to a reduced feature set $(1, 1, 2, 2)$. Because the full feature set is redundant, this sub-sampling is expected not to significantly affect the rejection performance of AdaBoost classifiers.

We carried out experiments to compare the performance of classifiers trained on these two feature sets: the full feature set $(1, 1, 1, 1)$ containing 134,736 features and the reduced feature set $(1, 1, 2, 2)$ containing 14,807 features (excluding features of the small size). Two classifiers were trained up to the maximum of 200 features. The classifier's threshold was changed to meet the detection rate of 99.7%. The training set contains 7,500 face patterns and 7,500 non-face patterns. Rejection performance was evaluated through the false positive rate on a validation test set that contains 500,000 non-face patterns. All non-face patterns were selected randomly from the training set mentioned above.

The results shown in Figure 3.9 indicates that the performances of these two classifiers were no different, especially when the number of features was large enough, for example, more than 50. As a result, by using the reduced feature set, the training time can be shortened to approximately one-ninth.

Another experiment we conducted showed that, for similar performance, an AdaBoost classifier trained on the reduced feature set that uses larger sampling step sizes requires more features than one trained on the full feature set. Therefore, only the sampling parameter $(Stepx, Stepy, StepDx, StepDy) = (1, 1, 2, 2)$ was used in training the 24×24 AdaBoost classifiers.

3.6.3 Efficiency of the Cascaded 36×36 Classifiers

In our system, the first stage is a cascade of classifiers that processes 36×36 patterns with a moving step size of 12 pixels. By taking advantage of simplification in training classifiers only for rejection, as demonstrated in section 3.6.2, training this cascade only uses the feature set

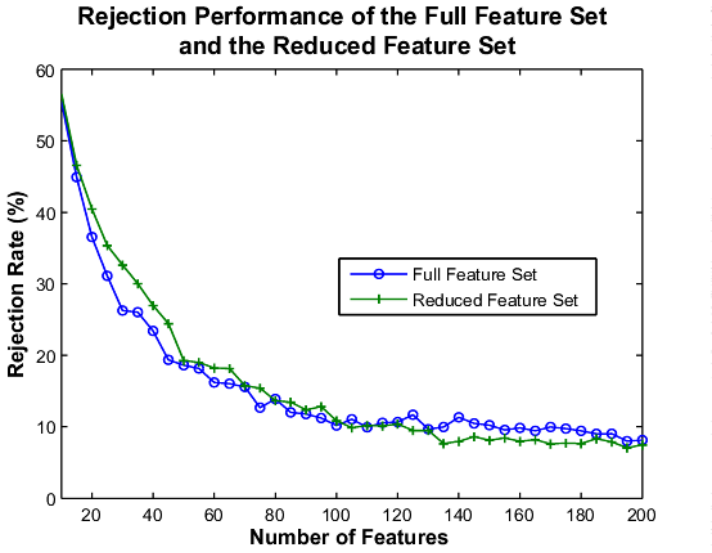

Figure 3.9: Rejection performance of classifiers trained on the full feature set and the reduced feature set.

generated from a 36×36 window with sampling parameters $(2.5, 2.5, 2.5, 2.5)$. As a result, 12,223 features are produced. The training set contains 12,000 face patterns and 12,000 non-face patterns. Since a 36×36 face sample contains a large portion of background outside the 24×24 face region and the classifier is required to be fast and to keep all possible face regions, a minimum detection rate of 99.9% and a maximum of false positive rate of 70.0% were set as the training parameters. In our experiments, after reaching 50 features, the classifier's performance did not significantly increase, so the maximum number of features for each layer is set to 50. To keep a balance between computation speed and robustness, the maximum number of layers is set to three because using more layers would degrade the overall detection rate dramatically.

Figure 3.10(a) shows several features of the first 36×36 layer classifier selected by AdaBoost. They look somehow similar to the features of the first 24×24 layer classifier as shown in Figure 3.10(b). In addition, Figure 3.11 shows an example of face candidate regions detected by using this cascade.

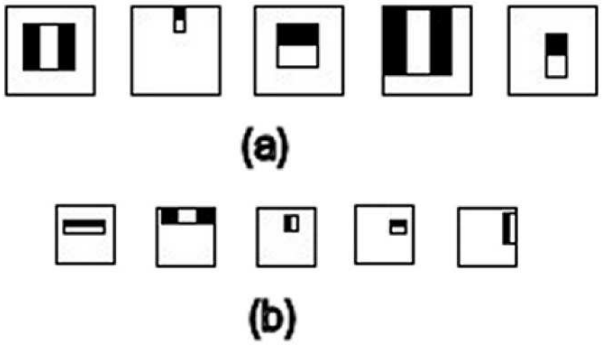

Figure 3.10: The features selected by AdaBoost of the first layer when training the 36×36 classifier (a) and training the 24×24 classifier (b).

3.6.4 Features Selected by AdaBoost for SVM

Two main issues surrounding the reuse of features selected by AdaBoost are: *(i) which layer's features should be reused for SVM? and (ii) how many features should be used?*

For comparison of the performance of SVM classifiers, 2,450 face patterns and 7,500 non-face patterns that were separated from the training set (section 3.6.1) were used. The SVM classifiers were trained with a RBF kernel whose parameter γ is 0.0625. The parameter ν was set to 0.15. These parameters were found by using cross-validation tests.

Figure 3.12 compares the performance of classifiers trained on 200-feature sets selected by different layers in the cascade (layers 14, 17, 20, and 25). These comparable performances suggest that the second stage (using AdaBoost) can be switched to the final stage (using SVM) at any time. As a result, the total training time of the system can easily be controlled.

To determine the number of features is that would be sufficiently robust, we used the 200-feature set selected in layer 17 to generate different subsets of features with different numbers of features. Features in each set were selected in the order in which they were added in the training process. For example, a 25-feature set consists of the first 25 features selected by AdaBoost when training layer 17. The results shown in Figure 3.13 indicate that with more than 100 features, the performance of the classifiers was comparable.

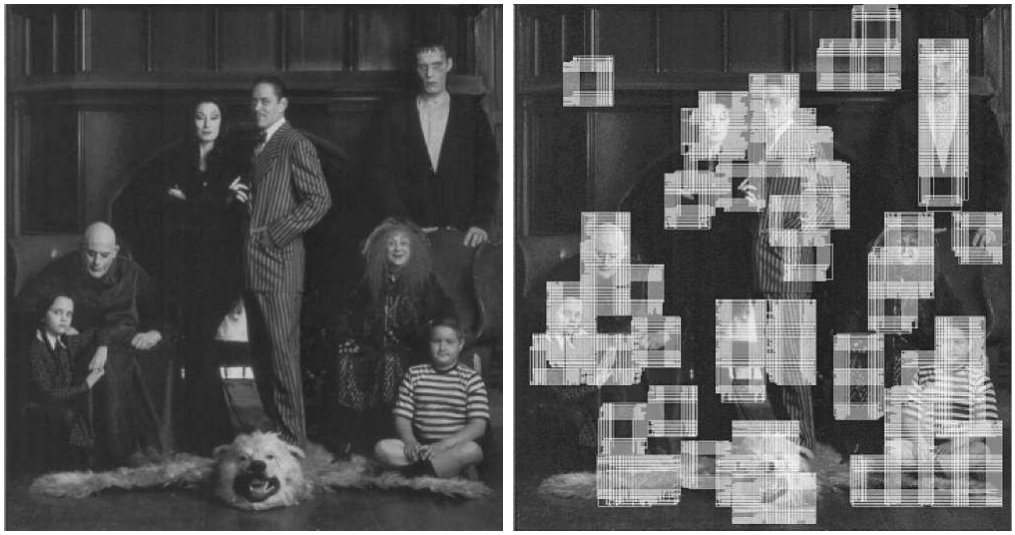

Figure 3.11: Face regions estimated by 36×36 classifiers: (left) original image and (right) candidate face regions.

Basically, the speed of a SVM classifier is proportional to the number of features used, so the greater the number of features used, the slower the classifier will be. Figure 3.14 shows the processing speed of SVM classifiers using different subsets of features. The SVM classifier using 25 features ran the fastest while the SVM classifier using 200 features was the slowest. The speeds of SVM classifiers using 100, 125 and 175 features were not importantly different because their difference in terms of number of features and number of support vectors were not large enough to have a significant impact. Therefore, 100 features might be the best trade-off between speed and performance.

3.6.5 Efficiency of SVM classifiers

We carried out an experiment to show the efficiency of a single SVM classifier over a cascade of AdaBoost classifiers. In this experiment, 40,000 false positives were gathered by running a cascade of 17 AdaBoost classifiers (*CAB17*) on the set of non-face images mentioned in section 3.6.1. These false positives then were used as hard non-face patterns to train and test the performance of two classifiers: a single RBF SVM classifier and a cascade of other

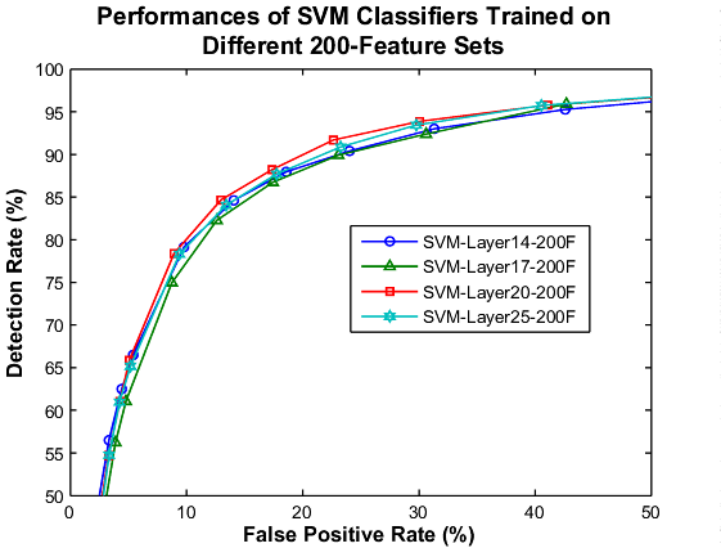

Figure 3.12: Performance of nonlinear SVM classifiers with different 200-feature sets.

18 AdaBoost classifiers. Of 40,000 non-face patterns, 7,500 non-face patterns were used along with 7,500 face patterns to train these two classifiers. The remaining 32,500 non-face patterns and other 2,450 face patterns were used to compare the accuracy of the classifiers. The cascade of AdaBoost classifiers were trained with the parameters set as in section 3.6.1. The RBF SVM classifier reused 100 features selected by the last layer of *CAB17* as the feature vector and was trained by an RBF kernel whose parameter γ is 0.0625. The parameter ν was set to 0.15. These parameters were found by using cross-validation tests.

The result shown in Figure 3.15 demonstrates that with hard classified patterns that later layers of the cascade will process, the single SVM classifier can achieve higher accuracy than the cascade of AdaBoost classifiers trained by roughly predefined training parameters. Furthermore, the training time of a single SVM (which takes several hours) is much shorter than that of a cascade of AdaBoost classifiers (which might take several weeks).

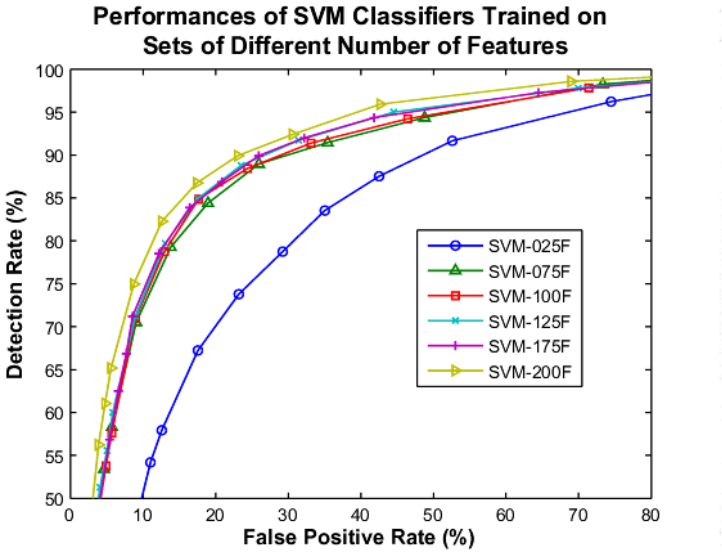

Figure 3.13: Performance of nonlinear SVM classifiers on different number of features.

3.6.6 Performance Comparison

The final system consists of three stages. In the first stage, the cascaded 36×36 classifiers consist of three layers, making for a total of 120 features. The second stage consists of 17 layers with 2,160 features. Compared to the system with 6,061 features used in [91], our system uses fewer features and, can thus save significant training time (which is approximately 27 times in total).

The final stage is a cascade of three SVM classifiers that takes 100 features of the last layer in the second stage as the feature vector. Each SVM classifier was trained by using 7,500 face patterns and 7,500 non-face patterns. The same 7,500 face patterns were used in training all these SVM classifiers. By running the cascade of AdaBoost classifiers of the second stage on the set of non-face images, 40,000 false positives were collected and used as non-face patterns to train the SVM classifiers. The 7,500 non-face patterns used to train the first SVM classifier were selected randomly from the 40,000 non-face patterns. Non-face patterns in the subsequent SVM classifiers were false positives collected by the partially cascaded SVM classifiers on these 40,000 non-face patterns. To control the number

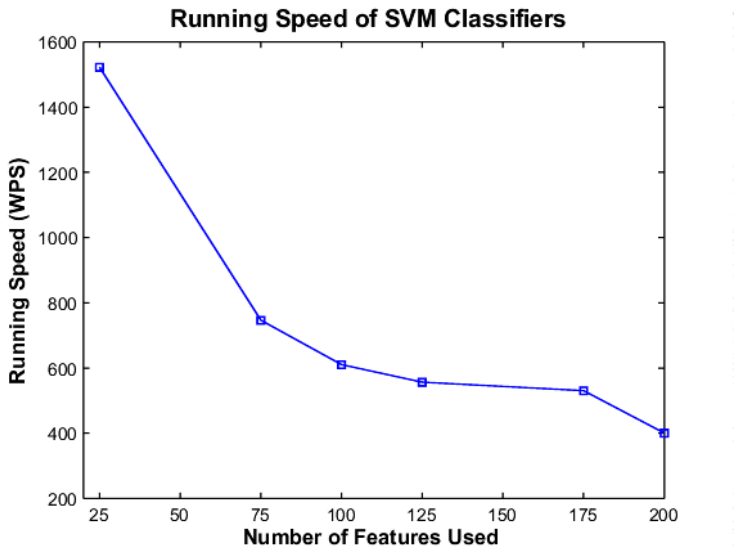

Figure 3.14: Pattern evaluation speed of nonlinear SVM classifiers.

of support vectors, the parameter $\nu = 0.15$ was used instead of the parameter C. All SVM classifiers were trained by using the RBF kernel with $\gamma = 0.0625$. All these parameters were found by using the cross validation test tool provided by LibSVM [15]. This training procedure yielded three SVM classifiers whose numbers of support vectors are 4,725, 5,043, and 4,847 respectively. The average evaluating speed of a SVM classifier is approximately 610 WPS (windows per second).

We tested our system on the MIT+CMU frontal-face standard test set [75] which consists of 124 images with 480 frontal faces (excluding images containing hand-drawn, cartoon and small faces). The configuration and rejection performance of the classifiers are listed in Tables 3.1 and 3.2. The first row presents the number of features of each layer, and the second row shows the fraction of the remaining patterns after each layer was processed. The last row indicates the fraction of time that each layer consumes. All these statistics were extracted by running the classifiers on the MIT+CMU test set.

The fraction of the remaining patterns on these two tables indicates that most of the non-face patterns, i.e., 70.5%, were rejected by the first stage, the cascade of 36×36 AdaBoost classifiers. When the first 24×24 layer classifier was added to the cascade of 36×36 classifiers,

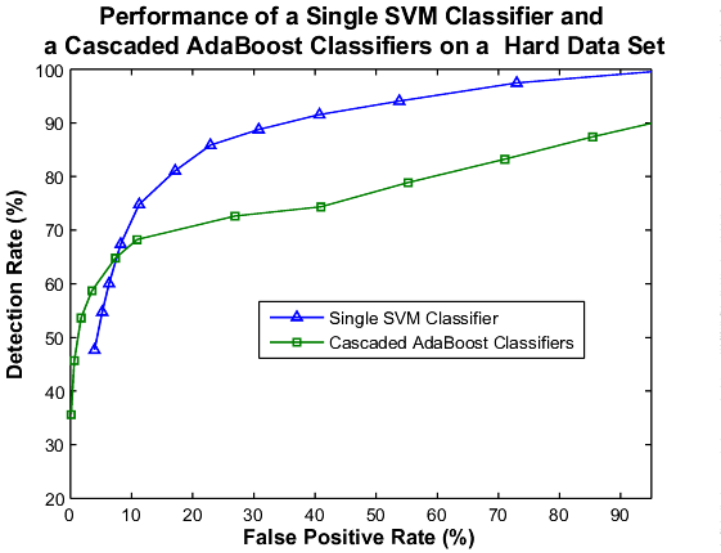

Figure 3.15: Performance of a single SVM classifier and a cascade of AdaBoost classifiers on hard classified patterns.

this combination rejected 85.91% of analyzed patterns compared to 73.22% of using only the first layer of the single cascade of 24×24 classifiers. Furthermore, the rejection of this very large number of patterns was done extremely quickly, only using 15.95% of the total processing time. It also showed that most of the processing time used by the AdaBoost+SVM system, 44.99%, was used for SVM classifiers.

The detection rate and speed of the classifiers with ten false positives are listed in Table 3.3. It is clear that our multi-stage system ran faster than the single cascade of 24×24 AdaBoost classifiers while achieving comparable detection rates. This performance was possible for three reasons. First, the cascade of 36×36 AdaBoost classifiers rejected many non-face patterns extremely quickly while slow SVM classifiers only processed a very small number of the remaining patterns. Second, many images in the MIT+CMU test set contain large portion of background, which [31] mentioned has a ratio of non-face to face patterns of about 50,000 to 1. Experimental results showed that the AdaBoost+SVM system ran faster than that of the original AdaBoost on 75% of the total number of images in this test set. Third, at a small number of false positives, some true face candidate regions rejected

Table 3.1: Configuration and rejection performance of a cascade of 24×24 AdaBoost classifiers with 38 layers

	Layer 01	Layer 02	Layer 03	...	Layer 17	...	Layer 38
Number of features	10	20	30	...	200	...	200
Remaining patterns (%) after each layer	36.78	12.50	6.02	...	0.03	...	0.01
Processing time (%) of each layer	36.97	24.14	10.99	...	0.24	...	0.15

Table 3.2: Configuration and rejection performance of final AdaBoost+SVM classifiers

	Layer 01*	Layer 02*	Layer 03*	Layer 01+	Layer 02+	Layer 03+	...	Layer 17+	Cascade SVMs
Number of features	20	50	50	10	20	30	...	200	100
Remaining patterns (%) after each layer	73.93	51.30	29.50	14.09	5.89	3.18	...	0.02	0.01
Processing time (%) of each layer	0.67	1.14	0.71	13.33	10.98	6.43	...	0.27	44.99

(*)Layers of the cascade of 36×36 *AdaBoost classifiers*
(+)Layers of the cascade of 24×24 *AdaBoost classifiers*

by 36×36 classifiers did not severely affect the final performance because they might also be rejected by 24×24 classifiers in later layers.

Some detection results are given in Figure 3.16.

3.6.7 Robustness to Face Variations

We used the Yale dataset A [7] and B [26] to show the robustness of our face detector to face variations such as lighting conditions, facial expressions and occlusions. The Yale A dataset contains 165 images of 15 subjects. There are 11 images per subject, one for each of the following facial expressions or configurations: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink. Subjects and images of one subject in the Yale A dataset are shown in Figure 3.17 and Figure 3.18. The Yale B

Table 3.3: Performance comparison at ten false positives

	Detection Rate (%)	Speed (WPS)
Cascades of AdaBoost + Cascade SVM	81.6	927,478
Cascade of AdaBoost - 38 layers	81.2	789,818
Viola and Jones[91]	78.3	N/A

dataset contains 9 subjects, each subject has 65 images with different lighting conditions (see Figure 3.19) which are changed due to the azimuth and elevation of the single light source direction. As for the Yale B dataset, we divided 65 lighting conditions into 7 symmetric subsets based on azimuth values (see Figure 3.20): [-10, +10] (117 images), [-15, -35] (81 images) and [+15, +35] (99 images), [-50, -70] (63 images) and [+50, +70] (63 images), [-85, -130] (81 images) and [+85, +130] (81 images).

For each dataset, we applied our face detector in two settings, with and without doing histogram equalization (HE) on the input image for handling illumination conditions. As shown in Figure 3.21for the Yale A dataset, our face detector can achieve high detection rate (92.7% recall and 9.4% precision). There is no significant difference between using or not using histogram equalization. It is reasonable because the lighting conditions have not seriously affected face appearances. Some detection results are shown in Figure 3.22.

However, for the Yale B dataset, using histogram equalization can help to improve the recall from 51.1% to 81.5% as shown in Figure 3.23 and Figure 3.24. This processing step is useful when lighting conditions seriously affect face appearances.

3.7 Conclusion

We have developed a method to build a fast and robust face detection system based on a multi-stage approach. The cascaded structure of AdaBoost-based classifiers in the two first stages allows the system to best adapt to various complexities of input patterns, while non-

linear SVM classifiers at the final stage are robust enough to achieve good results. Extensive experiments demonstrated that a significant computation time is devoted to potential face regions because almost all non-face patterns are rejected quickly by the two first stages, and only a very small number of face-like patterns are processed by the slow SVM classifiers. Discriminant Haar wavelet features selected from AdaBoost are used for all stage classifiers to take advantage of their efficient representation and fast evaluation.

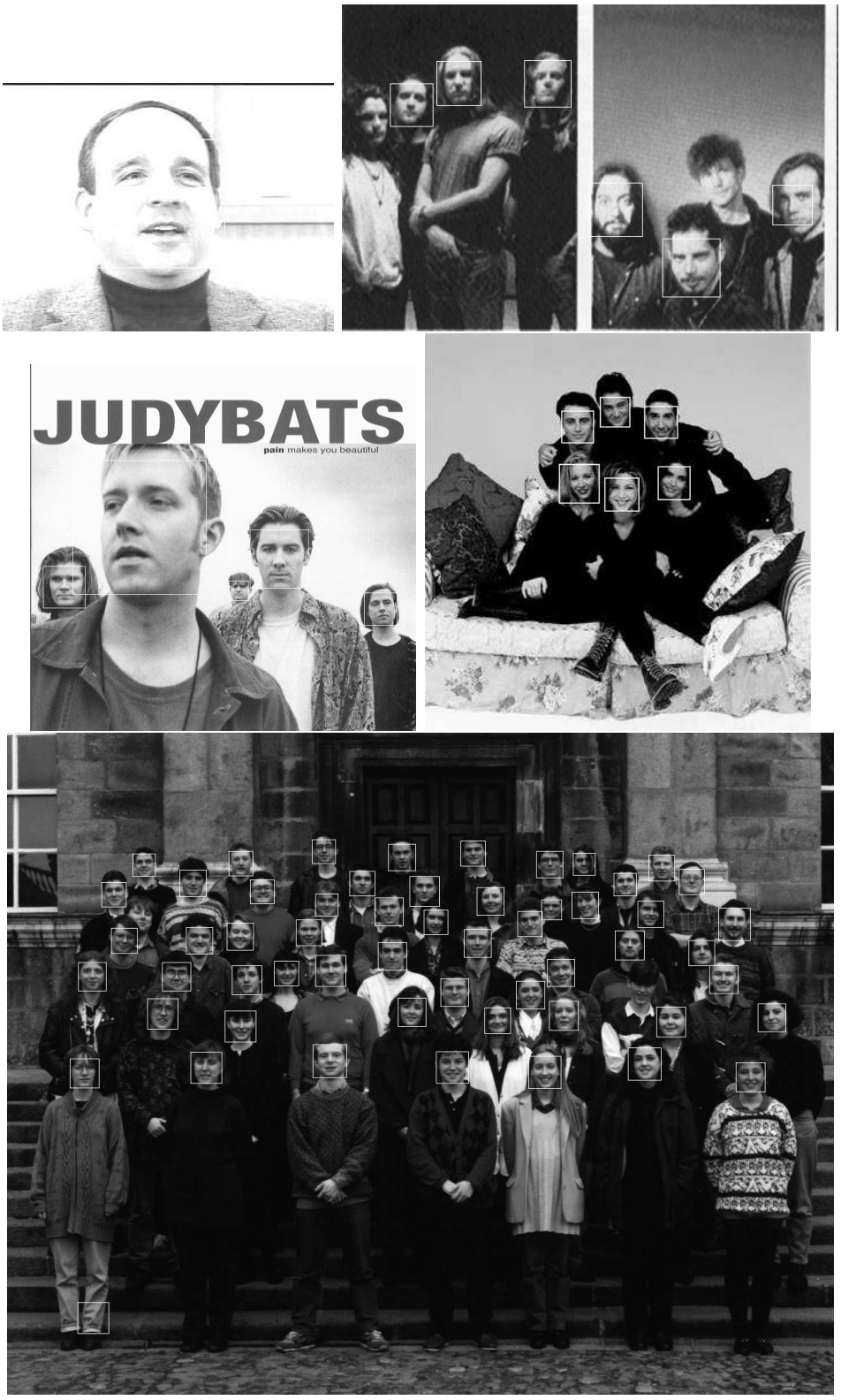

Figure 3.16: Detection results with our system on test images from the MIT+CMU test set.

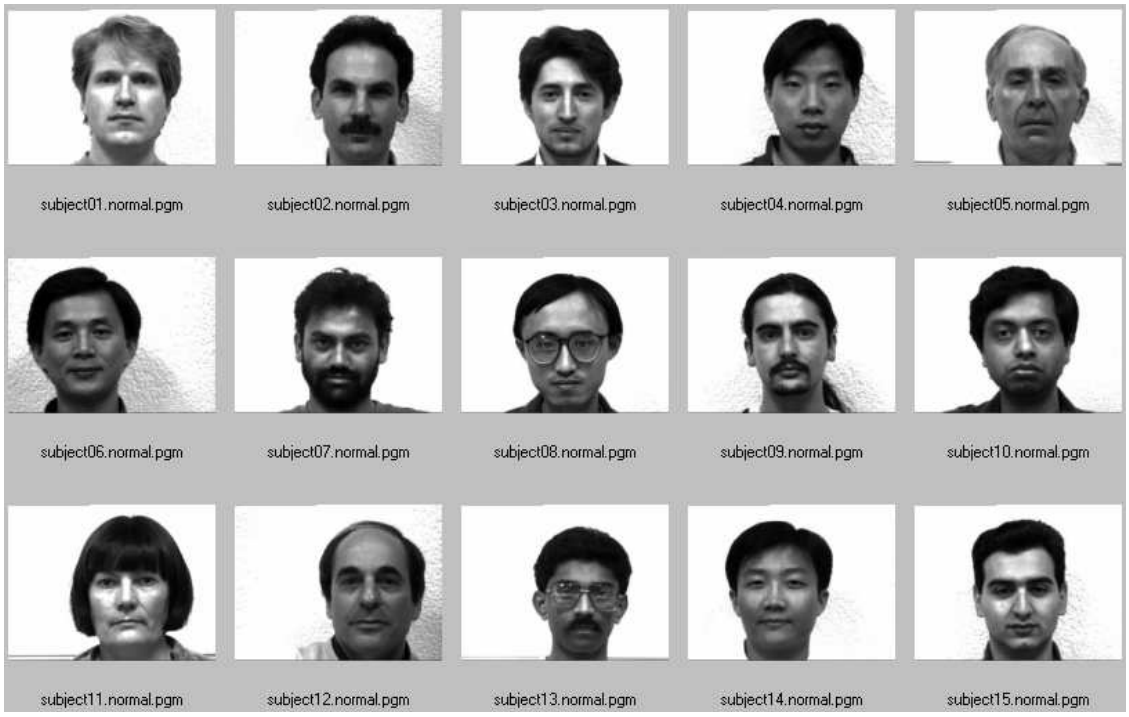

Figure 3.17: Subjects in the Yale A dataset.

Figure 3.18: Various face appearances of one subject in the Yale A dataset.

Figure 3.19: Different lighting conditions for acquiring face images in the Yale B dataset [26].

Figure 3.20: Face samples with different lighting conditions of divided subsets.

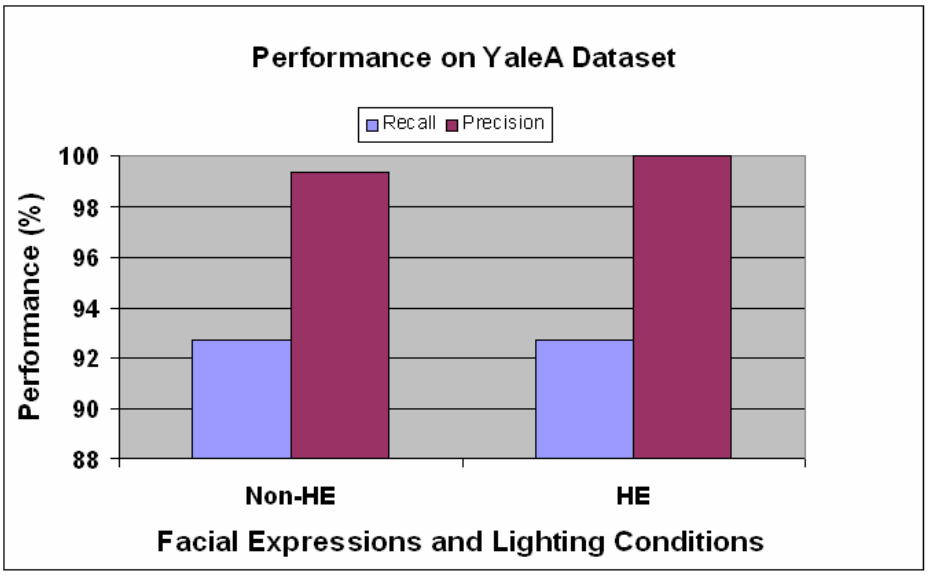

Figure 3.21: Performance on the Yale A dataset with and without using histogram equalization.

Figure 3.22: Detection results on the Yale A dataset: (left) without using histogram equalization, (right) with using histogram equalization.

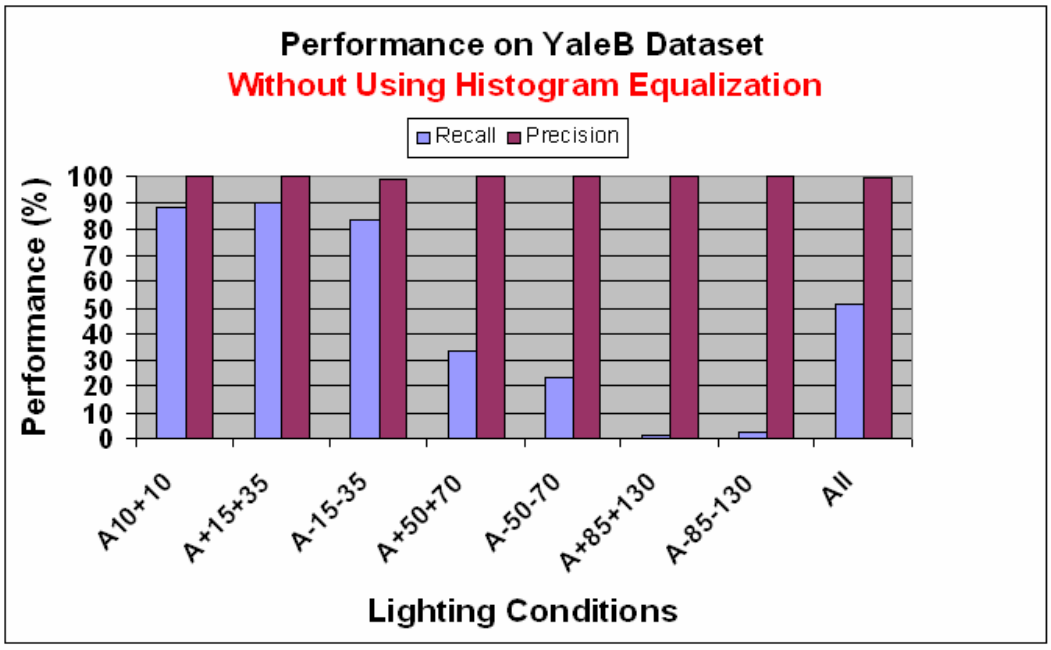

Figure 3.23: Performance on the Yale B dataset without using histogram equalization.

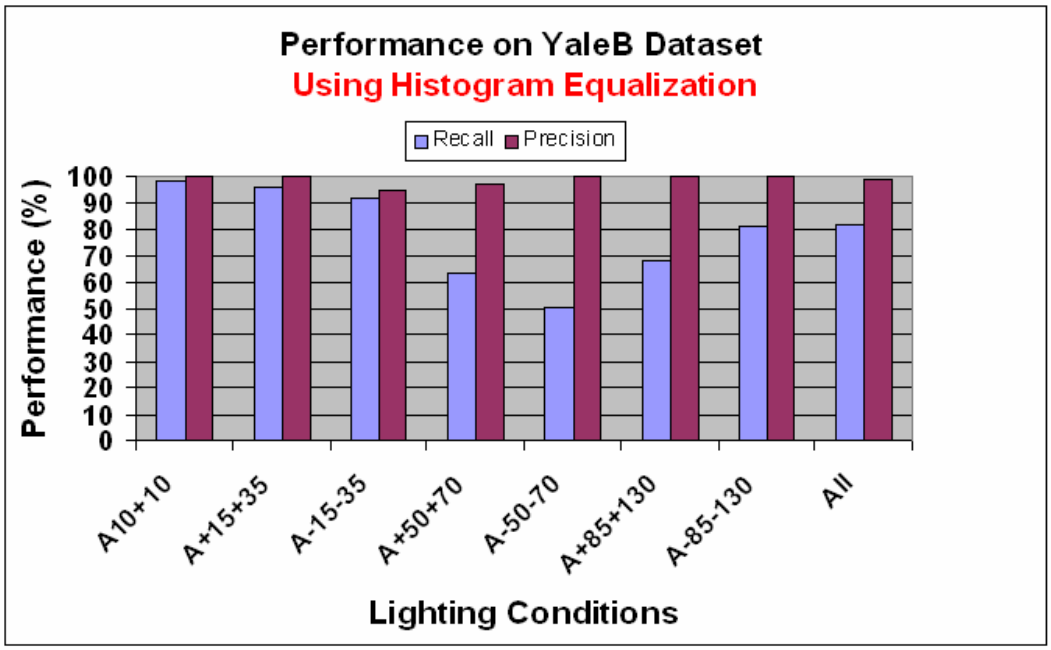

Figure 3.24: Performance on the Yale B dataset with using histogram equalization.

Chapter 4

Large Scale Video Indexing and Retrieval Using Human Faces

4.1 Introduction

Explosion of many multimedia databases needs to have effective and scalable tools for indexing and retrieving based on video contents. For example, in broadcasting news video databases, it is desirable for the system to be able to organize the video data into person-involved stories so that users can easily find and browse all events involving a specific person.

As described in section 1.2, there are many general and specific challenging problems for extracting and organizing faces from news video data. Several approaches have been proposed to handle these problems. For example, to eliminate face appearance variations, Sivic et al. [83] modeled a face sequence as a histogram of quantized facial features. Shots containing principal actors of a movie were retrieved using a similarity measure between two histograms. Other work [4, 21, 22] also showed good face retrieval results for movies by developing new similarity measures that are invariant to affine-transformations, partial occlusions. However, compared to news video, the number of persons of interest in movies is much smaller, although they appear more frequently and distinctively.

In [76, 77], low quality results for face recognition, name entity extraction from transcripts and video-caption recognition were integrated with temporal information to boost the overall accuracy of retrieval. However, their experiments were only carried out on a single small-sized video dataset. In [98], video shots related to a named individual were found by exploring various information sources from video data, such as names appearing in transcripts, face information, and most importantly, the temporal alignment of names and faces. Their results

were promising, but their use of face information was very limited, and additional reference images of the target face under various conditions were required to be provided in advance. In [99, 100], faces were labeled with their corresponding names using supervised learning methods such as SVM and multiple instance learning. With supervised learning methods, good generalization power can only be achieved if large training sets are provided; however, producing and annotating such training sets can be very labor intensive.

To avoid high variations of detected faces in video, Zhai et al. [104, 103], instead of using detected faces, used the 'body', an extended face region (e.g. the neck) for comparison of two faces in detecting anchor persons. However, this method is less robust for face retrieval.

In an effort to reduce the number of retrieved images presented to users and thereby to improve the precision, clustering can be used to generate representative examples. However, most clustering methods cannot be applied to large, high dimensional datasets such as those typically associated with video image processing. For such applications, k-means has been a favorite method due to its simplicity; however, it suffers from a number of serious drawbacks. First, it cannot be applied to general similarity measures. Second, the number of clusters must be provided in advance. Third, k-means optimizes according to a global criterion, often resulting in the formation of many clusters with relatively poor internal association. Finally, in the case of very large high-dimensional datasets, scalability and convergence problems make it difficult to obtain reasonable results [39].

Recently, Berg et al. [9, 8] proposed an impressed method to organize faces and names appearing in Internet news documents in meaningful face clusters in which each cluster corresponds to one individual shown by his name. However, compared to news video, faces and names in Internet news documents are less noisy, better image quality, strongly related.

In this study, we propose a face retrieval system that is distinguished from previous work by the following features.

First, representative faces are automatically organized in advance and available for users to browse by using the relevant set correlation (RSC) clustering model introduced in [33].

The GreedyRSC clustering heuristic based on this model avoids all the problems of k-means clustering listed above. An overview of the clustering model and heuristics is presented in section 4.2.

Second, faces appearing in video are aligned with possible corresponding names extracted from the closed caption text, by using a machine translation method [66]. By this way, important people with their identity and visual appearance can be mined from the video.

Third, our framework is general and has the potential to handle very large scale video datasets effectively and efficiently.

4.2 RSC Clustering Model

The clustering strategy employed in this paper is based on the *relevant-set correlation* (RSC) model proposed by Houle [33]. RSC clustering can be viewed as a generalized nearest-neighbor clustering strategy, in which distance information is used only to produce ranked lists of neighbors ('relevant sets') for items in the data set. Under the model, the quality of cluster candidates, the degree of association between pairs of cluster candidates, and the degree of association between clusters and data items are all assessed according to the statistical significance of a form of correlation among pairs of relevant sets and/or candidate cluster sets. In this section, the RSC significance measures are introduced and briefly discussed; full details can be found in [33].

4.2.1 Internal and External Association

For any data set S, any subset $A \subseteq S$ can be represented as a zero-one set membership vector of length $n = |S|$, where a given coordinate is set to 1 whenever its associated item is present in S. The RSC model assesses the degree of association between two non-empty sets $A, B \subseteq S$ by applying the standard Pearson correlation formula to the sequence of coordinate

pairs formed by the set membership vectors, yielding the following *set correlation* formula:

$$\textsc{r}(A,B) = \frac{|S|}{\sqrt{(|S|-|A|)(|S|-|B|)}}\left(\frac{|A\cap B|}{\sqrt{|A|\,|B|}} - \frac{\sqrt{|A|\,|B|}}{|S|}\right).$$

A set correlation of 1 is achieved only when A is identical to B; otherwise, the correlation value is strictly less than 1.

Intuitively speaking, for an item $v \in A$ to be considered well-associated with the remaining items of A, one would expect those items of S that are highly relevant to v to belong to set A as well. The RSC model assesses the internal association of a candidate cluster set A as the average of the correlations between A and all relevant sets of size $|A|$ based at an item of A. The *self-correlation* of A is thus defined as:

$$\textsc{sr}(A) \triangleq \frac{1}{|A|}\sum_{v\in A} \textsc{r}(A, \textsc{q}(v,|A|)),$$

where $\textsc{q}(v,|A|)$ is the relevant set for item v of size $|A|$. A self-correlation of 1 is achieved when the relevant sets of all members of A perfectly coincide with A.

4.2.2 Significance of Association

In general, when making inferences involving Pearson correlation, a high correlation value alone is not considered sufficient to judge the significance of the relationship between two variables. When the number of variable pairs is small, it is much easier to achieve a high value by chance than when the number of pairs is large. For this reason, to help interpret correlation scores, statisticians resort to tests of significance (such as the t-test) that account for variation in the number of pairs.

Under the RSC model, associations are measured against a null hypothesis in which all relevant sets of items are assumed to have been produced by means of random selection from the full data set. Under the 'randomness' hypothesis, the mean and standard deviation of

the self-correlation score can be calculated. Standard scores (also known as Z-scores) of two actual cluster candidate sets can then be generated and compared. The more significant candidate would be the one whose standard score is higher — that is, the one whose self-correlation score exceeds the expected value by the greatest number of standard deviations.

The RSC significance measure for cluster candidate A is given by:

$$Z(A) = \frac{\text{SR}(A) - \mathbf{E}[\text{SR}(A)]}{\sqrt{\mathbf{Var}[\text{SR}(A)]}} = \sqrt{|A|\,(|S|-1)}\,\text{SR}(A).$$

Since $|S|$ can be regarded as a constant, using $Z()$ to rank cluster candidates is equivalent to using the following 'normalized-squared' significance statistic:

$$Z_*(A) = \frac{Z^2(A)}{|S|} = |A|\,\text{SR}^2(A).$$

The normalized-squared statistic has the advantage of being easier to interpret. For integer $k > 1$, a significance score of $Z_*(A) = k$ is the level of significance attained by a perfectly-associated cluster of size k — that is, one for which the same-sized relevant set of every member coincides with the cluster.

For inter-set association , testing $\text{R}(A, B)$ against the null hypothesis that set B was generated by random selection from S gives the following standard score:

$$\frac{\text{R}(A,B) - \mathbf{E}[\text{R}(A,B)]}{\sqrt{\mathbf{Var}[\text{R}(A,B)]}} = \sqrt{|S|-1}\,\text{R}(A,B).$$

Since $|S|$ is regarded as constant, the significance measure used for RSC inter-set association is simply the set correlation $\text{R}(A, B)$ itself.

4.2.3 Cluster Reshaping

Within any highly-significant set A, the contributions of some relevant sets to the self-correlation may be substantially greater than others. Those items whose relevant sets contribute highly can be viewed as better associated with the concept underlying aggregation A than those whose contributions are small. It turns out that the contributions to the overall significance of A are partitionable among its constituent members according to the formula

$$Z(A) = \frac{1}{\sqrt{|A|}} \sum_{v \in A} Z(A, v), \quad \text{where} \quad Z(A, v) = \sqrt{|S| - 1}\, \mathrm{R}(A, \mathrm{Q}(v, |A|)).$$

Members within a cluster can be re-ranked in order of their contributions $Z(A, v)$, thereby enhancing the power of the underlying similarity measure. This also suggests that candidate A can be improved by modifying its membership to produce a new set A', for which the following significance score is maximized:

$$Z(A, A') = \frac{1}{\sqrt{|A'|}} \sum_{v \in A'} Z(A, v).$$

4.2.4 Clustering Strategy

The RSC-based clustering method presented in [33] seeks to generate as many clusters as possible, subject to the following restrictions:

- Every selected candidate item set A should meet a minimum threshold value of cluster quality, as measured by $Z(A)$.

- All pairs of selected cluster candidates (A, B) should meet maximum threshold values on cluster similarity, as measured by $\mathrm{R}(A, B)$.

If a region of the data is sufficiently well-associated for a subset to meet or exceed the minimum threshold on cluster quality, then a cluster should be chosen to represent the region.

However, if two or more highly-similar cluster candidates arise from within the region, then only one of the candidates should be retained.

The selection of cluster candidates can be viewed within the framework of the well-studied family of independent vertex set problems for graphs. Those cluster candidate item sets whose quality scores meet the minimum threshold are mapped onto vertices of a graph, with assigned weights equal to the quality scores. A vertex pair is joined by an edge wherever the inter-cluster similarity scores of the corresponding cluster candidates exceeds the maximum threshold. RSC clustering thus reduces to the problem of selecting a subset of graph vertices that maximizes some objective function involving such variables as subset size and vertex weights, subject to the restriction that no graph edge may have both of its endpoints selected.

The clustering heuristic presented in [33], GreedyRSC, employs a greedy strategy for cluster selection whereby candidates with the highest quality are selected first, and any candidates found to be overly-similar to a previously-selected candidate are declared to be redundant, and then eliminated. GreedyRSC also incorporates the following heuristic design choices:

- The quadratic cost of cluster quality evaluation is avoided by strictly limiting the size of all relevant sets considered to be at most some constant $b > 0$.

- The discovery of clusters of arbitrarily-large size is facilitated by first computing small tentative clusters with respect to a range of data samples of varying sizes. GreedyRSC treats the tentative clusters as *patterns* for the explicit generation of full-sized clusters, by reshaping them with respect to the full dataset as described above.

- The number of candidate clusters is restricted by considering only relevant sets of sample items as the eligible candidate patches or patterns.

- The cost of generating relevant sets in practice is reduced by using approximate neighborhoods as generated using the efficient and scalable SASH similarity search struc-

ture [34]. Experiments on a variety of large, very high-dimensional data sets (such as text, protein sequences, and images) have shown that the SASH consistently returns a high proportion of the true k-nearest neighbor set at speeds of roughly two orders of magnitude faster than sequential search. Furthermore, it offers better performance and significantly better control over the time-accuracy trade-off, than previous approximation methods based on metric indices.

The GreedyRSC heuristic also seeks to reduce the total size and number of candidate cluster sets generated, by eliminating redundant patterns and cluster candidates at intermediate stages of the clustering process.

For more details regarding the GreedyRSC clustering heuristic, its implementation using SASH, and its performance, see [33].

4.2.5 SASH-based Similarity Search

One of the difficulties faced for clustering of very large multi-dimensional data sets is the relatively high cost of performing similarity searches. Generally speaking, the use of search indices for high-dimensional data suffers from an effect often referred to as the 'curse of dimensionality' , where the cost of computing exact k-nearest neighbors with respect to meaningful distance measures approaches that of a sequential scan of the full data set. This has led researchers and practitioners to develop techniques for approximate similarity search in the hope of substantial speedups over sequential scan.

The SASH structure proposed by Houle [34] is an efficient and scalable data structure for approximate k-nearest neighbor search. Experiments on a variety of large, very high-dimensional data sets (such as text, protein sequences, and images) have shown that the SASH consistently returns a high proportion of the true k-nearest neighbor set at speeds of roughly two orders of magnitude faster than sequential search. Furthermore, it offers better

performance and significantly better control over the time-accuracy trade-off, than previous approximation methods based on metric indices.

The SASH organizes the data set S into a multi-level structure of random samples as follows. At construction time, half the data items are selected at random to form the bottom level set S_0 of the SASH. The items of $S \setminus S_0$ are then recursively organized into a smaller SASH (the 'sub-SASH'). Each item of S_0 is then integrated into the structure by searching for and then connecting to a small number of approximate nearest neighbors chosen from the bottom level of the sub-SASH.

At query time, an approximate k'-nearest neighbor query is first performed in the sub-SASH for some k' dependent on both k and the set size; the precomputed links are then followed to obtain candidate neighbors from the bottom level S_0. Finally, the closest items found are reported as the approximate nearest neighbor set.

In contrast with tree-based methods proposed to date, most query result objects are reachable via multiple paths through a relatively compact portion of the structure. The use of path redundancy and random sampling allows the SASH to automatically shape itself to the data set even when the underlying distribution is completely unknown, a greatly desirable feature for clustering applications [32, 33]. For more details and experimental results regarding the SASH structure and implementation, see [34].

4.2.6 Advantages of the RSC Clustering Model

The advantages of the RSC clustering model over typical clustering models include:

- It can be applied to any dataset for which ranked relevant sets can be efficiently generated whenever a dataset item is treated as a query-by-example. The model does not depend on the precise value of the underlying similarity measure except for the purpose of generating ranked relevant sets.

- Items can appear in more than one cluster. This allows the model to assess the association between two clusters according to the degree of correlation (overlap) between their set memberships.

- The model can assess the quality of internal association of clusters independently from other clusters. The model is not forced to accept a poorly-associated cluster in order to satisfy some global optimization criterion.

- Clustering heuristics based on the model can automatically determine an appropriate number of clusters over a large range of sizes (even as few as 3 or 4 items).

Heuristics based on the RSC model, supported by fast approximate similarity search techniques, SASH, have been shown to scale to handle dataset of millions of objects represented in thousands or even millions of dimensions [32, 33, 34, 47].

4.3 Implementation of a News Video Indexing and Retrieval System

4.3.1 TRECVID Dataset

We used TRECVID 2004 dataset [29, 1] for demonstration of our framework. The data set consists approximately 133 hours of CNN and ABC news from January 1998 to June 1998, with commercials, sports and graphics galore. A typical news program is 30 minutes long and consists of roughly 54,000 video frames. For efficient management, the news video programs were partitioned into news stories and shots in advanced. The average number of shots in one news program is about 100. There were 4,376 news stories extracted manually from 218 annotation files. The quality of frame images was quite low due to digitization process.

Figure 4.1: Face extraction from news video - face regions detected by a face detector (top), and faces after normalization (bottom).

4.3.2 Face Extraction and Normalization

We used our fast and robust face detector [42, 43, 45] presented in chapter 3 to detect all faces with minimum size of 32x37 pixels. The face detector also detects eye locations of the detected faces. To group all faces belonging to one person, we used a simple tracking method based on estimating sizes and locations of faces in consecutive frames. It produced 18,200 faces for which two eyes were clearly visible. On average, there are 4 faces detected from one news program. The running time was 72 hours on a 3.0GHz PC Pentium IV with 2GB RAM.

Eye positions provided by the face detector were used to align the faces to a predefined canonical pose. To compensate for illumination effects, the subtraction of the best-fit brightness plane followed by histogram equalization was applied as in [75]. Next, the faces were scaled to a size of 52x60 pixels, and an elliptical mask was applied so as to remove the background. The results of these steps are shown in Figure 4.1. The robustness of our face detector is shown in Figure 1.2, 1.3, 1.4.

We then used PCA [88] to reduce the number of dimensions of the feature vectors for face representation. Projection vectors were generated from 3,816 frontal faces with different variations taken from the FERET database [72]. The faces were normalized as described above, and then used to calculate the mean face and the eigenfaces corresponding with the largest 786 eigenvalues. This number was selected so as to retain 97% of the total energy. Some of the eigenfaces are shown in Figure 4.2.

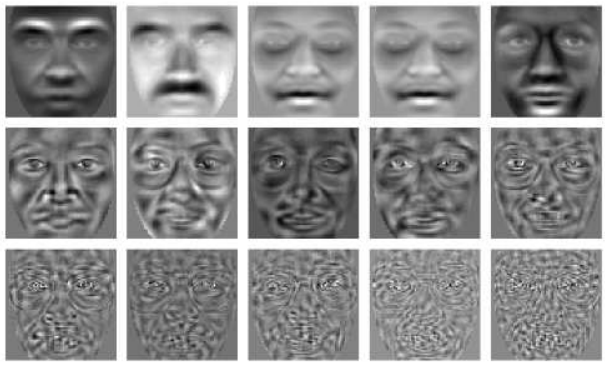

Figure 4.2: Some eigenfaces used to form the subspace for face representation.

4.3.3 Person Name Extraction

To extract person names from video closed caption texts, we used LingPipe, a state-of-the-art suite of natural language processing tools written in Java that performs tokenization, sentence detection, named entity detection, coreference resolution, classification, clustering, part-of-speech tagging, general chunking, fuzzy dictionary matching [2]. Since annotation texts of news stories provided by TRECVID are in lower case, we firstly used the tagging tool of LingPipe to find proper nouns in the texts and capitalize them. Next, the named entity recognition tool of LingPipe is used to extract all personal names from the news story. It produced 4,028 distinct names. Figure 4.3 shows an example of a news story with extracted names, faces and representative frames.

4.3.4 Performance of RSC clustering

Applying GreedyRSC to the TRECVID faces produced 661 clusters after 30 minutes of execution on a 3.0GHz PC Pentium IV with 2GB RAM. In order to produce approximate k-nearest neighbor lists for use by GreedyRSC, the SASH was tuned for an average accuracy of 98% at a speed of 6 times faster than sequential search. We set the parameter of norm-squared significance score to 0.6 in order to ensure the faces in each cluster highly relevant.

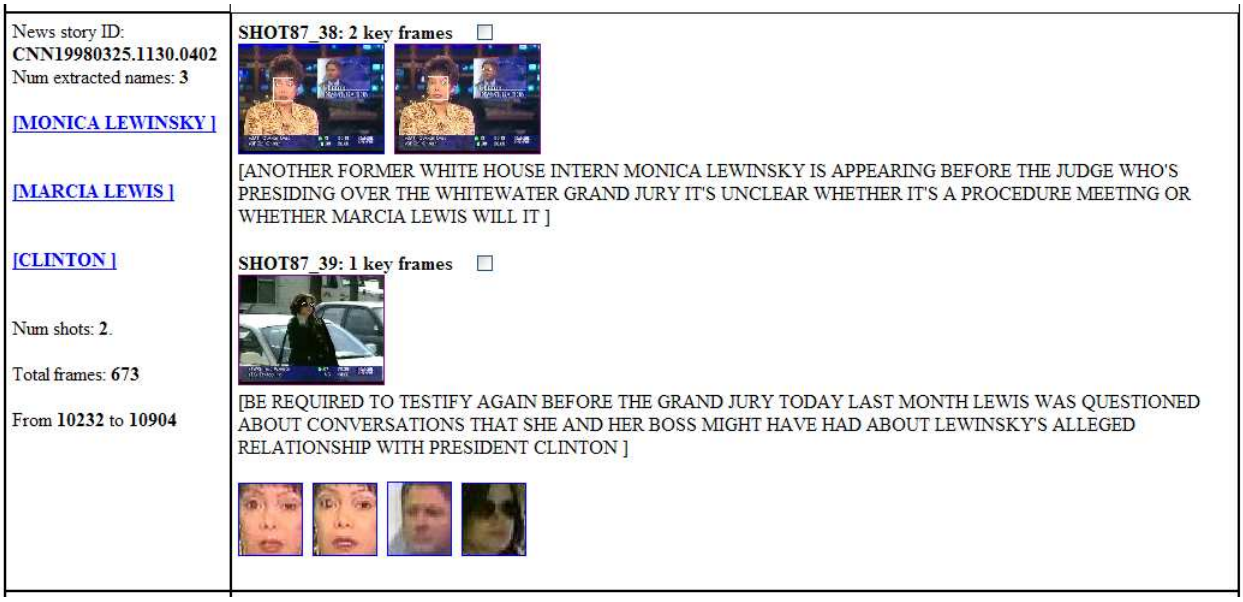

Figure 4.3: An example of a news story with extracted names, faces and representative frames.

The resulting clusters had sizes ranging from 3 to 72. Of the 18,200 faces, approximately 80% of faces were not assigned to any clusters. This is not unreasonable since many faces appeared fewer than four times in the dataset. Figure 4.4 shows faces in one cluster. Representative faces of several of the clusters are shown in Figure 4.5.

4.3.5 Faces and Names Association

We followed the approach that Duygulu et. al described in [19] to align faces with names. In this approach, the problem of face and name matching was modeled as a machine translation problem that translates visual elements to words. Given a set of pair sentences (one sentence in the source language and one sentence in the target language), several methods [13] were proposed to find correspondences between words in these languages.

We treated each news story as a basic unit to form a pair of sentences. In each news story, extracted faces represent for English language and extracted names as French language. The GIZA++ tool [66] was used to match names to face clusters. Only top three alignment candidates were used to show to users. This process is shown in Figure 4.6.

Statistics

Cluster index [0..269]:	57	Pattern base item index:	8477
Cluster size:	7	Pattern size:	9
Fringe size (incl. cluster):	7	Pattern sample size:	18200
Norm-squared significance:	6.49919	Pattern sample level:	-2
Normalized significance:	2.54935	Average item-to-pattern confidence:	0

Cluster 57 Summary

Cluster ID	Cluster Size	Norm-Squared Significance	Inter-set Correlation
57	7	6.49919	

Cluster 57 Members

Member Rank	Members	Correlation	Significance
1		1	1
2		0.888834	1.78385
3		0.888834	2.57181
4		0.777668	3.1601
5		0.777668	3.75498
6		0.777668	4.35316
7		0.666502	4.76796

Figure 4.4: Faces of one cluster found by GreedyRSC.

For each news story, we have $N \times M$ possible face-name associations where N is the number of faces and M is the number of names. By investigating all news stories in the database, the best correspondence of faces and names are found. For example, in the two example news story, the Clinton's face can be aligned with the names in the first news story: *Sam, Clinton and Albright* and the names in the second news story: *Clinton and Wolf Blitzer*. By taking the co-occurrence information of faces and names, the best match for the Clinton's face is the name *Clinton*.

Since extracted faces and names are still noisy, we used their occurrence frequency to remove unimportant faces and names before passing them to the matching process. In Figure 4.7 (top), we shows an example of our face and name association result in which the name *Clinton* with the highest occurrence frequency is assigned correctly to the cluster.

Figure 4.5: Representative faces of several clusters found by GreedyRSC.

Meanwhile, Figure 4.7 (bottom) shows another example where the name *Trip* that is assigned correctly to the cluster does not have the highest occurrence frequency.

4.4 Browsing and Navigating Video Contents by Names and Faces

The resulting system can allow users to navigate the video content as illustrated in Figure 4.8. In this system, people can start from either a list of news video programs, or a list of representative faces extracted from clusters. In each news program, we show extracted frames, faces and names. People can access to these faces to see what clusters they belong to, and name candidates that are aligned to them. For each cluster, one or several representative faces along with name candidates aligned to the faces in the cluster are shown. By this way, users can explore the video contents easily and friendly.

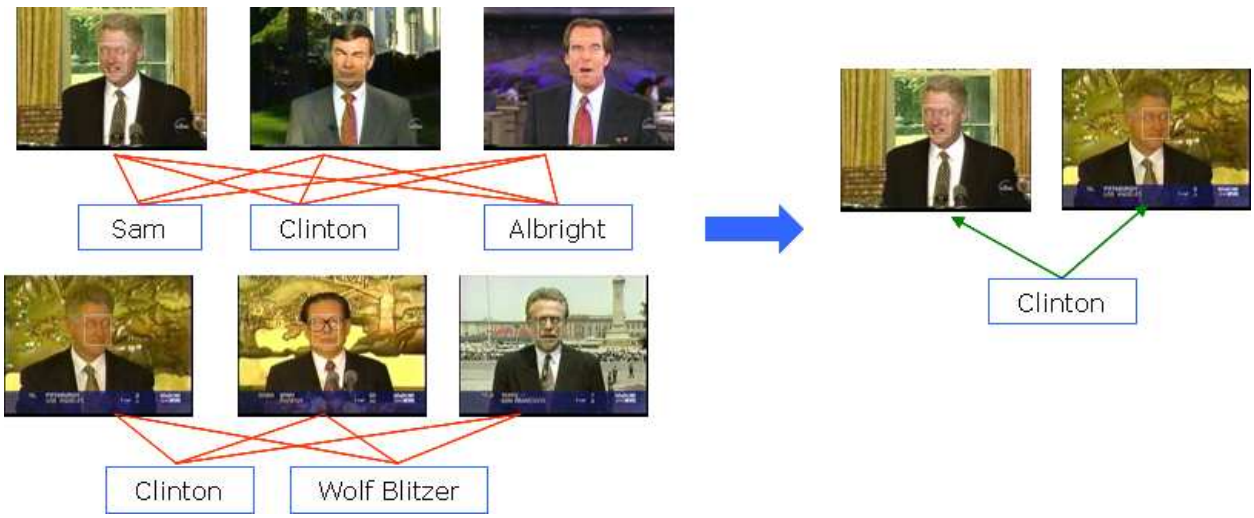

Figure 4.6: Face-name matching modeled as a translation problem in which faces are treated as words in the source language and names are treated as words in the target language.

4.5 Finding Important People By Multi-modal Analysis

As shown in Figure 4.5, our system can find important people such as *Bill Clinton, Kenn Starr and Monica Lewinsky*, who have gained many interests of audiences in that period when the scandal of Bill Clinton has gone to the public. Note that it is possible to have several face clusters corresponding to one person since their appearances have varied dramatically time by time. For example, the top row of Figure 4.5 shows 5 face clusters of Bill Clinton.

We judged manually the relevance of all face clusters returned by the system. Of 73 face clusters presented to users, 45 face clusters correspond to important people appearing more than one time in different news programs, resulting 61.6% precision. Of 45 relevant face clusters, 35 face clusters were labeled correctly by the names belonging the top-three, resulting 77.8% precision.

The demo of this system is available on the web at:
`http://satoh-lab.ex.nii.ac.jp/users/ledduy/Demo/`.

4.6 Discussion

Retrieving video segments related to a visual appearance of a person in real video data (such as broadcast news video) is useful but challenging. We have proposed and implemented a news video retrieval system using face information. By integrating human face processing techniques and RSC model-based clustering together with fast approximate similarity search, our method has the potential to handle very large scale video datasets effectively and efficiently. In the future, we plan to integrate other information sources from video data such as face positions and name entities extracted from transcripts to further improve the performance. More experiments and evaluations are also needed, particularly on larger datasets.

Figure 4.7: Examples of face and name association. (Top) The name with the highest occurrence frequency is assigned correctly to the cluster. (Bottom) The name that is assigned correctly to the cluster is not the one with the highest occurrence frequency.

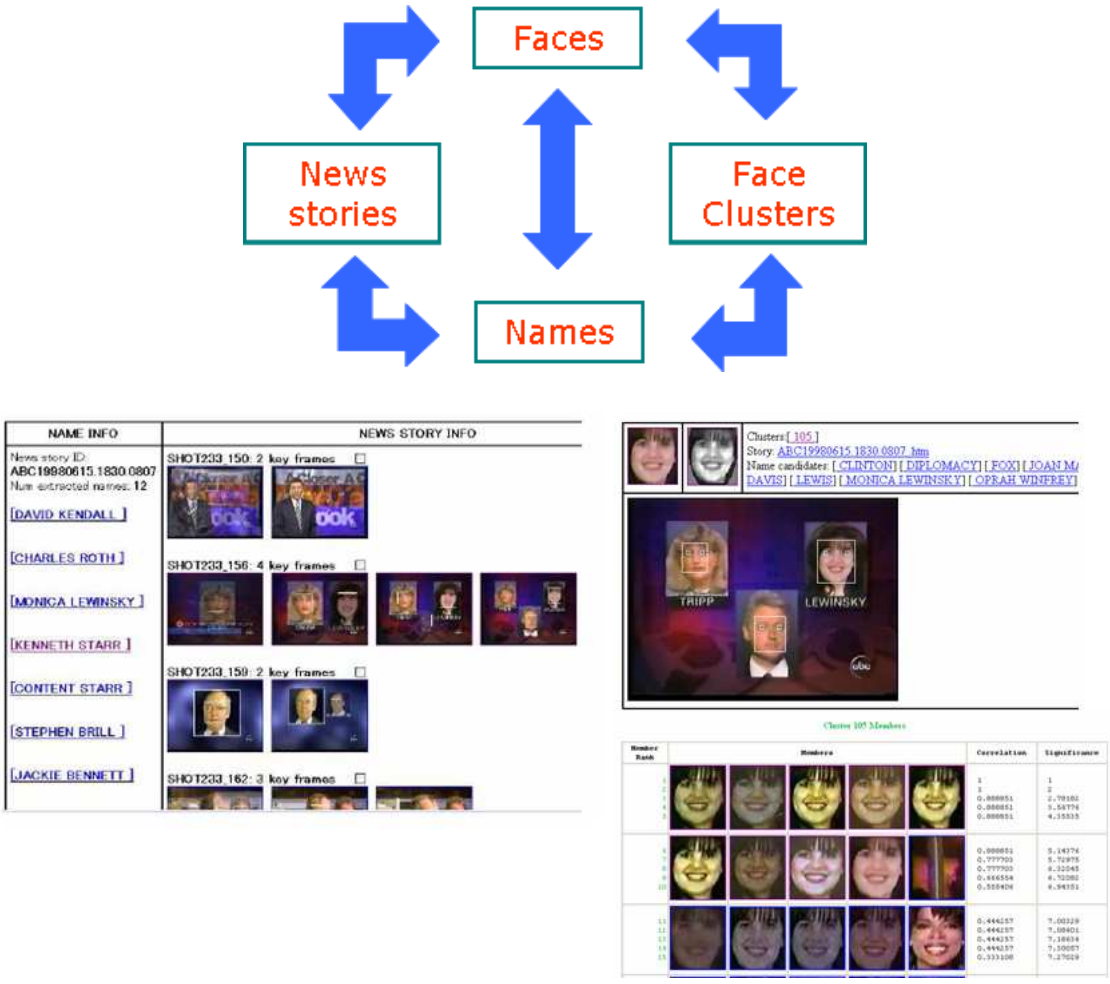

Figure 4.8: Navigation using faces and names.

Chapter 5

Discussion

5.1 Summary

Human face processing is significant in indexing large video archives, especially large broadcast news video database. In this thesis, we study several human face processing techniques focusing on developing video indexing and video retrieval systems. Our main contributions include:

- Propose two feature selection methods that can speed up the detection process and make the training process much easier and shorter while still maintaining the high accuracy. These approaches are general and can be integrated in building object detection systems. Furthermore, the feature selection method based on the conditional mutual information approach can handle huge feature sets that are currently used in many state of the art object detection systems.

- Propose a multi-stage approach to building a face detection system. The proposed system can achieve comparative to better speed and detection accuracy while realizing much faster training time. Reliable results of this face detection system are demonstrated in developing a news video retrieval system using face information.

- Propose a general framework for indexing large video datasets using high-level features such as human face. We have identified successfully the most suitable face representation and clustering technique for grouping similar faces from high dimensional and very large face sets. The state of the art clustering technique based on relevant set

correlation model is used and customized for organizing efficiently extracted faces from video. Furthermore we have shown that the integration of many immature techniques such as face processing, clustering and machine translation techniques can be used to develop a video browsing system on such large scale video databases as public benchmark TRECVID 2004 dataset. With this system, users can browse and navigate the video content by news stories, can quickly realize visually who appears frequently in some period. The results are available on the Internet and showed usefulness and effectiveness of our study.

5.2 Future Work

In the future, following work will be taken into account:

- Feature extraction: This is a fundamental problem in computer vision and object recognition. From our discussions described in 2.2.7, we plan to investigate how to combine strong points of wavelet features in representation and fast computation using integral image to design new features that are not only highly discriminant but also quickly extracted and normalized. More informative and discriminative features can help to improve clustering results.

- Face clustering: Study post-processing techniques to improve results returned by GreedRSC clustering. For example, to investigate how to reshape the resulting clusters by new similarity measures using temporal information, or to investigate how to perform classification based on clustering results.

- Semantic based video indexing and retrieval by using multimodal analysis: Study how to integrate available modalities from video data such as text, image, temporal information, etc to bridge semantic gaps in indexing and retrieval.

- Faces and names association: Study more robust methods in person name extraction and investigate models for efficiently labeling faces and names. Several open issues include: robust anchor person elimination and face modeling.

- Video summarization: Study how to extract significant phrases from text (e.g names, locations, organizations, keywords, etc) and link them to key image frames and key objects from video data to make a comprehensive summarization for important events. Information extraction techniques will be investigated and then modified to work with visual data.

- Video mining: Study how to apply data mining approaches to video databases to discover knowledge. Mined knowledge can be associations, highlights, unusual events, and so on.

References

[1] `http://www-nlpir.nist.gov/projects/trecvid/`.

[2] `http://www.alias-i.com/lingpipe/` .

[3] T. Ahonen, A. Hadid, and M. Pietikainen. Face recognition with local binary patterns. In *Proc. Intl. European Conference on Computer Vision*, volume 1, pages 469–481, 2004.

[4] O. Arandjelovic and A. Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 860–867, 2005.

[5] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski. Face recognition by independent component analysis. *IEEE Transactions on Neural Networks*, 13(6):1450–1464, Nov 2002.

[6] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, Jul 1994.

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

[8] T. L. Berg, A. C. Berg, J. Edwards, and D. A. Forsyth. Who's in the picture? In *Advances in Neural Information Processing Systems*, 2004.

[9] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. G. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 848–854, 2004.

[10] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful? In *Proc. Intl. Conf. on Database Theory*, page 217235, 1999.

[11] J. Bins and B. Draper. Feature selection from huge feature sets. In *Proc. Intl. Conf. on Computer Vision*, volume 2, pages 159–165, 2001.

[12] L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 236–243, 2005.

[13] P. Brown, S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, Jun 1993.

[14] C. J. C. Burges. Tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[15] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[16] C.-C. Chang and C.-J. Lin. Training *nu*-support vector classifiers: Theory and algorithms. *Neural Computation*, 13(9):2119–2147, 2001.

[17] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[18] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley Interscience, 2nd edition, 2000.

[19] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proc. Intl. European Conference on Computer Vision*, volume 4, pages 97–112, 2002.

[20] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. Intl. Joint Conference on Artificial Intelligence (IJCAI)*, pages 1022–1027, 1993.

[21] A. W. Fitzgibbon and A. Zisserman. On affine invariant clustering and automatic cast listing in movies. In *Proc. Intl. European Conference on Computer Vision*, volume 3, pages 304–320, 2002.

[22] A. W. Fitzgibbon and A. Zisserman. Joint manifold distance: a new approach to appearance based clustering. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 26–36, 2003.

[23] F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5(11):1531–1555, 2004.

[24] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, Sep 1999.

[25] C. Garcia and G. Tziritas. Face detection using quantized skin color regions merging and wavelet packet analysis. *IEEE Transactions on Multimedia*, 1(3):264–277, Sep 1999.

[26] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[27] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(3):1157–1182, 2003.

[28] A. Hadid, M. Pietikainen, and T. Ahonen. A discriminative feature space for detecting and recognizing faces. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 797–804, 2004.

[29] A. G. Hauptmann and M. G. Christel. Successful approaches in the trec video retrieval evaluations. In *Proc. ACM International Conference on Multimedia*, pages 668–675, 2004.

[30] B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. Technical Report A.I. Memo No. 1687, Massachusetts Institute of Technology, May 2000.

[31] B. Heisele, T. Serre, S. Prentice, and T. Poggio. Hierarchical classification and feature reduction for fast face detection with support vector machines. *Pattern Recognition*, 36(9):2007–2017, Sep 2003.

[32] M. E. Houle. Navigating massive data sets via local clustering. In *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pages 547–552, 2003.

[33] M. E. Houle. A generic query-based model for scalable clustering. Technical Report NII-2006-008E, National Institute of Informatics, May 2006.

[34] M. E. Houle and J. Sakuma. Fast approximate similarity search in extremely high-dimensional data sets. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 619–630, 2005.

[35] C. Huang, H. Ai, Y. Li, and S. Lao. Vector boosting for rotation invariant multi-view face detection. In *Proc. Intl. Conf. on Computer Vision*, volume 1, pages 446–453, 2005.

[36] C. Huang, H. Ai, B. Wu, and S. Lao. Boosting nested cascade detector for multi-view face detection. In *Proc. Intl. Conf. on Pattern Recognition*, volume 2, pages 415–418, 2004.

[37] X. Huang, S. Z. Li, and Y. Wang. Jensen-shannon boosting learning for object recognition. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 144–149, 2005.

[38] N. Ikizler and P. Duygulu. Person search made easy. In *Proc. Int. Conf. on Image and Video Retrieval*, pages 578–588, 2005.

[39] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

[40] N. Kwak and C. H. Choi. Input feature selection by mutual information based on parzen window. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1667–1671, Dec 2002.

[41] D.-D. Le and S. Satoh. An efficient feature selection method for object detection. In *Proc. Int. Conf. on Advances in Pattern Recognition*, volume 3686, pages 461–468, 2005.

[42] D.-D. Le and S. Satoh. Fusion of local and global features for efficient object detection. In *Proc. SPIE, Applications of Neural Networks and Machine Learning in Image Processing IX*, volume 5673, pages 106–116, 2005.

[43] D.-D. Le and S. Satoh. Multi-stage approach to fast face detection. In *Proc. British Machine Vison Conf.*, volume 2, pages 769–778, 2005.

[44] D.-D. Le and S. Satoh. Ent-boost: Boosting using entropy measure for robust object detection. In *Proc. Int. Conf. on Pattern Recognition*, volume 2, pages 602–605, 2006.

[45] D.-D. Le and S. Satoh. Multi-stage approach to fast face detection. volume 89, pages 2275–2285, Jul 2006.

[46] D.-D. Le and S. Satoh. Robust object detection using fast feature selection from huge feature sets. In *Proc. Int. Conf. on Image Processing*, volume 2, pages 602–605, 2006.

[47] D.-D. Le, S. Satoh, and M. Houle. Face retrieval in broadcasting news video by fusing temporal and intensity information. In *Proc. Int. Conf. on Image and Video Retrieval*, volume 4071, pages 391–400, 2006.

[48] K. Levi and Y. Weiss. Learning object detection from a small number of examples: The importance of good features. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 53–60, 2004.

[49] S. Z. Li and Z. Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):23–38, Sep 2004.

[50] Y.-Y. Lin, T. Liu, and C.-S. Fuh. Fast object detection with occlusions. In *Proc. Intl. European Conference on Computer Vision*, volume 3021, pages 402–413, 2004.

[51] Y.-Y. Lin and T.-L. Liu. Robust face detection with multi-class boosting. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 680–687, 2005.

[52] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch - a method for focus-of-attention. *International Journal of Computer Vision*, 11(3):283–318, Dec 1993.

[53] C. Liu. Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *IEEE Transactions on Image Processing*, 11(4):467–476, Nov 2002.

[54] C. Liu. Gabor-based kernel pca with fractional power polynomial models for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):572–581, May 2004.

[55] C. Liu and H. Y. Shum. Kullback-leibler boosting. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 587–594, 2003.

[56] C. Liu and H. Wechsler. A bayesian discriminating features method for face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):725–740, Jun 2003.

[57] H. Liu, F. Hussain, C. L. Tan, and M. Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6:393–423, 2002.

[58] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. Intl. Conf. on Computer Vision*, volume 2, pages 1150–1157, 1999.

[59] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91 – 110, Nov 2004.

[60] S. Lyu. Infomax boosting. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 533–538, 2005.

[61] A. Martinez and A. Kak. Pca versus lda. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, Feb 2001.

[62] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct 2004.

[63] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *Proc. Intl. European Conference on Computer Vision*, volume 3021, pages 69–82, 2004.

[64] T. Mita, T. Kaneko, and O. Hori. Joint haar-like features for face detection. In *Proc. Intl. Conf. on Computer Vision*, volume 2, pages 1619–1626, 2005.

[65] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710, Jul 1997.

[66] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[67] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29(1):5159, Jan 1996.

[68] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971987, Jul 2002.

[69] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face dectection. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[70] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, Jan 2000.

[71] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, Aug 2002.

[72] P. J. Phillips, H. J. Moon, S. A. Rizvi, and P. J. Rauss. The feret evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1094–1104, Oct 2002.

[73] A. K. R. Lienhart and and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *Proc. of the German 25th Pattern Recognition Symposium*, pages 297–304, 2003.

[74] S. Romdhani, P. H. S. Torr, B. Schlkopf, and A. Blake. Computationally efficient face detection. In *Proc. Intl. Conf. on Computer Vision*, volume 1, pages 695–700, 2001.

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

[76] S. Satoh and T. Kanade. Name-it: Association of face and name in video. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, pages 368–373, 1997.

[77] S. Satoh, Y. Nakamura, and T. Kanade. Name-it: Naming and detecting faces in news videos. *IEEE Multimedia*, 6(1):22–35, 1999.

[78] R. S. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[79] H. Schneiderman. Feature-centric evaluation for efficient cascaded object detection. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 20, pages 29–36, 2004.

[80] H. Schneiderman and T. Kanade. A statistical model for 3d object detection applied to faces and cars. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, pages 746–751, 2000.

[81] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56(3):151177,, Feb 2004.

[82] B. Scholkopf, A. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. *Neural Computing*, 12:1083–1121, 2000.

[83] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: Video shot retrieval for face sets. In *Proc. Int. Conf. on Image and Video Retrieval*, pages 226–236, 2005.

[84] C. Snoek and M. Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia Tools and Applications*, 25(1):5–35, 2005.

[85] C. Sun and D. Si. Fast reflectional symmetry detection using orientation histograms. *Real-Time Imaging*, 5:63–74, 1999.

[86] J. Sun, J. M. Rehg, and A. Bobick. Automatic cascade training with perturbation bias. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 276–283, 2004.

[87] K. K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, Jan 1998.

[88] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, 1991.

[89] S. Ullman, E. Sali, and M. Vidal-Naquet. A fragment-based approach to object representation and classification. In *Proc. Intl. Workshop on Visual Form*, pages 85–100, 2001.

[90] M. Vidal-Naquet and S. Ullman. Object recognition with informative features and linear classification. In *Proc. Intl. Conf. on Computer Vision*, pages 281–288, 2003.

[91] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. Intl. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

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

[93] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. Intl. Conf. on Very Large Data Bases*, page 194205, 1998.

[94] L. Wiskott, J. M. Fellous, N. Kuiger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779, Jul 1997.

[95] J. Wu, J. M. Rehg, and M. D. Mullin. Learning a rare event detection cascade by direct feature selection. In *Advances in Neural Information Processing Systems*, 2003.

[96] R. Xiao, M.-J. Li, and H.-J. Zhang. Robust multipose face detection in images. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):34–41, Jan 2004.

[97] R. Xiao, L. Zhu, and H.-J. Zhang. Boosting chain learning for object detection. In *Proc. Intl. Conf. on Computer Vision*, volume 1, pages 709–715, 2003.

[98] J. Yang, M. Chen, and A. G. Hauptmann. Finding person x: Correlating names with visual appearances. In *Proc. Int. Conf. on Image and Video Retrieval*, pages 270–278, 2004.

[99] J. Yang and A. G. Hauptmann. Naming every individual in news video monologues. In *Proc. ACM International Conference on Multimedia*, pages 580–587, 2004.

[100] J. Yang, R. Yan, and A. G. Hauptmann. Multiple instance learning for labeling faces in broadcasting news video. In *Proc. ACM International Conference on Multimedia*, pages 31–40, 2005.

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

[102] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5(10):1205–1224, 2004.

[103] Y. Zhai and M. Shah. Tracking news stories across different sources. In *Proc. ACM International Conference on Multimedia*, pages 2–10, 2005.

[104] Y. Zhai, A. Yilmaz, and M. Shah. Story segmentation in news videos using visual and text cues. In *Proc. Int. Conf. on Image and Video Retrieval*, pages 92–102, 2005.

[105] D. Zhang, S. Z. Li, and G. Perez. Real-time face detectioin using boosting in hierarchical feature spaces. In *Proc. Intl. Conf. on Pattern Recognition*, volume 2, pages 411–414, 2004.

[106] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang. Local gabor binary pattern histogram sequence (lgbphs): A novel non-statistical model for face representation and recognition. In *Proc. Intl. Conf. on Computer Vision*, volume 1, pages 786–791, 2005.

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

Index

List of Publications

Refereed Transactions and Journals

1. Duy-Dinh Le, Shin'ichi Satoh, *Multi-Stage Approach to Fast Face Detection*, In IEICE Transaction on Information and Systems, Vol. 89, No.7, pp. 2275-2285, Jul 2006.

2. Duy-Dinh Le, Shin'ichi Satoh, *Feature Selection By AdaBoost For Efficient SVM-Based Face Detection*, In Information Technology Letters, Vol.3, pp. 183-186, Kyoto, Japan, Sep 2004.

Refereed Conference Proceedings

1. Duy-Dinh Le, Shin'ichi Satoh, *Robust Object Detection Using Fast Feature Selection from Huge Feature Sets*, In Proc. 13th International Conference on Image Processing 2006 (ICIP06), pp. 961-964, USA, Oct 2006.

2. Duy-Dinh Le, Shin'ichi Satoh, *Ent-Boost: Boosting Using Entropy Measure for Robust Object Detection*, In Proc. 18th International Conference on Pattern Recognition 2006 (ICPR06), Vol. 2, pp. 602-605, Hong Kong, Aug 2006.

3. Duy-Dinh Le, Shin'ichi Satoh, Michael Houle, *Face Retrieval in Broadcasting News Video By Fusing Temporal and Intensity Information*, In Proc. 5th International Conference on Image and Video Retrieval 2006 (CIVR06), LNCS Vol. 4071, pp. 391-400, USA, Jul 2006.

4. Duy-Dinh Le, Shin'ichi Satoh, *Multi-Stage Approach to Fast Face Detection*, In 16th Proc. British Machine Vision Conference 2005 (BMVC05), UK, Sep 2005.

5. Duy-Dinh Le, Shin'ichi Satoh, *An Efficient Feature Selection Method for Object Detection*, In Proc. 3rd International Conference on Advances in Pattern Recognition (ICAPR05), LNCS Vol. 3686, pp. 461-468, UK, Sep 2005.

6. Duy-Dinh Le, Shin'ichi Satoh, *Fusion of Local and Global Features for Efficient Object Detection*, In Proc. SPIE Vol. 5673, pp. 106-116, Applications of Neural Networks and Machine Learning in Image Processing IX; Nasser M. Nasrabadi, Syed A. Rizvi; Eds., Feb 2005.

Presentations and Posters

1. Lizuo Jin, Shin'ichi Satoh, Fuminori Yamagishi, Duy-Dinh Le, Masao Sakauchii, *Person X Detector*, TRECVID 2004 Workshop, USA, Nov 2004.