HUMAN FACE ANALYSIS

ANALIZA FEŢEI UMANE

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea "Politehnica" din Timişoara
în domeniul INGINERIE ELECTRONICĂ ŞI
TELECOMUNICAŢII
de către

Ing. Ioan Buciu

Conducător ştiinţific: prof.dr.ing. Ioan Năforniţă
Referenţi ştiinţifici: prof. Ioannis Pitas
prof.dr.ing. Victor Emil Neagoe
prof.dr.ing. Radu Vasiu
prof.dr.ing. Cornelia Gordan

Ziua susţinerii tezei: 12.12.2008

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor

Universitatea „Politehnica" din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Foreword

The work presented in this PhD thesis was undertaken during my research activity at the Department of Electronics Engineering and Telecommunications of „Politehnica" University of Timisoara, Romania, as well as at the Department of Informatics, Artificial Intelligence and Information Analysis (AIIA) Laboratory, Aristotle University of Thessaloniki, Greece.

This thesis presents several original authors' contributions related to two topics of human face analysis, namely face detection task and facial expression classification task, respectively.

It is a pleasure to thank the many people who made this thesis possible.

First of all, I would like to express my sincere thanks to my PhD supervisor, Professor Ioan Nafornita, the head of the Communications Dept., Electronics and Communications Faculty, ``Politehnica'' University of Timisoara, without whose support this thesis would not haven been started and, especially, would not have went to an end. My utmost gratitude goes to him for his expertise, kindness, guidance, his capacity of insight, and most of all, for his patience.

I would also like to thank my colleague, Professor Cornelia Gordan, the head of the Electronics Dept., Faculty of Electrical Engineering and Information Technology, University of Oradea, who introduced me to the academic field and continuously supported me since the beginning of my career.

It is difficult to overstate my gratitude to my PhD co-supervisor, Professor Ioannis Pitas, the head of the Artificial Intelligence and Information Analysis (AIIA) Lab, Dept. of Informatics, Aristotle University of Thessaloniki. I thank him for allowing me to join his team and giving me the great opportunity to work with him and to acquire the knowledge and expertise. I benefited enormously from his breadth of knowledge. The most work presented in the thesis was developed during my staying at AIIA Lab where I was involved into two European Projects, namely the European Union Research Training Network "Multimodal Human-Computer Interaction" (HPRN-CT-2000-00111), and the "SIMILAR" European Network of Excellence on Multimodal Interfaces of the IST Programme of the European Union (www.similar.cc).

I am very grateful to Dr. Constantine Kotropoulos for his valuable support in many research aspects, especially those related to the support vector machine and independent component analysis topics. He provided encouragement, sound advice, good teaching, good company, and lots of good ideas. I would have been lost without his help.

I am indebted to Dr. Nikos Nikolaidis who worked with me for my last PhD period. His criticism and technical discussion helped me to improve my skills and expertise.

Many thanks go to my mother who encouraged me every time I was down. Special thanks are for my father who passed away letting emptiness in my soul while I wrote the thesis.

Lastly, and most importantly, I wish to thank my wife Adriana who stood beside me every day and encouraged me constantly throughout this endeavor, my thanks to my son, Darius Theodoros for giving me happiness and joy.

Timişoara, October 2008 Ioan Buciu

This thesis is dedicated to my family: Adriana and Darius Theodoros

Buciu, Ioan

Human Face Analysis

UPT, PhD Thesis, Series 7, No. 8, Politehnica Publishing House, 2008, 122 pages, 27 figures, 21 tabels.

ISSN: 1842-7014

ISBN: 978-973-625-750-6

Keywords:

Face Detection, Facial Expression Recognition, Subspace Image Decomposition, Feature Extraction, Classification, Human Visual System.

Abstract,

This thesis presents several original authors' contributions related to two topics of human face analysis, namely face detection task and facial expression classification task, respectively. The original work is presented as two distinct parts. In the first part of the thesis, a method for improving the accuracy of Support Vector Machines for face detection is introduced, followed by a rigorous statistical analysis of its stability in the attempt of using the bagging approach for gaining superior classification performance. The second and the biggest part of the thesis are dedicated to the feature extraction topic applied for facial expression recognition. Independent component analysis is a tool used in this regard. Several linear and non-linear independent component analysis methods are investigated and compared, and interesting conclusions are drawn. Next, two novel non-negative matrix factorization algorithms are described and their ability for providing useful features for classifying facial expression is proven through extensive experiments. By analogy to neurophysiology, the basis images discovered by non-negative matrix decomposition could be associated with the receptive fields of neuronal cells involved in encoding human faces. Taken from this point of view, an analysis of these three representations in connection to the receptive field parameters such as spatial frequency, frequency orientation, position, length, width, aspect ratio, etc, is undertaken. By analyzing the tiling properties of these bases some conclusions of how suitable these algorithms are to resemble biological visual perception systems can be drawn. The thesis ends up with a new feature extraction method using the phase congruency concept for measuring the similarity between image points, also applied for facial expression recognition.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Human Face Analysis as visual Pattern Recognition application

Human face analysis is a general term covering many aspects related to the analysis of faces. This analysis has emerged as topic having an interdisciplinary character. Nowadays, it involves research work coming from various fields, such as psychology, neurophysiology, image and video processing, computer vision or pattern recognition. From the computer vision point of view, the face analysis topics can be classified as follows:

- *Face detection* segments the face areas from the background. Given an arbitrary image or image sequence as input, a face detector is a system which is able to determine whether or not there is any human face in the image, and, if any, outputs an encoding of its location. Typically, the encoding in this system is to fit each face in a bounding box defined by the image coordinates of the corners.

- *Face recognition*. A face recognition system assists a human expert in determining the identity of a test face [1].

- *Face verification*. Although connected with the face recognition task and sometimes confused, the problem is conceptually different. A person verification system should decide whether an identity claim is valid or invalid.

- *Face encoding* refers to extracting valuable facial information from the whole face space. The information should obey some organic computing principles, such as efficient storing, organization and coding, by analogy with the Human Visual System. This topic is closely related to dimensionality reduction issue.

- *Facial expression recognition* deals with the interpretation and recognition of emotions expressed through facial expression, usually for the purpose of creating a friendly human-computer interface.

- *Facial expression modeling (synthesis)* aims at creating a synthetic "talking-head" able to simulate realistic human facial expressions. The artificial head should be able to recognize a human facial expression with a satisfactory classification rate and would reply to us according to our emotional state. Multimedia and film market are two commercial domains where this task has found important applications.

- *Face (facial features) tracking* appears in video sequences, especially for surveillance purposes (security) or face modeling. Here, the purpose is to accurately and robustly track fiducial points over time.

Although there is a clear distinction between the aforementioned topics, they are sometimes interconnected. For instance, the first step of any face recognition or facial expression recognition system is to detect the face in a digital image. Thus, face detection task should be a necessary prior step. However, most existing face recognition or facial facial expression recognition systems or methods perform with databases where the faces are assumed already detected, so the detection step is skipped. In this case the database contains faces that occupy the whole image space (i.e. the face is cropped from the uniform or complex background), or, at least, the face location is a priori known. A face tracking machine also should start with detecting the face, or at least to identify fiducial points to be tracked. Also, to synthesize an artificial face (able to simulate expressions), the face encoding is a must to extract appearance-based or geometrical facial information.

This thesis concerns only the first (face detection) and the fourth (facial expression recognition) human face analysis task, respectively. Both issues are a visual pattern recognition problem and can be analyzed using its tools.

Given a set of data samples, the ultimate goal of any recognition system is to automatically classify and group data samples into several classes, where the samples within the same group share common attributes. Typically, any automatic recognition system comprises two modules: *preprocessing and feature extraction* module and *classification* module, as illustrated in Figure 1.1. Consequently, its recognition performance is highly influenced by the efficiency of both modules. The object (human

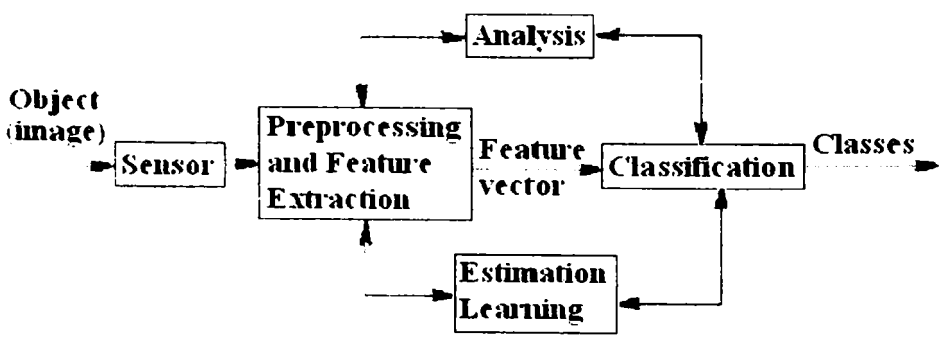


Figure 1.1: The Pattern Recognition issue

face or an environment containing human faces, in our case) is captured and recorded using a sensor. The sensor is basically a photo or video camera. The recorded data (digital image of the object or video frame sequences) are preprocessed (histogram equalization, noise removal, edge detection, etc) and transformed by extracting relevant features. Based on some training data (many recorded samples) a classifier analyzes the information and learns information characteristics. Also, the classifier adapts its parameters, so that, when the learning process is finished, the classifier to be able to accurately estimate (predict) the correct class for an unseen sample (not included in the training data).

The purpose of feature extraction is to transform the data in order to reduce the data dimensionality. A proper feature extraction technique will keep statistical relevant (discriminant) features and discard redundant information (or noise). Working with lower data dimensionality is twofold: decrease the computational load and increase the classifier performances in terms of its accuracy. If proper good set of features are extracted even the simplest classifiers based on basic metrics (such as, for example, Euclidean distance) may achieve satisfactory performance. There is a direct relationship between the number of features and the classifier's performance, i.e. the number of features greatly (positively or negatively) influences the classification accuracy. That is, the classification error decreases going from a small feature size to a moderate feature size followed by an increase for a large number of features (the so-called *peaking phenomenon*). This is a direct consequence of the so-called ``curse of dimensionality'' [2], that is, the time required for an algorithm grows dramatically, sometimes exponentially with the number of features involved, rendering the algorithm intractable in extremely high-dimensional problems. Thus, feature extraction step is a crucial step in any recognition system. The most part of this thesis is dedicated to the feature extraction step, employing methods such as Independent Component Analysis (ICA), or Non-negative Matrix Factorization algorithms [3] with direct application to facial expression recognition. From the classification point of view, the thesis also presents an improved version of Support Vector Machine for discriminate faces among non-face patterns.

1.2 Thesis content

The thesis consists of 6 chapters of which the latter 4 presents the original work developed by the author. Chapter 2 to Chapter 4 contains the research work accomplished while the author was with its second affiliation, i.e., Artificial Intelligence and Information Analysis (AIIA) Lab, Dept. of Informatics, Aristotle University of Thessaloniki, whilst the Chapter 5 deals with work performed at the Electronics Dept., Faculty of Electrical Engineering and Information Technology, University of Oradea.

Chapter 2 reviews face detection and facial expression recognition paradigms and their associated issues, followed by a short description of the existing face detection approaches.

Chapter 3 deals with the face detection task using an advanced classification scheme based on Support Vector Machines (SVMs). An approach to enhance the classification accuracy which uses a combination of SVMs is developed. Furthermore, a statistical analysis is undertaken to discover whether bagging (a technique utilized to enhance the classifier's accuracy) is suitable for SVMs.

Chapter 4 presents several linear and nonlinear Independent Component Analysis (ICA) techniques applied for extracting facial features further used to classify facial expressions. A statistical analysis is carried out and the methods are systematically analyzed with respect to their accuracy performance, their sparseness degree, etc., in comparison with the Principal Component Analysis method.

Chapter 5 covers a new algorithm named Non-negative Matrix Factorization along with three variants termed Local Non-negative Matrix Factorization, Discriminant Non-negative Matrix Factorization and Polynomial Non-negative Matrix Factorization. The latter two are developed by the author in order to extract relevant biological-inspired non-negative features for facial expression classification. It also presents the analogy of those algorithms to the Human Visual System principles, bringing some interesting

common features.

Chapter 6 provides a novel technique found to be effective in extracting facial features with application to facial expression classification. This approach is based on the phase congruency concept where discriminant features are extracted by measuring the similarity between image points.

CHAPTER 2

Face Detection and Facial Expression Recognition

2.1 Face Detection

2.1.1. Problem definition

Face detection plays an important role in multiple applications, such as teleconferencing, facial gesture recognition, and biometric access control to services, model-based coding, video content-based indexing, and video retrieval systems. Face detection is a preprocessing step in face recognition/verification tasks [4] - [7]. The goal of face detection is to determine if there are any human faces in a test image or not. Detecting a face in a complex scene is nontrivial problem. If a face exists, the face detector should be capable to locate a face regardless of its uniform or complex background, imaging formulation conditions, poses, scales, orientations or occlusions. Imaging formulation conditions refer to the lighting variation that can worsen the face detector's performance, especially for the appearance-based face detection approaches which are very sensitive to illumination's changes. Reducing the image resolution is another cause where the process of finding a face's location can fail. Usually, to detect potential faces at different scales, the face detector scans the whole image space with variable sized windows for matching. Most current face detection systems can only detect upright, frontal or slight pose variations under certain lighting conditions. Occluded faces can substantially differ in appearance from the non-occluded ones, resulting in a system's failure in detecting the face. Robust face detection methods that should handle various scenarios under different acquisition conditions have to be build to be reliable and useful as integrated part of a facial expression recognition system.

The human brain is highly trained for this task, therefore we find and analyze faces effortlessly under almost any conditions with a minimum of information, sometimes even when there is no face at all, like in cloud or rock pattern. The presence of hair, glasses or jewelry seems to be no problem for the Human Visual System , whereas automatic approaches are easily mislead. Moreover, for face acquisition, it is assumed that the face poses a frontal-view or near frontal-view, which is not always true.

Several works addressed these issues, though. Despite the importance of face detection, most researchers involved in human face analysis ignore this step and they exclusively focus on the other topics.

2.1.2 State-of-the-Art

Many approaches have been proposed for face detection. A first attempt to cope with both frontal and profile view faces was proposed by Kleck and Mendolia [8]. Three view perspectives were used (full-face, a 90° right profile, and a 90° left profile) for a number of 14 males and 14 females. These samples were shown to 24 male and 24 female decoders. It was found that positive expressions were more accurately identified in full-face and in right hemiface views as compared to left hemiface views, while the left hemiface was associated with better accuracy than the right hemiface for negative expressions. Essa and Pentland [9] performed face detection by using View-based and Modular Eigenspace methods proposed by Pentland and Moghaddam [10]. A face space is defined by carrying out the Principal Component Analysis (PCA) on a face database. To determine the face in a single image, the test image is projected in the resulting face space and the distance of the image from the face space is calculating from the projection coefficients. Further, to apply the technique to a video sequence, a spatio-temporal filtering is performed and the potential faces are described by the so-called ``motion blobs'' that are analyzed. A 3-D facial model with the help of a geometric mesh is developed to fit a face in an image. Given an input image, the system allows automatically detection of the position of eyes, nose and lips, proceeded by warping the face image to match the canonical face mesh. Further, additional ``canonical feature points'' on the image that correspond to the fixed (non-rigid) nodes on the proposed mesh are extracted. Yang and Huang have developed a system that attempts to detect a facial region at a coarse resolution and subsequently to validate the outcome by detecting facial features at the next resolution by employing a hierarchical knowledge-based pattern recognition system [11]. A probabilistic method to detect human faces using a mixture of factor analyzers has been proposed in [12]. Other techniques include neural networks [13], or algorithms where feature points are detected using spatial filters and then grouped into face candidates using geometric and gray level constrains [14]. Sung and Poggio report an example based-learning approach [15]. They model the distribution of human face patterns by means of few view-based face and non-face prototype clusters. A small window is moved over all portions on an image and determines whether a face exists in each window based on distance metrics. Huang and Huang [16] uses a point distribution model (PDM), where the initialization of the model is performed with the help of a Canny edge detector. This provides a rough estimation of the face's localization in the image. The position variations of certain designated points on the facial feature are described by 10 action parameters (APs). The face'location correspond to the valley in the pixel intensity map between the lips and the two symmetrical vertical edges associated to the outer vertical boundaries of the face. Hong et. al [17] developed a facial expression recognition system where the face detection step is accomplished by using the *PersonSpotter* module [18]. The system uses a spatio-temporal filtering of the input images. Within frames, the stereo disparities of the pixels whose changed their values due to the movement are analyzed by inspecting the local maximums of the disparity histogram and regions corresponding to a certain confidence interval are selected. A skin color detector along with a convex region detector is then applied for finer localization. A bounding box is finally drawn around the cluster's region found by the both detectors, with a maximum probability that the regions correspond to heads and hands. The system is limited so that no abrupt illumination variations, hair or glasses are allowed. It also can only detect frontal-view faces. Kumar and Poggio [19] uses skin segmentation and motion tracking to keep track of candidate regions in

the image corresponding to potential face candidates, followed by classification (face detection step) of the candidate regions into face and non-face, thus localizing the position and scale of the frontal face. Incorporating the skin segmentation procedure prior to face detection allows the system to perform face detection in real time. The skin model is obtained by training a support vector machine (SVM) using the red, green and yellow components of the pixel. Over 2000 skin samples of different people with widely varying skin tone and under differing lighting conditions are collected. The skin detector performs by scanning the input image in raster scan and classifying each pixel into skin or non-skin. The positions and velocities of the skin components are encoded an tracked to predict where the component will be seen in the next frame and thus helping to constrain the skin's search. Components that are smaller than a predefined threshold or those that have no motion at all are discarded from consideration. For face detection a number of 5,000 frontal face and 45,000 non-face patterns is used to train the SVM, each pattern being normalized to a size of 19×19 pixels. In the test phase, the SVM is applied at several scale of the active components for face face-like patterns searching. Their real-time face detection system works close to 25 frames per second. Pantic and Rothkrantz proposed an [20] expert system namely Integrated System for Facial Expression Recognition (ISFER), which performs recognition and emotional classification of human facial expression from a still full-face image. The system is composed by two major parts. The first one is the ISFER Workbench, which forms a framework for hybrid facial feature detection where, for robustness, multiple feature detection techniques are combined and applied in parallel. The second part comprises an inference engine called HERCULES, which converts low level face geometry into high level facial actions, followed by highest level weighted emotion encoding. The system can handle both frontal and profile view of the face for detection. The face acquisition was accomplished by two cameras mounted on the user's head. In their work, Oliver et al. [21] used coarse color and size/shape information to find and trace the face. More precisely, to detect and track faces in real time, the so-called 2D blob features (which are spatially-compact clusters of pixels that are similar in terms of low-level image properties) are extracted. Both the face and background classes are learned incrementally from the data by using the Expectation Maximization (EM) algorithm to obtain Gaussian mixture models for the spatio-chrominance feature vector comprising shapes and color patterns corresponding to faces. From the Gaussian mixture two to three components are usually sufficient to describe the face, while up to five components are required for the mouth. Given several statistical blob models that could potentially describe some particular image data, the membership decision is made by searching for the model with the Maximum A Posteriori (MAP) probability. Local pixel information retrieved after initial application of the MAP decision criterion is merged into connected and compact regions that correspond to each of the blobs. To grow the blob a connected component algorithm is employed that considers for each pixel the values within a neighborhood of a certain radius in order to determine whether this pixel belongs to the same connected region. The blobs are finally filtered to obtain the best candidate for being a face or a mouth. Due to the fact that the background may contain skin-like color that can affect the face detector's accuracy, to increase the robustness, geometric information, such as the size and shape of the face to be detected is combined with the color information to finally locate the face. Therefore, only those skin blobs whose size and shape (ratio of aspect of its bounding box are closest to the canonical face size and shape are taken into account. Bartlett et al. [22] proposed a system that automatically detects frontal faces in a video stream and codes them (in real time) according to the six basic emotions, i.e., anger, dis-

gust, fear, joy, sadness, surprise plus the neutral. The face finder module employs a cascade of feature detectors trained with boosting techniques similar to that proposed by Viola and Jones [23]. Each feature detector (classifier) contains a subset of filters reminiscent of Haar basis functions, which can be computed very fast at any location and scale in constant time. The system scans across all possible 24×24 pixel patches in the image and classifies each as face vs. non-face. For each feature detector in the cascade a subset of 2 to 200 of these filters are chosen by using a feature selection procedure based on Adaboost strategy for selecting the filter which achieves the best result in the training phase. The approach continues with refining the selection by finding the best performing single-feature classifier from a new set of filters generated by shifting and scaling the chosen filter by two pixels in each direction, as well as composite filters made by reflecting each shifted and scaled feature horizontally about the center and superimposing it on the original. While this approach requires binary classifiers, a second face detection technique based on Gentleboost [24] which uses real valued features is also proposed as alternative. The same face detection approach of Viola and Jones has been used by Tian in [25] for different image resolution who investigated the effect of image resolution in facial expression classification. A second face detection method based on neural networks (NN) and developed by Rowley at al. [26] is also taken into account. A preprocessing step that includes illumination correction and histogram equalization is carried out prior to feed the neural network with 20×20 pixel window of the image. To detect faces anywhere in the input, the filter is applied at every location in the image. To detect faces larger than the window size, the input image is repeatedly reduced in size (by subsampling), and the filter is applied at each size. The neural network has retinal connections to its input layer. There are three types of hidden units: 4 which look at 10×10 pixel subregions, 16 which look at 5×5 pixel subregions, and 6 which look at overlapping 20×5 pixel horizontal stripes of pixels. Each of these types was chosen to allow the hidden units to detect local features that might be important for face detection. The work of Viola and Jones was further extended by Isukapalli et al. [27]. They proposed the usage of a decision tree of classifiers (DCT). While standard cascade classification methods apply the same sequence of classifiers to each image, their DTC approach is able to select the most effective classifier at every stage, based on the outcomes of the classifiers already applied. They used DTC not only to detect faces in a test image, but to identify the expression on each face.

A comprehensive survey of face detection methods can be found in [28] and [29].

2.2 Facial Expression Recognition

2.2.1 Problem definition

Human facial expression analysis has captured an increasing attention from psychologists, anthropologists, and computer scientists [30]. The computer scientists try to develop complex human-computer interfaces that are capable of automatically recognizing and classifying human expressions or emotions and/or even to synthesize these expressions onto artificial talking-heads (avatars). Fasel and Luettin define facial expressions as temporally deformed facial features such as eye lids, eye brows, nose, lips and skin texture generated by contractions of facial muscles. They observed typical changes of muscular activities to be brief, ``lasting for a few seconds, but rarely more than five seconds or less than 250 ms'' [31]. They also point out

the important fact that felt emotions are only one source of facial expressions besides others like verbal and non-verbal communication or physiological activities. Though facial expressions obviously are not to equate with emotions (and the terms are many times wrongly interchanged), in the computer vision community, the term ``facial expression recognition'' often refers to the classification of facial features in one of the six so called basic emotions: happiness, sadness, fear, disgust, surprise and anger, as introduced by Ekman in 1971 [32]. This attempt of an interpretation is based on the assumption that the appearance of emotions are universal across individuals as well as human ethnics and cultures.

The task of automatic facial expression analysis can be divided into three main steps: face detection, facial feature extraction and classification into expressions. The detection issue has been discussed earlier. After localizing the face, as much information as possible about the displayed facial expression has to be extracted. Several types of perceptual cues to the emotional state are displayed in the face: relative displacements of featured (e.g. raised eyebrows), quasi textural changes in the skin surface (furrowing the brow), changes in skin hue (blushing) and the time course of these signals. Depending on how the face and its expression are modeled, features have to be designed that condense this information or a part of it to a set of numbers building the base for the classification, and therefore primarily deciding about the quality of the final analysis result. Most automatic facial expression analysis systems found in the literature directly classify in terms of basic emotions. This is an attempt of interpretation rather than the classification of really observed facial appearance. Some research groups therefore follow the idea of Ekman and Friesen [34] who, in the late 70-ies, postulated a system that categorizes all possible, visually detectable facial changes in 44 so-called Action Units (AUs). This system, known as Facial Action Coding System (FACS) has been developed to facilitate objective measurements of facial activity for behavioral studies. The interpretation of the AUs in terms of basic emotions then is based on a special FACS dictionary. FACS are an important tool in behavioral science, and the underlying study can be seen as the theoretical basis for any facial expression analysis. Nevertheless, the AU coding is skipped in most Human Computer Interaction (HCI) applications, because its insignificant contribution to the goal of interpreting nonverbal feedback from a user. Classification is complicated by the fact that despite cross cultural similarities, facial expressions and the intensity with which they are exhibited strongly vary between individuals. Also, it is doubtful that naturally expression can be unambiguously classified into one of the six basic categories. Quite often, facial expressions are blended and their interpretation mainly depends on the situational context. Automatic classification furthermore is confronted with a physiognomic variability due to gender, age and ethnicity.

2.2.2 State-of-the-Art

Facial Expression Analysis (FEA) dates back to the 19th century when Darwin [35] studied the anatomical and physiological basis of facial expressions of man and animal. Since the mid 1970s, automatic facial expression analysis has attracted the interest of many computer vision research groups. Surveys on automatic facial expression analysis can be found in [36, 37, 31]. Generally speaking, facial expression recognition methods can be classified into *appearance-based* methods and *geometry-based* ones. In the first category, fiducial points of the face are selected either manually [38] or automatically [39]. The face images are convolved with Gabor filters and the responses extracted at the fiducial points form vectors that are further used for facial

expression classification. Alternatively, Gabor filters can be applied to the entire face image instead of specific face regions. Regarding the geometry-based methods, the coordinates of the fiducial points form a feature vector that represents facial geometry. Although the appearance-based methods seem to yield a reasonable facial expression recognition accuracy, the highest recognition rate has been obtained when both the responses of Gabor wavelets and geometry-based features, like the coordinates of fiducial points, are combined [38, 40, 41]. The analysis can be performed wither on still images [38] or image sequences, where temporal information is considered [42]. Gabor and Independent Component Analysis (ICA) representations were described for the recognition of 6 single upper facial action units (AUs) and 6 lower face AUs in [43]. The action units correspond roughly to the movement of the individual 44 facial muscles. The best recognition rates were achieved by both Gabor wavelets and ICA representations [43]. The local properties of ICA representation were found to be important for identity recognition [44]. Identity and facial expression recognition performance were also investigated by directly comparing ICA versus Principal Component Analysis (PCA) in [45], where it was found that ICA outperformed PCA. On the contrary, insignificant performance differences between ICA and the PCA were reported on the same database in [46]. Guo and Dyer addressed facial expression classification, when a small number of training samples was only available [47]. A new linear programming-based technique was developed for both feature extraction and classification and a pairwise framework for feature selection was designed instead of using all classes simultaneously. Gabor filters were used to extract facial features and large margin classifiers such as support vector machines (SVMs) and AdaBoost were employed to recognize facial expressions. Their approach named ``feature selection via linear programming'' (FSLP) is able to automatically determine the number of selected features for each pair of classes in contrast to AdaBoost, which heuristically determines the number of features. Susskind et al. studied the nature of emotional space [30]. Evidence is presented justifying that emotion categories are not entirely discrete and independent, but they vary along underlying continuous dimensions. PCA has been successfully applied to recognize facial expressions [48, 49, 50]. A more recent paper [51] dealt with facial expression, where Gabor features were extracted from samples that belong to the Cohn-Kanade database. The Gabor features were then selected by AdaBoost and the combination of AdaBoost and SVMs (called AdaSVMs system) yielded the best classification performance of 93.3%.

CHAPTER 3

Support Vectors – based Face Detection

3.1 Improving the accuracy of SVMs applied for face detection

One method which has been applied successfully to face detection is based on Support Vector Machines [52]. Support Vector Machines (SVMs) is a state-of-the-art pattern recognition technique whose foundations stem from statistical learning theory [53]. However, the scope of SVMs is beyond pattern recognition, because they can handle also another two learning problems, i.e., regression estimation and density estimation. In the context of pattern recognition, the main objective is to find the optimal separating hyperplane, that is, the hyperplane that separates the positive and negative examples with maximal margin. SVM is a general algorithm based on guaranteed risk bounds of statistical learning theory, i.e., the so-called *structural risk minimization* principle. This principle is based on the fact that the error rate of learning machine on test data (i.e., the generalization error rate) is bounded by the sum of the training error rate and a term that depends on the Vapnik-Chervonenkis (VC) dimension [53]. We briefly describe linearly separable case followed by linearly non-separable case and the nonlinear one.

Consider the training data set

$$S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$$

of labeled training patterns, where $\mathbf{x}_i \in \mathbb{R}^d$ with m denoting the dimensionality of the training patterns, and

$$y_i \in \{-1, +1\}$$

We claim that S is linearly separable if for some $\mathbf{w} \in \mathbb{R}^m$, and b real

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \qquad \text{for} \quad i = 1, 2, \ldots, l \tag{3.1}$$

where $\mathbf{w}$ is the normal vector to the separating hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ and b is the bias (or offset) term [54]. The optimal separating hyperplane is the solution of the following quadratic problem:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ &\text{subject to} && y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \ldots, n \end{aligned} \tag{3.2}$$

In Figure 3.1 the optimal separating hyperplane is drawn in the case of linearly separable data. The optimal $\mathbf{w}^*$ is given by

$$\mathbf{w}^* = \sum_{i=1}^{n} \lambda_i^* y_i \mathbf{x}_i \tag{3.3}$$

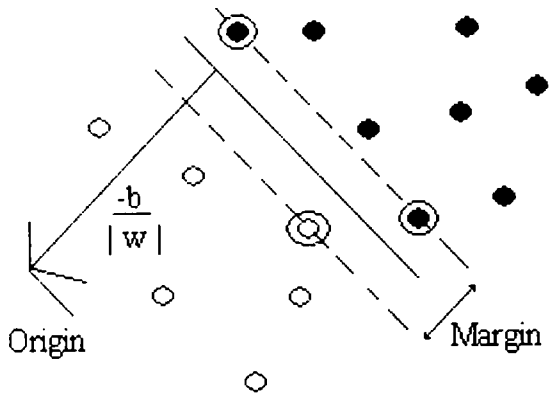


Figure 3.1: Optimal separating hyperplane in the case of linearly separable data. Support vectors are circled.

where $\boldsymbol{\lambda}^*$ is the vector of Lagrance multipliers obtained as the solution of the so-called Wolfe-dual problem

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \lambda_i - \boldsymbol{\lambda}^T \mathbf{D} \boldsymbol{\lambda} \\ \text{subject to} \quad & \sum_{i=1}^{n} y_i \lambda_i = 0 \\ & \lambda_i \geq 0 \end{aligned} \tag{3.4}$$

where $\mathbf{D}$ is an $n \times n$ matrix having elements $D_{ij} = y_i y_i \mathbf{x}_i^T \mathbf{x}_j$.

Thus $\mathbf{w}^*$ is a linear combination of the training patterns $\mathbf{x}_i$ for which $\lambda_i^* > 0$. These training patterns are called *support vectors*. Given a pair of support vectors $(\mathbf{x}^*(1), \mathbf{x}^*(-1))$ that belong to the positive and negative patterns, the bias term is found by [53]

$$b^* = \frac{1}{2}\left[\mathbf{w}^{*T}\mathbf{x}^*(1) + \mathbf{w}^{*T}\mathbf{x}^*(-1)\right]. \tag{3.5}$$

The decision rule implemented by the SVM is simply

$$f(\mathbf{x}) = \text{sign}\left(\mathbf{w}^{*T}\mathbf{x} - b^*\right). \tag{3.6}$$

If the training set S is not linearly separable, the optimization problem (3.4) is generalized to

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, n \\ & \xi_i \geq 0 \end{aligned} \tag{3.7}$$

where ξ_i are positive slack variables [54], and C is a parameter which penalizes the errors. The situation is summarized schematically in Fig 3.2. The Lagrange multipliers now satisfy the inequalities

$$0 \leq \lambda_i \leq C. \tag{3.8}$$

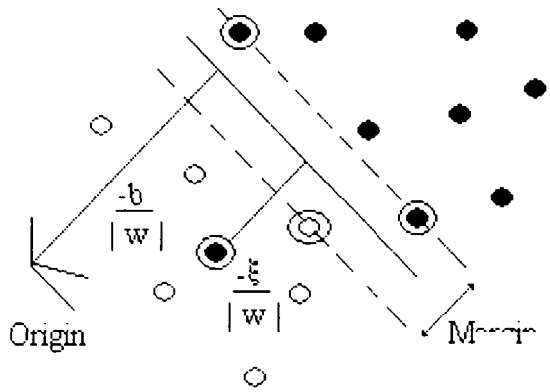


Figure 3.2: Separating hyperplane for non-separable data. Support vectors are circled.

The main difference is that support vectors do not necessarily lie on the margin.

Finally, SVMs can also provide nonlinear separating surfaces by projecting the data to a high dimensional feature space $\mathcal{H}$ in which a linear hyperplane is searched for separating all the projected data, $\phi : \mathbb{R}^m \longrightarrow \mathcal{H}$. If the inner product in space $\mathcal{H}$ had an equivalent kernel in the input space $\mathbb{R}^m$, i.e.:

$$\phi^T(\mathbf{x}_i)\phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) \tag{3.9}$$

the inner product would not need to be evaluated in the feature space, thus avoiding the curse of dimensionality problem. In such a case, $D_{ij} = y_i y_i K(\mathbf{x}_i, \mathbf{x}_j)$ and the decision rule implemented by the nonlinear SVM is given by

$$f(\mathbf{x}) = \text{sign}\left(\sum_{\substack{i=1 \\ \lambda_i^* \neq 0}}^{n} \lambda_i^* \, y_i \, K(\mathbf{x}, \mathbf{x}_i) - b^*\right). \tag{3.10}$$

3.1.1 Application of majority voting in the output of several SVMs

To increase the SVMs accuracy a combination scheme was proposed by Buciu et al. [55]. Let us consider five different SVMs defined by the kernels indicated in Table 3.1. The following kernels have been used: (1) Polynomial with q equal to 2; (2) Gaussian Radial Basis Function (GRBF) with $\sigma = 10$; (3) Sigmoid with κ equal to 0.5 and θ equal to 0.2; (4) Exponential Radial Basis Function having σ equal to 10. The penalty, C, in (3.7)was set up to 500. In Table 3.1, $||\cdot||_p$ denotes the vector p-norm, $p = 1, 2$. For brevity, we index each SVMs by k, $k = 1, 2, \ldots, 5$. To distinguish between training and test patterns, the latter ones are denoted by $\mathbf{z}_j$. Let $\mathcal{Z}_k$ be the set of test patterns classified as face patterns by the kth SVM during the test phase, i.e.,

$$\mathcal{Z}_k = \{\mathbf{z}_j : f_k(\mathbf{z}_j) = 1\}, \;\; k = 1, 2, \ldots, 5. \tag{3.11}$$

Let $\mathcal{Z} = \cup_{k=1}^{5} Z_k$. We define the histogram of labels assigned to all $\mathbf{z}_j \in \mathcal{Z}$ as

$$h(\mathbf{z}_j) = \#\{f_k(\mathbf{z}_j) = 1, \;\; k = 1, 2, \ldots, 5\} \tag{3.12}$$

Table 3.1: Kernel functions used in SVMs.

k	SVM type	Kernel function $K(\mathbf{x},\mathbf{y})$
1	Linear	$\mathbf{x}^T\mathbf{y}$
2	Polynomial	$(\mathbf{x}^T\mathbf{y}+1)^q$
3	GRBF	$\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|_2^2}{2\sigma^2})$
4	Sigmoid	$\tanh(\kappa\cdot\mathbf{x}^T\mathbf{y}-\theta)$
5	ERBF	$\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|_1}{2\sigma^2})$

where # denotes the set cardinality. We combine the decisions taken separately by the SVMs indexed by $k = 1, 2, \ldots, 5$ as follows:

$$g(\mathbf{z}_i) = \begin{cases} 1 & \text{if} \quad i = \arg\max_j \{h(\mathbf{z}_j)\} \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

Let us define the quantities:

$$\begin{aligned} F_k &= \#\{f_k(\mathbf{z}_j) = 1, \ \mathbf{z}_j \in \mathcal{Z}_k\} \\ G_k &= \#\{g(\mathbf{z}_j) = 1, \ \mathbf{z}_j \in \mathcal{Z}_k\} \end{aligned} \tag{3.14}$$

To determine the best SVM, we simply choose

$$m = \arg\max_k \{\frac{G_k}{F_k}\}. \tag{3.15}$$

3.1.2 Bagging approach

Bagging is a method for improving the prediction error of classifier learning system by generating replicated bootstrap samples of the original training set [56]. Given a training set a $\mathcal{S}^*$ bootstrap replicate of it is built by taking l samples with replacement from the original training set $\mathcal{S}$. The learning algorithm is then applied to this new training set. This procedure is applied B times yielding $\mathcal{S}^{*1}, \ldots, \mathcal{S}^{*B}$. Finally, those B new models are aggregating by uniform voting and the resulting class is that one having the most votes over the replicas. Notice that in the bootstrap replica an original pattern may not appear on it while others may appear more than once, on average 63% of he original patterns appearing in the bootstrap replica. A more detailed description of the bagging approach is provided in the next Section.

3.1.3 Performance assessment

For all experiments the Matlab SVM toolbox developed by Steve Gunn was used [57]. For a complete test, several auxiliary routines have been added to the original toolbox.

A training data set of 96 images, 48 images containing a face and another 48 images with non-face patterns, is built. The images containing face patterns have been derived from the face database of IBERMATICA where several sources of degradation are modeled, such as varying face size and position and changes in illumination. All images in this database are recorded in 256 grey levels and they are of dimensions 320 $\times$ 240. These face images correspond to 12 different persons. For each person

four different frontal images have been collected. The procedure for collecting face patterns is as follows. From each image a bounding rectangle of dimensions 160 $\times$ 128 pixels has been manually determined that includes the actual face. The face region included in the bounding rectangle has been subsampled four times. At each subsampling, non-overlapping regions of 2×2 pixels are replaced by their average. Accordingly, training patterns $\mathbf{x}_i$ of dimensions 10×8 are built. The ground truth, that is, the class label $y_i = +1$ has been appended to each pattern. Similarly, 48 non-face patterns have been collected from images depicting trees, wheels, bubbles, and so on, by subsampling four times randomly selected regions of dimensions 160×128. The latter patterns have been labeled by $y_i = -1$.

We have trained the five different SVMs indicated in Table 3.1. The trained SVMs have been applied to six face images from the IBERMATICA database that have not been included in the training set. Each test image corresponds to a different person. The resolution of each test image has been reduced four times yielding a final image of dimensions 15×20. Scanning row by row the reduced resolution image, by a rectangular window 10×8, test patterns are classified as non-face ones (i.e., $f(\mathbf{z}) = -1$) or face patterns (i.e., $f(\mathbf{z}) = 1$). When a face pattern is found by the machine, a rectangle is drawn, locating the face in image.

We have tabulated the ratio G_k/F_k in Table 3.2. From Table 3.2, it can be seen that

Table 3.2: Ratio G_k/F_k achieved by the various SVMs.

SVM type	Test Image numbers					
k	1	2	3	4	5	6
1	0.83	0.20	0.57	0.66	**1**	0.74
2	0.52	0.28	0.57	0.44	**1**	0.71
3	0.67	0.25	0.44	0.44	0.80	0.83
4	0.64	0.14	0.15	0.11	0.22	0.13
5	**1**	**0.50**	**0.80**	**0.80**	0.80	**1**

ERBF is found to maximize the ratio in (3.15) for the five test images. On the contrary the machine built using the sigmoid kernel attains the worst performance with respect to (3.15). Interestingly, the ERBF machine experimentally yields the greatest number of support vectors, as can be seen in Table 3.3.

Table 3.3: Number of support vectors found in the training of the several SVMs studied.

SVM type	Test Image numbers					
k	1	2	3	4	5	6
1	11	11	11	11	10	11
2	14	13	14	14	14	13
3	12	10	12	16	12	12
4	13	11	11	11	11	11
5	39	41	41	40	39	40

To assess the performance of the majority voting procedure, we have manually annotated each test pattern $\mathbf{z}_i$ with the ground truth that is denoted as $z_{i,81}$. Two quantitative measurements have been used for the assessment of the performance of

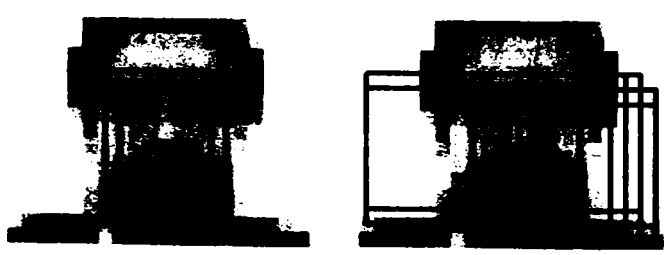
each SVM, namely, the *false acceptance rate* (FAR) (i.e., the rate of false positives) and the *false rejection rate* (FRR) (i.e., the rate of false negatives) during the test phase. We have measured FAR and FRR for each SVM individually as well as after majority voting. We have found that FRR is always zero while FAR varies. For each of the five different SVM we used bagging. The number of bootstrap replicas was 21. Unfortunately, for this set of data, the method did not work well. Moreover, perturbing the distribution of the original data bagging slightly degrades the performance of the initial classifier. The values of FAR attained by each SVM individually and after applying majority voting along with the values obtained with bagging are shown in Table 3.4. The FAR after bagging are in parentheses. It is seen that application of majority voting

Table 3.4: False acceptance rate (in %) achieved by the various SVMs individually, with bagging and after applying majority voting. In parentheses are the values corresponding to bagging

SVM type	Test Image numbers					
k	1	2	3	4	5	6
1	3.9 (4.7)	10.5 (12.1)	6.5 (7.6)	5.2 (6.5)	2.6 (3.5)	6.5 (7.8)
2	6.5 (10.1)	6.5 (9.3)	6.5 (7.6)	9.2 (9.2)	2.6 (3.5)	6.5 (10.8)
3	5.2 (7.7)	7.8 (10.1)	9.2 (10.6)	9.2 (13.5)	3.9 (4.5)	5.2 (8.8)
4	7.8 (23.7)	17.1 (29.2)	31.5 (44.6)	44.7 (78.5)	21.0 (46.5)	47.3 (88.8)
5	2.6 (2.6)	2.6 (3.1)	3.9 (6.5)	3.9 (6.5)	3.9 (4.5)	3.9 (4.8)
combining	2.6	1.3	2.6	2.6	2.6	3.9

reduces the number of false positives in all cases and particularly when $F_k \neq G_k$.

Figure 3.3 depicts 2 extreme cases observed during a test. It is seen that majority voting helps to discard many of the candidate face regions returned by a single SVM (Fig. 3.3(b)) yielding the best face localization (Fig. 3.3(a)).



(a) (b)

Figure 3.3: (a) Best and (b) worst face location determined during a test.

3.2 Can bagging strategy enhance the SVMs accuracy for detection ?

A performance measure of a classifier is the so called *accuracy*, which is usually represented by the ratio of correct classifications. The accuracy measured on the training set generally differs from the accuracy measured on the test set, especially if the statistics of training and test sets are different. From a practical point of view, the latter is more important. The general method to estimate the accuracy is as follows. First, we use a part of the given data (namely the *training set*) to train the classifier by possibly exploiting the class membership information. The trained classifier is then tested on the remaining data (the *test set*) and the results are compared to the actual classification that is assumed to be available. The percentage of correct decisions in the test set is an estimate of the accuracy of the trained classifier, provided that the training set is randomly sampled from the given data. There are many methods which can be used to enhance the accuracy of a classifier for artificially generated data sets or real ones, such as *bagging, boosting, stacking*, and their variants. The accuracy of a classifier as a result of any of the previously mentioned methods is of primary concern and the classifier performance is often examined from this perspective. Improving the accuracy is equivalent to reducing the *prediction error*, which is defined as $1 - accuracy$.

A well known method for estimating the prediction error is the so-called *bootstrap*, where sub-samples of the original data set are analyzed repeatedly [58]. Bagging is a variant of the bootstrap technique, where each sub-sample is a random sample created with replacement from the full data set [56]. Other procedures of this type include boosting [60] and stacking [61]. Ensembling multiple classifiers can yield a more accurate classifier [55]. Bagging has produced a superior performance for many classifiers, such as decision trees [63] and perceptrons [64]. However, there are several classifiers for which this method has either a little effect or may slightly degrade the classifier performance (e.g. k-nearest neighbor, linear discriminant analysis) [65]. From this point of view, classifiers can be split into *stable* and *unstable* ones. A classifier is considered as being stable if bagging does not improve its performance. If small changes of the training set lead to a varying classifier performance after bagging, the classifier is considered to be an unstable one. The unstable classifiers are characterized by a high variance although they can have a low bias. On the contrary, stable classifiers have a low variance, but they can have a high bias. Bias and variance are defined in the next Section.

It turns out that bagging, along with the decomposition of the prediction error into its variance and bias components, is a suitable tool for the investigation of the stability of a classifier. We also explore the aggregation effect, which indicates whether bagging is useful to a given problem or not. The stability of regularization networks has been proved in [66]. Since these networks and Support Vector Machines (SVMs) are closely related [67], it is expected that SVMs will be stable as well. This Chapter provides numerical evidence that a two-class SVM classifier can be included in the class of stable classifiers, the analysis fully described by Buciu et al. in [68]. To support this claim, the concepts of bias, variance, and aggregation effect are considered.

3.2.1 Bias and variance decomposition of the average prediction error

A *labeled instance* or *training pattern* is a pair $\mathbf{z} = (\mathbf{x}, y)$, where $\mathbf{x}$ is an element from feature domain $\mathcal{X}$ and y is an element from class domain $\mathcal{Y}$. The probability distribution over the space of labeled instances is denoted with $\mathcal{F}$.

The instances of the training set $\mathcal{L} = \{\mathbf{z}_i \mid i = 1, \ldots, n\}$ are assumed to be independent and identically distributed, that is, $\mathbf{Z}_1, \ldots, \mathbf{Z}_n \sim \mathcal{F}(\mathbf{x}, y)$, where capital letters denote random variables. Without loss of generality, we consider a two-class problem. Therefore, $y_i \in \{-1, +1\}$. In such a classification problem, we construct a classification rule $C(\mathbf{x}, \mathcal{L})$ by training on the basis of $\mathcal{L}$. The output of the classifier will be then $c \in \{-1, +1\}$. Let $Q[y, c]$ indicate the loss function between the predicted class label c and the actual class label y. A plausible choice is $Q[y, c] = 1$ if $y \neq c$ and 0 otherwise.

Let $\mathbf{Z}_o = (\mathbf{X}_o, Y_o)$ be another independent draw from $\mathcal{F}$ called the *test pattern* with value $\mathbf{z}_o = (\mathbf{x}_o, y_o)$. The *average prediction error* for the rule $C(\mathbf{X}_o, \mathcal{L})$ is defined as:

$$err(C) = E_{\mathcal{F}}\{E_{\mathcal{OF}}\{Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]\}\} \tag{3.16}$$

where $E_{\mathcal{F}}$ indicates expectation over the training set $\mathcal{L}$ and $E_{\mathcal{OF}}$ refers to expectation over the test pattern $\mathbf{Z}_o \sim \mathcal{F}$. Note that the expression (3.16) is consistent with the risk functional defined in statistical learning theory [53]. Indeed $Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]$ is the loss function and $E_{\mathcal{F}}\{E_{\mathcal{OF}}\{Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]\}\}$ is a bootstrap estimate of the risk functional.

The average prediction error can be decomposed into components to allow for a further investigation. Several decompositions of the prediction error into its bias and variance have been suggested. In [65], an exact additive decomposition of the prediction error into the Bayes error, bias, and variance is performed. Another decomposition method allows for negative variance values [69]. Decomposing the prediction error in three terms, namely the squared bias, the variance, and a noise term is suggested in [70]. In [71], the decomposition is related to the estimated probabilities, whereas in [72] the decomposition into the bias and variance is done for the classification rule. A bias/variance decomposition for any kind of error measure, when using an appropriate probabilistic model is derived in [73]. A low-biased SVMs is built based on bias-variance analysis in [74], [75]. Due to the fact that we would like to decompose the average prediction error in terms that employ the ``1/0'' loss function, we are motivated to adopt the approach proposed in [72].

In the following, we confine our analysis to a two-class pattern recognition problem. Let us define:

$$P(y_j \mid \mathbf{x}) = P(Y = y_j \mid \mathbf{X} = \mathbf{x}), \quad \text{for} \quad y_j \in \{-1, +1\}, \quad j = 1, 2. \tag{3.17}$$

It is well known that the Bayes classifier C_{opt} given by:

$$C_{opt}(\mathbf{x}) = \arg \max_{y_j \in \{-1, +1\}} P(y_j \mid \mathbf{x}) \tag{3.18}$$

yields the minimum prediction error:

$$err(C_{opt}) = 1 - \int_{\mathcal{X}} \max_{y_j \in \{-1, +1\}} \{P(y_j \mid \mathbf{x})\} p(\mathbf{x}) d\mathbf{x}. \tag{3.19}$$

If the probability density function $p(\mathbf{x})$ and the a priori probabilities $P(y_j)$ were known, $C_{opt}(\mathbf{x})$ could be computed by the Bayes rule:

$$P(y_j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y_j) P(y_j)}{p(\mathbf{x})}, \quad j = 1, 2, \tag{3.20}$$

where $p(\mathbf{x}) = \sum_{j=1}^{2} P(y_j) p(\mathbf{x} \mid y_j)$. Unfortunately, in real life, it is very difficult to

have an exact knowledge of either of them. However, some methods in the literature estimate the minimum decision error (3.19). For instance, given enough training data, the prediction error of the nearest neighbor rule, err_{NN}, is sufficiently close to the Bayes (minimum) prediction error. It has been shown that, as the size of the training set increases to infinity, the nearest neighbor prediction error is bounded from below by the Bayes minimum prediction error and from above as follows [76]:

$$err(C_{opt}) \leq err_{NN} \leq err(C_{opt})\left(2 - \frac{p}{p-1}err(C_{opt})\right) \leq 2 \cdot err(C_{opt}), \tag{3.21}$$

where p is the number of classes (e.g. $p = 2$ in our case). In other words, the nearest neighbor rule is asymptotically at most twice as bad as the Bayes rule, especially for small $err(C_{opt})$. Having this in mind and having computed err_{NN} we can obtain an upper bound estimate of $err(C_{opt})$.

Let us form B quasi-replicas of the training set $\mathcal{L}_1, \ldots, \mathcal{L}_B$, each consisting of n instances, drawn randomly, but with replacement. An instance $(\mathbf{x}, y)$ may not appear in a replica set, while others could appear more than once. Due to the fact that the n-th outcome being selected $0, 1, 2, \ldots$ times is approximately Poisson - distributed with parameter 1 when n is large, on average 63% of the original training set will appear in the bootstrap sample [58]. The learning system then generates the classifiers C_b, $b = 1, \ldots, B$, from the bootstrap samples and the final classifier C_A is formed by aggregating the B classifiers. C_A is called the *aggregated classifier*. In order to classify a test sample $\mathbf{x}_o$, a voting between the class labels y_{ob} derived from each classifier, $C_b(\mathbf{x}_o, \mathcal{L}_b) = y_{ob}$, is performed and $C_A(\mathbf{x}_o)$ is the class received the most votes. In other words, the aggregated classifier is given by:

$$C_A(\mathbf{x}_o) \triangleq \text{sign}\{E_{\mathcal{F}}\{C(\mathbf{x}_o, \mathcal{L}^*)\}\}, \tag{3.22}$$

where $\mathcal{L}^* = \{\mathcal{L}_1, \ldots, \mathcal{L}_B\}$. For example, suppose that for $(\mathbf{x}_o, y_o)$, $C(\mathbf{x}_o, \mathcal{L}^*)$ outputs the class $\{-1\}$ with a relative frequency $3/10$ and class the $\{+1\}$ with a relative frequency $7/10$, respectively. Then $C_A(\mathbf{x}_o)$ predicts the $\{+1\}$ class label. The aggregated classifier is also named as *bagging predictor* [65]. In the following, we deal with the bias and the variance of a classifier. Let us define the *bias* of classifier C as:

$$bias(C) = E_{\mathcal{F}}E_{\mathcal{OF}}\{Q[C_{opt}(\mathbf{X}_o, \mathcal{L}), C_A(\mathbf{X}_o)]\} = err(C_A) - err(C_{opt}), \tag{3.23}$$

where the dependence of the Bayes classifier on $\mathcal{L}$ is explicitly stated. Therefore, the bias of classifier C is the average number of mismatches in the classifications produced by the Bayes classifier and the aggregated classifier. C is called unbiased if its aggregated classifier C_A predicts the same class as the Bayes classifier with probability 1 over the inputs. The *variance* of classifier C is expressed by [72]:

$$var(C) = E_{\mathcal{F}}E_{\mathcal{OF}}\{Q[C(\mathbf{X}_o, \mathcal{L}), C_A(\mathbf{X}_o)]\}. \tag{3.24}$$

The variance measures the dispersion of C_A around C due to the variations from one bootstrap replica to another. Another quantity of interest is the *aggregation effect* defined by:

$$ae(C) = err(C) - err(C_A) = (\delta - 1) \cdot err(C_A) \tag{3.25}$$

where:

$$\delta \triangleq \frac{err(C)}{err(C_A)}. \tag{3.26}$$

Having defined the bias, the variance, and the aggregation effect of a classifier, it can be easily shown that the following decomposition is valid [72]:

$$err(C) = err(C_{opt}) + bias(C) + ae(C). \tag{3.27}$$

3.2.2 Bootstrap error estimate for the bagged classifier

Using the leave-one-out strategy, a sample-based estimate of the prediction error decomposition can be derived according to [72]. By doing so, we can draw the numerical evidence in order to demonstrate if a classifier is stable or not.

We can estimate the aggregated predictor expressed in (3.22) by:

$$\widehat{C}_A(\mathbf{x}) \equiv \text{sign}\{E_{\widehat{\mathcal{F}}}\{C(\mathbf{x}, \mathcal{L}^*)\}\} \tag{3.28}$$

where $\widehat{\mathcal{F}}$ is the empirical probability distribution over $\mathbf{Z}$. The computation of (3.28) is performed as follows:

1. Create ordinary bootstrap samples $\mathcal{L}_1^* = \{\mathbf{z}_1^*, \mathbf{z}_2^*, \ldots, \mathbf{z}_n^*\}$ with replacement from $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$;
2. Create B bootstrap samples. Let the bootstrap samples be $\mathcal{L}_b^*$, $b = 2, 3, \ldots, B$;
3. Let N_i^b be the number of times $\mathbf{z}_i$ appears in the b-th bootstrap sample and:

$$I_i^b = \begin{cases} 1 & \text{if} \quad N_i^b = 0 \\ 0 & \text{if} \quad N_i^b > 0. \end{cases} \tag{3.29}$$

If $\widehat{\mathcal{F}}_{(i)}$ is the distribution assigning probabilities $1/(n-1)$ to all training observations, except $\mathbf{x}_i$, where it assigns zero probability, then the aggregated classifier can be estimated by:

$$\widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)}) = \text{sign}\left\{ \frac{\sum_{b=1}^B I_i^b C(\mathbf{x}_i, \mathcal{L}_b^*)}{\sum_{b=1}^B I_i^b} \right\}. \tag{3.30}$$

An estimate of the classifier variance is:

$$\widehat{var}(C) = \frac{1}{n} \sum_{i=1}^n \left\{ \frac{\sum_{b=1}^B I_i^b Q[C(\mathbf{x}_i, \mathcal{L}_b^*), \widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)})]}{\sum_{b=1}^B I_i^b} \right\}. \tag{3.31}$$

Subsequently, we determine the estimate for the prediction error of classifier C. Using the leave-one-out cross validation technique, the average prediction error (3.16) can be estimated in the following manner:

$$\widehat{err}(C) = E_{\widehat{\mathcal{F}}}\{E_{\widehat{\mathcal{F}}_{(i)}}\{Q[Y_o, C(\mathbf{X}, \mathcal{L}_{(i)})]\}\}, \tag{3.32}$$

where the set $\mathcal{L}_{(i)} = \mathcal{L} - \{(\mathbf{x}_i, y_i)\}$ contains the samples drawn from $\mathcal{F}_{(i)}$. The leave-one-out bootstrap estimate of the prediction error for C and $\widehat{C}_A$ is:

$$\widehat{err}(C) = \frac{1}{n} \sum_{i=1}^n \left\{ \frac{\sum_{b=1}^B I_i^b Q[y_i, C(\mathbf{x}_i, \mathcal{L}_b^*)]}{\sum_{b=1}^B I_i^b} \right\} \tag{3.33}$$

and

$$\widehat{err}(\widehat{C}_A) = \frac{1}{n} \sum_{i=1}^n \left\{ \frac{\sum_{b=1}^B I_i^b Q[y_i, \widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)})]}{\sum_{b=1}^B I_i^b} \right\}. \tag{3.34}$$

respectively. Now, we can estimate the aggregation effect as:

$$\widehat{ae}(C) = \widehat{err}(C) - \widehat{err}(\widehat{C}_A) = (\widehat{\delta} - 1) \cdot \widehat{err}(\widehat{C}_A), \tag{3.35}$$

where

$$\widehat{\delta} \triangleq \frac{\widehat{err}(C)}{\widehat{err}(\widehat{C}_A)}. \tag{3.36}$$

Notice that the leave-one-out bootstrap estimate is equivalent with the $.632$ bootstrap estimator [58]. The minimum (optimal) prediction error can be estimated, as suggested in [76], by the lower bound of the inequality:

$$err(C_{opt}) \geq \alpha - [\alpha(\alpha - err_{NN})]^{1/2} \tag{3.37}$$

where err_{NN} is the prediction error of the NN classifier. In the case of a two-class problem, $\alpha = 1/2$. Then, the bias estimate is upper bounded by:

$$\widehat{bias}(C) \leq \widehat{err}(\widehat{C}_A) - [\alpha - [\alpha(\alpha - err_{NN})]^{1/2}]\Big|_{\alpha=1/2}. \tag{3.38}$$

Another upper bound of the bias can be obtained if a k - nearest neighbor (k - NN) classifier is employed. It is known that [77]:

$$\widehat{bias}(C) \leq \widehat{err}(\widehat{C}_A) - err_{kNN}\Big|_{k=5}, \tag{3.39}$$

where we used that

$$err(C_{opt}) \geq err_{kNN}\Big|_{k=5}. \tag{3.40}$$

Finally, the bootstrap estimate of prediction error is obtained by

$$\widehat{err}(C) = \widehat{err}(C_{opt}) + \widehat{bias}(C) + \widehat{ae}(C) \tag{3.41}$$

where $\widehat{err}(C_{opt})$ is estimated by the lower bound of (3.40). A classifier is said to be stable, if the aggregation effect is negative or zero, or, equivalently if:

$$\widehat{\delta} \leq 1. \tag{3.42}$$

The stability indicator $\widehat{\delta}$ can be viewed as a bagging *gain* in the sense that, if it is less than or equals 1 bagging does not yield any improvement in the classification performance.

3.2.3 Experimental results

We draw numerical evidence to support our claim on the stability of SVMs for the face detection task.

3.2.3.1 *Data description*

Three image databases are employed in our experiments. They contain facial and non-facial patterns. The first database, the so called IBERMATICA database, consists of 464 images in total. It was collected within the framework of M2VTS project. The facial

patterns extracted from this database contain several degradations, such as changes in illumination, varying expressions, scale variations, etc [78]. The spatial resolution of images is 320 × 240 pixels. The images were recorded in 256 grey levels. Each face image has been cropped with a rectangle of 160 × 128 pixels, which includes the major fiducial points such as the eyebrows, eyes, nose, mouth, and chin, as shown in Figure 3.4. Each image has been downsampled four times, finally yielding an image of 10 ×

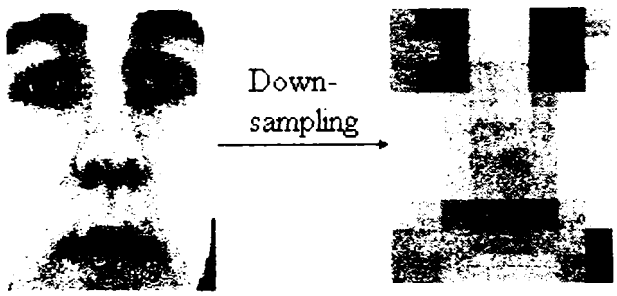


Figure 3.4: Example of a cropped face from the IBERMATICA database. Left: an original image of size 320 × 240 pixels. Right: a downsampled facial image to 10 × 8 pixels, properly magnified for visualization purposes.

8 pixels. This preprocessing step was used to reduce the dimension of input patterns. The ground truth (i.e., the class label $y_i = +1$) was appended to each facial pattern. Non-facial patterns have been collected from images depicting wheels, bubbles, trees, etc., in a similar manner to that described in [79]. That is:

1. Start with a small set of manually selected non-facial patterns in the training set.
2. Train an SVM classifier with the current training set.
3. Choose randomly an image that does not contain any face. Divide this image into blocks of size 10 × 8 and apply the SVM on each block. Collect all the blocks that the current system wrongly classifies as faces, if any. Add these non-facial patterns to the training set as new negative examples. This process is repeated for several times. Such misclassified non-facial patterns as facial ones are indicated by black rectangles in Figure 3.5. The non-facial patterns have been labeled by $y_i = -1$.

The AT&T (former Olivetti) [80] database was used to build the second data set of facial and non-facial patterns. This database contains 10 different images per person for 40 different persons. The images have dimensions 92 × 112 pixels. They have been recorded at different times, with variations in the lighting, facial expression, and facial details (glasses/nonglasses). They undergo the same preprocessing steps as the images of the IBERMATICA database. The final pattern size was 17 × 14. Note that the just mentioned pattern size for this data set is different than that of the first image data set due to the scaling variations between the face images in the AT&T face database and those in the IBERMATICA face database. The image set contains 306 facial patterns chosen randomly from the available face images and 294 non-facial patterns. Figure 3.6 shows several cropped facial images along with the corresponding downsampled versions.

While the first two image data sets can be considered as small ones, the third image data set is a combination of images from the AT&T face database and the face detection database collected by Rowley, Baluja, and Kanade [26]. The images has been downsampled so that facial and non-facial patterns of dimensions 17 × 14 are obtained. A set of 435 facial and 5,722 non-facial patterns has been created and

Figure 3.5: Patterns wrongly classified as faces by an SVM are appended as negative examples in the training set. Such patterns are marked with black rectangles.

used only in the test phase of our experiments. We refer to this image data set as the extended image data set. No further preprocessing was applied (e.g. masking, illumination gradient correction, or histogram equalization).

3.2.3.2 *Training phase*

We trained an aggregated SVM classifier on a set of 50 training samples extracted from the IBERMATICA face database augmented by non-facial patterns determined by the bootstrapping procedure. Another aggregated SVM classifier was trained on a second set of 50 training samples from the AT&T face database. A polynomial kernel of degree 2 was chosen. Since bagging can potentially be very useful, especially when the available amount of training data is small, we intentionally kept only 50 patterns from each set for training. We calculated the empirical distribution $\widehat{\mathcal{F}}$ and we computed

Table 3.5: Estimated prediction error (%) and its decomposition into bias and variance terms for an SVM with a quadratic kernel ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) and a 5-NN trained on the IBERMATICA database (21 bootstrap samples). The number in parenthesis refers to the equation used to compute the quantity in question.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (3.33)	0.5400 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (3.31)	0.0000 [0.0000]	0.0084 [0.0083]
$\widehat{bias}(C)$ (3.39)	0.5400 [0.0000]	0.0084 [0.0083]
$\widehat{err}(\widehat{C}_A)$ (3.34)	0.5400 [0.0000]	0.0084 [0.0083]
$\widehat{ae}(C)$ (3.35)	0.0000 [0.0000]	-0.0084 [0.0083]
$\widehat{\delta}$ (3.36)	1	0

the leave-one-out bootstrap estimate of the prediction error of SVM, the leave-one-out bootstrap estimate of the prediction error for the aggregated SVM classifier, the bias, and the variance. The number of bootstrap replicas was initially 21. We repeated the

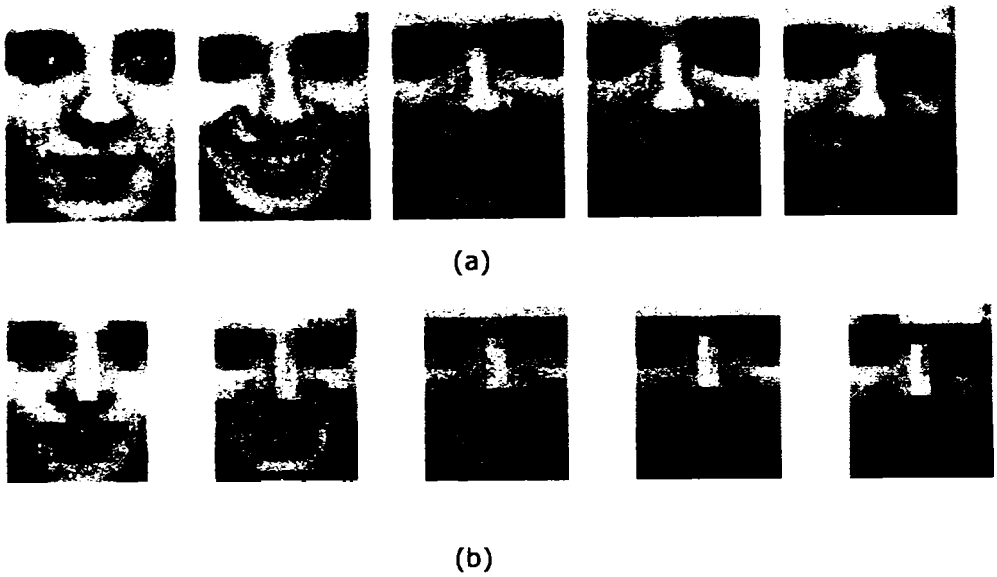


Figure 3.6: (a) Five different cropped face images of a person from the AT&T face database. (b) Downsampled face images corresponding to the original images in (a), properly magnified for visualization purposes.

experiment 10 times by forming other replicas of the training set. For comparison, we experimented also with a 5 - NN classifier. The aforementioned figures of merit are collected in Table 3.5 for the IBERMATICA database and Table 3.6 for the AT&T database, respectively, when the number of bootstrap replicas equals 21. The values depicted in Tables 3.5 and 3.6 are averaged over 10 runs. The prediction errors are expressed in percentage. The standard deviation for each figure of merit is given in brackets. Since $\widehat{err}_{5NN} = 0$, a lower bound for $err(C_{opt})$ is zero, according to (3.40). Note that we used the upper bound (3.39) to estimate the bias. Eq. (3.33) was used to compute the bootstrap estimate of the prediction error $\widehat{err}(C)$.

Table 3.6: Estimated prediction error (%) and its decomposition into bias and variance terms for an SVM with a quadratic kernel ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) and a 5-NN trained on the AT&T data set (21 bootstrap samples). The number in parenthesis refers to the equation used to compute the quantity in question.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (3.33)	0.5200 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (3.31)	0.0000 [0.0000]	0.0257 [0.0053]
$\widehat{bias}(C)$ (3.39)	0.5200 [0.0000]	0.0044 [0.005]
$\widehat{err}(\widehat{C}_A)$ (3.34)	0.5200 [0.0000]	0.0043 [0.005]
$\widehat{ae}(C)$ (3.35)	0.0000 [0.0000]	-0.0043 [0.005]
$\hat{\delta}$ (3.36)	1	0

From Tables 3.5 and 3.6 we notice that the prediction error of SVM after bagging does not change from the value it had before bagging. Due to the fact that err_{5NN} is zero, the bias equals $\widehat{err}(\widehat{C}_A)$. A zero or negative aggregation effect is characteristic of

a stable classifier. In the case of a 5 - NN classifier, bagging degrades the performance of 5 - NN.

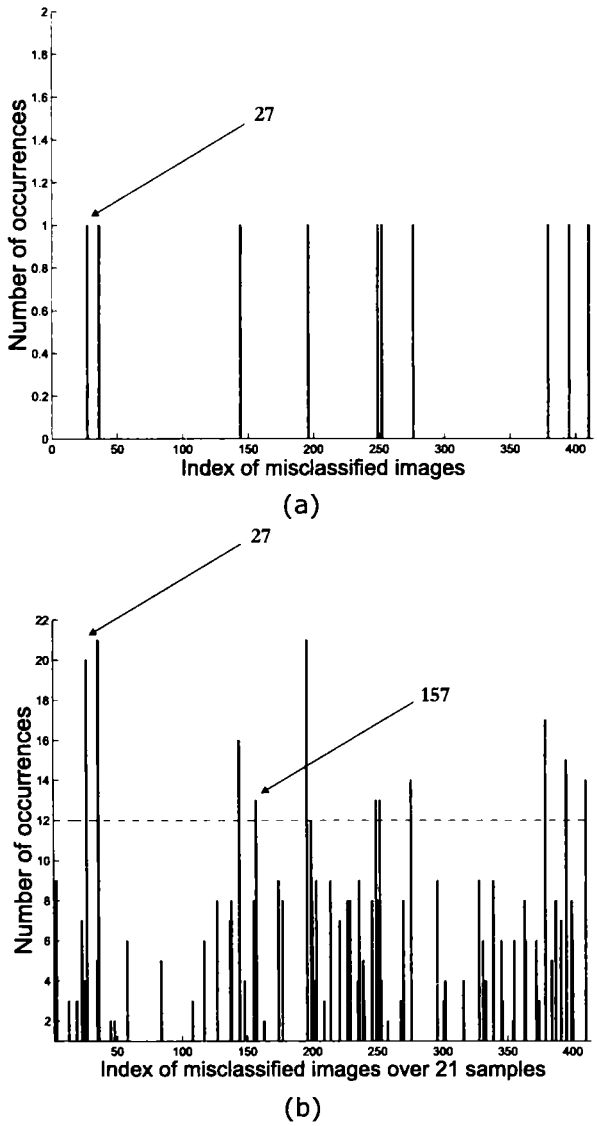


Figure 3.7: Face detection using a quadratic SVM on the IBERMATICA face database. (a) Histogram of the misclassified patterns before bagging. (b) Histogram of misclassified patterns when 21 SVMs are trained on 21 bootstrap samples and aggregation is performed.

Figure 3.7a depicts the histogram of the misclassified pattern indices without bagging for the experiment conducted on the IBERMATICA database. Figure 3.7b shows the histogram of misclassified pattern indices after bagging with 21 bootstrap replicas. One can observe that the classification accuracy does not improve with bagging. Therefore, $C(\mathbf{x})$ and $\widehat{C}_A(\mathbf{x})$ tend to make the same errors, as can be seen from the histogram bins that exceed the dashed line in Figure 3.7b. The same patterns are misclassified even when the training sets are changed. For example, the misclassified pattern with index 27 that is misclassified before bagging, is misclassified after bagging 20 out of the 21 times. In addition, a new misclassified pattern appears at index

Table 3.7: Average prediction error (%) in the test phase for SVMs applied to the IBERMATICA and AT&T face databases.

Database	Kernel	$B = 21, m = 60$			$B = 61, m = 20$		
		$\overline{err}(C)$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(C)$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$
IBERMATICA	linear	3.93	4.24	0.92	3.23	3.72	0.89
	quadratic	3.25	3.27	0.99	2.88	3.05	0.94
	ERBF	2.75	3.01	0.91	1.40	2.30	0.61
AT&T	linear	4.87	4.48	**1**.09	4.72	4.52	**1**.05
	quadratic	4.86	5.78	0.84	5.03	5.67	0.89
	ERBF	2.93	3.04	0.96	2.86	2.93	0.98

157 for the aggregated classifier. We observe that the aggregated classifier does not commit less errors, as one might expect. This could be attributed to the stability of SVMs.

3.2.3.3 *Test phase*

While the values of the SVMs parameters D and γ were arbitrarily chosen in the experiments (IBERMATICA, AT&T, and PID databases), for the extended image data set these values were determined by a cross-validation approach in a such a way that they yield the best accuracy in the training phase. The values of D and γ that yield the worst accuracy were also indicated. We run the SVM classifier with an ERBF kernel for $\gamma = \{0.001, 0.01, 0.05, 0.1, 1, 10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000\}$ and $D = \{0.1, 1, 10, 100, 500, 1000, 10000, 100000\}$. The same values of D were tested for the linear and quadratic kernel, respectively. The worst accuracy was obtained for $D = 0.1$ in the case of the linear kernel and for $(D = 500, \gamma = 0.01)$ in the case of the ERBF kernel. The best accuracy was obtained for $D = 500$ in the case of the linear kernel, and for $(D = 500, \gamma = 1)$ in the case of the ERBF kernel. For a polynomial kernel the same accuracy was obtained for all values of D. Accordingly, we chose to use $D = 500$. In the test phase, we are concerned only with the prediction error on the test set of a trained SVM classifier with/without bagging. We included also the extended image data set in our experiments besides the IBERMATICA and the AT&T databases. The following steps were followed:

1. Divide the initial database (e.g. IBERMATICA, AT&T) randomly into a training set of 50 images and a large test set comprised of the remaining images. That is, 414 and 550 samples for the IBERMATICA and the AT&T databases, respectively. Train the SVM with the training set and then apply the trained SVM on the test set.

2. Build $B = 21$ bootstrap replicas from the initial training set. Train the SVM on each replica, thus obtaining B classifiers.

3. Apply each of the B classifiers on the test set and aggregate these B classifiers for a final decision.

4. Repeat steps 1 - 3 for m= 60 times.

By averaging over m iterations, we obtain $\overline{err}(C)$ and $\overline{err}(\widehat{C}_A)$. We repeated also steps 1 - 4 for B = 61 and m = 20. The results for (B = 21, m = 60) and (B = 61, m = 20) are given in Table 3.7 for the two databases when SVMs with different kernels are used. All prediction errors are expressed in percentage. One can see from Table 3.7, that, after many trials, on average, $\widehat{\delta}$ is less than unity for the IBERMATICA database,

regardless of the kernel function used. Bagging dramatically degrades the prediction error for ERBF kernel function for $B = 61$ bootstrap samples and $m = 20$ iterations. This is the worst performance achieved in the test phase. An analogous degradation of SVM performance is also observed in the AT&T database for the polynomial and the ERBF kernels functions. The linear kernel is an exception, since $\widehat{\delta}$ exceeds unity by a small amount. Notice that the bootstrap estimate of the average prediction error is now measured during the test phase. This prediction error is different than the one obtained in the training phase (Tables 3.5 and 3.6), because the test set is disjoint to the bootstrap replicas of the training set.

For the extended image database we repeated the same steps, but the number of training samples was set to 200 and the remaining $5,957$ samples were used for testing. In addition the number of bootstrap replicas varies from 21 up to 141. Moreover, we compared the stability with the so-called Q statistics diversity measure [81]. Q varies between -1 and +1. Classifiers that tend to recognize the same patterns correctly will admit positive values of Q and those which commit errors on different patterns will lead to a negative Q. The closer to 1 is Q the more and the same patterns will be correctly or falsely classified by the ensemble of classifiers. Hence, the higher the value of Q is the worse is the ensemble (bagged classifier). For a high value of Q close to 1 the ensemble does not provide any advantage in accuracy over the single classifier, which in our case is similar to dealing with a stable classifier. The results are shown in Table 3.8 along with $\widehat{\delta}$ and average Q. We have reported both results corresponding to the parameters that provided the worst and the best performance to verify the statement of Evgeniou et al. [82] about tuning the SVMs parameters. They found that, when the parameters of a single SVM are tuned such as to yield the best performance, a bagged SVM does not improve the accuracy over the single SVM. Indeed, as it seen from Table 3.8, $\widehat{\delta}$ admits its largest value for an SVM with a linear kernel in the worst case. For a polynomial kernel and an ERBF kernel the marginal improvements in $\widehat{\delta}$ are correlated with the high values of Q, a fact that amplifies our claim on the stability of SVMs. The linear SVM although underperforming the quadratic SVM with respect to the average prediction error yields $\widehat{\delta}$ above 1. By increasing the number of bootstrap samples the classifier performance deteriorates for all kernels. The more bootstrap samples are used the worse classifier performance is obtained.

3.2.3.4 *Discussions*

In this Chapter, the behavior of SVM by applying bagging in the light of the bias and variance decomposition of the prediction error was investigated. Although bagging, which perturbs the initial training set and then combines the classifications produced on several replicas of the training set, has successfully improved the performance of many classifiers, there are several cases where this algorithm either does not help too much or may slightly degrade the pattern recognition performance. This happens to the class of stable classifiers. Here, we reported experimental evidence that the SVM classifiers can be included in the class of stable classifiers. We estimated the prediction error by means of a leave-one-out strategy and drew conclusions about the stability of the aforementioned classifiers by examining the values of the prediction error components in the training phase. For the face detection task, bagging SVMs is found to be useless. Even when, after many iterations on average, we may slightly obtain better results (see, for example, Table 3.7, AT&T database, linear kernel), bagging is not a good idea, because the price paid for a slight performance improvement is the huge processing time. To conclude, we can state that the empirical results collected from

Table 3.8: Average prediction error (%) before and after bagging in the test phase for the extended image database.

Kernel	without bagging	with bagging	B			
			21	61	101	141
Linear	worst case	$\overline{err}(\hat{C}_A)$	3.63	3.90	3.90	4.31
	$\overline{err}(C) = 4.35$	$\hat{\delta}$	**1**.19	1.11	1.11	1.01
		Q	0.98	0.97	0.89	0.99
	best case	$\overline{err}(\hat{C}_A)$	2.41	2.46	2.67	2.67
	$\overline{err}(C) = 2.73$	$\hat{\delta}$	**1**.13	1.10	1.02	1.02
		Q	0.98	0.88	0.85	0.80
Quadratic		$\overline{err}(\hat{C}_A)$	2.40	2.41	2.47	2.56
	$\overline{err}(C) = 2.60$	$\hat{\delta}$	**1**.08	1.07	1.05	1.01
		Q	0.95	0.97	0.99	0.99
ERBF	worst case	$\overline{err}(\hat{C}_A)$	5.9	6.5	6.5	6.8
	$\overline{err}(C) = 5.9$	$\hat{\delta}$	**1**.00	0.90	0.90	0.86
		Q	0.95	0.94	0.96	0.98
	best case	$\overline{err}(\hat{C}_A)$	1.34	1.39	1.64	1.74
	$\overline{err}(C) = 1.36$	$\hat{\delta}$	**1**.01	0.97	0.82	0.78
		Q	0.91	0.98	0.99	0.99

our experiments indicate that SVMs tend to behave like weakly stable classifiers when applied to face detection task.

CHAPTER 4

ICA applied for Facial Expression Recognition

4.1 Independent Component Analysis as a feature extraction method

One of the most popular techniques for dimensionality reduction is Principal Component Analysis (PCA). This technique is based on second-order statistics of the data and performs dimensionality reduction by retaining the components that correspond to the largest eigenvalues of the covariance matrix, while discarding those components that have insignificant contribution to data representation. In principle, PCA yields uncorrelated components. When the data have a Gaussian distribution, the uncorrelated components are independent as well. However, if the data are mixtures of non-Gaussian components, PCA fails to extract the components having a non-Gaussian distribution. On the contrary, Independent Component Analysis (ICA) takes into account higher-order statistics of the data in an attempt to recover the non-Gaussian components.

From a statistical point of view, the least interesting structure is the Gaussian one. In one dimension, two moments, the mean and the variance, completely define the probability density function (pdf). Moreover, the Gaussian distribution has the highest entropy among all distributions with a given covariance matrix [83]. Taking the Gaussian distribution as a reference, any quantity that measures the level of "interestingness" of the data, is a quantity that measures the non-Gaussian structure of the data. A principled measure of nongaussianity is the negentropy. The negentropy of a standardized random variable (i.e. one that has zero-mean and unit variance) can be approximated by the third-order moment and the fourth-order cumulant (i.e. the kurtosis) in a computationally simple manner. Therefore, we need moments and cumulants of order higher than 2 to capture the non-Gaussian structure of data [83]. All these quantities are closely related to the methods employed in order to find statistically independent components. Seeking non-Gaussian components is related to looking for statistical independence [83]. A measure of non-Gaussianity of a random variable (RV) s is its normalized kurtosis estimated as:

$$\text{kurt}(s) = \frac{\sum_i (s_i - \bar{s})^4}{\left[\sum_i (s_i - \bar{s})^2\right]^2} - 3 \qquad (4.1)$$

where s_i are the observations of s and $\bar{s}$ denotes its sample mean. The normalized kurtosis for Gaussian RVs is zero. Super-Gaussian RVs have a positive kurtosis. A typical super-Gaussian pdf is the Laplacian pdf. Sub-Gaussian RVs have negative

kurtosis with a typical example being the uniform RV in the interval $[-\alpha, \alpha] \in \mathbb{R}$.

ICA can be formulated by considering the following statistical model:

$$\mathbf{x} = \mathbf{As} \tag{4.2}$$

where $\mathbf{s} = (s_1, s_2, \ldots, s_n)^T$ is a latent random vector with independent components that are combined via a mixing $m \times n$ matrix $\mathbf{A}$ to form a zero-mean observation vector $\mathbf{x} = (x_1, x_2, \ldots, x_p)^T$. The task of ICA is to estimate a demixing matrix $\mathbf{W}$ of dimensions $n \times p$ that will recover the original components of $\mathbf{s}$ as:

$$\mathbf{u} = \mathbf{Wx} = \mathbf{WAs} \tag{4.3}$$

where $\mathbf{u} = [u_1, u_2, \ldots, u_i, \ldots, u_n]$ is an estimate of $\mathbf{s}$. Given a batch of m observation data $\mathbf{x}_j$, $j = 1, \ldots, m$ we can form $\mathbf{X}$ whose columns are $\mathbf{x}_j$. Then (4.3) becomes:

$$\mathbf{U} = \mathbf{WX} = \mathbf{WAS} \tag{4.4}$$

where $\mathbf{X}$ and $\mathbf{U}$ are $p \times m$ and $n \times m$ matrices, respectively. Usually, we call the columns of $\mathbf{U}$ (and implicitly the columns of $\mathbf{S}$) *independent sources*. The columns of $\mathbf{X}$ are measurements from a number of sensors that capture the sources. Usually, the number of observed components is equal to the number of independent components ($p = n$). There are ICA methods that cope with cases $p < n$ or $p > n$, called *overcomplete* or *undercomplete* ICA, respectively. Basically, the ICA algorithms attempt to obtain an estimate of $\mathbf{W}$ by using an objective (contrast) function that must be maximized or minimized, depending on the formulation.

4.2 ICA approaches

Let $p = n$. The *InfoMax* algorithm performs ICA based on the information maximization approach proposed by Bell and Sejnowski [84]. This approach relies on the maximization of the entropy of the joint distribution $f(\mathbf{u})$. The demixing matrix $\mathbf{W}$ is updated through an iterative process. At iteration $k + 1$, $\mathbf{W}$ is updated according to:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta[\mathbf{I} + (\mathbf{1} - 2\mathbf{z}_k)\mathbf{u}_k^T]\mathbf{W}_k, \tag{4.5}$$

where η is the learning rate controlling the convergence speed of the algorithm, $\mathbf{1}$ is a $n \times 1$ vector of ones, $\mathbf{I}$ is the $n \times n$ identity matrix, and $\mathbf{z}$ is a $n \times 1$ vector having elements:

$$z_i = g(u_i) \qquad i = 1, \ldots, n \tag{4.6}$$

with $g(.)$ being a component-wise nonlinearity applied to all elements of the demixer output $\mathbf{u}$, at each iteration k. The form of the nonlinearity must be chosen to match the cumulative distribution function of the input. In the Infomax algorithm [84], this non-linearity is approximated by the logistic transfer function:

$$g(u_i) = 1/(1 + e^{-u_i}) \qquad i = 1, \ldots, n. \tag{4.7}$$

The just described approximation works well when it comes to recover super-Gaussian components, but fails to extract the components having a sub-Gaussian distribution if such components exist in the mixture of non-Gaussians. Therefore, Lee et al. have extended the InfoMax algorithm to the *extended-InfoMax* approach by

employing a new learning rule that is able to separate both sub- and super-Gaussian distributions [85]. The learning rule that is able to switch between these distributions iteratively updates the demixing matrix as follows:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta[\mathbf{I} - \Gamma \tanh(\mathbf{u}_k)\mathbf{u}_k^T - \mathbf{u}_k\mathbf{u}_k^T]\mathbf{W}_k, \tag{4.8}$$

where Γ is an $n \times n$ diagonal matrix whose ii-th element, ξ_{ii}, takes the value 1 for a super-Gaussian source and the value -1 for a sub-Gaussian one, and tanh() denotes the hyperbolic tangent function that is applied to the elements of $\mathbf{u}_k$ in a component-wise fashion. The adaptation of γ_{ii} is given by:

$$\xi_{ii} = \text{sign}(E\{\text{sech}^2(u_{ki})\}E\{u_{ki}^2\} - E\{[\tanh(u_{ki})]u_{ki}\}). \tag{4.9}$$

where $i = 1, \ldots, n$, u_{ki} is the i-th element of $\mathbf{u}_k$, and sign() and sech() denote the sign and hyperbolic secant functions, respectively.

Another approach for separating sources, the so called *Joint Approximate Diagonalization of Eigen-matrices* (JADE) was proposed by Cardoso and Souloumiac [86]. The main advantage of JADE is the fact that it does not need a learning step for its tuning. Its drawback is the relatively small number of components that can be extracted, making it inadequate for a large number of mixture components. JADE has the following steps [86]:

1. Form the sample covariance matrix $\widehat{\mathbf{D}}_x = \frac{1}{m}\mathbf{X}\mathbf{X}^T$ and compute a whitening matrix $\widehat{\mathbf{V}}$.

2. Form the sample 4th-order cumulant tensor $\{cum(z_i, z_j, z_k, z_l) \mid 1 \leq i, j, k, l \leq n\}$, where z_i are the elements of $\mathbf{z} = \widehat{\mathbf{V}}\mathbf{A}\mathbf{s}$ and n is the number of sources/measurements.

3. Compute the eigenmatrices of the cumulant tensor.

4. Minimize the sum of the squared cross-cumulants of z_i.

The fourth approach employed in the paper is *fastICA* developed by Hyvarinen [87], which is an algorithm that maximizes negentropy. The fastICA algorithm steps for estimating several independent components with deflationary orthogonalization are the following [83]:

1. Center the data to zero their mean.

2. Choose the number n of independent components to be estimated. Set $p = 1$. Whiten the data to obtain $\mathbf{z} = \mathbf{V}\mathbf{x} = \mathbf{V}\mathbf{A}\mathbf{s}$.

3. Choose randomly an initial vector of unit norm for $\mathbf{w}_p$.

4. Let $\widehat{\mathbf{w}}_{p,k+1} = E\{\mathbf{z}_k g(\mathbf{w}_{p,k}^T\mathbf{z}_k)\} - E\{g'(\mathbf{w}_{p,k}^T\mathbf{z}_k)\}\mathbf{w}_{p,k}$, where $g(\xi) = (1/a)\log(\cosh(a\xi))$ is the contrast function and its derivative is given by $g'(\xi) = \tanh(a\xi)$.

5. Do the following orthogonalization $\widetilde{\mathbf{w}}_{p,k+1} = \widehat{\mathbf{w}}_{p,k+1} - \sum_{j=1}^{p-1}(\widehat{\mathbf{w}}_{p,k+1}^T\mathbf{w}_j)\mathbf{w}_j$.

6. Let $\mathbf{w}_{p,k+1} = \frac{\widetilde{\mathbf{w}}_{p,k+1}}{\|\widetilde{\mathbf{w}}_{p,k+1}\|}$.

7. If $\mathbf{w}_p$ has not converged, go back to step 4.

8. Set $p \longleftarrow p + 1$. If $p \leq n$, go back to step 3.

A major advantage of fastICA is its speed, making it even 100 times faster than the previously described approaches.

For all ICA approaches described so far, it has been assumed that the number of components equals the number of sensors. If the number of sources is very large, the application of ICA is limited by memory constraints. Therefore, the preprocessing PCA step is not only intended to decorrelate the data, but also to lower their dimension. By keeping only $l < p$ appropriately chosen dimensions the demixing matrix $\mathbf{W}$ becomes of size $l \times l$. When discarding the $(p - l)$ dimensional subspace with the smallest variance, there is a risk to throw away the independent components (ICs) that might be contained in this subspace, since there is no guarantee that ICs exist only in the l dimensional subspace defined by the principal components (PCs) with the largest eigenvalues. For instance, an IC a with very small variance was found to be associated with the form of the ``on-off'' experimental protocol when analyzing fMRI data [88]. To address the weakness of the previously described ICA methods, Stone and Porrill have developed the *undercomplete Independent Component Analysis* (uICA) for preserving the information that might be lost during PCA and established the following contrast function for maximizing the entropy [89]:

$$h(\mathbf{W}) = \frac{1}{2}\log|\mathbf{W}\mathbf{D}_x\mathbf{W}^T| + E\left\{\sum_{i=1}^{n}\log\left(\frac{\partial z_i}{\partial u_i}\right)\right\}. \tag{4.10}$$

allowing to have a non-square $n \times p$ demixing matrix without applying PCA for data dimensionality reduction. $\mathbf{D}_x$ is the sample covariance matrix of the input data $\mathbf{x}$. If $z_i = g(u_i) = \tanh(u_i)$, (4.10) can be maximized using, for example, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton method. The derivative of (4.10) is given by:

$$\frac{\partial h}{\partial \mathbf{W}} = \mathbf{W}^{\#T} - 2E\{\mathbf{u}\mathbf{x}^T\}. \tag{4.11}$$

where $\mathbf{W}^{\#} = (\mathbf{D}_x\mathbf{W}^T)(\mathbf{W}\mathbf{D}_x\mathbf{W}^T)^{-1}$ is the pseudoinverse of $\mathbf{W}$ with respect to the positive definite sample covariance matrix $\mathbf{D}_x$. However, when considering the whitened data, the covariance matrix equals the identity matrix, simplifying the first term of (4.10) to $\frac{1}{2}\log|\mathbf{W}\mathbf{W}^T|$ and $\mathbf{W}^{\#}$ to $\mathbf{W}^T(\mathbf{W}\mathbf{W}^T)^{-1}$.

All the aforementioned approaches treat the mixture $\mathbf{X}$ of independent components $\mathbf{S}$ as a linear one. It may happen to have components that are mixed using nonlinear functions. A kernel Hilbert space is used by Bach and Jordan to come up with the so called *kernel-ICA* algorithm to extract such sources that are mixed by using nonlinear functions [90]. Two contrast functions that rely on canonical correlations in this reproducing space have been defined namely the *kernel ICA-KCCA* (where KCCA stands for Kernel Canonical Correlation Analysis) and the *ICA-KGV* (where KGV stands for Kernel Generalized Variance). Kernel ICA-KCCA minimizes the first kernel canonical correlation that depends on the data $\mathbf{x}_j$, $j = 1, \ldots, m$ only through the centered Gram matrices for l ICs. Kernel ICA-KGV minimizes the kernel generalized variance. Both contrast functions are related to a generalized eigenvector problem $K_\kappa\alpha = \lambda D_\kappa\alpha$, where κ is a regularization parameter and K_κ and D_κ are block matrices constructed from the Gram matrices. Kernel ICA-KCCA deals with the minimal eigenvalue of the aforementioned problem whereas kernel ICA-KGV deals with the entire spectrum. The interested reader may consult [90] for more details.

4.3 Two architectures for performing ICA on images

Donato suggests that ICA features contain suitable and powerful discriminative information for classifying facial action units [43]. Facial expressions are combinations of such facial action units. Hence, ICA features may also be suitable for facial expression classification. In this paper, ICA is applied to facial images for feature extraction towards facial expression classification. We have m images containing human facial expressions, each image being of size $r \times c$ pixels, vectorized into a $p = rc$-dimensional vector by lexicographic ordering. There are at least two ways in which ICA can be applied to this problem namely the Architectures I and II [91].

4.3.1 Architecture I

The observation matrix $\mathbf{X}$ is formed by treating the face images as row vectors. Thus $\mathbf{X}$ is an $m \times p$ matrix. By doing so, ICA recovers m independent images. There are two preprocessing steps applied before ICA. The first step is PCA.

Let $\mathbf{D}_x$ be the covariance matrix of the original images, $\mathbf{D}_x = \frac{1}{m}\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}} = \mathbf{P}\square\mathbf{P}^T$, where $\widetilde{\mathbf{X}} = [\mathbf{x}_1 - \psi | \ldots | \mathbf{x}_m - \psi]^T$ with $\psi = \frac{1}{m}\sum_{k=1}^{m}\mathbf{x}_k$. Let us choose $l < p$ eigenvectors of $\mathbf{D}_x$ (those with the largest eigenvalues) and form $\mathbf{P}_l \in \mathbb{R}^{p\times l}$ whose columns are the eigenvectors. Each training face image $\mathbf{x}_k$ can be projected to the eigenvectors (called here eigenfaces) and be represented by $\mathbf{y}_k = \mathbf{P}_l^T(\mathbf{x}_k - \psi)$. Let us construct $\mathbf{Y} = [\mathbf{y}_1 | \ldots | \mathbf{y}_m]^T = [(\mathbf{x}_1 - \psi) | \ldots | (\mathbf{x}_m - \psi)]^T\mathbf{P}_l = \widetilde{\mathbf{X}}\mathbf{P}_l$. The original images can be reconstructed as linear combinations of the basis images $\mathbf{P}_l$ as $\mathbf{X}_{recPCA} = \mathbf{Y}\mathbf{P}_l^T$. In the following, we assume that $\psi = 0$ and accordingly $\widetilde{\mathbf{X}} = \mathbf{X}$. Whitening the data is the second preprocessing step. The whitening process transforms the original observation data by filtering them with $\mathbf{W}_s = 2(\frac{1}{m}\mathbf{P}_l^T\mathbf{P}_l)^{-1/2}$, such that the data are now given by $\mathbf{P}_{wl}^T = \mathbf{W}_s\mathbf{P}_l^T$. The transformed data constitute the input of the ICA process. By applying ICA to $\mathbf{P}_{wl}^T$ instead to the original observation matrix $\mathbf{X}$, a number of l ICs can be recovered into the columns of basis $\mathbf{U}$:

$$\mathbf{U} = \mathbf{W}\mathbf{P}_{wl}^T = \mathbf{W}(\mathbf{W}_s\mathbf{P}_l^T) = \mathbf{W}_I\mathbf{P}_l^T \tag{4.12}$$

where $\mathbf{W}_I = \mathbf{W}\mathbf{W}_s$. Hence, we have $\mathbf{P}_l^T = \mathbf{W}_I^{-1}\mathbf{U}$ and the ICA reconstruction of the original data is given by the approximation:

$$\mathbf{X}_{recICA} = \mathbf{Y}\mathbf{P}_l^T = \mathbf{Y}(\mathbf{W}_I^{-1}\mathbf{U}) = (\mathbf{X}\mathbf{P}_l\mathbf{W}_I^{-1})\mathbf{U}. \tag{4.13}$$

The rows of $\mathbf{B} = \mathbf{X}\mathbf{P}_l\mathbf{W}_I^{-1}$ contain the ICA coefficients of the linear combination of independent basis $\mathbf{U}$, where the training images are represented by the matrix $\mathbf{X}$. The rows of $\mathbf{B}$ are used further for classification. The ICA coefficients of a zero-mean test image $\mathbf{x}_{test}$ are obtained as:

$$\mathbf{b}_{test}^T = \mathbf{x}_{test}^T\mathbf{P}_l\mathbf{W}_I^{-1}. \tag{4.14}$$

4.3.2 Architecture II

Now consider $\mathbf{X}^T$. In this case, the pixels are assumed to be independent [91]. The columns of $\mathbf{X}$ are linear combinations of basis vectors obtained from the columns of matrix $\mathbf{W}_I$. In Architecture II, ICA is performed on the projected data $\mathbf{Y}^T = \mathbf{P}_l^T\mathbf{X}^T$. Therefore, the basis images obtained by performing PCA and ICA can be represented as $\mathbf{P}_l\mathbf{W}_I^{-1}$ and the coefficients needed for ICA reconstruction are expressed in the

columns of $\mathbf{U} = \mathbf{W}_I \mathbf{Y}^T$. The reconstructed images are:

$$\mathbf{X}_{recICA}^T = (\mathbf{P}_l \mathbf{W}_I^{-1})(\mathbf{W}_I \mathbf{Y}^T). \tag{4.15}$$

A zero-mean test image is represented as:

$$\mathbf{u}_{test} = \mathbf{W}_I \mathbf{P}_l^T \mathbf{x}_{test}. \tag{4.16}$$

4.4 Data description

The experiments have been performed using two databases. The first database has been derived from the Cohn-Kanade (C-K) AU-coded facial expression database [92] that contains single or combined action units. Facial action units have been converted to emotions according to [20]. Thirteen persons (expressers) who are able to express the six basic emotions create the database. Each subject from C-K database delivers an expression over time starting from a neutral pose and ending with a very intense expression, thus having several frames with different expression intensities. We picked up three poses with low (close to neutral), medium, and high (close to the maximum) intensity of facial expression, respectively. By doing so, the statistical variability of facial emotions is roughly captured. Therefore, the total number of images is 234 in the first database. The second database contains 213 images of Japanese female facial expressions (JAFFE) [33]. Ten expressers produced 3 or 4 examples for each of the 6 basic facial expressions (anger, disgust, fear, happiness, sadness, surprise) plus a neutral pose, thus producing a total of 213 images of facial expressions.

Each raw image $\mathbf{x}$ has been manually aligned with respect to the upper left face corner. The registration was performed by clicking the eyes - thus retrieving the eyes coordinates, followed by rotating the image to horizontally align the face according to eyes, cropping the face to remove the image borders and, finally, downsampling the image to a final size of 60 $\times$ 45 pixels for computational purposes. Figure 4.1 presents samples of facial expressions of one person from the JAFFE database posing 7 facial expressions and another person from the C-K database posing 6 facial expressions.

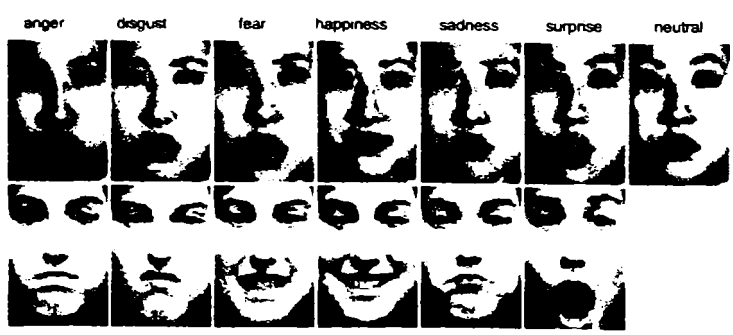


Figure 4.1: An example of one expresser from the JAFFE database posing 7 facial expressions (first row) and another one from the Cohn-Kanade database posing 6 facial expressions (second row).

4.5 Classifiers

Let us enumerate the 7 facial expressions (i.e. *anger, disgust, fear happiness, sadness, surprise, neutral*) by $j = 1, \ldots, 7$. The 7 expressions form 7 classes O_j, $j = 1, \ldots, 7$, where O_j is the class that corresponds to facial images depicting this particular expression. This setup is suitable for the JAFFE database that contains all the 7 facial expressions. In the case of the C-K database, we have only 6 expressions, therefore the enumeration ends at 6. In the experiments, two different classifiers are employed.

We used the *Cosine Similarity Measure* (CSM) classifier, since such a classifier was reported to yield a good classification performance [43]. The classification method is based on the nearest neighbor rule and uses the angle between a test vector $\mathbf{b}_{test}$ and the facial expression class center $\mathbf{b}_j$ as a similarity measure:

$$d_j = \frac{\mathbf{b}_{test}^T \mathbf{b}_j}{\|\mathbf{b}_{test}\| \|\mathbf{b}_j\|} \qquad j = 1, \ldots, N_e, \tag{4.17}$$

where $N_e = 7$ for JAFFE ($N_e = 6$ for C-K database) and chooses the class that corresponds to the maximal cosine similarity

$$\arg\max_{j=1,\ldots,N_e} \{d_j\}. \tag{4.18}$$

In the case of Architecture II, $\mathbf{b}$ is replaced by $\mathbf{u}$. From (4.17) it is seen that CSM is an 1-nearest neighbor classifier for normalized feature vectors.

SVMs [53] were employed for facial expression recognition, too. The sequential minimal optimization technique developed by Platt [93] was used to train SVMs having $\mathbf{b}$ and $\mathbf{u}$ as input, respectively. Since classical SVM theory was intended to solve a two class classification problem, we chose the Decision Directed Acyclic Graph (DDAG) learning architecture proposed by Platt et al. to cope with the multi-class classification [94]. It is worth noting that CSM and SVMs are the most popular classifiers for facial expression recognition, as they have been extensively used in [42], [43], [45].

The classifier accuracy, defined as the percentage of the correctly classified test images, is used to assess the performance of the facial expression recognition systems that employ the six ICA approaches in order to extract features, which subsequently feed the aforementioned classifiers.

4.6 ICA assessment

The six ICA approaches were applied to create feature vectors $\mathbf{b}_j, \mathbf{b}_{test}$ or $\mathbf{u}_j, \mathbf{u}_{test}$. We split the data into disjoint training and test sets. We used 164 and 150 images for training and we left out 70 and 63 images for testing in the C-K and JAFFE database, respectively. Both training and test set images were chosen randomly from the database. However, we ensured that both training and test data sets contain samples from all expressers and expressions. In the case of SVMs, five kernels were used namely the linear kernel, the polynomial kernel of degree 2,3, and 4, and the radial basis function (RBF). For all SVMs the penalizing parameter was set to 10 and the width of RBF kernel is $\sigma = 0.005$ [53]. Only the two among the five kernels that yield the highest accuracy are retained, except for the JAFFE database and Architecture II, where the linear kernel performed equally well to the polynomial kernel of degree 3.

The first objective is to find which ICA image representation performs best with respect to the classifier accuracy. The experiments were conducted by varying the number of principal components (PCs) from 5 to 160 (for the C-K database) and from 5 to 145 (for the JAFFE database) accounting from 24% to 99.8% of the trace of the covariance matrix. Due to the limited memory capacity and the algorithmic complexity, we were able to extract up to a maximum of 80 components in the JADE and the kernel-ICA approaches.

In order to see if the accuracy differences for the various classifiers and feature extraction approaches involved in experiments are statistically significant, we apply the approximate analysis described in [95]. We have examined if accuracy differences are statistically significant for pairs of the same classifier fed by features extracted by two different ICA approaches as well as for pairs of different classifiers fed by the best performing ICA approaches. The analysis is repeated for each database and architecture. Let us assume that the accuracies p_1 and p_2 are binomially distributed random variables. Let $\hat{p}_1, \hat{p}_2$ denote the empirical accuracies, and $\bar{p} = \frac{\hat{p}_1 + \hat{p}_2}{2}$. The hypothesis $H_0 : p_1 = p_2 = \bar{p}$ is tested at 95% level of significance. The difference of accuracies has variance $\beta = var(p_1 - p_2) = 2\frac{\bar{p}(1-\bar{p})}{N}$, where N is the number of test facial expression images. If

$$\hat{p}_1 - \hat{p}_2 \geq 1.65\sqrt{\beta} \tag{4.19}$$

we reject H_0 with risk 5% of being wrong. Then, we may claim that the accuracy difference is statistically significant at 95% level of significance.

The second issue investigated in the paper is related to the variation of recognition accuracy with respect to the mutual information of the basis images or their coefficients. The statistical dependencies of face representations were measured by computing the average mutual information between pairs of basis images that correspond to the maximum recognition accuracy achieved. The mutual information of two RVs u_1, u_2 is given by:

$$I(u_1, u_2) = H(u_1) + H(u_2) - H(u_1, u_2) \tag{4.20}$$

where H(u) is the differential entropy of the RV u [91]. The average mutual information calculated over all possible pairs of basis images is a good measure of the independence of basis images.

The nature of the independent components (ICs) and the influence of the discarded PCs in the recognition accuracy are investigated as well. The super- and sub-Gaussian nature of the basis images was tested by measuring their normalized kurtosis (4.1). Furthermore, non-linear mixtures of independent components were also investigated.

To obtain a better quantitative insight on how well the accuracy is correlated to the mutual information and the kurtosis over the number of PCs, we have computed the correlation coefficient and the corresponding p-value. Mutual information, kurtosis and accuracy were computed for various numbers of components from the set $\{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160\}$ for the C-K database and $\{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140\}$ for the JAFFE database. Accordingly, we have 17/15 values of the aforementioned quantities (mutual information, kurtosis, accuracy) for varying numbers of components that are stored in three 17/15-dimensional vectors. The correlation was then calculated between the elements of the vector comprising the mutual information values and the vector comprising the accuracy values as well as between the vector having as elements the kurtosis values and the vector of accuracies.

4.6.1 Cohn-Kanade database

4.6.1.1 Architecture I

The experimental results are presented in Table 4.1. The number of PCs varies between 5 and 160 and admits the values in the set $\{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160\}$. For each number of PCs, features are extracted by the several ICA approaches and the classifier accuracy is measured over the test facial expression images. The maximum accuracy obtained along with the corresponding number of PCs are listed in columns numbered by ``1'' and ``2''. For both the CSM classifier and the SVM with a polynomial kernel of degree 3, a small number of PCs yields a good classification accuracy. The classification accuracy obtained by the CSM classifier, when it employs features extracted by the InfoMax, the JADE, the fastICA, and the kernel-ICA was found to be identical. A decrease of approximately 3 % in accuracy was found, when features extracted by the extended-Infomax and the uICA. Overall, the best recognition accuracy was 82.9 % and was obtained by the linear SVM with fastICA, when 110 PCs were used. While such a large number of PCs is needed for the linear SVM in order to achieve the highest accuracy, 30 PCs are adequate for the SVM with a polynomial kernel of degree 3 in order to attain an accuracy of 81.43 %, which is reasonable compromise between accuracy and dimensionality reduction. It is worth noting that 140 PCs are necessary for the uICA and the linear SVM in order to reach an accuracy of 82.7 %, very close to the best accuracy. In Table 4.1, the highest accuracy appears in bold.

Table 4.1: Experimental results for the C-K database and Architecture I. The letters in column ``Met.'' (Method) refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA. The columns numbered from 1 to 10 represent: 1) classification accuracy (%), 2) Number of PCs, 3) average basis image mutual information, 4) and 5) normalized average positive and negative kurtosis of the basis images, 6) coefficient kurtosis, 7) and 8) correlation coefficient between the classification accuracy and the mutual information with its corresponding p-value, 9) and 10) correlation coefficient between the classification accuracy and the positive kurtosis with its corresponding p-value.

Clas.	Met.	1 (%)	2	3	4	5	6	7	8	9	10
CSM	A	74.3	10	0.07	4.1	NA	1.0	-0.03	0.91	0.01	0.95
	B	71.4	10	0.07	3.4	-0.8	1.4	-0.44	0.14	0.42	0.16
	C	74.3	30	0.03	14.1	NA	1.1	-0.44	0.22	0.27	0.47
	D	74.3	30	0.03	13.8	-0.5	0.9	-0.44	0.14	0.36	0.24
	E	71.4	50	0.00	32.9	-0.7	0.7	-0.31	0.38	0.27	0.31
	F	74.3	30	0.06	1.38	NA	0.5	-0.55	0.12	0.82	0.006
SVM linear	A	80	110	0.00	34.8	NA	1.4	-0.97	0	0.84	0.0006
	B	81.4	130	0.00	46.3	-1.5	1.3	-0.98	0	0.80	0.0001
	C	78.6	70	0.00	27.6	NA	0.9	-0.99	0	0.92	0.0003
	D	**82.9**	110	0.00	49.9	0	1	-0.97	0	0.78	0.0002
	E	82.7	140	0.03	1.2	NA	0.5	-0.78	0.0002	0.68	0.002
	F	78.6	70	0.04	1.4	NA	0.5	-0.80	0.007	0.67	0.012
SVM poly	A	80	20	0.04	8.4	NA	1.4	-0.56	0.053	0.34	0.27
	B	81.4	30	0.03	13.6	-0.9	1.1	-0.63	0.026	0.40	0.19
	C	80	20	0.05	9.2	NA	1.7	-0.60	0.020	0.56	0.28
	D	80	20	0.04	8.6	-0.7	1.7	-0.47	0.12	0.26	0.39
	E	78.3	100	0.04	1.0	NA	0.6	-0.49	0.10	0.38	0.21
	F	80	20	0.07	1.1	NA	0.6	-0.50	0.28	0.52	0.30

For each classifier, the accuracy differences due to different ICA approaches are not statistically significant at 95 % level of significance. The accuracy differences between the several pairs of classifiers that employ the best performing ICA approaches, such

as (CSM & fastICA, SVM linear & fastICA), (SVM linear & fastICA, SVM cubic & extended ICA) etc., are not statistically significant at 95 % level of significance as well.

One merit of ICA is that it produces independent and sparse basis images or coefficients depending on the architecture employed. For Architecture I, the basis images are expected to be independent and sparse. Their independence is measured by the average mutual information listed in the third column of Table 4.1.

The presence of a super- or a sub-Gaussian distribution in the basis images is tested in columns ``4'' and ``5'' of Table 4.1. These columns show the average positive and negative kurtosis of the basis images indicating a super-Gaussian and a sub-Gaussian distribution, respectively, and constitute a measure of sparseness of the basis images. ``NA'' in the column ``5'' stands for ``Not Available'', i.e. when a sub-Gaussian distribution of basis images is not detected. The average negative kurtosis listed in column ``5'' shows that the presence of sub-Gaussian components does not necessarily enhance the classifier performance.

Ten basis images extracted from the C-K database during training with each method in the case of Architecture I are depicted in Figure 4.2. As one can notice from Figure

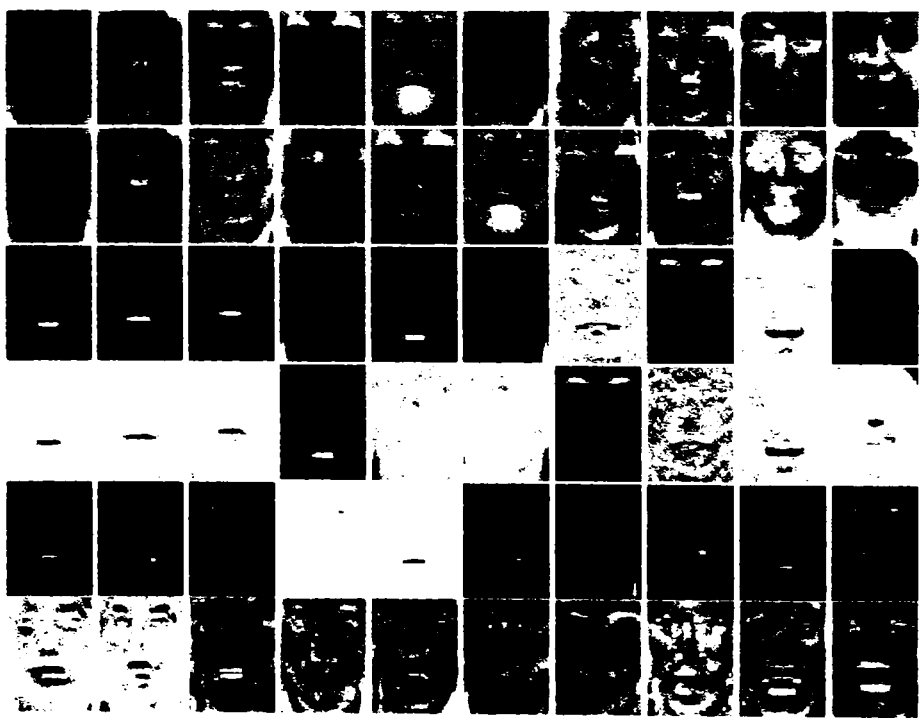


Figure 4.2: First ten basis images for Architecture I obtained by InfoMax (1st row), extended InfoMax (2nd row), JADE (3rd row), fastICA (4th row), undercomplete ICA (5th row), and kernel-ICA (6th row). The images are depicted in decreasing order of normalized kurtosis.

4.2, the basis images for JADE, fastICA, and uICA are more sparse than the basis images derived by the remaining methods.

The normalized kurtosis of the image representation coefficients was used to measure their sparseness. Column ``6'' in the Table 4.1 shows the coefficient sparseness measured by kurtosis is not as high as that of the basis images (column ``4'').

Columns ``7'' and ``8'' in Table 4.1 record the correlation coefficient between the accuracy and the average mutual information over all possible pairs of basis images

extracted for each number of PCs (mutual information for short, hereafter) and the corresponding p-value. The last two columns list the correlation coefficient between the classification accuracy and the average positive kurtosis of the basis images (positive kurtosis for short, hereafter). The strongest correlation between accuracy and mutual information was found for the linear SVM. The minus sign achieved for all classifiers indicates a negative correlation, meaning that a decrease in mutual information (hence greater independence) correlates with an increase of the classifier accuracy. The correlation is weak in the case of the CSM classifier and the SVM with a polynomial kernel of degree 3. Indeed, for the CSM classifier, the p-value exceeds 0.05, a fact that indicates that the correlation coefficient is not statistically significant. For the SVM with a polynomial kernel of degree 3, the extended InfoMax and the JADE exhibit a correlation coefficient between accuracy and mutual information that is statistically significant. A similar behavior was observed for the correlation between the basis image sparseness and accuracy. For an SVM with a linear kernel, a strong statistically significant correlation between accuracy and the positive kurtosis values is found.

The uICA was used in order to avoid discarding PCs having a small variance, but might contain ICs. The experiments have shown that, for the CSM classifier and the SVM with a polynomial kernel of degree 3, applying PCA for input dimensionality reduction is a good practice, since it yields the best performance for a small number of PCs. The uICA was not able to improve the accuracy by processing the original image data.

The linear SVM is the only classifier for which the details count, since a large number of PCs is needed in order to achieve the highest accuracy. However, this is due to the linear separating hyperplane which performs best in high-dimensional spaces.

The last investigated aspect was the assessment of the descriptive power of nonlinear IC mixtures. By applying kernel-ICA to this end, it was observed that the nonlinear ICA does not enhance the recognition performance.

4.6.1.2 Architecture II

The experimental findings are summarized in Table 4.2. All ICA approaches with the CSM classifier yield the same accuracy (72.9%), as one can see from column ``1''. The best accuracy (80%) was obtained by the SVM with an RBF kernel that employs features extracted by the extended Infomax. However, the accuracy difference between 80% and 72.9% is not statistically significant for 95% level of significance. Moreover, the pairwise performance differences within each classifier due to different ICA approaches are not statistically significant at the same level of significance. This is also valid for all pairs of classifiers that employ the ICA approach yielding the highest accuracy.

The second architecture derives coefficients that are as independent and sparse as possible. The mutual information and the average positive and negative kurtosis was measured for coefficients, as shown in columns ``3''--``5'' of Table 4.2, while column ``6'' quantifies the sparseness (kurtosis) of the basis images. By comparing the column ``6'' in Table 4.2 and column ``4'' in Table 4.1, one can notice that the basis images in Architecture II are not as sparse as in Architecture I. Ten basis images corresponding to C-K database which are obtained after training each method in Architecture II are depicted in Figure 4.3. They have a rather holistic appearance compared with the sparse basis images of Figure 4.2.

As for Architecture I, a weak correlation between the CSM classifier accuracy and mutual information was found. Only Infomax and Extended Infomax yield a statis-

Table 4.2: Experimental results for the C-K database and Architecture II. The letters in column ``Met.'' (Method) refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA. The columns numbered from 1 to 10 represent: 1) classification accuracy (%), 2) Number of PCs, 3) average coefficient mutual information, 4) and 5) normalized average kurtosis of super- and sub-Gaussian coefficients, 6) basis kurtosis, 7) and 8) correlation coefficient between the classification accuracy and the mutual information with its corresponding p-value, 9) and 10) correlation coefficient between the classification accuracy and the positive kurtosis with its corresponding p-value.

Clas.	Met.	1 (%)	2	3	4	5	6	7	8	9	10
CSM	A	72.9	40	0.02	14.7	NA	1.7	-0.70	0.01	0.26	0.41
	B	72.9	10	0.13	2.3	-1.3	1.1	-0.57	0.049	0.64	0.02
	C	72.9	10	0.13	1.1	NA	0.7	-0.49	0.176	0.09	0.08
	D	72.9	10	0.08	3.5	-1.7	1.1	-0.21	0.50	0.08	0.78
	E	72.9	60	0.00	0.1	-1.8	0.9	-0.21	0.49	0.32	0.308
	F	72.9	10	0.13	1.1	-0.5	0.7	-0.36	0.337	0.03	0.92
SVM linear	A	75.7	90	0.00	38.6	NA	4.5	-0.91	0	0.60	0.003
	B	72.8	110	0.00	5.2	-1.5	3.3	-0.98	0	0.80	0.001
	C	72.8	60	0.01	42.1	NA	1	-0.94	0.0004	-0.06	0.88
	D	75.2	110	0.00	30.2	0	71.7	-0.98	0	-0.9	0.005
	E	73.3	100	0.00	10.5	-0.5	2.2	-0.70	0.1	0.65	0.02
	F	75.7	40	0.02	0.4	-0.8	1.7	-0.75	0.1	0.48	0.4
SVM poly	A	71.4	20	0.00	8.9	NA	1.4	-0.11	0.73	0.71	0.008
	B	74.3	10	0.13	2.3	-1.3	1.1	-0.08	0.79	0.03	0.91
	C	75.7	20	0.04	0.8	NA	0.8	-0.10	0.80	0.40	0.09
	D	75.7	20	0.00	8.5	-0.3	1.4	0.27	0.38	0.76	0.004
	E	75.7	90	0.00	9.1	-0.3	0.6	-0.23	0.46	0.76	0.47
	F	75.7	20	0.04	0.8	-0.5	0.8	-0.20	0.3	0.45	0.10
SVM RBF	A	74.3	30	0.01	12.1	NA	1.4	-0.54	0.069	0.10	0.75
	B	**80**	120	0.00	6.8	-1.4	1.5	-0.96	0	0.88	0
	C	75.7	70	0.05	51.8	NA	1.7	-0.78	0.008	0.74	0.009
	D	78.6	100	0.06	76.3	0	1.7	-0.99	0	0.74	0.005
	E	71.8	60	0.00	0.1	-1.8	0.6	-0.17	0.59	0.65	0.019
	F	75.7	70	0.00	1.7	-0.3	0.6	-0.41	0.3	0.57	0.09

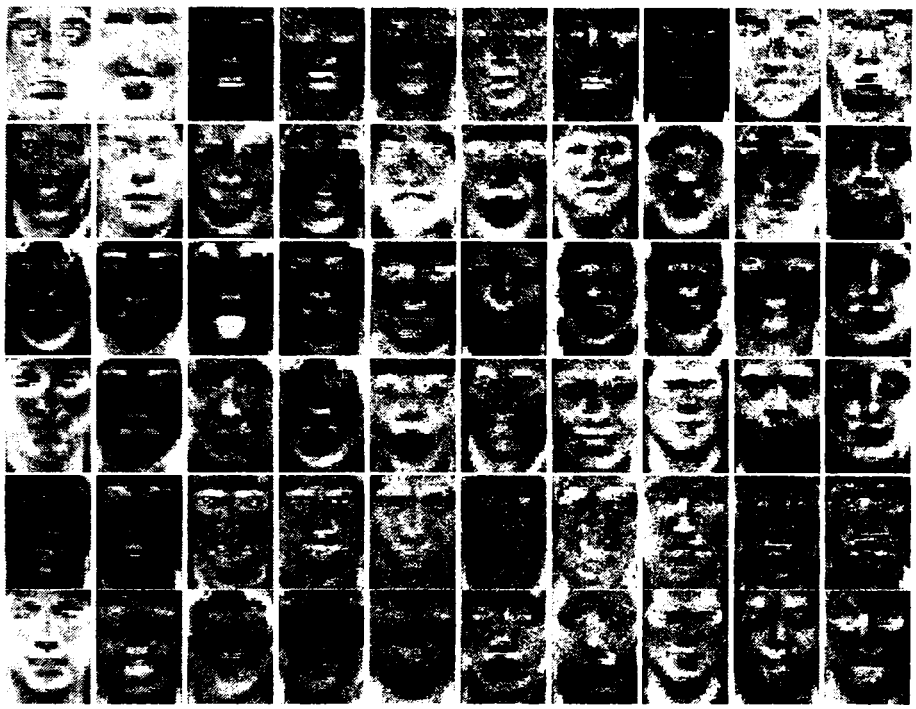


Figure 4.3: First ten basis images for Architecture II obtained by InfoMax (1st row), extended InfoMax (2nd row), JADE (3rd row), fastICA (4th row), undercomplete ICA (5th row), and kernel-ICA (6th row). The images are depicted in decreasing order of normalized kurtosis.

tically significant correlation. In contrast, strong statistically significant correlations between the accuracy of the SVM classifier with an RBF kernel and mutual information were measured. In this case, 3 out of the 6 ICA approaches yield statistically significant correlations and the best performing classifier (i.e., SVM-RBF with Extended Infomax) shows the second highest correlation. The linear SVM shows a strong correlation between mutual information and accuracy at least for 4 out of the 6 ICA approaches (i.e., Informax, Extended Infomax, JADE, fastICA) consistently in Tables 4.1 - 4.4. This suggests that independence is associated with a more linearly separated feature space.

Overall, the Architecture II yields a smaller classification accuracy than the Architecture I.

4.6.2 JAFFE database

4.6.2.1 Architecture I

The experimental results are summarized in Table 4.3. The facial expressions in JAFFE database are a little bit harder to be recognized than those recorded in the C-K database due to the fact that the human expressers in the former database were less expressive than those in the latter database. As a consequence, a larger number of PCs had to be retained in order to obtain the maximum recognition rate of 66.67% for the CSM classifier. This rate was obtained by all ICA approaches with Architecture I. However, the accuracy differences between all possible pairs of classifier employing

Table 4.3: Experimental results for the JAFFE database and Architecture I. The letters in column ``Met.'' (Method) refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA. The columns numbered from 1 to 10 represent: 1) classification accuracy (%), 2) Number of PCs, 3) average basis image mutual information, 4) and 5) normalized average positive and negative kurtosis of the basis images, 6) coefficient kurtosis, 7) and 8) correlation coefficient between the classification accuracy and the mutual information with its corresponding p-value, 9) and 10) correlation coefficient between the classification accuracy and the positive kurtosis with its corresponding p-value.

Clas.	Met.	1 (%)	2	3	4	5	6	7	8	9	10
CSM	A	66.7	40	0.00	15.5	NA	1.0	-0.75	0.004	0.62	0.030
	B	66.7	50	0.00	16.4	NA	1.0	-0.85	0.0004	0.66	0.017
	C	66.6	50	0.00	19.8	NA	0.7	-0.81	0.007	0.68	0.040
	D	66.7	50	0.00	17.0	-0.5	1.3	-0.88	0	0.70	0.010
	E	66.7	60	0.00	5.6	-0.2	0.5	-0.41	0.183	0.69	0.011
	F	66.7	50	0.01	2.2	NA	0.5	-0.84	0.003	0.72	0.027
SVM linear	A	76.2	60	0.00	19.6	NA	1.3	-0.98	0	0.92	0
	B	**79.4**	110	0.00	29.5	NA	1.5	-0.99	0	0.92	0
	C	73.2	80	0.00	31.8	NA	1.0	-0.77	0.008	0.74	0.09
	D	**79.4**	110	0.00	27.4	NA	1.1	-0.97	0.001	0.91	0
	E	77.2	110	0.01	1.4	NA	0.5	-0.83	0.001	0.62	0.009
	F	76.2	80	0.00	2.3	NA	0.6	-0.60	0.3	0.26	0.2
SVM RBF	A	71.4	70	0.00	22.6	NA	1.2	-0.92	0	0.83	0.007
	B	60.3	20	0.02	7.4	NA	2.2	-0.51	0.08	0.74	0.005
	C	63.4	20	0.02	8.7	NA	0.8	-0.62	0.36	0.71	0.09
	D	63.4	20	0.02	8.1	NA	1.7	-0.42	0.17	0.14	0.65
	E	62.5	40	0.01	22.9	-0.2	0.5	-0.39	0.20	0.21	0.50
	F	63.5	20	0.03	1.9	NA	0.7	-0.45	0.09	0.40	0.19

all ICA approaches are not statistically significant at 95 % level of significance.

In JAFFE database, a statistically significant correlation coefficient between mutual information and the accuracy of the CSM classifier for all ICA approaches except uICA was found. Moreover, the correlation coefficient between the accuracy of the CSM classifier and kurtosis was found to be statistically significant for all ICA approaches. This was not the case for the correlation coefficient between the accuracy of the CSM classifier and either mutual information or kurtosis for the C-K database. The linear SVM classifier yields the highest accuracy 79.4 %, when the extended-InfoMax and the fastICA approaches are employed. From the inspection of Table 4.3, it is seen that very strong statistically significant correlations between the classification accuracy and the mutual information of basis images as well as the classification accuracy and the positive kurtosis of the basis images are measured for the best performing ICA approaches with the linear SVM.

4.6.2.2 Architecture II

The experimental findings are collected in Table 4.4. The highest accuracy of 79.4 % was obtained with the linear SVM and fastICA. It is worth mentioning for the SVM-RBF classifier that the accuracy difference when Extended Infomax is employed instead of uICA is indeed statistically significant at the 95% level of significance. All other pairwise accuracy differences either within the same classifier due to different ICA approaches employed or across different classifiers are statistically insignificant at the same level of significance.

In the case of the SVM with a linear kernel, a statistically significant strong correlation between the classification accuracy and the mutual information was found for features extracted by InfoMax, Extended InfoMax, JADE, and fastICA, as can be seen

Table 4.4: Experimental results for the JAFFE database and Architecture II. The letters in column ``Met.'' (Method) refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA. The columns numbered from 1 to 10 represent: 1) classification accuracy (%), 2) Number of PCs, 3) average coefficient mutual information, 4) and 5) normalized average kurtosis of super- and sub-Gaussian coefficients, 6) basis kurtosis, 7) and 8) correlation coefficient between the classification accuracy and the mutual information with its corresponding p-value, 9) and 10) correlation coefficient between the classification accuracy and the positive kurtosis with its corresponding p-value.

Clas.	Met.	1 (%)	2	3	4	5	6	7	8	9	10
CSM	A	69.8	20	0.00	7.4	NA	7.4	-0.47	0.12	0.17	0.59
	B	68.3	40	0.01	1.9	-0.9	1.6	-0.77	0.14	0.54	0.06
	C	68.3	40	0.03	21.2	NA	21.2	-0.73	0.02	0.45	0.22
	D	68.3	40	0.03	16.6	-1.5	1.3	-0.53	0.07	0.21	0.5
	E	68.3	40	0.02	0.1	-0.8	0.5	-0.47	0.11	0.30	0.18
	F	68.3	40	0.00	0.7	-0.3	0.5	-0.76	0.01	0.91	0.000
SVM linear	A	74.6	100	0.06	41.4	NA	41.4	-0.97	0	0.77	0.003
	B	77.8	100	0.00	3.1	-1.2	1.7	-0.97	0	0.79	0.001
	C	76.2	60	0.05	42.1	NA	42.1	-0.88	0.009	0.63	0.07
	D	**79.4**	110	0.07	82.8	0	82.8	-0.95	0	-0.8	0.001
	E	77.2	70	0.00	100.1	-1.8	1.0	-0.58	0.17	0.49	0.10
	F	76.2	60	0.00	0.5	-0.2	1.2	-0.49	0.09	0.62	0.14
SVM RBF	A	69.8	40	0.03	14.6	NA	14.6	-0.21	0.49	0.32	0.30
	B	74.6	80	0.00	2.8	-1.6	1.8	-0.95	0	0.76	0.003
	C	68.3	70	0.05	44.9	NA	44.9	-0.70	0.08	0.49	0.008
	D	69.8	80	0.05	41.8	NA	41.8	-0.89	0.0001	0.52	0.07
	E	60.3	70	0.00	100.1	-1.8	1.0	-0.61	0.03	0.49	0.45
	F	69.8	80	0.00	0.5	-0.2	1.2	-0.71	0.05	0.33	0.25

in Table 4.4. In the case of fastICA with the linear SVM, an interesting phenomenon is the negative correlation between the accuracy and the positive kurtosis (i.e. -0.8 with p-value 0.001) indicating that the accuracy decreases with an increase in sparseness. However, the correlation between the accuracy of the linear SVM and mutual information is in the expected direction for mutual information, namely that performance increases as mutual information decreases. This is in par, with the measurement obtained for linear SVM with fast ICA in the C-K database.

4.6.3 Performance enhancement using leave-one-set of expressions-out

One possible way of improving accuracy is by exploiting maximally the available data set. To do so, we repeated the experiments by employing the leave-one-set of expressions-out (leave-one-out for short, [LVO]) strategy. That is, one set of expressions was left out for test in a cyclic fashion. During one rotation, the number of training images is 228 and the number of test images is 6 and by performing 39 rotations overall 234 test images are produced for the C-K database. In a similar way, the rotations yield 214 test images for the JAFFE database.

For both databases, the accuracy of all classifiers employing different ICA approaches was increased substantially, as can be seen in Table 4.5. For example, an impressive performance enhancement was noticed for the kernel-ICA with the linear SVM in Architecture I applied to the C-K database. Its accuracy was raised from 78.6 % to 86.6% with LVO.

The statistical significance of accuracy differences at 95% level of significance was studied for each Architecture and each database: (i) within the same classifier for all possible pairs due to different ICA approaches; (ii) across different classifiers employing the best performing ICA approaches.

For the C-K database and Architecture I, the only statistically significant accuracy difference is that between the accuracy of the CSM classifier that employs Infomax (81.4 %)and the SVM with a cubic kernel that employs fastICA (87.6 %). For the C-K database and Architecture II, the use of fastICA instead of uICA within the SVM classifier with an RBF kernel yields statistically significant performance improvement. The reader can verify that the accuracy differences between 84% and 77.3% as well as between 84% and 77% are also statistically significant.

For the JAFFE database and Architecture I, it can easily be checked that the accuracy differences between the CSM classifier and the SVM linear classifier are statistically significant irrespective of the ICA approach employed for feature extraction. Similarly the Infomax within the SVM classifier with an RBF kernel yields a statistically significant performance than the other ICA approaches. The accuracy differences between the CSM classifier and the SVM classifier with an RBF kernel, when Infomax is used, are also statistically significant. However, between the SVM classifier with a linear kernel that employs fastICA and the SVM classifier with an RBF kernel that employs Infomax there is no statistically significant performance difference. For the JAFFE database and Architecture II, the use of fastICA instead of uICA within the SVM classifier with a linear kernel yields a statistically significant accuracy difference. Similarly statistically significant performance differences are obtained between the SVM classifier with a liner kernel and fastICA (or the SVM classifier with an RBF kernel and Extended Infomax) and the CSM classifier irrespective of the ICA approach that feeds the latter classifier.

4.6.4 Subspace selection

Unlike PCA, there is no inherent ordering into the independent components [43]. An ordering parameter could be the class discriminability of each component [91] defined as the ratio

$$r = \frac{\sigma_{between}(k)}{\sigma_{within}(k)} \tag{4.21}$$

where

$$\sigma_{between}(k) = \sum_{j} (\bar{b}_k^j - \bar{b}_k)^2 \tag{4.22}$$

$$\sigma_{within}(k) = \sum_{j} \sum_{i} (b_k^{ij} - \bar{b}_k^i)^2 \tag{4.23}$$

with $\bar{b}_k$ denoting the gross me an of coefficient b_k, $\bar{b}_k^j$ being the jth facial expression class mean of coefficient b_k, and b_k^{ij} standing for the kth coefficient of the ith training image in the jth facial expression class.

It has been found that, by ordering the independent components with respect (4.21), ICA can outperform the PCA approach [43]. We have repeated the experiments with the CSM classifier in Architecture I, when feature selection is done according to (4.21) and compared the accuracy obtained with that reported previously (i.e. without subspace selection). We conducted the experiments for the maximum number of components and then we selected as many independent components according to (4.21), so that the maximum accuracy was obtained. The results are summarized in Table 4.6. Notice that Table 4.6 depicts results for the test set. By comparing the results in Table 4.6 and those in Table 4.3, one can see that, in JAFFE database, the

Table 4.5: Averaged accuracy obtained with leave-one-out. The letters in column ``Met.'' (Method) refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA. (NA stands for accuracy results that are not available).

Clas.	Met.	C-K database		JAFFE database	
		Arch. I	Arch. II	Arch. I	Arch. II
CSM	A	81.4	77.3	69.6	72.6
	B	82	80	69.6	70
	C	79	81.5	69.6	71
	D	81.3	81.5	69.6	71
	E	80.1	81.5	69.6	68.3
	F	81.1	80	69.6	67.8
SVM linear	A	81.3	NA	80.3	77.5
	B	81.3	NA	83.5	80
	C	82.3	NA	82.5	78
	D	84.6	NA	**84**	**81**
	E	83.3	NA	82.6	66
	F	86.6	NA	82.1	78
SVM poly	A	83.7	80	NA	NA
	B	84.6	77	NA	NA
	C	82.4	80	NA	NA
	D	**87.6**	80	NA	NA
	E	83.3	77.3	NA	NA
	F	85.7	78.2	NA	NA
SVM RBF	A	NA	81.5	79	79
	B	NA	83.8	64.7	**81**
	C	NA	80	68.3	77.5
	D	NA	**84**	69.3	74
	E	NA	70	65.2	72.5
	F	NA	79	68.3	77

accuracy obtained by each ICA approach after subspace selection is higher than that reported without subspace selection with the extended ICA being an exception. By cross-examining Tables 4.6 and 4.1, this observation is roughly valid for the accuracy obtained by each ICA approach with the exception of kernel-ICA in C-K database. However, accuracy differences are not statistically significant neither for the C-K database nor for the JAFFE one.

Table 4.6: Accuracy (%) for the CSM classifier in Architecture I on both databases along with the number of components corresponding to the maximum accuracy (in parenthesis and italics), retrieved by employing subspace selection. The letters in column ``Method'' refer to the ICA approach used: A) InfoMax, B) Extended Infomax, C) JADE, D) fastICA, E) uICA, and F) kernel-ICA

Database	Method					
	A	B	C	D	E	F
C-K	77.1 (*80*)	77.1 (*90*)	74.2 (*40*)	78.5 (*110*)	72.5 (*80*)	70 (*30*)
JAFFE	69.8 (*70*)	66.6 (*80*)	68.2 (*50*)	69.8 (*130*)	67.7 (*40*)	68.2 (*50*)

We should also mention that a supervised ICA technique, the so called ICA-FX [96], was developed in order to obtain features that are not only independent from each other, but also convey class information, contrary to the other ICA approaches studied in this paper, which are unsupervised ones and do not utilize the class information. Unlike the method described in [91], ICA-FX allows an intrinsic class information embedding. To examine to what extent the classification performance is affected by incorporating the class information inside the training procedure, we ran the ICA-FX approach on the C-K database and compared it with the classical ICA approach previously exploited. Due to the fact that the Architecture I does not allow us to make a comparison against ICA-FX, since ICA is performed on the PCA projection matrix implying loss of the class label, we chose ICA Architecture II [91], where class label is preserved. Table 4.7 shows that the CSM classifier yields a higher accuracy when it is fed by features extracted by ICA-FX than those extracted by the other six ICA approaches. The difference in accuracies is found to be statistically significant at 95 % confidence level.

Table 4.7: Accuracy results by employing subspace selection with the help of the ICA-FX approach. The results are shown for the Architecture II on Cohn-Kanade database using the CSM and the SVM classifiers.

C-K database, Architecture II		
Classifier	CSM	SVM RBF
Accuracy	84.28	78.8

4.6.5 Discussion and conclusions

A systematic comparative study of six ICA approaches was performed for facial expression classification in order to select the one that provides the best recognition

rate using two databases, two facial feature extraction architectures, and two classifiers. Regarding the classification performance, overall, the fastICA combined with SVMs yields a reasonable compromise between accuracy and fast run time for feature extraction. In our study we addressed the following issues:

1. *Performance variation with the number of PCs*: We found that a small number of PCs can produce a reasonable recognition performance for a CSM classifier. Although the present paper exhibits many commons issues with the work described in [43], we must notice that the present study differs in too many aspects with that in [43] that does not allow for a fair comparison between the results reported here and in [43].

2. *Implications of applying PCA prior to ICA to reduce data dimensionality*: We found that the use of uICA does not yield a higher classification accuracy than preprocessing observations by PCA.

3. *Features having super- and sub-Gaussian distribution did not improve facial expression classification accuracy*.

4. *Independent features obtained by non-linear unmixing of observations using kernel-ICA, do not improve the classification performance*. This fact indicates that either there is no such a non-linear mixture in the our data, or, if any non-linear mixture exists, its contribution to the classification performance is minimal.

5. The main conclusion drawn from the experiments is that, overall, as can be seen from Tables 4.1- 4.4, *there is a strong correlation between the average mutual information of independent components and accuracy*. *A similar finding was obtained for sparseness*. For the linear SVM classifier, this relationship is consistently statistically significant, when Infomax, extended Infomax, or fastICA is used for feature extraction. However, the degree of the correlation varies with the classifier and database involved.

6. *Statistically significant accuracy differences are measured only when the leave-one-set of expressions-out is used*. The LVO set-up enabled us to detect statistically significant accuracy differences as is detailed in Section 4.6.3.

ICA yields an efficient coding by performing a sparse image representation and removing the higher order correlations. Whether this is necessary for efficient image representation and pattern recognition purposes, it is still an open problem. It seems (and this is known to the scientific community) that SVMs are more affected by the outliers and noise which is the case of holistic representation. The outliers and ``noise'' are characterized by those parts of the face that are not essential for facial expression recognition and are present in a holistic representation that has a low degree of sparseness. As more localized features are obtained by ICA by employing more PCs and reducing the mutual information, thus increasing the degree of sparseness, the ``noise'' is eliminated and the performance of SVM improves. In many cases, we found that obtaining more sparse basis images (or coefficient) does not necessary lead to a more accurate facial expression classification. These results can be related to the work conducted by Petrov and Li [97]. They investigated local correlation and information redundancy in natural images and they found that the removal of higher-order correlations between the image pixels increased the efficiency of image representation insignificantly. Accordingly, their results suggest that the reduction

of higher-order redundancies than the second-order ones is not the main cause of receptive field properties of neurons in V1.

Although we do not deny the role of sparse image representations in visual cortex, we argue that a more important characteristic of an efficient image representation is feature orientation. Thus, a sparse representation alone does not seem to be sufficient in achieving the maximum recognition performance. This observation comes from [112], where ICA and Gabor filter representation applied to facial expression recognition were compared. Both ICA and Gabor filters approaches gave sparse representations and a highly kurtotic (non-Gaussian) feature distribution. However, the Gabor images that contain important spatially oriented features led to a higher accuracy than the ICA features.

CHAPTER 5

Face Feature Extraction based on NMF approaches

5.1 Face encoding and representation: holistic and sparse features

Two main opposite theories exist with respect to the face encoding and representation in the Human Visual System (HVS). The first one refers to the dense (holistic) representation of the face, where the faces have "holon"-like appearance. The second one claims that a more appropriate human face representation would be given by a sparse code, where only a small fraction of the neural cells corresponding to face encoding is activated. Despite plenty of research work done in order to assess which is the correct paradigm, no consensus was found yet among neuroscientists. Nowadays, the theoretical and experimental evidence suggests that the HVS performs face analysis (encoding, storing, face recognition, facial expression recognition) in a structured and hierarchical way, where both representations have their own contribution and goal. Basically, according to neuropsychological experiments, it is believed that, for face recognition, the encoding relies on the holistic image representation, while, a sparse image representation is preferred when it comes to facial expression analysis and classification. Face and facial expression analysis is not only a concern of the neuropsychology experts. Applications where the human face plays a central role are facial biometrics and facial expression analysis. From the computer vision perspective, the various techniques developed by the computer scientists in order to cope with face and facial expression recognition fall in the same two image representation approaches. In this regard, the findings from neuroscience are well correlated with the nature of image representation provided by the mathematical models of these techniques, i.e. the techniques which were found to perform better for face recognition yield a holistic image representation, contrary to those techniques which are more suitable for facial expression recognition and lead to a sparse or local image representation. The proposed mathematical models of image formation and encoding try to simulate the *efficient storing, organization* and *coding* of data in the human cortex. This is equivalent with embedding constraints in the model design regarding the dimensionality reduction, redundant information minimization, mutual information minimization, non-negativity constraints, class information, etc. While holistic representation treats an image as a whole (global feature), where each pixel has a major contribution to representation, sparse representation is characterized by a highly kurtotic distribution, where a large number of pixels have zero value, and, small number of pixels have positive or negative values (local features).

In its extreme, sparse representation provides a local image representation having only just a few contributing pixels. Image representation is closely related to feature selection. For example, Principal Component Analysis (PCA) models the second order statistics of the data by keeping those eigenvectors that correspond to the largest eigenvalues, while discarding those components that have insignificant contribution for data representation. Human facial image representation based on principal components give us a dense representation and whose basis images have holistic (``ghost'' - like) appearance.

Another image representation approach is based on Independent Component Analysis (ICA) that looks for components that are as independent as possible and produces image features whose properties are related to the ones of V1 receptive fields and have orientation selectivity, bandpass nature and scaling ability. ICA produces either a sparse or a holistic image representation , depending on the architecture used (i.e. the independence is either assumed over images or pixels). This approach has been successfully applied to recognize facial actions by Donato et al. [43]. The work of Donato et al. shows that the extraction of local features from the entire face space by convolving each image with a set of Gabor filters having different frequencies and orientations can outperform other methods that invoke the holistic representation of the face, when it comes to classify facial actions. They achieved the best recognition results by using ICA and Gabor filters. However, they also found that other local spatial approaches, like local PCA and PCA jets provide worse accuracy than, for example, Fisher Linear Discriminant (FLD), which is a holistic approach.

A relatively new approach for feature extraction is provided by the Non-negative matrix factorization (NMF) which decomposes a given data set into two nonnegative more or less sparse factors. The rationale of retrieving nonnegative factors is motivated by at least two reasons. One is the biological fact that the firing rates in visual perception neurons are non-negative. The other reason comes from the image processing field, where the pixels in a grayscale image have nonnegative values. NMF has been already applied on a variety of applications, such as image classification [98], chemometry [99], sound recognition [100], musical audio separation [101] or extraction of summary excerpts from audio and video [102], air emission quality studies [103], identification of object materials from spectral reflectance data at different optical wavelengths [104], or text mining [105]. A particular image processing task where NMF has been used is face recognition [106]. An comprehensive survey of NMF methods and their most important applications can be found in Buciu et al. [3].

The next Sections describe four non-negative matrix factorization algorithms for extracting features further used for facial expression classification.

5.2 Non-negative matrix factorization (NMF)

Non-negative matrix factorization (NMF) has been proposed by Lee and Seung [107] as a method that decomposes a given $m \times n$ non-negative matrix $\mathbf{X}$ into non-negative factors $\mathbf{W}$ and $\mathbf{A}$ such as $\mathbf{X} \approx \mathbf{WA}$, where $\mathbf{W}$ and $\mathbf{A}$ are matrices of size $m \times p$ and $p \times n$, respectively [107]. Suppose that $i = 1, \ldots, m$, $j = 1, \ldots, n$ and $k = 1, \ldots, p$. Then, each element x_{ij} of the matrix $\mathbf{X}$ can be written as $x_{ij} \approx \sum_k w_{ik} a_{kj}$. The quality of approximation depends on the cost function used. Two cost functions were proposed by Lee and Seung in [108]: the Euclidean distance between $\mathbf{X}$ and $\mathbf{WA}$ and Kullback-

Leibler (KL) divergence . In this case, KL has the following expression:

$$D_{NMF}(\mathbf{X} \| \mathbf{WA}) \triangleq \sum_{i,j} \left(x_{ij} \ln \frac{x_{ij}}{\sum_k w_{ik} a_{kj}} + \sum_w z_{ik} a_{kj} - x_{ij} \right), \tag{5.1}$$

This expression can be minimized by applying multiplicative update rules subject to $\mathbf{W}, \mathbf{A} \geq 0$. The positivity constraints arise in many real image processing applications. For example, the pixels in a grayscale image have non-negative intensities. In the NMF approach, its proposers find appropriate to impose non-negative constraints, partly motivated by the biological aspect that the firing rates of neurons are non-negative. Since both matrices **W** and **A** are unknown, we need an algorithm which is able to find these matrices by minimizing the divergence (5.1). By using an auxiliary function and the Expectation Maximization (EM) algorithm [109] , the following update rule for computing h_{kj} is found to minimize the KL divergence at each iteration t [108]:

$$a_{kj}^t = a_{kj}^{t-1} \frac{\sum_i w_{ki} \frac{x_{ij}}{\sum_k w_{ik} a_{kj}^{t-1}}}{\sum_i w_{ik}}. \tag{5.2}$$

By reversing the roles of **W** and **A** in (5.2), a similar update rule for each element w_{ik} of **W** is obtained:

$$w_{ik}^t = w_{ik}^{t-1} \frac{\sum_j \frac{x_{ij}}{\sum_k w_{ik}^{t-1} a_{kj}} h_{jk}}{\sum_j a_{kj}}. \tag{5.3}$$

Both updating rules are applied alternatively in an EM manner and they guarantee a nonincreasing behavior of the KL divergence.

It has been shown that, if the matrix **X** contains images from an image database one in each matrix column, then the method decomposes them into basis images (columns of **W**) and the corresponding coefficients (or hidden components) (rows of **A**) [107]. The resulting basis images contain parts of the original images, parts that are learned thorough the iterative process in the attempt of approximating **X** by the product **WA**. In this context, m represents the number of pixels in the image, n is the total number of images and p is the number of the subspaces in which basis images lay.

5.3 Local non-negative matrix factorization (LNMF)

Local non-negative matrix factorization has been developed by Li et al [106]. This technique is a version of NMF which imposes more constraints on the cost function that are related to spatial localization. Therefore, the localization of the learned image features is improved. If we use the notations $[\mathbf{u}_{ij}] = \mathbf{U} = \mathbf{W}^T\mathbf{W}$ and $[\mathbf{v}_{ij}] = \mathbf{V} = \mathbf{A}\mathbf{A}^T$, the following three additional constraints can be imposed on the NMF basis images and decomposition coefficients:

1. $\sum_i u_{ii} \longrightarrow$ min. This guarantees the generation of more localized features on the basis images **W**, than those resulting from NMF, since we impose the constraint that basis image elements are as small as possible.

2. $\sum_{i \neq j} u_{ij} \longrightarrow$ min. This enforces basis orthogonality, in order to minimize the redundancy between image bases. It must be noted that, while LNMF enforces basis orthogonality, NMF does not necessarily do so.

3. $\sum_i v_{ii} \longrightarrow$ max. By means of this constraint, the total ``activity'' on each

retained component (total squared projection coefficients summed over all training images) is maximized.

Therefore, the new cost function takes the form of the following divergence:

$$D_{LNMF}(\mathbf{X}||\mathbf{WA}) \triangleq D_{NMF}(\mathbf{X}||\mathbf{WA}) + \alpha \sum_{ij} u_{ij} - \beta \sum_{i} v_{ii}, \tag{5.4}$$

where α, $\beta > 0$ are constants. A solution for the minimization of relation (5.4) can be found in [106]. Accordingly, if we use the following update rules for image basis and coefficients:

$$a_{kj}^{(t)} = \sqrt{a_{kj}^{(t-1)} \sum_{i} w_{ki}^{(t)} \frac{x_{ij}}{\sum_{k} w_{ik}^{(t)} a_{kj}^{(t-1)}}}. \tag{5.5}$$

$$w_{ik}^{(t)} = \frac{w_{ik}^{(t-1)} \sum_{j} \frac{x_{ij}}{\sum_{k} w_{ik}^{(t-1)} a_{kj}^{(t)}} a_{jk}^{(t)}}{\sum_{j} a_{kj}^{(t)}}. \tag{5.6}$$

$$w_{ik}^{(t)} = \frac{w_{ik}^{(t)}}{\sum_{i} w_{ik}^{(t)}}, \quad \text{for all } k \tag{5.7}$$

the KL divergence is nonincreasing.

5.4 Discriminant non-negative matrix factorization (DNMF)

Let us suppose now that we have $\mathcal{Q}$ distinctive image classes and let n_c be the number of training samples in class $\mathcal{Q}$, $c = 1, \ldots, \mathcal{Q}$. Each image from the image database corresponding to one column of matrix $\mathbf{X}$, belongs to one of these classes. Therefore, each column of the $p \times n$ matrix $\mathbf{A}$ can be expressed as image representation coefficients vector $\mathbf{a}_{cl}$, where $c = 1, \ldots, \mathcal{Q}$ and $l = 1, \ldots, n_c$. The total number of coefficient vectors is $n = \sum_{c=1}^{\mathcal{Q}} n_c$. We denote the mean coefficient vector of class c by $\mu_c = \frac{1}{n_c} \sum_{l=1}^{n_c} \mathbf{a}_{cl}$ and the global mean coefficient vector by $\mu = \frac{1}{n} \sum_{c=1}^{\mathcal{Q}} \sum_{l=1}^{n_c} \mathbf{a}_{cl}$. Both NMF and LNMF consider the database as a whole and treat each image in the same way. There is no class information integrated into the cost function. A novel approach termed *Discriminant Non-negative Matrix Factorization* (DNMF) was developed by Buciu and Pitas [110]. The decomposition coefficients encode the image representation in the same way for each image. Therefore, by modifying the expression for the coefficients in a such a way that the basis images incorporate class characteristics, we obtain a class-dependent image representation. We preserve the same constraints on basis as for LNMF and we only introduce two more constraints on the coefficients:

1. $\mathbf{S}_w = \sum_{c=1}^{\mathcal{Q}} \sum_{l=1}^{n_c} (\mathbf{a}_{cl} - \mu_c)(\mathbf{a}_{cl} - \mu_c)^T \longrightarrow \min$. $\mathbf{S}_w$ represents the within-class scatter matrix and defines the scatter of the coefficient vector samples corresponding to the class around their mean. The dispersion of samples that belong to the same class around their corresponding mean should be as small as possible.

2. $\mathbf{S}_b = \sum_{c=1}^{\mathcal{Q}} (\mu_c - \mu)(\mu_c - \mu)^T \longrightarrow \max$. $\mathbf{S}_b$ denotes the between-class scatter matrix and defines the scatter of the class mean around the global mean μ. Each cluster formed by the samples that belong to the same class must be as far as possible from the other clusters. Therefore, $\mathbf{S}_b$ should be as large as possible.

We modify the divergence by adding these two more constraints. The new cost

function is expressed as:

$$D_{DNMF}(\mathbf{X}||\mathbf{WA}) \triangleq D_{LNMF}(\mathbf{X}||\mathbf{WA}) + \gamma \sum_{c=1}^{\mathcal{Q}} \sum_{l=1}^{n_c} (\mathbf{a}_{cl} - \mu_c)(\mathbf{a}_{cl} - \mu_c)^T - \delta \sum_{c=1}^{\mathcal{Q}} (\mu_c - \mu)(\mu_c - \mu)^T, \tag{5.8}$$

where γ and δ are constants. Since DNMF is based on LNMF formulation according to (5.8), the orthogonality of the basis images is enforced. Following the same EM approach used by NMF and LNMF techniques, we come up with the following update expression for each element a_{kl} of coefficients from class c:

$$a_{kl(c)}^{(t)} = \frac{2\mu_c - 1 + \sqrt{(1-2\mu_c)^2 + 8\xi a_{kl(c)}^{(t-1)} \sum_i w_{ki}^{(t)} \frac{x_{ij}}{\sum_k w_{ik}^{(t)} a_{kl(c)}^{(t-1)}}}}{4\xi}. \tag{5.9}$$

The elements h_{kl} are then concatenated for all $\mathcal{Q}$ classes as:

$$a_{kj}^{(t)} = [a_{kl(1)}^{(t)} \mid a_{kl(2)}^{(t)} \mid \ldots \mid a_{kl(\mathcal{Q})}^{(t)}] \tag{5.10}$$

where `` | " denotes concatenation. The expression (5.11) and (5.12) for updating the image basis remains unchanged from LNMF:

$$w_{ik}^{(t)} = w_{ik}^{(t-1)} \sum_j \frac{x_{ij}}{\sum_k w_{ik}^{(t-1)} a_{kj}^{(t)}} \frac{a_{jk}^{(t)}}{\sum_j a_{kj}^{(t)}} \tag{5.11}$$

$$w_{ik}^{(t)} = \frac{w_{ik}^{(t)}}{\sum_i w_{ik}^{(t)}}, \quad \text{for all } k \tag{5.12}$$

The derivation of (5.9) is given in the Appendix. The DNMF approach is a supervised method that preserves the sparseness of basis images through (5.11), while enhancing the class separability by the minimization of $\mathbf{S}_w$ and the maximization of $\mathbf{S}_b$. Note that this idea is similar with the FLD method. However, the difference is fundamental: whilst FLD preserves the class discriminatory information on the original images, DNMF performs on the decomposition coefficients.

5.5 Facial expression recognition experiment

The DNMF approach has been tested along with PCA, FLD [111], NMF, LNMF, FNMF, ICA, Gabor and SVMs [53] approaches for recognizing the six basic facial expressions namely, anger, disgust, fear, happiness, sadness and surprise from face images from Cohn-Kanade AU-coded facial expression database [92]. The registration of each original image $\mathbf{x}$ was performed by mouse clicking on the eyes, thus retrieving the eyes coordinates, followed by an image shift step for centering the eyes. Furthermore, the images are rotated to horizontally align the face according to eyes. In the next step, the face region is cropped in order to remove the image borders, while keeping the main facial fiducial points (as eyebrows, eyes, nose and chin). Finally, each image of a resulting size 80 × 60 pixels was downsampled to a final size of 40 × 30 pixels for computational purposes (except for the Gabor case). The face image pixels were stored into a m = 1200 - dimensional vector for each image. These vectors form the columns of matrix $\mathbf{X}$ for PCA, FLD, DNMF, LNMF, NMF, FNMF and ICA approaches. In

the case of the Gabor feature method, each 80 × 60 image was convolved with 12 Gabor filters, corresponding to the low frequency range for three frequencies ν = 2,3,4 and four orientations μ = 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3\pi}{4}$. Each resulting image was further downsampled by a factor of 3 to an image of 20 × 15 pixels, which was scanned row-wise to form a final feature vector of dimension 300 for each Gabor filter output. The 12 outputs have been concatenated to form a new longer feature vector of dimension 3600. Hence, in the case of Gabor filter approach, the final matrix $\mathbf{X}$ is of size 3600 × n, where n is the number of facial images [112]. The resulting feature vectors have been stored in the columns of $\mathbf{X}$ and were directly used for classification. We used only the magnitude of Gabor filter output, because it varies slowly with the pixel position, while the phase is very sensitive with respect to position. In the case of the SVM method, there are two approaches that can be taken into account. In the first one, the SVM is applied on the gray level values, i.e. directly on the face images, without extracting any feature. In the second approach, the SVM is applied on the features extracted by the aforementioned image representation methods. We employed here both approaches for SVMs. The sequential minimal optimization technique developed by Platt [93] was used to train SVMs having the original images as input. Since classical SVM theory was intended to solve a two class classification problem, we chose the Decision Directed Acyclic Graph (DDAG) learning architecture proposed by Platt et al. to cope with the multi-class classification [94].

5.5.1 Training procedure

In the classical facial expression classification context, the n face images are split into a training set containing $n_{(tr)}$ images and a disjoint test set containing $n_{(te)}$ ones, with the corresponding matrices denoted by $\mathbf{X}_{(tr)}$ and $\mathbf{X}_{(te)}$, respectively. The training images $\mathbf{X}_{(tr)}$ are used in the expression for updating $\mathbf{Z}$ and $\mathbf{H}$. To form the training set, $n_{(tr)} = 164$ face images were randomly chosen from the Cohn-Kanade derived database, while the remaining $n_{(te)} = 70$ images were used for testing, thus forming the test face image set. Both the training and the test set contains all expressions. This has been checked before we proceeded further to processing. Out of the training images we formed the basis images corresponding to NMF, LNMF, FNMF, DNMF (by executing the algorithms described in this paper) and to ICA (by using the so-called architecture I approach described in [91]). The training procedure was applied eleven times for various numbers of basis images.

5.5.2 NMF feature extraction and image representation

By imposing only non-negativity constraints, the features extracted by NMF have a rather holistic appearance. LNMF greatly improves the sparseness and minimizes redundant information by imposing other constraints. DNMF also minimizes redundant information, but the degree of sparseness is limited by those retrieved features that are crucial for maximizing class separability. Figure 5.1 shows the creation of a sample basis image after a number of iterations. The features are automatically selected according to their discriminative power. For comparison, a number of 25 basis images out of 144 for NMF, LNMF, FNMF and DNMF, respectively, are depicted in Figure 5.7. It can be noticed by visual inspection that the basis images retrieved by DNMF are not as sparse as those extracted by LNMF but are sparser than the basis images found by NMF. The basis images extracted by FNMF are almost as sparse as those corresponding to LNMF. To quantify the degree of sparseness, we measured the normalized

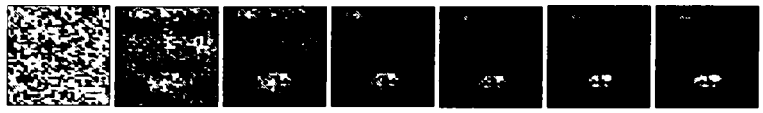


Figure 5.1: Creation of a sample basis image by DNMF algorithm after 0 (random initialization of basis images matrix **Z**), 300, 600, 900, 1200, 1500 and 1800 iterations, respectively.

kurtosis of a base image **w** (one column of **W**) defined as $k(\mathbf{w}) = \frac{\sum_i (w_i - \overline{w})^4}{(\sum_i (w_i - \overline{w})^2)^2} - 3$, where w_i are the elements of **w** (pixels of the basis image) and $\overline{w}$ denotes the sample mean of **w**. It was found experimentally that the average kurtosis over the maximum number of 144 basis images is: $\overline{k}_{NMF} = 7.51$, $\overline{k}_{LNMF} = 152.89$, $\overline{k}_{FNMF} = 151.46$, $\overline{k}_{DNMF} = 22.57$. Therefore, in terms of basis image sparseness, DNMF is a compromise between NMF and LNMF. We have noticed in our experiments that the degree of sparseness corresponding to basis images extracted by DNMF did not increase after a number of iterations. We believe this is caused by those patterns in the basis images that encode meaningful class information and they cannot be disregarded as the iterations proceed further. Probably, the most important issue concerning DNMF algorithm is the fact that almost all features found by its basis images are represented by the salient face features such as eyes, eyebrows or mouth, features that are of great relevance for facial expressions. While discarding less important information, conveyed by nose and cheek (which is not the case for NMF), or putting less stress on it, DNMF preserves spatial topology of salient features (which are mostly absent in the case of LNMF or FNMF) by emphasizing them. The features retrieved by LNMF and FNMF have rather random positions.

For PCA, FLD, NMF, LNMF, FNMF and DNMF, the image data are then projected onto the image basis in an approach similar to the one used in classical PCA, yielding a feature vector $\mathbf{F}_{(tr)} = \mathbf{W}^T(\mathbf{X}_{(tr)} - \Psi)$, where Ψ is a matrix whose columns store the average face $\Psi = \frac{1}{n_{(tr)}} \sum_{j=1}^{n_{(tr)}} \mathbf{X}_{j(tr)}$. Since $\mathbf{X}_{(tr)} = \mathbf{ZA}$, a more natural way to compute $\mathbf{F}_{(tr)}$ would be $\mathbf{F}_{(tr)} = \mathbf{W}^{-1}(\mathbf{X}_{(tr)} - \Psi)$. However, in our previous experiments we found that, by projecting the face images into the basis images instead of working directly with the coefficients $\mathbf{F}_{(tr)}$ given by the above expression, we can have slightly better results. Moreover, due to the fact that **Z** is not a square matrix, we would be forced to use its pseudoinverse, which may suffer from numerical instability. In any case, we can not use the coefficient matrix **H** computed directly by (5.9) in the training phase, since we do not have any expression for calculating a representation of test images. Let us enumerate the six facial expressions so that ``1'' is anger, ``2'' is disgust, ``3'' is fear, ``4'' is happiness, ``5'' is sadness and ``6'' is surprise. To have a first visualization on how efficient is to project the facial images onto the basis images, Figure 5.3 displays the projection of images coming from three expression classes (anger, disgust, surprise) on the first two basis images shown in Figure 5.7. Let us denote by $M1$, $M2$ and $M6$ the mean of the three clusters formed by these projections and the distance between the means by d_{12}, d_{16} and d_{26}, respectively. Then, for this metric space we have $d_{12} = 5.9$, $d_{16} = 6.1$, and $d_{26} = 3.4$ in the case of NMF, $d_{12} = 9$, $d_{16} = 21.8$ and $d_{26} = 20.4$ for LNMF, $d_{12} = 11.1$, $d_{16} = 26.2$ and $d_{26} = 21.7$ for FNMF and $d_{12} = 12.5$, $d_{16} = 27.9$ and $d_{26} = 28$ for DNMF approaches, respectively. For simplicity, Figure 5.3 shows only M2 and M6. The distance between them is depicted by a line segment. It can be noted that the classes do not overlap in the case of DNMF as

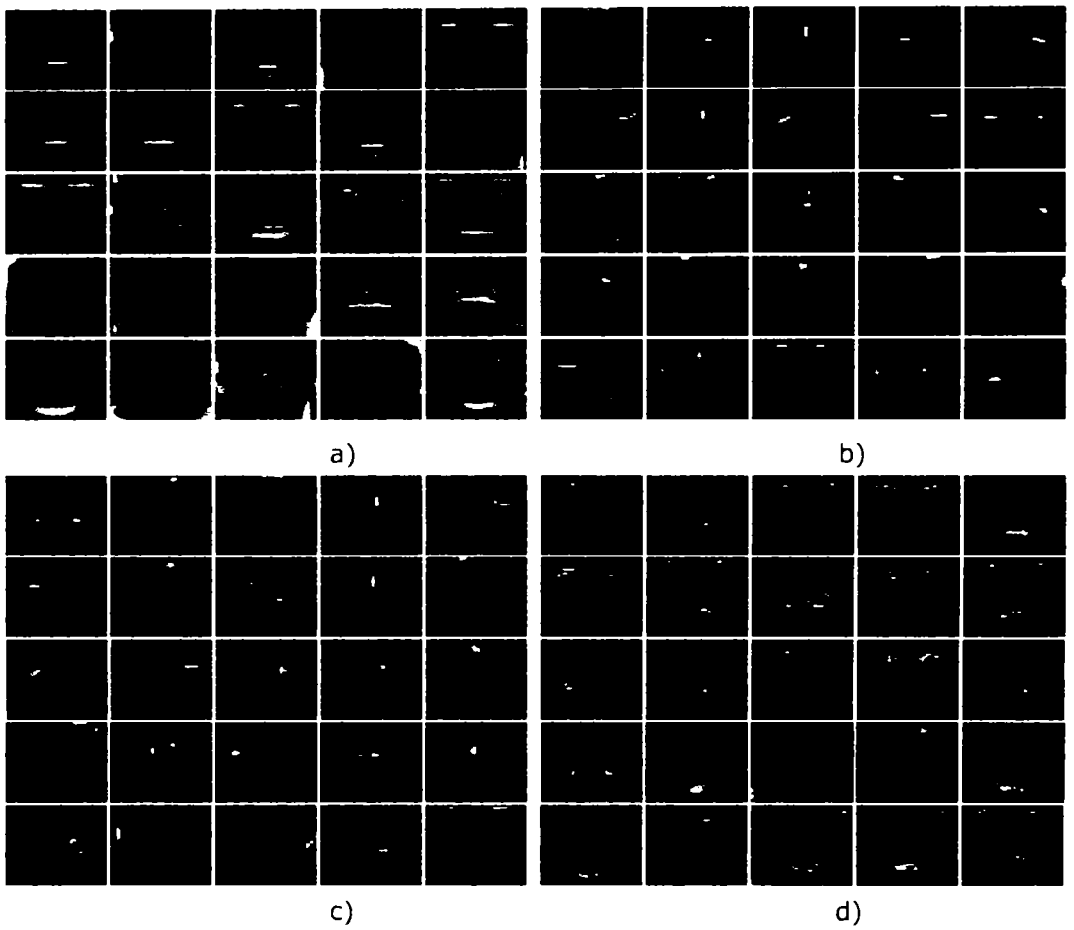


Figure 5.2: A set of 25 basis images out of 144 for a) NMF, b) LNMF, c) FNMF and d) DNMF. They were ordered according to their decreasing degree of sparseness.

much as they do in the case of NMF, LNMF or FNMF methods. The distance between the means corresponding to the four NMF derived algorithms for all expressions are tabulated in Table 5.1. For all expressions, the best between-class separability is obtained by DNMF, followed by FNMF, LNMF and NMF.

For the ICA approach, we used the first architecture described in [91] that gives us the coefficients to be applied for classification. The coefficients of each image form essentially a row of the matrix $\mathbf{F}_{(tr)} = (\mathbf{X}_{(tr)} - \Psi)\mathbf{P}_p\mathbf{A}_{un}^{-1}$. Here $\mathbf{P}_p$ is the projection matrix resulting from PCA procedure applied a priori to ICA and $\mathbf{A}_{un}$ is the unmixing matrix found by ICA algorithm. The number of independent components is controlled by the first p eigenvectors [91]. Note that the training phase is related to the process of finding $\mathbf{W}$, $\mathbf{A}$ and $\mathbf{A}_{un}$, in order to form the new feature vector, which is further used in the classification procedure. In the case of the Gabor approach, there is no training step and the feature vectors $\mathbf{f}_{(tr)}$ used for classification comprise in the columns of $\mathbf{X}$,

5.5.3 Test procedure

In the test phase, for PCA, FLD, DNMF, LNMF, FNMF and NMF, for each test face image $\mathbf{x}_{(te)}$, a test feature vector $\mathbf{f}_{(te)}$ is then formed by $\mathbf{f}_{(te)} = \mathbf{W}^T(\mathbf{x}_{(te)} - \psi)$. For the ICA

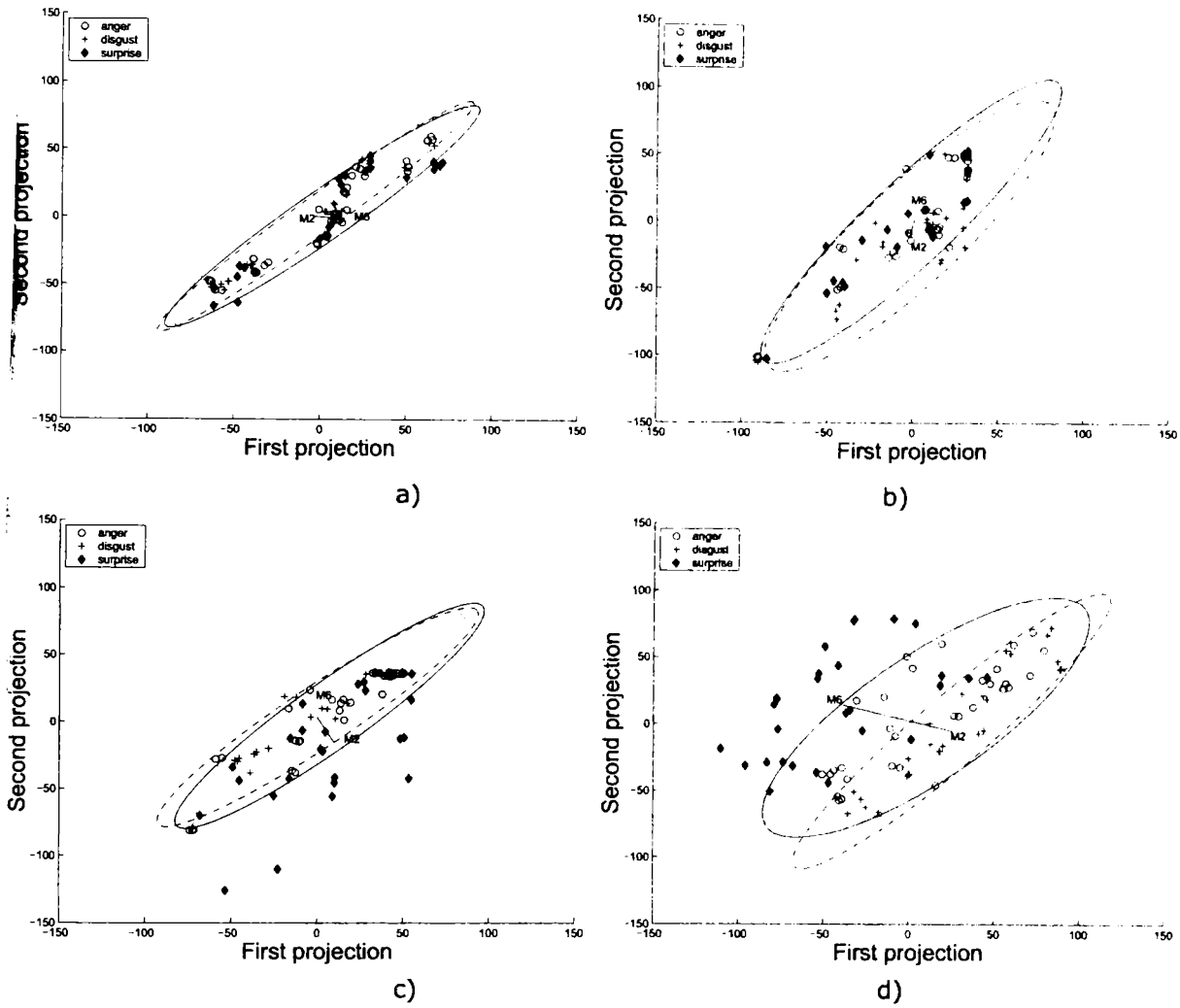


Figure 5.3: Scatter plot of the clusters formed by the projection of three expression classes (anger, disgust, surprise) on the first two basis images shown in Figure 5.7 for a) NMF, b) LNMF, c) FNMF, and d) DNMF. $M2$ and $M6$ represent the mean of the clusters corresponding to ``disgust'' and ``surprise'' classes and the distance between they is depicted by a line segment. The ellipse encompasses the distribution with a confidence factor of 90 %.

approach, the test feature vector is formed by $\mathbf{f}_{(te)} = (\mathbf{x}_{(te)} - \psi)\mathbf{P}_p\mathbf{A}_{un}^{-1}$. In the case of Gabor approach, the classification procedure is applied directly to the columns of matrix $\mathbf{X}_{(te)}$ that contain Gabor features, obtained as described previously.

5.5.4 Classification procedure

The six basic facial expressions i. e. anger (*an*), disgust (*di*), fear (*fe*), happiness (*ha*), sadness (*sa*) and surprise (*su*), available for the facial image database form the six expression classes. If we construct a classifier whose class label output for a test sample $\mathbf{f}_{(te)}$ is $\widetilde{l}$, the classifier accuracy is defined as the percentage of the correctly classified test images when $\{\widetilde{l}(\mathbf{f}_{(te)}) = l(\mathbf{f}_{(te)})\}$, where $l(\mathbf{f}_{(te)})$ is the correct class label. Once we have formed $c = 6$ classes of new feature vectors (or prototype samples), a

Table 5.1: Distance between the means of the database projection onto the first two basis images corresponding to the four NMF derived algorithms for all six facial expressions.

	NMF	LNMF	FNMF	DNMF
d_{12}	5.9	9	11.1	12.5
d_{13}	8.9	9.2	16.0	17.4
d_{14}	9.5	14.0	30.3	40.8
d_{15}	5.9	10.3	16.9	18.6
d_{16}	6.1	21.8	26.2	27.9
d_{23}	6	7.6	9.5	25.6
d_{24}	6.8	12.3	23.3	46.5
d_{25}	3.6	8.8	10.9	26.1
d_{26}	3.4	20.4	21.7	28
d_{34}	9.5	12.9	29.4	53.6
d_{35}	5.9	8.9	15.7	34
d_{36}	6.3	20.2	24.4	35.2
d_{45}	6.8	13.6	29.9	57
d_{46}	6.9	24.6	37.2	62
d_{56}	3.4	21.7	26	38.6

nearest neighbor classifier is employed to classify the new test sample by using the following similarity measures:

1. *Cosine similarity measure* (CSM). The description of this distance metric was given in Chapter 4.

2. *Maximum correlation classifier* (MCC). The second classifier is a minimum Euclidean distance classifier. The Euclidean distance from $\mathbf{f}_{(te)}$ to $\mathbf{f}_{l(tr)}$ is expressed as $\|\mathbf{f}_{(te)} - \mathbf{f}_{l(tr)}\|^2 = -2h_l(\mathbf{f}_{(te)}) + (\mathbf{f}_{(te)})^T\mathbf{f}_{(te)}$, where $h_l(\mathbf{f}_{(te)}) = (\mathbf{f}_{l(tr)})^T\mathbf{f}_{(te)} - \frac{1}{2}\|\mathbf{f}_{l(tr)}\|^2$ is a linear discriminant function of $\mathbf{f}_{(te)}$. A test image is classified by this classifier by computing c linear discriminant functions and choosing $MCC = \text{argmax}_{l=1,\ldots,c}\{h_l(\mathbf{f}_{(te)})\}$.

Besides, as already mentioned, SVMs were used as classifiers where, either the original gray level values or the features extracted by the presented algorithms are considered as input.

5.5.5 Performance evaluation and discussions

We have tested the algorithms for several numbers of basis images (subspaces) and for all three classifiers. The results are shown in Figure 5.4 and Figure 5.5.

Unfortunately, the accuracy does not increase monotonically with the number of basis images (for any of the methods and classifiers). Table 5.2 depicts the maximum, mean classification accuracy and its standard deviation over the number of basis images for all methods involved in experiment and for the three classifiers (CSM, MCC and SVM). In this Table 5.2, SVM1 SVM2 denote the Support Vector Machine applied to the features extracted by the image representation methods involved in the experiment or to the downsampled original gray level images, respectively. For SVMs, the best accuracy was obtained with a polynomial kernel having degree 3 and setting up the penalizing term to 100 in the case of PCA, PDA, Gabor and LNMF image representations. When NMF and DNMF are combined with SVMs, the best accuracy is

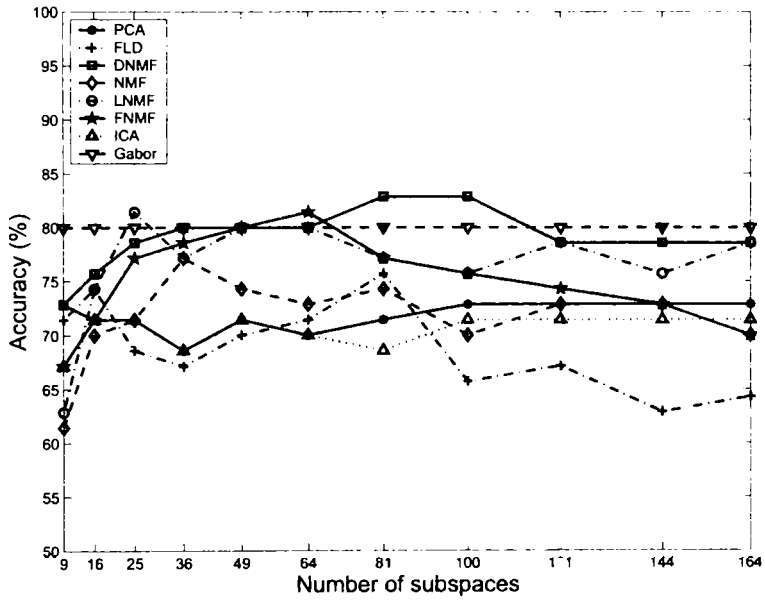


Figure 5.4: Accuracy achieved in the case of CSM classifier for DNMF, NMF, LNMF, FNMF, ICA and Gabor methods versus number of basis images (subspaces).

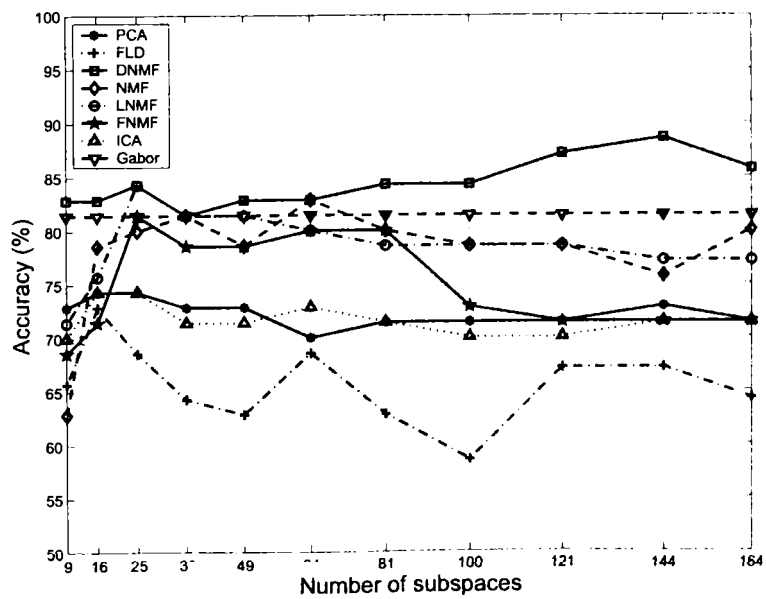


Figure 5.5: Accuracy achieved in the case of MCC classifier for DNMF, NMF, LNMF, FNMF, ICA and Gabor methods versus number of basis images (subspaces).

Table 5.2: Maximum, mean and standard deviation of the classification accuracy (%) calculated over the number of basis images.

		PCA	FLD	DNMF	NMF	LNMF	FNMF	ICA	Gabor
CSM	max	72.85	75.71	82.85	77.14	81.42	81.42	71.42	80
	mean	71.68	68.96	79.04	71.58	76.34	75.06	70.15	x
	std	1.40	4.04	3.19	4.42	5.53	4.39	1.66	x
MCC	max	74.28	72.85	**88.57**	82.85	84.28	81.42	74.28	81.40
	mean	72.33	65.71	83.65	77.93	78.88	75.06	71.74	x
	std	1.32	3.77	1.61	5.85	3.69	4.62	1.71	x
SVM1	max	81.42	84.28	87.14	78.57	81.42	84.28	80.00	82.85
	mean	78.57	78.5	83.71	71.96	65	68.28	78.09	x
	std	3.84	6.55	5.18	5.06	24.55	22.11	1.74	x
SVM2	max	x	x	x	x	x	x	x	x
FLD	max	x	x	x	87.14	84.85	x	75.71	85.71

provided by an RBF kernel having parameter value $\sigma = 0.00005$ (width of RBF) and penalizing term value 500. Except for DNMF, the classifier accuracy is better for SVM1 than for CSM, MCC and SVM2. However, none of the image representations combined with the three classifiers reaches the maximum accuracy 88.57 % achieved by DNMF combined with the MCC classifier. The maximum classification accuracy obtained by DNMF is followed by DNMF plus SVM1 and LNMF plus SVM1, respectively. NMF extracts ``noisy'' features that can decrease its classification accuracy. Features that might be crucial for facial expression recognition are lost by LNMF in its attempt to obtain a local image representation. DNMF balances between NMF and LNMF. Despite the fact that FNMF is based on the same discriminant criteria as DNMF, the accuracy corresponding to this algorithm is comparable with the one yielded by LNMF but lower than the one obtained by DNMF. When combined with SVM, FNMF outperforms LNMF, but its performance does not reach the maximum accuracy corresponding to DNMF. For this data set the poorest performance is achieved by FLD. This can be caused either due to an insufficient data size or to the highly non linear class separability.

Moreover, DNMF algorithm has larger mean accuracy and smaller standard deviation than NMF and LNMF for CSM, as can be seen in Table 5.2. The DNMF mean accuracy is greatly improved when MCC is applied, achieving the biggest average classification accuracy (83.65 %) and the smallest standard deviation (1.61 %).

To establish to what degree the performance benefit is due to adding class-specific information to NMF or LNMF and to what degree it is due to putting this information directly in the feature LNMF learning stage (as DNMF does), we performed FLD on top of either NMF (FLD+NMF) or LNMF (FLD+LNMF), respectively. Also, this approach has been used for ICA (FLD+ICA) and Gabor representations (FLD+Gabor). The last row of the Table 5.2 shows the comparison results. For all these image representations, the use of FLD on top of them seems to be a good idea, since the results show an increase in the accuracy compared with the case when MCC and CSM were applied directly to those image representations. The biggest gain was obtained for FLD+NMF, where the accuracy increased from 82.85% to 87.14%, a value that is still smaller by 1.43% then the best case (DNMF with 88.57%). As DNMF is built on LNMF by incorporating the discriminant information in the learning process, the comparison of the result of FLD+LNMF with that of LNMF and DNMF is of particular interest. In this case, FLD+LNMF improves the accuracy insignificantly, compared with LNMF (from 84.28% to 84.85%) and did not reach the maximum of 88.57% obtained by DNMF with

MCC.

As far as the ICA approach is concerned, we should mention that a supervised ICA technique, called ICA - FX, has been developed [96] to obtain features that are not only independent from each other, but also convey class information, contrary to the classical ICA, which is an unsupervised approach and does not utilize class information. In order to establish to what extent the classification performance is affected by incorporating class information, we ran the ICA - FX approach and compared it with classical ICA in our experiments. Due to the fact that the first ICA architecture does not allow us to make a comparison against ICA - FX, since this architecture performs ICA on the PCA projection matrix (thus performing ICA on the reduced data size and loosing the class labels), we have chosen to run the experiments for comparison with the second ICA architecture [91]. In this case, ICA operates on the PCA coefficients where the data dimensionality is reduced and the class label is preserved. For 49 basis images we obtained an accuracy of 70% and 71.1% with CSM classifier corresponding to ICA and ICA - FX approach, respectively. When the MCC classifier is involved, we have yielded an accuracy of 61.5% and 72.9% corresponding to ICA and ICA - FX.

The parameter ξ governs the convergence speed for minimizing $\mathbf{S}_w$ while maximizing $\mathbf{S}_b$. However, it also interferes with the expression that minimizes the approximation $\mathbf{X} \approx \mathbf{WA}$, i.e., the term $D_{NMF}(\mathbf{X} \| \mathbf{WA})$. An overly small value of ξ will speed up the decrease of $\mathbf{S}_w$, the increase of $\mathbf{S}_b$ and the minimization of $D_{NMF}(\mathbf{X} \| \mathbf{WA})$). However, the algorithm may stop too early and the number of iterations might not be sufficient to reach a local minimum for $D_{DNMF}(\mathbf{X} \| \mathbf{WA})$. A premature stop can affect the process of correctly learning the basis images that might not be sparse anymore. On the other hand, the algorithm may converge very slowly if an overly large value of ξ is chosen. By keeping γ and δ fixed at value one, experimentally, we have chosen a value of $\xi = 0.5$ in our experiments that gave us a good trade-off between sparseness and convergence speed. Besides, it keeps the value of $D_{NMF}(\mathbf{X} \| \mathbf{WA})$ low.

It is worth noting that DNMF shares some common characteristics with the biological visual models proposed by neuroscience. In the light of the sparse image coding theory, the neural interpretation of this model is that the neural cell performs sparse coding on the visual input, having its receptive fields closely related to the sparse coding basis images while its firing rates are proportional to the representation coefficients. Compared to NMF (holistic) and LNMF or FNMF (local), the sparse representation given by DNMF is preferred, having some advantages over holistic and local representations [113] . A detailed analysis regarding the interpretation of DNMF algorithm in neurophysiology terms will be given in Section 5.7 Another important aspect is related to the nature of features extracted by these methods. Obviously, the human face has some salient features such as eyebrows, eyes, and mouth. DNMF emphasizes salient face features and diminishes other features, as opposite to NMF approach, which puts approximately the same weight on each image pixel. In contrary, the features discovered by LNMF or FNMF have rather random position and they do not always correspond to salient facial image features.

5.6 Polynomial non-negative matrix factorization (PNMF)

5.6.1 The necessity of retrieving nonlinear features

NMF, LNMF and DNMF all are linear models in the sense that an image is decomposed as a linear mixture of basis images. However, as suggested and evidenced by nu-

merous works, the receptive fields exhibit nonlinear behavior [114], [115]. In other words, the response of the visual cells is a nonlinear function of their stimuli, where the response is characterized and analyzed on a low dimensional subspace [116], [117]. On the other hand, it was recently argued and proved that, in order to achieve an efficient perceptual coding system, a nonlinear image representation should be developed [118]. As described in that paper, employing an adaptive nonlinear image representation algorithm results in a reduction of the statistical and the perceptual redundancy amongst representation elements. As far as the pattern recognition (and, in particular, the face and facial expression recognition) task is concerned, the underlying features most useful for class discrimination may lie within the higher order statistical dependencies among input features. For example, Bartlett et al. [91] have demonstrated that the ICA is superior to PCA in human face recognition in that ICA learns the higher-order dependencies in the input besides the correlations. However, whether the facial expression is composed of a set of independent components is not clear yet. The aspects described above bring arguments in favor of developing a nonlinear counterpart of the NMF. Therefore, the aim of a nonlinear NMF variant is twofold: (a) to yield a model compatible with the neurophysiology paradigms (non-negativity constraints and nonlinear image decomposition) and (b) to discover higher-order correlation between image pixels that lead to more powerful (in discriminative terms) latent features.

One way to handle nonlinear correlation can be provided by using kernel theory. Kernel-based subspace methods have been extensively investigated in the literature. Nonlinear methods based on the kernel theory, such as Kernel PCA and Kernel Fisher Linear Discriminant were used for face recognition or denoising purposes and they were found to outperform their linear variants [119]. In [120] kernels are decomposed in order to obtain posterior probabilities for the class membership in a data clustering application. The kernel theory was pushed further and was applied for retrieving independent features from a non-linear mixture of sources. This has led to a kernel-based Independent Component Analysis proposed by Bach and Jordan [90]. Hyperkernels have been introduced in [121], where the kernel is defined on the space of kernels itself, an approach which allows the adaptation of the kernel function instead of its parameters. An efficient way to adapt such hyperkernels using second-order cone programming is described in [122]. Recently, a combination of kernel theory and Fisher Linear Discriminant criterion has been proposed in [123] to extract the most discriminant nonlinear features and select a suitable kernel simultaneously.

A kernel-based NMF approach was developed by Buciu et al. [124] where the discovered features (encompassed by the basis images) posses non-linear dependencies, while the decomposition factors remain non-negative. In the light of the kernel theory, a new formulation of NMF is proposed, where, although the decomposition is still linear, the discovered features have non-linear dependencies. Here, the nonlinearity aspect refers only to the relation between the pixels of basis images. In principal, the original data residing in a given space are firstly transformed by a nonlinear polynomial kernel mapping into a higher dimensional space, the so called reproducing kernel Hilbert space (RKHS) and then a nonnegative decomposition is accomplished in the feature space. The nonlinear mapping enables implicit characterization of the data high-order statistics. By using a polynomial kernel function, the basis image features are higher-order correlated, as we shall demonstrate in subsequent sections. The proposed approach is named Polynomial kernel Non-negative Matrix Factorization (PNMF).

Another important issue appears when the samples from the database are recor-

ded under varying lighting conditions which can cause the linear approach to perform poorly [125]. It is known that when the Lambertian assumption regarding the illumination is violated (i.e., the change in illumination is drastic) the linear subspace methods may fail. Although no systematic experiments were run that involve an in-depth analysis of the PNMF performance in the case of illumination changes, some preliminary results on a database containing samples recorded under varying lighting conditions, where PNMF outperformed other methods can be reported.

5.6.2 Non-negative matrix factorization in polynomial feature space

Before defining a non-negative matrix factorization in a polynomial feature space, we give the following two definitions:

Definition 1: A *kernel* is a function κ that satisfies $\kappa(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$, for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$, where ϕ is a mapping from $\mathcal{X}$ to an (inner product) feature space $\mathcal{F}$, $\phi : \mathbf{x} \longrightarrow \phi(\mathbf{x}) \in \mathcal{F}$ [126].

Here $\langle ., . \rangle$ denotes the inner product.

Definition 2: Given two matrices $\mathbf{X}$ and $\mathbf{Y}$ of dimensions $m \times n$ and $m \times p$, respectively, the *kernel matrix* $\mathbf{K} \in \mathcal{X}^{n \times p}$ has elements $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{y}_j)$ for the data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathcal{X}$, $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_p \in \mathcal{X}$ and some kernel function k.

To have a first idea about the role of the kernel function let us consider an example of a two-dimensional input space $\mathbf{x} = (x_1, x_2) \in \mathcal{X}^{2 \times 1} \subseteq \mathbb{R}^2$ together with the feature map $\phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2) \in \mathcal{F} \subseteq \mathbb{R}^3$ [126]. The space of linear functions in $\mathcal{F}$ would be of the form:

$$g(\mathbf{x}) = \alpha_{11} x_1^2 + \alpha_{22} x_2^2 + \alpha_{12} \sqrt{2} x_1 x_2. \tag{5.13}$$

As one can see, the feature map maps the data from a two dimensional to a three dimensional space in such way that the linear relations in the feature space correspond to quadratic relations in the input space. The use of kernel functions eliminates the need for an explicit definition of the nonlinear mapping Φ, because the data appear in the feature space only as dot products of their mappings:

$$\begin{aligned} \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle &= \langle (x_1^2, x_2^2, \sqrt{2} x_1 x_2), (z_1^2, z_2^2, \sqrt{2} z_1 z_2) \rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= \langle \mathbf{x}, \mathbf{z} \rangle^2. \end{aligned} \tag{5.14}$$

Frequently used kernel functions are the polynomial ones, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$, and the Gaussian ones , $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / (2\sigma^2))$. This paper deals with the polynomial kernels.

Let us assume now that our input data $\mathbf{X} \in \mathcal{X} \subseteq \mathbb{R}^{m \times n}$ are transformed to the higher dimensional space $\mathcal{F} \subseteq \mathbb{R}^{l \times n}$, $l \gg m$. We denote the set of the transformed input data with $\mathbf{F} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \ldots, \phi(\mathbf{x}_n)]$, where the l - dimensional vector $\phi(\mathbf{x}_j) = [\phi(\mathbf{x})_1, \phi(\mathbf{x})_2, \ldots, \phi(\mathbf{x})_s, \ldots, \phi(\mathbf{x})_l]^T \in \mathcal{F}$. We can find a matrix $\mathbf{Y} = [\phi(\mathbf{z}_1), \phi(\mathbf{z}_2), \ldots, \phi(\mathbf{z}_p)]$, $\mathbf{Y} \in \mathcal{F}$, that approximates the transformed data set, such that $p < n$. Therefore, each vector $\phi(\mathbf{x})$ can be written as a linear combination as $\phi(\mathbf{x}) \approx \mathbf{Y}\mathbf{b}$. We introduce the following squared Euclidean distance in the space $\mathcal{F}$ between the mapping of the vector $\mathbf{x}_j$ and its decomposition factors as being our cost function:

$$q_j = \frac{1}{2} \|\phi(\mathbf{x}_j) - \mathbf{Y}\mathbf{b}_j\|^2. \tag{5.15}$$

The aim is now to minimize q_j subject to $b_r, Z_{ir} \geq 0$, and $\sum_{i=1}^{m} Z_{ir} = 1$.

For the polynomial kernels of degree d, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$ the cost function $Q = \|\phi(\mathbf{X}) - \mathbf{YB}\|^2$ is non-increasing under the following updating rules, for each iteration t:

$$\mathbf{B}^{(t)} \Leftarrow \mathbf{B}^{(t-1)} \otimes \mathbf{K}_{zx}^{(t-1)} \oslash (\mathbf{K}_{zz}^{(t-1)} \mathbf{B}^{(t-1)}) \tag{5.16}$$

$$\mathbf{Z}^{(t)} \Leftarrow \mathbf{Z}^{(t-1)} \otimes [(\mathbf{X}\mathbf{K}_{xz}^{'(t-1)}) \oslash (\mathbf{Z}^{(t-1)} \Omega \mathbf{K}_{zz}^{'(t-1)})] \tag{5.17}$$

$$\mathbf{Z}^{t} \Leftarrow \mathbf{Z}^{t} \oslash \mathbf{S} \tag{5.18}$$

where $\mathbf{K}_{zx} := \langle \phi(\mathbf{z}_i), \phi(\mathbf{x}_i) \rangle$ and $\mathbf{K}_{xz} := \langle \phi(\mathbf{x}_i), \phi(\mathbf{z}_i) \rangle$ are kernel matrices of dimensions $p \times n$ and $n \times p$, respectively, containing values of kernel functions of $\mathbf{z}_i \in \mathbf{Z}$ and $\mathbf{x}_i \in \mathbf{X}$, and $\mathbf{K}_{zz} = \langle \phi(\mathbf{z}_i), \phi(\mathbf{z}_o) \rangle$ is a $p \times p$ kernel matrix of any vectors $\mathbf{z}_i, \mathbf{z}_o \in \mathbf{Z}$. Ω is a diagonal matrix whose diagonal elements are $\omega_{rr} = \sum_{j=1}^{n} B_{rj}$, $r = 1, \ldots, p$. The columns of $\mathbf{S}$ are given by $\mathbf{s}_r = \sum_{i=1}^{m} Z_{ir}$, $r = 1, \ldots, p$.

The proof of (5.16) and (5.17) is given in Appendix. The sign `` ' '' denotes the derivative of matrix elements (functions). For the polynomial kernel $k'(\mathbf{x}_i, \mathbf{x}_j) = d \cdot k(\mathbf{x}_i \cdot \mathbf{x}_j)^{d-1}$. Note that, if the non-negativity constraint is not imposed in the decomposition coefficients, then, the coefficients can be computed as (see equation (A-11) in Appendix):

$$\mathbf{B} = (\mathbf{K}_{zz})^{-1} \mathbf{K}_{zx}. \tag{5.19}$$

The choice of the Euclidean distance as a cost function for the non-linear feature space was motivated by the fact that we want to avoid to explicitly express the nonlinear mapping $\phi(\mathbf{x})$ and $\phi(\mathbf{z})$. Indeed, if we expand equation (5.15), taking into account equation (5.14), we have:

$$Q = \mathbf{k}(\mathbf{x}, \mathbf{x}) - 2\mathbf{k}(\mathbf{x}, \mathbf{z}_i)\mathbf{b} + \mathbf{b}^T \mathbf{k}(\mathbf{z}_i, \mathbf{z}_o)\mathbf{b}, \tag{5.20}$$

In other words, the problem can be easily solved by invoking only the kernel function. The polynomial kernel corresponds to an inner product in the space of d - th order monomials of the input space. If $\mathbf{x}$ represents an image, then, we work in the feature space which is spanned by the products of any d pixels. If we would need to work with the mapped value $\phi(\mathbf{x})$, the dimensionality would be, for example, $l = 10^{10}$ for a 16×16 pixels image and $d = 5$. Thus, by using polynomial kernel we can take into account higher-order image statistics without being concerned about the "curse of dimensionality". The PNMF's complexity is $O(mnpd)$ compared to $O(mnp)$ corresponding to NMF. Unfortunately, the updating scheme does not guarantee a global minimum due to its non-convex optimization structure (applied simultaneously to B and Z), but only a local minimum. The local minimum that is reached depends on the initialization, i.e. the initial values of $\mathbf{B}$ and $\mathbf{Z}$, usually chosen randomly. PNMF algorithm suffers from the same optimization drawback as NMF. One way to partially overcome this problem, i.e. prevent the algorithm from getting ``stuck'' in a ``shallow'' local minimum is to run it several times with different initializations.

It should be noted that the way the development of the iterative approach for updating the decomposition factors was carried out does not permit a non-negative decomposition for a RBF kernel. This is due to the negative solution resulting from the derivative associated to the RBF kernel . Other approaches have to be found for allowing a more flexible kernel.

The developed algorithm is closely related to the reduced set methods applied to Support Vector Machines (SVMs) [127], [128] . These approaches were developed in order to increase the speed of the SVMs and to reduce the computational complexity

of kernel expansion by approximating them by using fewer terms without a significant loss in accuracy. The same Euclidean cost function was used, but no non-negativity constraint was imposed on the computation of the reduced set and their coefficients. Also, the input data of the reduced set method comes from the SVM output, which is a decision function depending on the Lagrangian computed by the SVMs optimization procedure and the kernel formed from the training and test data, while, in our case the input is formed only from the non-linear mapping of the original data.

5.6.3 Experimental performance and evaluation setup

To asses the performance of the PNMF method, experiments on face and facial expression recognition were conducted. For comparison purpose NMF and LNMF have been involved in the experiments. Also, ICA [83] and PCA [129], along with their nonlinear variants namely kernel ICA (KICA) [90] and kernel PCA (KPCA), respectively, were used. The same Cohn-Kanade and JAFFE database, respectively, were employed. We ran the PNMF algorithm for various number of basis images p and different values of the polynomial degree $d = \{2,3,4,5,6,7,8,9,10\}$. Also, the same polynomial degree range was used for KPCA and the results presented are the ones that correspond to the degree that gave the best results. Several basis images discovered by PNMF are depicted in Figure 5.6 for the Cohn-Kanade database and for different values of d. The basis images corresponding to $d = 2$ and $d = 3$ are noisy. As the degree increases more pixels are taken into account. This leads to a ``finer'' image representation and an emphasis on the expression. The phenomenon can be easily observed especially in the third basis image of Figure 5.6, where the happiness expression passes through different intensities (from a vague ``smile'' to an intense ``smile'').

Three classifiers were employed for classifying the features extracted by the algorithms. The first classifier is a nearest neighbor classifier based on the cosine similarity measure (CSM). The second classifier is a two layer neural network (RBFNN) based on radial basis functions (RBFs) $g(x) = \exp(-\|\mathbf{f}_i - \mathbf{f}_j\|^2/(2\sigma^2))$, where $\mathbf{f}$ is the feature vector associated to either training or test image. Finally, the third classifier is based on SVMs [53] with different kernels (linear, polynomial, and RBF). The classification results for the facial expression recognition task for different image representations and classifiers (CSM, RBFNN, SVM) involved in the experiment are shown in Table 5.3. The minimum number of basis images p corresponding to the maximum classification accuracy is also tabulated. The results of the six other subspace image representations (NMF, LNMF, PCA, KPCA, ICA and KICA) are also presented. For all three databases and all classifiers, PNMF outperformed all other methods. Generally, for both Cohn-Kanade (C-K) and JAFFE databases, the best results are provided when the features are classified by SVM followed by CSM and RBFNN. As far as the feature extraction algorithm is concerned, in the case of C-K, the best classification performance was achieved by PNMF, while the second best performance was attributed to the LNMF approach. A greater difference in performance between the best (PNMF) and the second best algorithm (NMF and LNMF) was obtained for the JAFFE database, where PNMF outperforms NMF by almost 3 %, in the case of SVM classifier. Both KPCA and KICA have shown superior performance compared to PCA and ICA, respectively. However, they performed worse than PNMF. Interestingly, KPCA and KICA achieved lower accuracy than PCA and ICA when they classified facial expressions from the C-K database. Compared with the Cohn-Kanade database, the JAFFE database leads to lower classification performance, due to the fact that the subjects posing for this database are not as expressive as those in the Cohn-Kanade, making facial expression harder

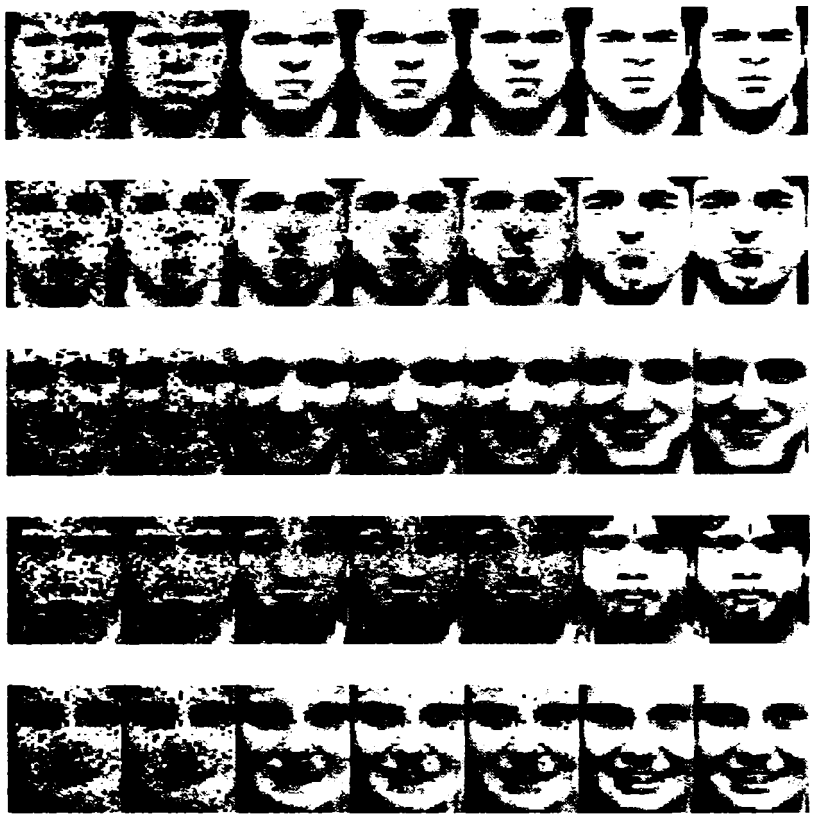


Figure 5.6: Five different basis images retrieved by the PNMF with d = {2,3,4,5,6,7,8} (left to right) for the Cohn-Kanade database.

to be recognized. As experiments showed, the difference between the classification performance of the PNMF (best one) and the second best one is larger in the case of the JAFFE database than in the Cohn-Kanade database. This fact is an indication that the benefit from using PNMF is more prominent in cases where classes are difficult to separate.

It has been argued [43], [130] that, by performing the processing on difference images obtained by subtracting each expression image from its corresponding neutral pose, when available, the classification accuracy is much improved. Thus an experiment involving difference images was conducted. The difference images were formed for the JAFFE database and the new database was denoted by JAFFE$_{diff}$. The same procedure as above was then applied on the new database. Indeed, the accuracy increased for all image representation approaches and all classifiers. An impressive gain was achieved in the case of the PNMF with CSM, where the accuracy increased from 69.8% in JAFFE up to 93.8% in JAFFE$_{diff}$. In terms of classifiers the highest accuracy is obtained by CSM followed by NN and SVM. However, regardless of classifier used, again, PNMF performed better than all other approaches, including KPCA and KICA. A slight accuracy improvement was observed for the KPCA over PCA.

The approach we have developed for updating the basis images and the coefficients relies on iterative minimization. Obviously, other optimization techniques such as, for example, Sequential Quadratic Programming or the interior-reflective Newton method can be used. However, due to the fact that we deal here with a large-scale optimization (taking into account the vector dimension) these approaches can be prohibitive in

Table 5.3: Maximum accuracy (%) obtained for the various methods used in the facial expression classification experiments. The degree of the polynomial PNMF is given in parenthesis. The best result is shown in bold.

Database	Classifier	Maximum accuracy (%)/number of basis images						
		NMF	LNMF	PNMF	ICA	PCA	KICA	KPCA
C-K	CSM	77.4	81.4	**81.8** (d = 6)	71.4	72.9	74.3	72.9
	RBFNN	67.1	77.1	**78.6** (d = 2)	72.9	74.3	72.9	74.3
	SVM	78.6	81.4	**83.9**/100 (d = 2)	80	81.4	82.9	82.9
JAFFE	CSM	66.3	62.4	**69.8** (d = 5)	63.4	61.3	66.7	58
	RBFNN	61.9	60.3	**65** (d = 4)	61.9	60.3	61.9	55
	SVM	74.6	74.6	**77.8** (d = 6)	74.6	74.6	76.2	71.4
JAFFE$_{diff}$	CSM	70	89.3	**93.8** (d = 7)	91	90.1	89.3	91
	RBFNN	76.8	82.1	**87.5** (d = 7)	82.1	85.7	82.1	86
	SVM	78.6	82.1	**83.9** (d = 5)	82.1	82.1	82.1	82.5

terms of computational cost or memory requirements as was shown in the following experiment. Having evaluated the analytical expression for the derivative and the constraints for the cost function, we run the MATLAB [131] routine ``fmincon'' with the large-scale optimization option to tackle our problem and compared it with our iterative solution starting with the same initial random matrices **B** and **Z**. The routine ``fmincon'' for large-scale optimization uses the interior-reflective Newton method with the help of preconditioned conjugate gradients. The initial value of the cost function was found to be $Q_{initial} = 3.4610 \cdot 10^8$. Table 5.4 shows the final value Q_{final} of the cost function and the time necessary to reach the minimum for 9 basis images of 20 $\times$ 15 pixels. The methods provided slightly different values for the final cost function. This is due to the fact that both methods are only able to find local minima and they rely on different minimization procedure. The proposed algorithm which was also implemented in MATLAB, was executed almost 431 times faster than ``fmincon''. We must also notice that we were not able to run ``fmincon'' with images having the dimension of 40 $\times$ 30 pixels and for more than 5 basis images due to the memory limitations.

5.6.4 Conclusions

The underlying idea of the new factorization algorithm, named PNMF, is the usage of the polynomial kernel function, which causes the decomposition to take place in a feature space instead of the input space. The algorithm has been applied on two databases for the facial expression classification task. For comparison purposes six reference feature extraction algorithms (NMF, LNMF, PCA, KPCA, ICA and KICA) have been also used. The features retrieved by the aforementioned approaches have been

Table 5.4: Convergence time (in seconds), initial and final value for the cost function Q for the iterative (PNMF) and ``fmincon'' methods, respectively. The number of basis images is 9 and the dimension of the basis image is 20 $\times$ 15 pixels.

Method	Time ($seconds$)	$Q_{initial}$	Q_{final}
PNMF	50	$3.4610 \cdot 10^8$	$2.3077 \cdot 10^4$
fmincon	21548	$3.4610 \cdot 10^8$	$2.2270 \cdot 10^4$

classified by three classifiers CSM, NN, and SVM, respectively. PNMF outperforms the other approaches for all classifiers. The benefit of the proposed approach is evident in problems where the classes are difficult to separate, as in the case of the JAFFE database. One can state with confidence that PNMF is always better than its linear counterpart, i.e. the NMF algorithm, in terms of retrieving more powerful latent variables for pattern classification, as evidenced by the experimental results.

The way the development of the iterative approach for updating the decomposition factors was carried out in this paper, does not permit a non-negative decomposition for another kernel type, such as, for instance, the RBF kernel. This is due to the negative solution resulting from the derivative associated to the RBF kernel. Other approaches that allow a more flexible kernel have to be found. Using other kernel types could be a potential way to improve the performance of the kernel non-negative matrix factorization approach.

5.7 NMF, LNMF, and DNMF modeling of neural receptive fields

Understanding how the image is processed at each level of the Human Visual System in order to be transformed into this signal and the type of signal encoding at the receptive fields (RFs) of the neural cells is one of the primary concerns of the neuropsychologists and neurophysiologists. Nowadays, the theoretical and experimental evidence suggests that the Human Visual System performs object (including face) recognition processing in a structured and hierarchical approach in which neurons become selective to process progressively more complex features of the image structure [132]. Whereas neurons from visual area 1 (V1) are responsible for processing simple visual forms, such as edges and corners (leading to a very sparse image representation), neurons from the visual area 2 (V2) process a larger visual area representing feature groups. As we further proceed to the visual area 4 (V4) and the inferotemporal cortex (IT), we meet neurons having large receptive fields that respond to high- level object descriptions such as ones describing faces or objects. This is equivalent with a decrease in image representation sparseness. Finally, the IT area of the temporal lobe contains neurons whose receptive fields cover the entire visual space. It also contains specialized neurons (face cells) that are selectively tuned for faces. There is now good evidence that there are dedicated areas in temporal cortical lobe that are responsible to process information about faces [133], [134], [135]. Moreover, it was found that there are neurons (located in TE areas) with responses related to facial identity recognition, while other neurons (located in the superior temporal sulcus) are specialized only to respond to facial expressions [136].

Models of receptive fields of neuronal cells have been proposed by numerous researchers. There are two types of neural cells: simple and complex ones. It has been

shown by Olsahusen and Field [137] that in V1 area the simple cells produces a sparse coding of natural images. Their receptive fields respond differently to visual stimuli having different spatial frequencies, orientations, and directions. Marcelja [138] and Daugman [139] have noticed that the the receptive fields of simple cells can be well described by 2D Gabor functions. The main drawback of Gabor function models is that they have many free parameters to be tuned ``by hand'' in order to tile the joint space or spatial frequency domain to form a complete basis for image representation. Other attempts to model the structure of V1 receptive fields were based on Principal Component Analysis (PCA), which leads to holistic image representation [140], [141] and Independent Component Analysis (ICA) [142].

Although the receptive fields of V1 seem to be well described by the models proposed above, there is no conclusive model for cells of the higher cortical levels, especially for face cells. In this Chapter an analysis of NMF, LNMF and DNMF approaches is undertaken, the original work being included in the work of Buciu and Pitas [143]. The NMF model associates the basis images with the receptive fields of neural cells and the coefficients with their firing rates. By analyzing the tiling properties of these bases we can have an insight of how suitable these algorithms are to resemble biological visual perception systems. In particular we are interested in the representation of facial expression images. A biological plausible model for the facial neurons responsible for biological facial expression recognition is proposed. From the computer vision point of view, an analysis of the parameters of the resulting basis images, such as spatial frequency, frequency orientation, position, length, width, aspect ratio, etc., in analogy to the parameters of the spatial neural receptive fields is carried out. The analysis of the basis images characteristics is motivated by the performance of DNMF algorithm in classifying facial expressions [130]. However, since some constraints are common for these three algorithms, NMF and LNMF are analyzed as well. The results whether these algorithms can model biological facial perception systems.

5.7.1 Receptive fields modeled by NMF, LNMF and DNMF

NMF, LNMF and DNMF are trained on a database consisting of facial expressions derived from Cohn-Kanade AU-coded facial expression database. A subspace of 144 basis images (matrix $\mathbf{W}$)($p = 144$) is considered. Once the basis images are calculated we computed the 144 inverse filters $\mathbf{Z} = \mathbf{W}^{-1}$ (to be called receptive field (RF) masks) corresponding to the basis images for all three algorithms. Twenty five receptive field masks for NMF, LNMF and DNMF are shown in Figure 5.7. As can be seen from the Figure 5.7 a), NMF produces neither oriented nor localized masks. The features discovered by NMF have a larger space coverage than those obtained by LNMF or DNMF, thus capturing redundant information. On the contrary, the LNMF receptive field masks are oriented and localized. Mask domain denotes the mask region where mask coefficients are large (above a certain threshold). Some of them have domain of almost a single pixel. Neurophysiologically, one single pixel representation is similar of having a grandmother cell where a specific image is represented by one neuron (with a very small receptive field size). Furthermore, the features discovered by LNMF have rather random position in the image domain. Receptive field masks produced by DNMF are sparse but contain less localized and oriented domain than LNMF. In addition it contains non-oriented features. Probably the most important issue related to the DNMF RFs masks is the fact that almost all their domain correspond to salient face features such as eyes, eyebrows or mouth that are of great relevance to facial expressions. While discarding less important information (e.g. nose and cheeks, which is not the

case for NMF), DNMF preserves local spatial information of salient facial features (that are almost absent in the case of LNMF). The preservation of the spatial facial topology correlates well with the findings of Tanaka et al. [144] who argued that some face cells require the correct spatial facial feature configuration in order to be activated for facial expression recognition. We have noticed in our experiments that the degree of sparseness corresponding to basis images extracted by DNMF did not increase after a number of iterations. We believe this is caused by those patterns in the basis images that encode meaningful class information (such as those corresponding to salient facial features) and they cannot be disregarded as the iterations proceed further. The degree of RF masks sparseness can be quantified by measuring the normalized kurtosis of a base image. The average kurtosis for the three representations over 144 basis images are: $\bar{k}_{NMF} = 7.51$, $\bar{k}_{LNMF} = 152.89$, $\bar{k}_{DNMF} = 22.57$.

We have described the spatial distribution of the receptive field masks in terms of 4 spatial parameters: average domain location (x, y), domain orientation (0, 90, 45 and 135 degrees, respectively) directions, and aspect ratio. The aspect ratio is defined as l/w, where l and w are the length and width of the receptive fields calculated as follows [145]:

$$l_k \equiv \sqrt{\sum_{x,y}(x\sin(\theta) + y\cos(\theta))^2 \bar{\mathbf{z}}_k^2}$$
$$w_k \equiv \sqrt{\sum_{x,y}(x\cos(\theta) - y\sin(\theta))^2 \bar{\mathbf{z}}_k^2}, \tag{5.21}$$

over (x, y) image space. These RF masks domain parameters calculated over the facial image database are represented in Figure 5.8. We can notice in Figure 5.8a that the RF masks do not cover the entire space. For NMF and DNMF they are centrally distributed and cover the image center which is in par with a similar characteristic of V4 receptive fields. LNMF features are rather distributed marginally as shown in Figure 5.8a. Unlike NMF, where domain orientation is at oblique angles (45 and 135 degree), LNMF emphasizes more horizontal and vertical features. DNMF puts approximately the same emphasis on horizontal and oblique features and slightly less stress on vertical ones. The oblique features are represented due to the chin contour (as it can be seen from Figure 5.7c) where DNMF acts like a local edge detector.

The aspect ratio of NMF ranges from 0.6 to 1.6 with a mean at 1.09 and a standard deviation of 0.19. LNMF aspect ratios range from 2 to 11 with a mean at 1.65 and standard deviation 2.04. DNMF aspect ratios range from 0.5 to 2.2 with mean 1.03 and standard deviation 0.26. The higher average aspect ratio of LNMF indicates that its receptive fields are more elongated horizontally then those of NMF or DNMF.

To characterize the frequency distribution of RF masks we have computed their spatial frequency and orientations from their Discrete Fourier Transform: $F_k(u, v) = \frac{1}{NM}\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\mathbf{z}(x.y)exp[-j2\pi(ux/N + vy/M)]$ where $u = 1, \ldots, N-1$ and $v = 1, \ldots, M-1$ are spatial frequency coordinates in the horizontal and vertical directions, respectively, expressed in cycles/image and N and M are the number of rows and columns in the basis image, respectively. The two dimensional spatial frequency are represented in polar coordinates (r, φ) where r denotes the absolute spatial frequency, φ orientation, $u = r\cos(\varphi)$ and $v = r\sin(\varphi)$. Thus, the optimal spatial frequency (orientation) is defined as the spatial frequency (orientation) of the peak in the amplitude (phase) spectrum.

Figure 5.9 presents the optimal spatial frequency and optimal orientation for NMF,

LNMF and DNMF receptive field masks found by taking the peak of the spectrum. Figures 5.9 a), b), c) indicate that the features are evenly spread in all orientation in the frequency domain for all three representation studied. Regarding radial spectrum distribution, NMF shows peak at a high spatial frequency bands (approximately between 0.7 and 0.9 cycles/image) as shown in Figure 5.9 d). LNMF features are distributed within a lower frequency band (of 0.25 - 0.45 cycles/image) as shown in Figure 5.9 e). A bandpass spectrum shape is shown by DNMF in Figure 5.9 f) The RFs power spectrum covers a larger spatial frequency band at [0.45,0.8] cycles/image, capturing a larger radial spectrum.

NMF, LNMF and DNMF receptive fields show a low, high and bandpass frequency spectrum, respectively. Redundancy reduction is also obtained by suppressing the low spatial frequency in order to whiten the power spectrum of images, therefore this is done by highpass filtering [146]. This is consistent with what LNMF performs through $\sum_{i \neq j} u_{ij} \longrightarrow \min$, and, thus having receptive fields similar to highpass filters (see Figure 5.9a and Figure 5.9d). On the other hand, the high frequency components contain only little power from the image source and, therefore, it is not robust to noise. To avoid this, highpass frequency must be eliminated. The combination of noise and redundancy reduction optimizes the information transfer, resulting a bandpass filtering. However, as noticed in [146], the balance between highpass and lowpass filtering depends on the signal to noise ratio of the input signal, which depends on the ambient light level.

5.7.2 Discussion and conclusion

There are many models proposed for biological facial analysis in the Human Visual System . On one side, the computer scientists try to find reliable methods that give satisfactory results for face or facial expression recognition. On the other side, psychologists and neurophysiologists try to understand how the human face is perceived by the Human Visual System , and develop models based on various experiments. Not surprisingly, some models proposed by the computer scientists, such as PCA, ICA or Gabor image decomposition, have been accepted as biologically plausible, since they share common properties with biological vision models. In this paper, three other models (NMF, LNMF and DNMF) were investigated. Although the main goal of this paper was to analyze their receptive field masks, it is worthwhile to mention common properties and differences between these three methods in order to draw a general conclusion. Table 5.5 summarizes several common and specific characteristics of these models.

Table 5.5: Characteristics of NMF, LNMF and DNMF methods

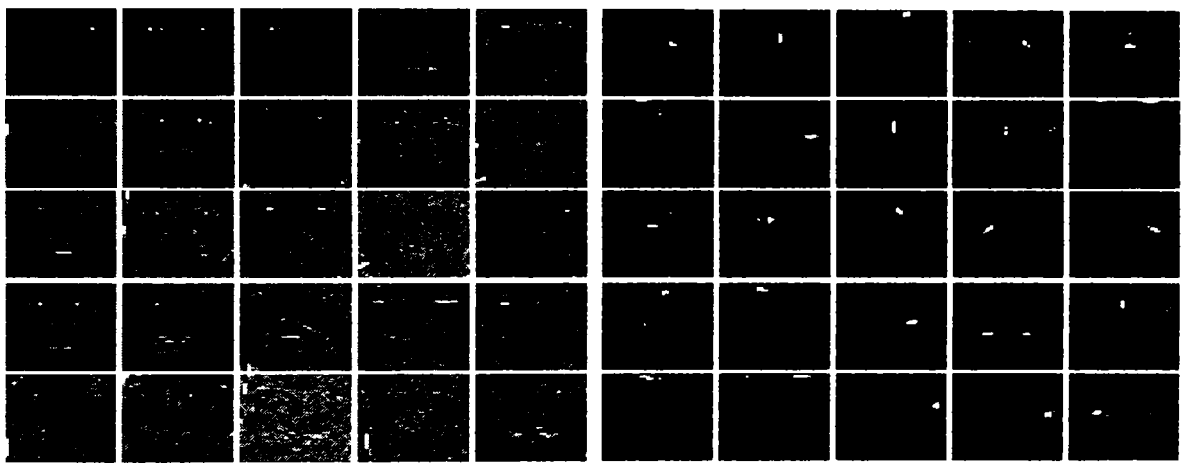
	Decomposition method		
	NMF	LNMF	DNMF
Non-negative constraints	yes	yes	yes
Redundancy reduction	no	yes	yes
Sparseness degree	holistic	local	sparse
Class-dependent learning	no	no	yes
Learning type	unsupervised	unsupervised	supervised
Salient feature extraction	yes	no	yes
Spat. freq. bandwidth	lowpass	highpass	bandpass

The basic principle of efficient information transfer (and hence efficient coding) is to reduce the redundancy of the input signal. It is well known that the natural stimuli (images) contain a large amount of redundant information that loads the dynamic range of the transmission channel without transferring information [147], [137]. Generally, the term efficient coding and information redundancy reduction was associated with finding principal or independent components in representing a set of images. One fundamental difference between the methods mentioned in the Introduction and these three algorithms analyzed in this paper is that neither NMF, LNMF nor DNMF assume features independence. ICA and other methods that rely on this assumption work well when they are applied on natural scenes. Definitely, natural images can contain more independent features than facial images. Here, each image has the same features (eyes, mouth, etc) spatially located in approximately the same position. This might be a reason why ICA performed worse than NMF, LNMF and DNMF when it comes to classify facial expressions [130].

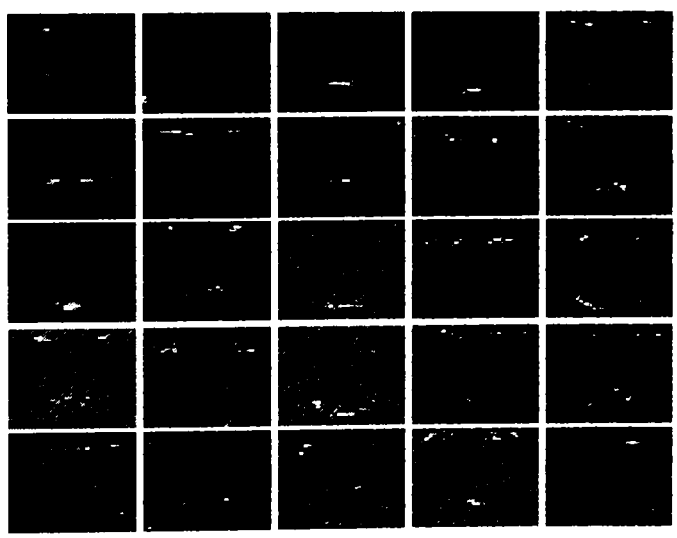
Sparsity is another important issue that comes from neurophysiological field and has several advantages over holistic or local representations [113]. It is argued that the tuning of the neurons in the temporal cortex that respond preferentially to faces represents a trade-off between fully distributed encoding (holistic or global representation, as NMF result) and a grandmother cell type of encoding (local representation, achieved by LNMF) [148]. This trade-off seems to be provided by DNMF representation.

The next three characteristics, namely class-dependent learning, training type and salient feature extraction are closely related to each other. NMF and LNMF are unsupervised approaches while DNMF is supervised one. In a feature extraction framework supervised learning is often necessary to guide feature development. Forcing a class-dependent learning by means of new constraints on coefficients expression, combined with the sparsity constraint on basis images, leads to a DNMF sparse image representation where the salient facial features (emotion-specific patterns that contribute most to expression recognition) are selected from the entire face image while the contribution of irrelevant features is diminished. However, it should be noticed that this class-dependent approach is rather a condition which comes from pattern recognition domain.

As a general conclusion, when comparing these three matrix factorization algorithms with each other, we favor DNMF since it fulfills several requirements: its enhances the class separability (which a pattern recognition issue) compared to the first two approaches, minimizes the redundancy over basis images (similar to efficient coding principle) and leads to a moderate sparse image representation (a neurophysiological issue). We found that, when DNMF is applied to faces, the receptive fields obtained by its basis images are bandpass filters covering the entire frequency orientation domain. Neurophysiology studies must be performed in order to validate the values of the parameters of the DNMF receptive fields.



a) b)



c)

Figure 5.7: Sample receptive field masks corresponding to basis images learned by a) NMF, b) LNMF and c) DNMF. They were ordered according to a decreasing degree of sparseness.

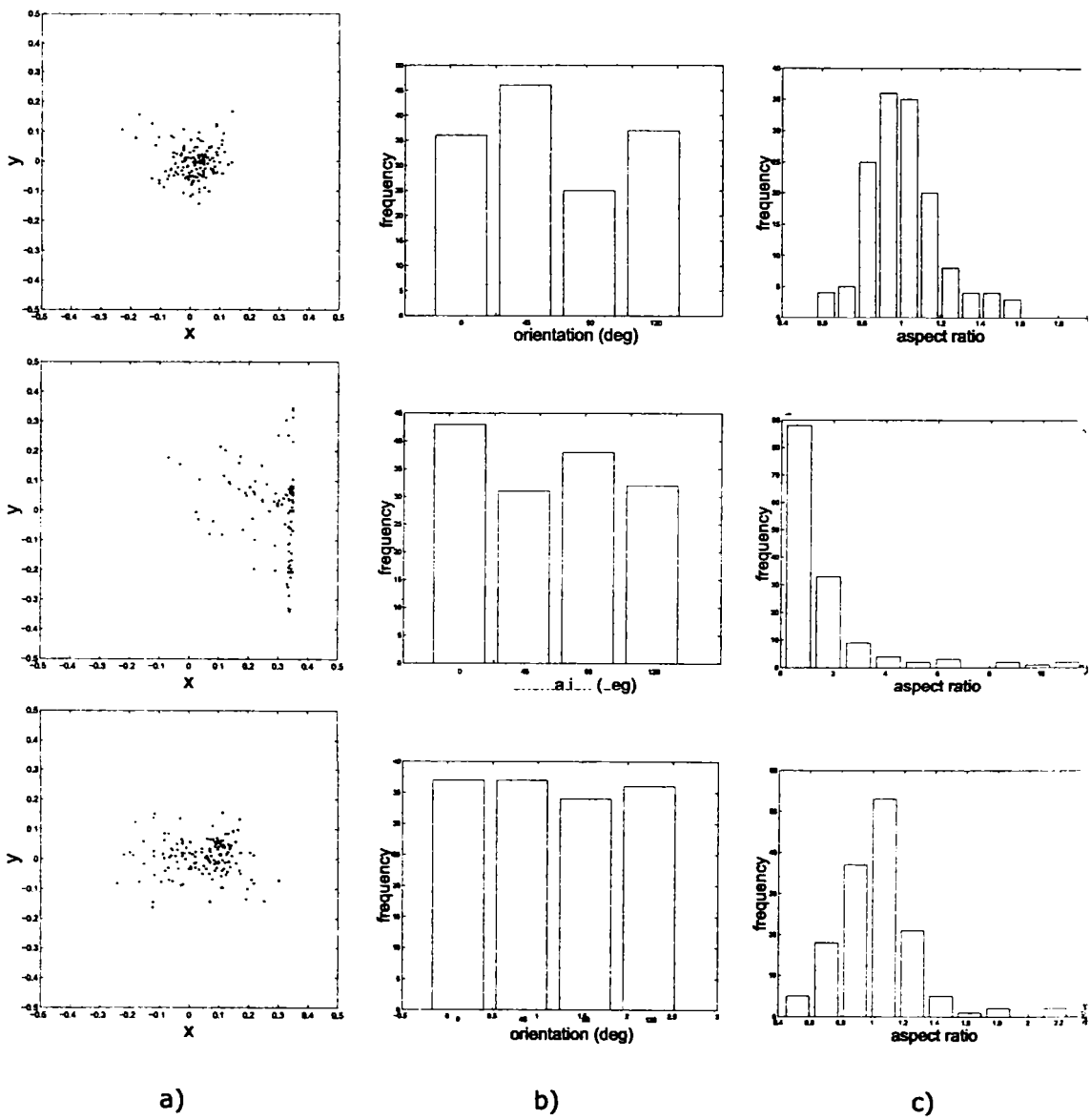


Figure 5.8: Spatial characteristics or FS masks domain for NMF (top), LNMF (middle) and DNMF (bottom) receptive fields (RFs): a) average location of RF domain; b) histogram of RF domain orientations in degrees (0°, 45°, 90°, 135°) and c) length-to-width aspect ratio of RF spatial domain.

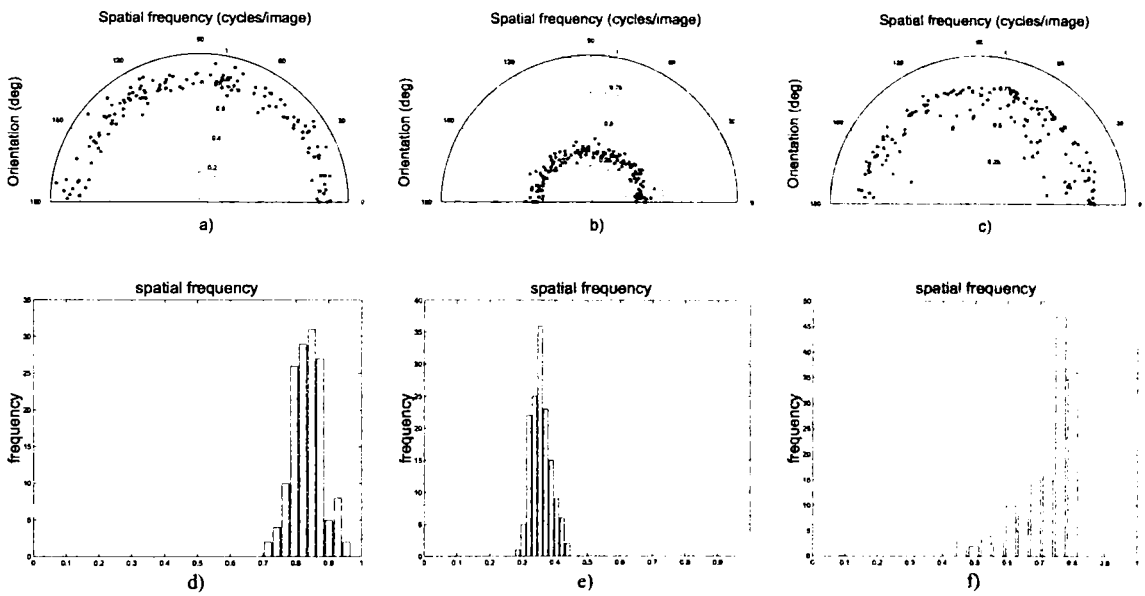


Figure 5.9: The optimal orientation and optimal spatial frequency for RF masks corresponding to (a) NMF, (b) LNMF and (c) DNMF receptive fields. The histogram of the distribution of 144 RFs in the spatial-frequency corresponding to (d) NMF, (e) LNMF and (f) DNMF approaches.

when there is no phase coherence the ratio drops down to zero. Computing the phase congruency quantity is equivalent to search for peaks in the local energy function [156]. For a one-dimensional signal $I(x)$, the local energy is given by:

$$E(x) = \sqrt{F^2(x) + H^2(x)}, \tag{6.2}$$

where $F(x)$ is the Fourier transform of the signal $I(x)$ with its DC component removed, and $H(x)$ represents its Hilbert transform. Further, the local energy can be expressed in terms of the cosine of the deviation of each phase component from the mean, yielding:

$$PC_1 = \frac{\sum_n A_n(cos(\phi(x) - \bar{\phi}(x))}{\sum_n A_n(x)} \tag{6.3}$$

where $\phi(x)$ is the phase component at location x and $\bar{\phi}(x)$ is the amplitude weighted mean local phase angle of all the Fourier terms at location x. Geometrically, the relations between the phase congruency, local energy and the sum of the Fourier amplitude components is illustrated in Figure 6.1. The relation (6.3) does not offer

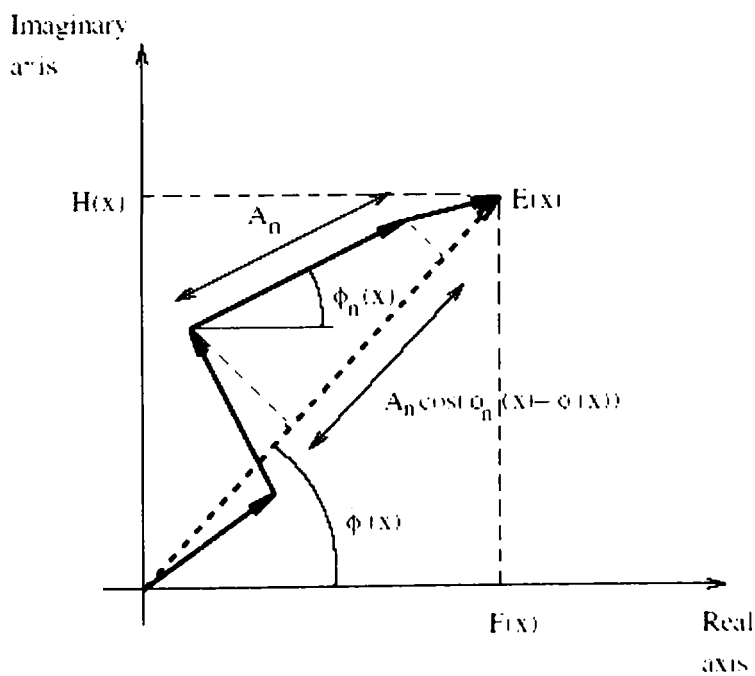


Figure 6.1: The relation between the phase congruency, local energy and the sum of the Fourier amplitude components.

satisfactory localized features. Moreover, it is sensitive to noise. Therefore, Kovesi [152] proposed a modified version for the phase congruency quantity:

$$PC_2 = \frac{\sum_n W(x)\lfloor A_n(cos(\phi(x) - \bar{\phi}(x)) - |sin(\phi(x) - \bar{\phi}(x))| - T)\rfloor}{\sum_n A_n(x) + \varepsilon}. \tag{6.4}$$

The term $W(x)$ is a weight term to moderate the frequency spread. The constant term ε is only introduced to avoid division by zero. T is a noise threshold and represents the

estimated noise influence. The symbol $\lfloor \rfloor$ denotes that the enclosed quantity is equal to itself when its value is positive, and zero otherwise. In practice the local phase information is obtained using banks of Gabor wavelets at different scales. Logarithmic Gabor functions can also be used [152] described by:

$$G(\omega) = exp\left[- \frac{(log(\omega/\omega_0)^2}{2(log(\sigma/\omega_0)^2} \right], \tag{6.5}$$

where ω_0 is the filter's center frequency and σ is a constant. Performing a convolution between the signal $I(x)$ and a pair of quadrature logarithmic Gabor filters, the following responses are obtained at location x:

$$e_n(x) = I(x) * M_n^e \tag{6.6}$$

and

$$o_n(x) = I(x) * M_n^o, \tag{6.7}$$

where n denotes the filter scale, M_n^e and M_n^o represents the even symmetric (cosine) and odd-symmetric (sine) wavelets at a scale n, respectively. Then, the amplitude $A_n(x)$ can be written in terms of filters response as:

$$A_n(x) = \sqrt{e_n(x)^2 + o_n(x)^2} \tag{6.8}$$

while the phase is given by:

$$\phi_n(x) = atan2(e_n(x), o_n(x)) \tag{6.9}$$

We further have $F(x) \simeq \sum_n e_n(x)$, $H(x) \simeq \sum_n o_n(x)$, and $\sum_n A_n(x) \simeq \sum_n \sqrt{e_n(x)^2 + o_n(x)^2}$. The weighted mean phase angle corresponding to each filter is expressed as:

$$\overline{\phi}_e(x) = \frac{F(x)}{\sqrt{F(x)^2 + H(x)^2}} \tag{6.10}$$

$$\overline{\phi}_o(x) = \frac{H(x)}{\sqrt{F(x)^2 + H(x)^2}} \tag{6.11}$$

By replacing the proper quantities and after some computation, the expression $A_n(cos(\phi(x) - \overline{\phi}(x)) - |sin(\phi(x) - \overline{\phi}(x))|)$ of relation (6.4) can be rewritten in terms of filter responses as following:

$$(e_n(x) \cdot \overline{\phi}_e(x) + o_n(x) \cdot \overline{\phi}_o(x)) - |e_n(x) \cdot \overline{\phi}_0(x) - o_n(x) \cdot \overline{\phi}_e(x)| \tag{6.12}$$

6.2 Facial feature extraction

Before applying the phase congruency to an image set let us consider, without loss of generality, the simplest case of two phase-shifted sinusoidal signals I_1 and I_2 depicted in Figure 6.2 a). The phase displacement, also drawn in Figure 6.2 b), represents the disparity used for measuring the features similarity which will be performed through the phase congruency. Each image of dimension $r \times s$ from a set is lexicographically scanned so that it is transformed into a $r \times s = m$ - dimensional column vector representing 1D signal. Each such column vector is stored in the columns of a $m \times n$ matrix

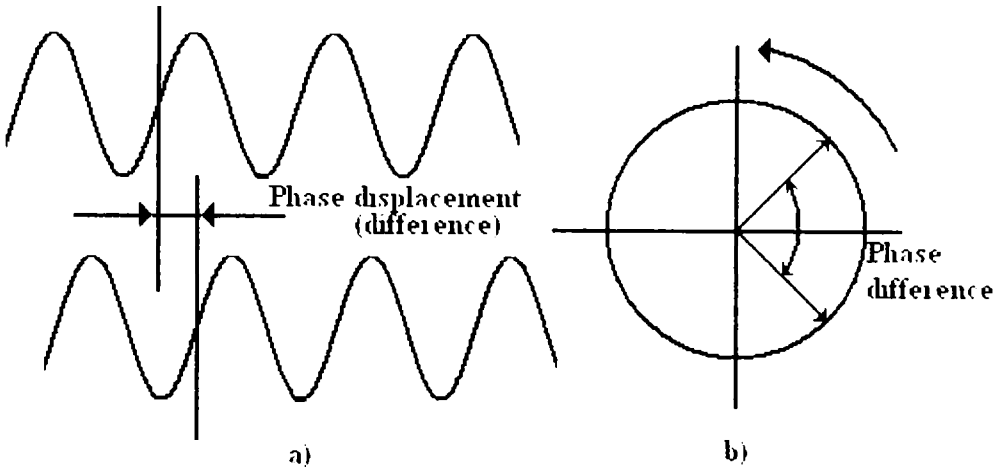


Figure 6.2: a) Two phase-shifted sinusoidal signals; b) Polar coordinates of the phase angle for the two points in the signals.

X with n the number of images. Phase congruency is applied across the columns using the relation (6.4) so that, the dissimilarity between any $x_{1,i}$ and $x_{1,j}$, $x_{2,i}$ and $x_{2,j}$, $x_{3,i}$, and $x_{3,j}$, ... , $x_{k,i}$ and $x_{k,j}$ is measured, for $i, j = 1 \ldots n$, $i \neq j$, and $k = 1 \ldots m$. The procedure is depicted in Figure 6.3. The degree of similarity between set image is thus computed leading to discriminant features incorporating phase information. Applying the phase congruency approach the matrix **X** is transformed into the matrix $\mathbf{X}_{PC_2}$. The resulting phase congruency feature maps are illustrated in Figure 6.4 for samples from the two sets involved in the experiments, namely Cohn-Kanade and JAFFE facial expression database, respectively. Here, each image represents a reshaped column of the matrix $\mathbf{X}_{PC_2}$.

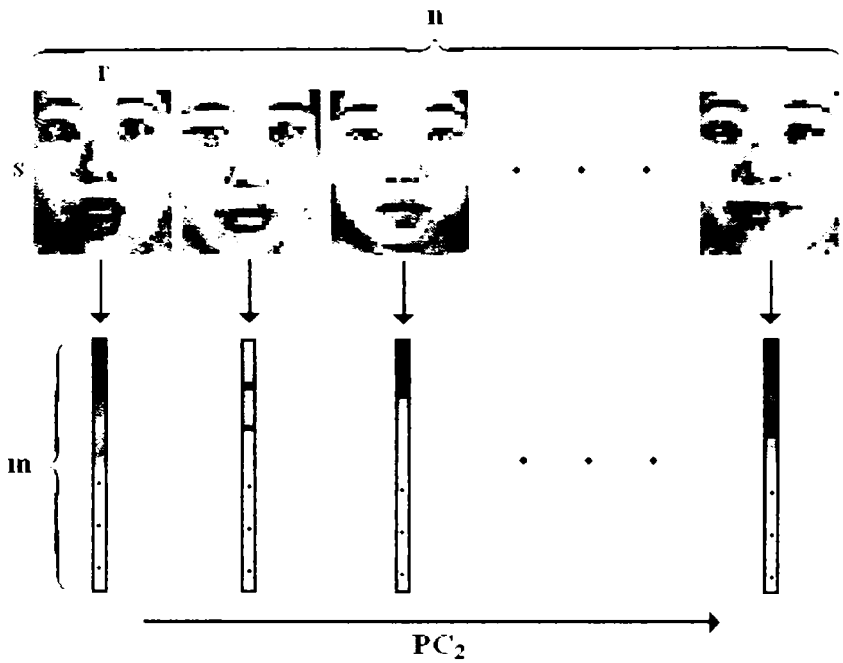


Figure 6.3: Facial features extracted by applying phase congruency approach to the training set from Cohn-Kanade (top row) and JAFFE (bottom row) facial expression database, respectively.

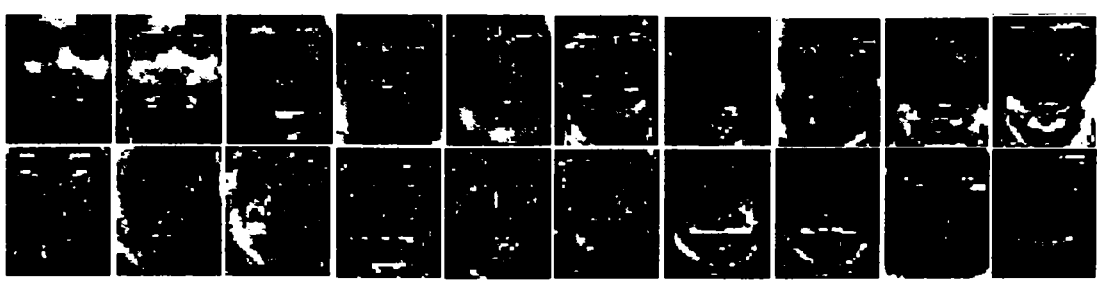


Figure 6.4: Facial features extracted by applying phase congruency approach to the training set from Cohn-Kanade (top row) and JAFFE (bottom row) facial expression database, respectively. Notice how the fiducial facial features that incorporate prominent discriminant phase information are emphasized.

6.3 Performance evaluation and discussions

Both Cohn-Kanade and JAFFE databases were used in the experiments. For the phase congruency approach the original matrix $\mathbf{X}$ is transformed into the feature matrix $\mathbf{X}_{PC_2(tr)}$, procedure described in Section 6.2. To reduce the data dimensionality, PCA is further applied to $\mathbf{X}_{PC_2(tr)} - \Psi$, where Ψ is a matrix whose columns store the average face $\Psi = \frac{1}{n_{(tr)}} \sum_{j=1}^{n_{(tr)}} \mathbf{X}_{j(tr)}$. New reduced feature vectors $\mathbf{f}_{(tr)}$ are then formed comprising the columns of a matrix $\mathbf{F}_{(tr)}$.

The experiments were carried out for four different Gabor wavelets scales ($scale = \{1, 2, 3, 4\}$) and three standard deviations of the Gaussian, $k = \{0.41. 0.55, 0.75\}$, where $k = \sigma/\omega_0$. These values have physical meanings. A $k = 0.75$ corresponds to a filter bandwidth of approximately one octave, $k = 0.55$ results in a two-octave bandwidth, while $k = 0.41$ resembles a three-octave bandwidth. We have chosen these values as the filter bandwidth corresponding to 1 to 3 octaves matches well with measurements obtained on mammalian visual cells [157, 158].

The results corresponding to the Cohn-Kanade database for varying PCs ($PCs = \{5, 10, 20, \ldots, 150\}$) are shown in Figure 6.5. The best results for $scale = 1$ are obtained when $k = 0.41$. Similar performances are noticed for the same k and $scale = 2$, but only for large number of Pcs (> 80). However, the features extracted using a two-octave bandwidth lead to close classification performances. As the filter's $scale$ increases so the accuracy corresponding to $k = 0.75$. The overall maximum performance (80.00 %) is obtained for $scale = 4$, $k = 0.75$, and 90 PCs.

Figure 6.6 depicts the results for the JAFFE database. In this case the classifier follows approximately the same behavior to that corresponding to the other database, for low k, as noted from Figure 6.6 a) and b). The same tendency remains at large $scale$, where better accuracy is achieved for small k, as shown in Figure 6.6 c) and d). The overall maximum accuracy (69.84 %) is yielded for $scale = 4$ and $k = 0.55$.

Finally, the results for PC_2 in comparison with the other approaches are drawn in Figure 6.7. For the Cohn-Kanade database, PC_2 clearly outperforms the other approaches. The second best feature extraction method is provided by the PCA, followed by ICA and LDA. For the JAFFE database a peak of 69.84 in accuracy is obtained for PC_2 with 50 PCs. For more principal components the PC_2 accuracy drops down under 63%. Here, the second best feature extraction method is provided by the LDA method with a maximum accuracy of 63.49.

Table 6.1 summarizes the results for all methods along with the corresponding number of principal components. As one can see, PC_2 conducts to the best facial expression recognition results for both facial expression databases, outperforming the other methods.

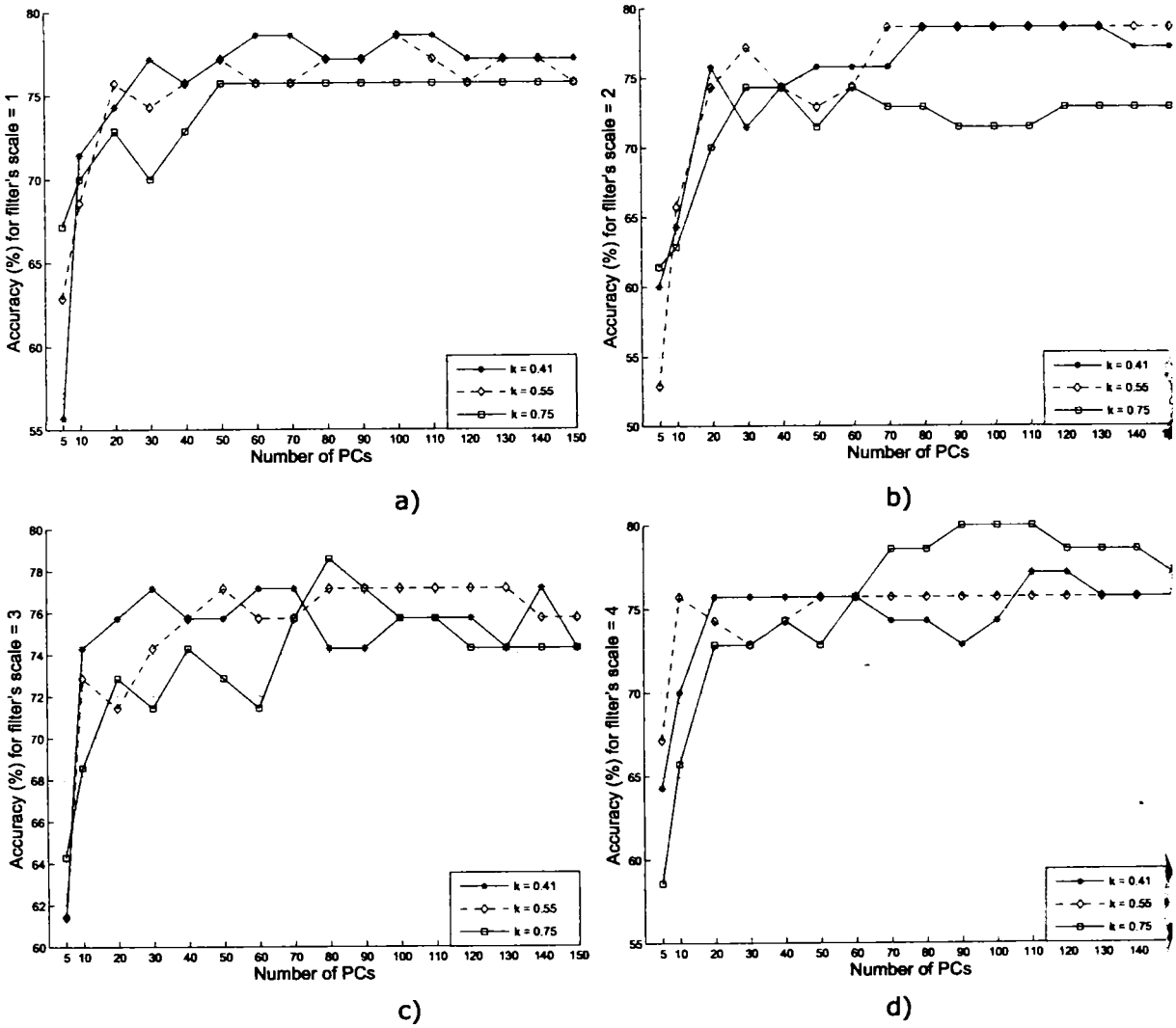


Figure 6.5: Experimental results for PC_2 corresponding to the Cohn-Kanade database for varying number of PCs, k, and $scale$.

Table 6.1: Maximum accuracy (%) for PC_2, LDA, ICA and PCA.

Database	Cohn-Kanade				JAFFE			
Method	PC_2	PCA	ICA	LDA	PC_2	PCA	ICA	LDA
Max. accuracy (%)	**80**	74.29	71.43	72.86	**69.84**	60.32	60.32	63.49
Number of PCs	80	90	50	60	50	10	130	20

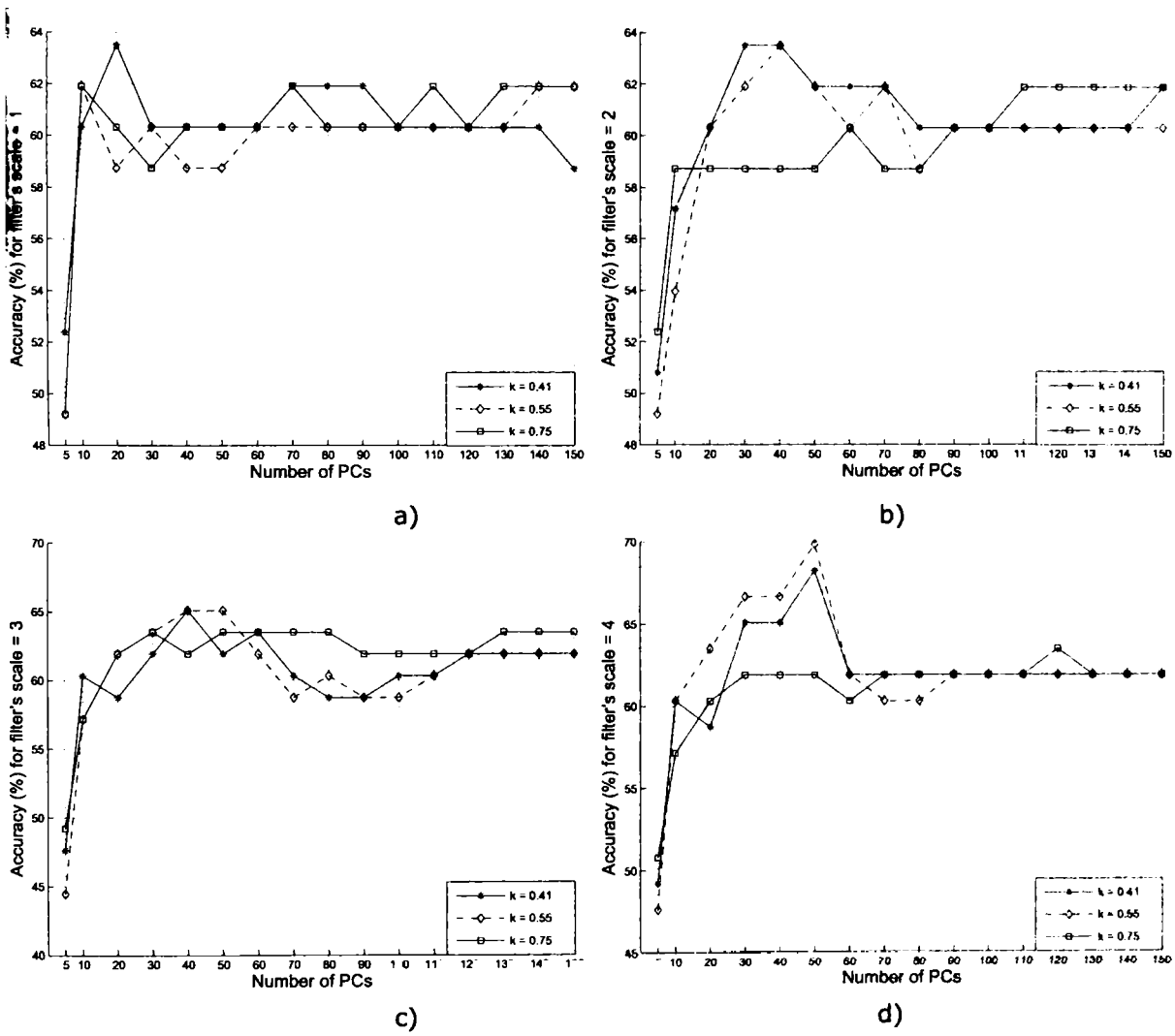
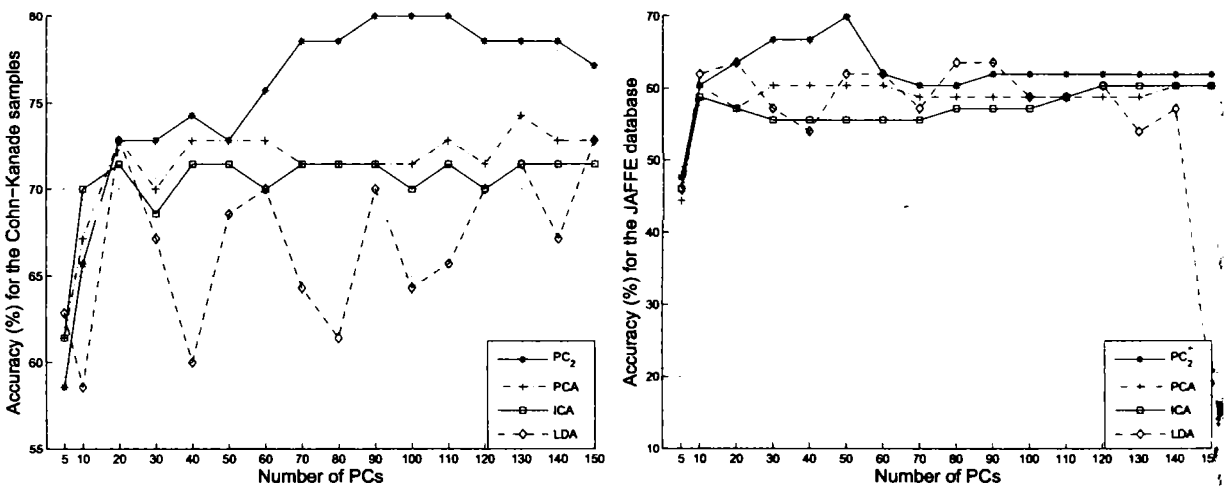


Figure 6.6: Experimental results for PC_2 corresponding to the JAFFE database for varying number of PCs, k, and $scale$.



a) b)

Figure 6.7: Experimental results for all methods involved in the experiment corresponding to a) C-K database, b) JAFFE database.

6.4 Conclusions

The underlying approach proposed in this paper is based on the phase congruency (PC_2) information extracted from a set of aligned images. These features seem to contain more discriminant power, as evidenced by experiments, than those retrieved by PCA, ICA or LDA technique, when applied to classify facial expressions. As far as the image registration is concerned, this preprocessing step is crucial as improper alignment would drastically reduce the method's performance. Since the local phase information is technically computed using banks of Gabor filters with different scales and standard deviation, the classification performances are highly influenced by these parameters. Based on the experimental findings, some remarks can be drawn with respect to the values of Gabor filter's parameters in relation to the facial expression database involved. For the Cohn - Kanade database, a narrow filter's bandwidth is necessary for high $scale$, while, for the JAFFE database and high $scale$, a larger bandwidth is required for leading to superior classification performances. It is worth mentioning that, the expressers from the JAFFE database are less expressive than the ones coming from the Cohn-Kanade database. It turns out that this issue leads to the necessity of using a larger filter's bandwidth for the expressions associated to the JAFFE database in order to capture more relevant information about the facial features corresponding to a particular expression.

Appendix A

Derivation of the DNMF updating rules

The expressions (5.9) and (5.11) can be proved by using an auxiliary function similar to those used in EM algorithm [109]. G is said to be an auxiliary function for $Y(\mathbf{F})$ if $G(\mathbf{F}, \mathbf{F}^{(t-1)}) \geq Y(\mathbf{F})$ and $G(\mathbf{F}, \mathbf{F}) = Y(\mathbf{F})$. If G is an auxiliary function for Y, then Y is nonincreasing under the update $\mathbf{F}^{(t)} = \text{argmin}_{\mathbf{F}} G(\mathbf{F}, \mathbf{F}^{(t-1)})$. With the help of G taking **W** and **A** as argument, a learning algorithm that alternates between updating basis images and updating coefficients can be derived. Since we did not impose any other constraints on the basis than those required by LNMF the derivation of (5.11) can be found in [106]. We have modified the coefficient expression, therefore we only give the derivation of its formulae. By fixing **W**, **A** is updated by minimizing $Y(\mathbf{A}) = D_{DNMF}(\mathbf{X}||\mathbf{WA})$. Let us define:

$$G(\mathbf{A}, \mathbf{A}^{(t-1)}) = \sum_{i,j}(x_{ij}\ln x_{ij} - x_{ij}) + \sum_{i,j,k} x_{ij}\frac{w_{ik}a_{kj}^{(t-1)}}{\sum_k w_{ik}a_{kj}^{(t-1)}}\left(\ln(w_{ik}a_{kj}) - \ln\frac{w_{ik}a_{kj}^{(t-1)}}{\sum_k w_{ik}a_{kj}^{(t-1)}}\right) +$$

$$+\sum_k w_{ik}a_{kj} + \alpha\sum_{i,j} u_{ij} - \beta\sum_i v_{ii} + \gamma\sum_{c=1}^{Q}\sum_{l=1}^{n_c}(a_{cl} - \mu_c)(a_{cl} - \mu_c)^T - \delta\sum_{c=1}^{Q}(\mu_c - \mu)(\mu_c - \mu)^T.$$

This function is an auxiliary function for $Y(\mathbf{A})$. It is straightforward to show that $G(\mathbf{A}, \mathbf{A}) = Y(\mathbf{A})$. In order to prove that $G(\mathbf{A}, \mathbf{A}^{(t-1)}) \geq Y(\mathbf{A})$, since $\ln(\sum_k w_{ik}a_{kj})$ is convex, the following inequality holds:

$$-\ln\left(\sum_k w_{ik}h_{kj}\right) \leq -\sum_k h_{ijk}\ln\frac{w_{ik}a_{kj}}{h_{ijk}}, \tag{A.2}$$

for all non-negative h_{ijk} that satisfy $\sum_k h_{ijk} = 1$. By denoting $h_{ijk} = \frac{w_{ik}a_{kj}^{(t-1)}}{\sum_k w_{ik}a_{kj}^{(t-1)}}$ we obtain:

$$-\ln\left(\sum_k w_{ik}h_{kj}\right) \leq -\sum_k \frac{w_{ik}h_{kj}^{(t-1)}}{\sum_k w_{ik}a_{kj}^{(t-1)}}\left(\ln w_{ik}h_{kj} - \ln\frac{w_{ik}a_{kj}^{(t-1)}}{\sum_k z_{ik}a_{kj}^{(t-1)}}\right). \tag{A.3}$$

From this inequality it follows that $G(\mathbf{A}, \mathbf{A}^{t-1}) \geq Y(\mathbf{A})$.

By setting $\frac{\partial G(\mathbf{A}, \mathbf{A}^{(t-1)})}{\partial a_{kl}}$ to zero for all kl, $l = 1, \ldots, n_c$ the partial derivative of G

with respect to a_{kl} gives us:

$$-\sum_i x_{il}\frac{w_{ik}a_{kl}^{(t-1)}}{\sum_k w_{ik}a_{kl}^{(t-1)}}\frac{1}{a_{kl}}+\sum_i w_{ik}-2\beta a_{kl}+2\gamma(a_{kl}-\mu_c)=0. \tag{A.4}$$

The equation can be rearranged as:

$$2\xi a_{kl}^2+\left(\sum_i w_{ik}-2\mu_c\right)a_{kl}-\sum_i x_{il}\frac{w_{ik}a_{kl}^{(t-1)}}{\sum_k w_{ik}a_{kl}^{(t-1)}}=0, \tag{A.5}$$

where $\xi=\gamma-\beta$.

This is a quadratic equation in a and it has the solution:

$$a_{kl}=\frac{2\mu_c-\sum_i w_{ik}+\sqrt{(\sum_i w_{ik}-2\mu_c)^2+8\xi a_{kl}^{(t-1)}\sum_i w_{ki}\frac{x_{il}}{\sum_k w_{ik}a_{kl}^{(t-1)}}}}{4\xi}. \tag{A.6}$$

Taking into account that $\sum_i w_{ik}=1$, we obtain (5.9).

Appendix B

Derivation of the PNMF updating rules

B.1 Derivation of the polynomial KNMF coefficients update

For updating the expression of the polynomial KNMF coefficients we present two approaches which lead to the same updating rule. The first approach derives the multiplicative rule (5.16) based on finding an upper bound minimizer which iteratively moves towards tighter upper bounds of the cost function involved. The second approach utilizes a gradient descent optimization procedure.

Definition 2 The function $G(b, b^{(t)})$ is an upper bound for $Q(b)$ if, for any b and $b^{(t)}$ we have $G(b, b) = Q(b)$ and $G(b, b^{(t)}) \geq Q(b)$, $\forall b \neq b^{(t)}$ [108].

Lemma 1 If G is an upper bound for Q, then Q is decreasing under the update $b^{(t+1)} = \text{argmin}_b G(b, b^{(t)})$.

Proof. $Q(b^{(t+1)}) = G(b^{(t+1)}, b^{(t+1)}) \leq G(b^{(t+1)}, b^{(t)}) \leq G(b^{(t)}, b^{(t)}) = Q(b^{(t)})$ □

Lemma 2 Let δ_{ij} denote the Kronecker delta function and let **L** be a diagonal matrix with elements $L_{ij} = \delta_{ij}(\mathbf{K}_{zz}\mathbf{b})_i / b_i^{(t)}$. Then the following theorem holds:

Theorem A.1: The upper bound of the function

$$Q(b) = \frac{1}{2}\sum_{j=1}^{n}\left(\Phi(\mathbf{x}_q) - \sum_{r=1}^{p} b_r \Phi(\mathbf{z}_r)\right)^2 \qquad \text{(A-1)}$$

is given by:

$$G(b, b^{(t)}) = G(b^{(t)}) + (b - b^{(t)})^T \nabla Q(b^{(t)}) + \\ + \frac{1}{2}(b - b^{(t)})^T L(b^{(t)})(b - b^{(t)}), \qquad \text{(A-2)}$$

where $\nabla Q(b^{(t)}) = \frac{\partial Q(b^{(t)})}{\partial b^{(t)}}$ is the first partial derivative with respect to $b^{(t)}$.

Proof. The cost function $Q(b)$ can be written as Taylor expansion in the neighborhood of the fixed point $b^{(t)}$ as follows:

$$Q(b) = Q(b^{(t)}) + (b - b^{(t)})^T \nabla Q(b - b^{(t)}) + \\ + \frac{1}{2}(b - b^{(t)})^T \nabla^2 Q(b^{(t)})(b - b^{(t)}), \qquad \text{(A-3)}$$

where $\nabla^2 Q(b^{(t)}) = \frac{\partial^2 Q(b^{(t)})}{\partial b'^2}$ is the second partial derivative with respect to $b^{(t)}$. Obviously when $b = b^{(t)}$ we have $G(b,b) = Q(b)$. For $b \neq b^{(t)}$, $G(b, b^{(t)}) \geq Q(b)$ is explicitly given by:

$$(b - b^{(t)})^T (L(b^{(t)}) - K_{zz})(b - b^{(t)}) \geq 0, \tag{A-4}$$

taking into account that $\frac{\partial^2 Q(b^{(t)})}{\partial b'^2} = \mathbf{K}_{zz}$. The relation (A-4) is equivalent with the statement that the matrix $\mathbf{L} - \mathbf{K}_{zz}$ is positive semidefinite. In order to prove that, consider first the matrix $\mathbf{P}$ whose elements are of the form:

$$P_{ij} = b_i^{(t)} (L - K_{zz})_{ij} b_j^{(t)}. \tag{A-5}$$

The matrix $\mathbf{P}$ is generated by rescaling elementwise the elements of $\mathbf{L} - \mathbf{K}_{zz}$. Therefore, $\mathbf{L} - \mathbf{K}_{zz}$ is positive semidefinite if $\mathbf{P}$ is positive semidefinite. For $\mathbf{P}$ and for any $\mathbf{b}$ we have:

$$\begin{aligned}
\mathbf{b}^T \mathbf{P} \mathbf{b} &= \sum_{i,j} b_i P_{ij} b_j \qquad \text{(A-6)} \\
&= \sum_{i,j} b_i b_j b_j^t \delta_{ij} (\mathbf{K}_{zz}\mathbf{b})_i - \sum_{i,j} b_i^t b_j^t b_i b_j K_{ij}^{zz} \\
&= \sum_{i,j} b_i^t b_j^t K_{ij}^{zz} b_i^2 - \sum_{i,j} b_i^t b_j^t b_i b_j K_{ij}^{zz} \\
&= \frac{1}{2} \sum_{i,j} b_i^t b_j^t b_i^2 K_{ij}^{zz} + \frac{1}{2} \sum_{i,j} b_i^t b_j^t K_{ij}^{zz} b_j^2 - \sum_{i,j} b_i^t b_j^t b_i b_j K_{ij}^{zz} \\
&= \frac{1}{2} \sum_{i,j} K_{ij}^{zz} b_i^t b_j^t (b_i - b_j)^2 \geq 0.
\end{aligned}$$

□

Here, K_{ij}^{zz} is the $\{i,j\}$ element of the matrix $\mathbf{K}_{zz}$.

Derivation of eq. (5.16), first solution.

Proof. Since $G(b, b^{(t)})$ is un upper bound for $Q(b)$ and $b^{(t+1)} = \text{argmin}_b G(b, b^{(t)})$ we find its minimum by taking the derivative and setting it to zero:

$$\frac{\partial G(b, b^{(t)})}{\partial b} = \nabla Q(b^{(t)}) + L(b^{(t)})(b - b^{(t)}) = 0. \tag{A-7}$$

This gives us:

$$L(b^{(t)}) b = L(b^{(t)}) b^{(t)} - \nabla Q(b^{(t)}). \tag{A-8}$$

Multiplying on the left by $L(b')^{-1}$, we get:

$$b = b^{(t)} - L(b^{(t)})^{-1} \nabla Q(b^{(t)}). \tag{A-9}$$

The partial derivative of $\nabla Q(b^{(t)})$ with respect to $b^{(t)}$ is given by:

$$\frac{\partial Q(b)}{\partial b_q} = -\Phi(\mathbf{z}_q)\sum_{j=1}^{n}\left(\Phi(\mathbf{x}_q) - \sum_{r=1}^{p} b_r \Phi(\mathbf{z}_r)\right) = \tag{A-10}$$

$$-\left(\sum_{j=1}^{n}\Phi(\mathbf{z}_q)\Phi(\mathbf{x}_q) - \sum_{j=1}^{n}\sum_{r=1}^{p} b_r \Phi(\mathbf{z}_q)\Phi(\mathbf{z}_r)\right) =$$

$$= -(\mathbf{k}_{zx} - \mathbf{K}_{zz}\mathbf{b})$$

Since $\mathbf{L}(b^{(t)})$ is a diagonal matrix,

$$L_{ij}(b^{(t)})^{-1} = b_i^{(t)} \frac{1}{\delta_{ij}(\mathbf{K}_{zz}\mathbf{b})_i}. \tag{A-11}$$

By substituting (A-11) and (A-11) in (A-9), we obtain

$$\begin{aligned} b_i &= b_i^{(t)} + b_i^{(t)} \frac{1}{(\mathbf{K}_{zz}\mathbf{b})_i}((\mathbf{k}_{zx})_i - (\mathbf{K}_{zz}\mathbf{b})_i) \\ &= b_i^{(t)} + b_i^{(t)} \frac{(\mathbf{k}_{zx})_i}{(\mathbf{K}_{zz}\mathbf{b})_i} - b_i^{(t)} \frac{(\mathbf{K}_{zz}\mathbf{b})_i}{(\mathbf{K}_{zz}\mathbf{b})_i} \\ &= b_i^{(t)} \frac{(\mathbf{k}_{zx})_i}{(\mathbf{K}_{zz}\mathbf{b})_i}. \end{aligned} \tag{A-12}$$

Putting it in a matrix form, we obtain the expression (5.16). □

Derivation of eq. (5.16), second solution.

Proof. An alternative solution can be found if we use a gradient descent optimization such as:

$$b = b^{(t)} - \eta(\nabla Q(b^{(t)})), \tag{A-13}$$

with $0 < \eta < \frac{1}{\beta}$,where η is the learning step and $\beta > 0$. Taking the Taylor expansion (A-3) and substituting b from (A-13), we finally have:

$$Q(b) - Q(b^{(t)}) = \eta(\nabla^2 Q(b^{(t)}))\left(1 - \frac{1}{2}\beta\eta\right). \tag{A-14}$$

Choosing an appropriate value for η and α such as $\eta = L_{ij}$ and $\beta = K_{zz}$, we have $\eta < \frac{1}{\beta}$, therefore $\left(1 - \frac{1}{2}\beta\eta\right) > 0$ for any element $z \in [0, 1]$, hence $Q(b) > Q(b^{(t)})$. However, this approach leads to the same solution since the relation (A-13) is equivalent with (A-9) after substituting η and β. □

B.2 Derivation of the polynomial KNMF basis images update, i.e. of eq. (5.17)

Proof. The same rationale is followed for obtaining an update rule for the basis images by employing eq. (5.20). Taking all images, the partial derivative of $\nabla Q(z)$ with

respect to z is given by:

$$\frac{\partial Q(z)}{\partial z_{\mu i}} = -\sum_{j=1}^{n} b_\mu \mathbf{K}'(\mathbf{x}_j \cdot \mathbf{z}_\mu) x_{ji} + \sum_{r=1}^{p} b_r b_\mu \mathbf{K}'(\mathbf{z}_r \cdot \mathbf{z}_\mu) z_{ri}. \tag{B-1}$$

In this case, the relation $G(z, z^{(t)}) \geq Q(z)$ translates into the following:

$$\frac{1}{2}\sum_{ij}[dzbK_{zz}^{d-1} - d(d-1)z^2 bK_{zz}^{d-2} + \tag{B-2}$$
$$+d(d-1)x^2 K_{xz}^{d-2}]z_i^{(t)} z_j^{(t)}(z_i - z_j)^2 \geq 0,$$

which is equivalent with:

$$x^2 K_{xz}^{d-2} \geq z^2 bK_{zz}^{d-2}. \tag{B-3}$$

Finally, the following inequality holds:

$$x^2 K_{xz}^{d-2} \geq x^2 K_{zz}^{d-2} \geq z^2 bK_{zz}^{d-2}, \tag{B-4}$$

since $(\mathbf{x}^T\mathbf{z})^{d-2} \geq (\mathbf{z}^T\mathbf{z})^{d-2}$, $\forall x \in [0, 255], z \in [0, 1]$ and $d \geq 2$, with equality for $d = 2$. Further, by choosing $L_{ij} = \delta_{ij}(\mathbf{z}\omega\mathbf{K}_{zz})_i / z_i^{(t)}$ we come up with the updating expression for basis images in (5.17). □

References

[1] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas, ``Exploiting discriminant information in non-negative matrix factorization with application to frontal face verification'', *IEEE Trans. on Neural Networks*, vol. 17, no. 3, pp. 683--695, 2006.

[2] I. Buciu and I. Nafornita, ``Linear and nonlinear dimensionality reduction techniques'', *Journal of Studies in Informatics and Control*, vol. 16, no. 4, pp. 431--444, December, 2007.

[3] I. Buciu, I. Nafornita and I. Pitas, ``Non-negative matrix factorization methods and their applications'', *IEEE Trans. on Signal Processing*, under review.

[4] H. P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen, and E. Petajan, ``Multimodal system for locating heads and faces,'' in *Proc. Second IEEE Int. Conf. Automatic Face and Gesture Recognition*, pp. 88--93, 1996.

[5] M. -H. Yang and N. Ahuja, ``Extracting gestural motion trajectory,'' in *Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition*, pp. 10--15, 1998.

[6] K. I. Kim, K. Jung, and H. J. Kim, ``Face recognition using kernel principal component analysis,'' *IEEE Signal Processing Letters*, vol. 9. no. 2, pp. 40--42, February, 2002.

[7] H. Rowley, S. Baluja, and T. Kanade, ``Human face detection in visual scenes,'' in *Advances in Neural Information Processing Systems*, vol. 8, pp. 875 - 881, 1997.

[8] R. E. Kleck and M. Mendolia, ``Decoding of profile versus full-face expressions of affect,'' *Journal of Nonverbal Behavior*, vol. 14, no. 1, pp. 35--49, 1990.

[9] I. Essa and A. Pentland, ``Coding, analysis, interpretation, and recognition of facial expressions,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 757--763, 1997.

[10] A. Pentland, B. Moghaddam, and T. Starner, ``View-based and modular eigenspaces for face recognition,'' in *Computer Vision and Pattern Recognition Conference*, pp. 84--91, 1994.

[11] G. Yang and T.-S. Yang, ``Human face detection in complex backround,'' *Pattern Recognition*, vol. 27, no. 1, pp. 53 -- 63, 1994.

[12] M.-H.Yang, N. Ahuja, and D. Kriegman ``Face detection using a mixture of factor analyzers,'' in *Proc. of the 1999 IEEE Int. Conf. on Image Processing*, vol. 3, pp. 612--616, 1999.

[13] R. Vaillant, C. Monrocq, and Y. Len Cun, ``Original approach for the localisation of objects in images ,'' *IEE Proc. Vis. Image Signal Processing*, vol. 141, no. 4, August 1994.

[14] K.-C Yow and R. Cipolla, ``Feature-based human face detection ,'' *Image and Vision Computing*, vol. 15, no. 9, pp. 713--735, 1999.

[15] K.-K. Sung, and T. Poggio, ``Example-based learning for view-based human face detection,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39--51, January 1998.

[16] C. Huang and Y. Huang, ``Facial expression recognition using model based feature extraction and action parameters classification,'' *Journal of Visual Communication and Image Presentation*, pp. 278--290, 1997.

[17] H. Hong, H. Neven, and C. V. der Malsburg, ``Online facial expression recognition based on personalized Galleries,'' in *Second Int. Conf. on Automatic Face and Gesture Recognition*, pp. 354--359, 1998.

[18] J. Steffens,E. Elagin, H. Neven, and C. V. der Malsburg, ``PersonSpotter - fast and robust system for human detection, tracking and recognition,'' in *Third Int. Conf. on Automatic Face and Gesture Recognition*, pp. 516--521, 1998.

[19] V. P. Kumar and T. Poggio, ``Learning-based approach to real time tracking and analysis of faces ,'' in *Fourth Int. Conf. on Automatic Face and Gesture Recognition*, pp. 96--101, 2000.

[20] M. Pantic and L. J. M. Rothkrantz, ``Expert system for automatic analysis of facial expressions,'' *Image and Vision Computing*, no. 18, pp. 881--905, March, 2000.

[21] N. Oliver, A. Pentland, and F. Berard, ``Lafter: a real-time face and lips tracker with facial expression recognition,'' *Pattern Recognition*, vol. 33, pp. 1369--1382, 2000.

[22] M. S. Bartlett, G. Littlewort, I. Fasel, and J. R. Movellan, ``Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction,'' in *2003 Conference on Computer Vision and Pattern Recognition Workshop*, vol. 5, pp. 53--59, 2003.

[23] P. Viola, and M. Jones, ``Robust real-time object detection,'' *Technical Report* CRL 20001/01, Cambridge Research-Laboratory, 2001.

[24] J. Friedman, T. Hastie, and R. Tibshirani, ``Additive logistic regression: A statistical view of boosting,'' *Annals of Statistics*, vol. 28, no. 2, pp. 337--374, 2000.

[25] Y. Tian, ``Evaluation of face resolution for expression analysis,'' in *Proc. of CVPR Workshop on Face Processing in Video (FPIV'04)*, 2004.

[26] H. Rowley, S. Baluja, and T. Kanade, ``Neural network - based face detection,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23--38, 1998.

[27] R. Isukapalli, R. Greiner, and A. Elgammal, ``Learning a dynamic classification method to detect faces and identify facial expression,'' *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, pp. 70--84, 2005.

[28] M. -H. Yang, D. Kriegman, and N. Ahuja ``Detection faces in images: A survey,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34--58, January, 2002.

[29] M. - H. Yang and N. Ahuja, ``Face detection and gesture recognition for human computer interaction,'' in *Kluwer Academic Publishers*, 2001.

[30] J. M. Susskind, G. Littlewort, M. S. Bartlett, J. Movellan, and A. K. Anderson, ``Human and computer recognition of facial expressions of emotion,'' *Neuropsychologia*, vol. 45, no.1, pp. 152--162, 2007.

[31] B. Fasel and J. Luettin, ``Automatic facial expression analysis: A survey'', *Pattern Recognition*, vol. 36, no. 1, pp. 259-□275, 2003.

[32] P. Ekman andW. Friesen, ``Constants across cultures in the face and emotion'', *Journal of Personality and Social Psychology*, vol. 17, no. 2, pp. 124□-129, 1971.

[33] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, ``Coding facial expressions with gabor wavelets'', In *Proc. Third IEEE Int. C. on Automatic Face and Gesture Recognition*, pp. 200-□205, 1998.

[34] P. Ekman and W. Friesen, "The Facial Action Coding System", *Consulting Psychologists* Press Inc., Palo Alto, Calif., 1978.

[35] C. Darwin, ``The Expression of the Emotions in Man and Animal'', J. Murray, London, 1872.

[36] M. Pantic and L .J. M. Rothkrantz, ``Automatic analysis of facial expressions: The state-of-the-art,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424--1445, Dec., 2000.

[37] M. Pantic and L .J. M. Rothkrantz, ``Facial action recognition for facial expression analysis from static face images,'' *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 34, no. 3, pp. 1449--1461, June, 2004.

[38] Z. Zhang, M. Lyons, M. Schuster, and S. Akamatsu, ``Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron,'' *in Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition*, April 14-16 1998, Nara Japan, pp. 454-459, 1998.

[39] L. Wiskott, J. -M. Fellous, N. Kruger, and C. von der Malsburg, ``Face recognition by elastic bunch graph matching,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 775--779, July 1997.

[40] Y.-Li Tian, T. Kanade, and J. Cohn, ``Evaluation of Gabor-wavelet-based facial action unit recognition in image sequences of increasing complexity,'' in *Proc. Fifth IEEE Int. Conf. Automatic Face and Gesture Recognition*, May, pp. 229--234, 2002.

[41] Y.-Li Tian, T. Kanade, and J. Cohn, ``Recognizing action units for facial expression analysis,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no.2, pp. 97--115, Feb. 2001.

[42] G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan, ``Dynamics of facial expression extracted automatically from video,'' *Image and Vision Computing*, vol. 24, no. 6, pp. 615--625, 2006.

[43] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman, and T. J. Sejnowski, ``Classifying facial actions,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 974--989, October 1999.

[44] J. Kim, J. Choi, J. Yi, and M. Turk, ``Effective representation using ICA for face recognition robust to local distortion and partial occlusion,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1977-1981, December 2005.

[45] B. A. Draper, K. Baek, M. S. Bartlett and J. R. Beveridge, ``Recognizing faces with PCA and ICA,'' *Computer Vision and Image Understanding*, vol. 91: Special issue on Face Recognition, pp. 115--137, 2003.

[46] B. Moghaddam, ``Principal manifolds and Bayesian subspaces for visual recognition,'' in *Int. Conf. Computer Vision (ICCV'99)*, pp. 1131--1136, 1999.

[47] G. Guo and C. R. Dyer, ``Learning from examples in the small sample, case: Face expression recognition,'' *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 35, no. 3, pp. 477--488, 2005.

[48] G. Cottrell and J. Metcalfe, ``Face, gender and emotion recognition using holons,'' *Advances in Neural Information Processing Systems*, vol. 3, pp. 564--571, 1991.

[49] C. Padgett and G. Cottrell, ``Representing face images for emotion classification ,'' *Advances in Neural Information Processing Systems*, vol. 9, pp. 894--900, 1997.

[50] A. J. Calder, A. M. Burton, P. Miller, A. W. Young and S. Akamatsu, ``A principal component analysis of facial expressions ,'' *Vision Research* , vol. 41, pp. 1179--1208, 2001.

[51] G. Littlewort, M. Bartlett, I. Fasel, J. Chenu, T. Kanda, H. Ishiguro, and J. Movellan, ``Towards social robots: Automatic evaluation of human-robot interaction by face detection and expression classification,'' *Advances in Neural Information Processing Systems*, vol. 16, pp. 1563--1570, 2004.

[52] E. Osuna, R. Freund, and F. Girosi, ``Training support vector machines: An application to face detection,'' in *Proc. of the IEEE Computer Society Computer Vision and Pattern Recognition Conf.*, pp. 130--136, 1997.

[53] V.N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer Verlag, 1995.

[54] C. Burges, ``A Tutorial on support vector machines for pattern recognition,'' *Data Mining and Knowledge Discovery*, vol. 2, pp. 1--43, 1998.

[55] I. Buciu, C. Kotropoulos, and I. Pitas, ``Combining support vector machine for accurate face detector,'' *2001 IEEE International Conference on Image Processing*, pp. 1054--1057, 2001.

[56] L. Breiman, ``Bagging Predictors,'' *Machine Learning*, vol. 24 , pp. 123--140, 1996.

[57] S. Gunn, "Support Vector Machines for Classification and Regression", ISIS Technical Report ISIS-1-98, Image Speech & Intelligent Systems Research Group, University of Southapton, May. 1998.

[58] B. Efron and R.Tibshirani, *An Introduction to the Bootstrap*. New York: Chapman & Hall, 1993.

[59] L. Breiman, ``Bagging predictors,'' *Machine Learning*, vol. 24, pp. 123--140, 1996.

[60] Y. Freund and R. E., Schapire, ``Experiments with a new boosting algorithm,'' in L. Saitta, ed., *Machine Learning: Proc. Thirteenth Int. Conf. Machine Learning*, pp. 148--156, Morgan Kaufmann, 1996.

[61] D. Wolpert, ``Stacked generalization,'' *Neural Networks*, vol. 5, pp. 241--259, 1992.

[62] I. Buciu, C. Kotropoulos, and I. Pitas, ``Combining support vector machines for accurate face detection,'' in *Proc. 2001 IEEE Int. Conf. Image Processing*, pp. 1054--1057, 2001.

[63] T. G. Dietterich, ``An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization,'' *Machine Learning*, vol. 40, no. 2, pp. 139--157, 2000.

[64] R. Avnimelech and N. Intrator, ``Boosted mixture of experts: an ensemble learning scheme,'' *Neural Computation*, vol. 11, pp. 483--497, 1999.

[65] L. Breiman, ``Bias, variance and arcing classifiers,'' Technical Report 460, Statistics Department, University of California at Berkeley, Berkeley, 1996.

[66] O. Bousquet and A. Elisseeff, ``Stability and generalization,'' *Journal Machine Learning Research*, vol. 2, pp. 499-526, 2002.

[67] T. Evgeniou, M. Pontil, and T. Poggio, ``Regularization networks and support vector machines,'' in *Advances in Large Margin Classifiers*, pp. 171-204, Cambridge, MA, 2000. MIT Press.

[68] J1. I. Buciu, C. Kotropoulos, and I. Pitas, ``Demonstrating the stability of support vector machines for classification,'' *Signal Processing*, vol. 86, no. 9, pp. 2364--2380, 2006.

[69] E. B. Kong and T. G. Dietterich, ``Error-correcting output coding corrects bias and variance,'' in *Proc. Twelfth Int. Conf. Machine Learning*, pp. 313--321, 1995.

[70] R. Kohavi and D. H. Wolpert, ``Bias plus variance decomposition for zero-one loss functions, in L. Saitta, ed., *Machine Learning: Proc. Thirteenth Int. Conf. Machine Learning*, pp. 275--283, Morgan Kaufmann, 1996.

[71] J. Friedman, ``Bias, variance, 0-1 loss and the curse of dimensionality,'' Technical Report, Stanford University, 1996.

[72] R. Tibshirani, ``Bias, variance and prediction error for classification rules,'' Techincal Report, Department of Statistics, University of Toronto, Toronto, Canada, 1996.

[73] T. Heskes, ``Bias/variance decompositions for likelihood-based estimators,'' *Neural Computation*, vol. 10, no. 6, pp. 1425--1433, MIT Press, 1998.

[74] G. Valentini and T. G. Dietterich, ``Low bias bagged support vector machines,'' in *Proc. Twentieth Int. Conf. Machine Learning*, Washington, D.C., USA, pp. 752--759, 2003.

[75] G. Valentini and T. G. Dietterich, ``Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods,'' *Journal of Machine Learning Research*, vol. 5, pp. 725--775, 2004.

[76] T. Cover and P. Hart, ``Nearest neighbor pattern classification,'' *IEEE Trans. Inform. Theory*, pp. 21--27, 1967.

[77] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer-Verlag, New York, 1996.

[78] C. Kotropoulos, A. Tefas, and I. Pitas, ``Morphological elastic graph matching applied to frontal face authentication under well-controlled and real conditions,'' *Pattern Recognition*, vol. 33, no. 12, pp. 31-43, October 2000.

[79] K. K. Sung and T. Poggio, ``Example-based learning for view-based human face detection,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39--51, 1998.

[80] ftp://ftp.uk.research.att.com/pub/data/att_faces.zip

[81] L. I. Kuncheva and C. J. Whitaker, ``Measures of diversity in classifier ensembles,'' *Machine Learning*, vol. 51, pp. 181--207, 2003.

[82] T. Evgeniou, M. Pontil, and A. Elisseff, ``Leave one out error, stability, and generalization of voting combination of classifiers,'' *Machine Learning*, vol. 55, pp. 71-97, 2004.

[83] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, N. Y: J. Wiley, 2001.

[84] A. J. Bell and T. J. Sejnowski, ``An information-maximization approach to blind separation and blind deconvolution,'' *Neural Computation*, vol. 7, no. 6, pp. 1129--1159, 1995.

[85] T.-W. Lee, M. Girolami, and T. J. Sejnowski, ``Independent component analysis using an extended Infomax algorithm for mixed sub-Gaussian and super-Gaussian sources,'' *Neural Computation*, vol. 11, no. 2, pp. 417--441, 1999.

[86] J. F. Cardoso and A. Souloumiac, ``Blind beamforming for non Gaussian signals,'' *IEE Proceedings-F*, vol. 140, no. 6, pp. 362-370, 1993.

[87] A. Hyvarinen, ``Fast and robust fixed-point algorithms for independent component analysis,'' *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp. 626--634, 1999.

[88] M. McKeown, S. Makeig, G. Brown, T. Jung, S. Kindermann, and T. Sejnowski, ``Spatially independent activity patterns in functional magnetic resonance imaging during the stroop color-naming task,'' in *Proc. Nat. Acad. Sci.*, vol. 95, pp. 803--810, 1998.

[89] J. V. Stone and J. Porrill, ``Undercomplete independent component analysis for signal separation and dimension reduction,'' Technical Report, 1998.

[90] F. R. Bach and M. J. Jordan, ``Kernel independent component analysis,'' *Machine Learning Research*, vol. 3, pp. 1--48, 2002.

[91] M. S. Bartlett, J. R. Movellan, and T. K. Sejnowski, ``Face recognition by independent component analysis,'' *IEEE Trans. Neural Networks*, vol. 13, no. 6, pp. 1450--1464, 2002.

[92] T. Kanade, J. Cohn, and Y. Tian, ``Comprehensive database for facial expression analysis,'' in *Proc. Fourth IEEE Int. Conf. Face and Gesture Recognition*, pp. 46-53, March, 2000.

[93] J. C. Platt, ``Fast training of support vector machines using sequential minimal optimization,'' *Advances in Kernel Methods - Support Vector Learning*, vol. 12, pp. 185--208, 1999.

[94] J. C. Platt, N. Cristianini, and J. S.-Taylor, ``Large margin DAGs for mutliclass classification,'' *Advances in Neural Information Processing Systems*, vol. 12, pp. 547--553, 2000.

[95] I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik, ``What size test set gives good error rate estimates?,'' *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 52--64, January 1998.

[96] N. Kwak, C. - H. Choi, and N. Ahuja ``Face recognition using feature extraction based on independent component analysis,'' in *Proc. 2002 IEEE Int. Conf. Image Processing*, pp. 337--340, 2002.

[97] Y. Petrov and Z. Li, ``Local correlations, information redundancy, and the sufficient pixel depth in natural images,'' *Journal Optical Society of America A*, vol. 20, no. 1, pp. 56--66, 2003.

[98] D. Guillamet, B. Schiele, and J. Vitri, ``Analyzing non-negative matrix factorization for image classification,'' in *Proc. of 16th Int. Conf. on Pattern Recognition*, vol. II, pp. 116--119, 2002.

[99] P. Paatero and U. Tapper, ``Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values,'' *Environmetrics*, vol. 5, pp. 111--126, 1994.

[100] T. Kawamoto, K. Hotta, T. Mishima, J. Fujiki, M. Tanaka, and T. Kurita ``Estimation of single tones from chord counds using non-negative matrix factorization,'' in *Neural Network World*, vol. 3, pp. 429--436, 2000.

[101] B. Wang and M. D. Plumbley, ``Musical audio stream separation by non-negative matrix factorization,'' in *Proc. of DMRN Summer Conference*, Glasgow, 2005.

[102] M. Cooper and J. Foote, ``Summarizing video using non-negative similarity matrix factorization,'' in *Proc. IEEE Workshop on Multimedia Signal Processing*, pp. 25--28, 2002.

[103] E. Kim, P. K. Hopke, and E. S. Edgerton, ``Source identification of Atlanta aerosol by positive matrix factorization,'' *Journal Air Waste Manage. Assoc.*, vol. 53, no. 1, pp. 731--739, 1977.

[104] J. Piper, V. P. Pauca, R. J. Plemmons, and M. Giffin, ``Unmixing spectral data for space objects using independent component analysis and non-negative matrix factorization,'' in *Proc. Amos Technical Conf.*, 2004.

[105] P. Pauca, F. Shahnaz, M. Berry and R. Plemmons, ``Text mining using non-negative matrix factorization,'' in *Proc. SIAM Inter. Conf. on Data Mining*, 2004.

[106] S. Z. Li, X. W. Hou and H. J. Zhang, ``Learning spatially localized, parts-based representation,'' *Int. Conf. Computer Vision and Pattern Recognition*, pp. 207--212, 2001.

[107] D D. Lee and H. S. Seung, ``Learning the parts of the objects by non-negative matrix factorization,'' *Nature*, vol. 401, pp. 788--791, 1999.

[108] D D. Lee and H. S. Seung, ``Algorithms for non-negative matrix factorization,'' *Advances Neural Information Processing Systems*, vol. 13, pp. 556--562, 2001.

[109] A. P. Dempster, N. M. Laird and D. B. Rubin, ``Maximum likelihood from incomplete data via the EM algorithm,'' *Journal of Royal Statistical Society*, vol. 39, no. 1, pp. 1--38, 1977.

[110] I. Buciu and I. Pitas, ``A new sparse image representation algorithm applied to facial expression recognition,'' *Proc. IEEE Workshop on Machine Learning for Signal Processing*, pp. 539--548, Sao Luis, Brazil, 2004.

[111] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, ``Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection,'' *ECCV*, vol. 1, pp. 45--58, 1996.

[112] I. Buciu, C. Kotropoulos and I. Pitas, ``ICA and Gabor representation for facial expression recognition,'' in *Proc. 2003 IEEE Int. Conf. Image Processing*, pp. 855--858, 2003.

[113] P. Foldiak, ``Sparse coding in the primate cortex,'' *The Handbook of Brain Theory and Neural Networks*, Second Edition, pp. 1064--1068, MIT Press, 2002.

[114] O. Schwartz and E. P. Simoncelli, ``Natural signal statistics and sensory gain control,'' *Nature Neuroscience*, vol. 4, no. 8, pp. 819--825, 2001.

[115] E. Simoncelli, ``Vision and the statistics of the visual environment,'' *Current Opinion in Neurobiology*, vol. 13, pp. 144--149, 2003.

[116] J. Touryan, G. Felsen, and Y. Dan, ``Spatial structure of complex cell receptive fields measured with natural images,'' *Neuron*, vol. 45, pp. 781--791, 2005.

[117] J. Rapela, J. M. Mendel, and N. M. Grzywacz, ``Estimating nonlinear receptive fields from natural images,'' *Journal of Vision*, vol. 6, no. 4, pp. 441--474, 2006.

[118] J. Malo, E. P. Simoncelli, I. Epifanio, and R. Navarro, ``Non-linear image representation for efficient perceptual coding,'' *IEEE Trans. on Image Processing*, vol. 15, no. 1, pp. 68--80, 2006.

[119] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda and B. Schölkopf, ``An introduction to kernel-based learning lagorithms,'' *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp. 181--201, 2001.

[120] A. S. Have, M. A. Girolami and J. Larsen, ``Clustering via kernel decomposition,'' *IEEE Trans. on Neural Networks*, vol. 17, no. 1, pp. 48--58, 2006.

[121] C. S. Ong, A. J. Smola, and R. C. Williamson, ``Learning the kernel with hyperkernels,'' *Journal of Machine Learning Research*, vol. 6, pp. 1043--1071, 2005.

[122] I. Wai-Hung Tsang and J. Tin-Yau Kwok, ``Efficient hyperkernel learning using second-order cone programming,'' *IEEE Trans. on Neural Networks*, vol. 17, no. 1, pp. 48--58, 2006.

[123] S. Yang, S. Yan, C. Zhangand X. Tang, ``Bilinear analysis for kernel selection and nonlinear feature extraction,'' *IEEE Trans. on Neural Networks*, vol. 18, no. 5, pp. 1442--1452, 2007.

[124] I. Buciu, N. Nikolaidis, and I. Pitas, ``Non-negative matrix factorization in polynomial feature space,'' *IEEE Trans. on Neural Networks*, vol. 19, no. 6, pp. 1090--1100, 2008.

[125] R. Basri and D. W. Jacobs, ``Lambertian reflectance and linear subspaces,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 218--233, 2003.

[126] J. S. - Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[127] C. J. C. Burges, ``Simplified support vector decision rules,'' in *Int. Conf. on Machine Learning*, pp. 71--77, 1996.

[128] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola, ``Input space versus feature space in kernel-based methods,'' *IEEE Trans. on Neural Networks*, vol. 10, no. 5, pp. 1000--1017, 1999.

[129] I. T. Jolliffe, *Principal Component Analysis*, (2nd ed.), New York: Springer-Verlag, 2002.

[130] I. Buciu and I. Pitas, ``Application of non-negative and local non-negative matrix factorization to facial expression recognition,'' *Int. Conf. on Pattern Recognition*, pp. 228--291, 2004.

[131] http://www.mathworks.com/

[132] M. Riesenhuber and T. Poggio, ``Hierarchical models of object recognition in cortex,'' *Nature Neuroscience*, vol. 2, pp. 1019--1025, 1999.

[133] R. Desimone, ``Face selective cells in the temporal cortex of monkey,'' *Journal of Cognitive Neuroscience*, no. 3, pp. 1--8, 1991.

[134] D. I. Perret, E. T. Rolls, and W. Caan, ``Visual neurons responsive to faces in the monkey temporal cortex,'' *Experimental Brain Research*, no. 47, pp. 329--342, 1982.

[135] N. Kanwisher, J. McDermott, and M. M. Chun, ``The fusiform face area: A module in human extrastriate cotrex specialized for face perception,'' *Journal of Neuroscience*, no. 17, pp. 4302--4311, 1997.

[136] M. E. Hasselmo, E. T. Rolls, G. C. Baylis, and V. Nalwa, ``The role of expression and identity in the face-selective responses of neurons in the temporal visual cortex of the monkey,'' *Behavioral Brain Research*, no. 32, pp. 203--218, 1989.

[137] B. A. Olshausen and D. J. Field, ``Natural image statistics and efficient coding,'' *Network Computation in Neural Systems*, vol. 7, no. 2, pp. 333--339, 1996.

[138] S. Marcelja, ``Mathematical description of the responses of simple cortical cells,'' *Journal of the Optical Society of America*, vol. 70 A, no. 11, pp. 1297--1300, 1980.

[139] J. G. Daugman, ``Two-dimensional spectral analysis of cortical receprive field profile,'' *Vision Research*, vol. 20, pp. 847--856, 1980.

[140] P. J. B. Hancock, R. J. Baddeley, and L. S. Smith, ``The principal components of natural images,'' *Network Computation in Neural Systems*, vol. 3, no. 1, pp. 61--70, 1992.

[141] C. Fyfe and R. Baddeley, ``Finding compact and sparse-distributed representations of visual scenes,'' *Network Computation in Neural Systems*, vol. 6, no. 3, pp. 333--344, 1995.

[142] A. J. Bell and T. J. Sejnowski, ``The `independent components' of natural scenes are edge filters'' *Vision Research*, no. 37, pp. 3327--3338, 1997.

[143] I. Buciu and I. Pitas, ``NMF, LNMF, and DNMF modeling of neural receptive fields involved in human facial expression perception'', *Journal of Visual Communication and Image Representation*, vol. 17, no. 5, pp. 958--969, October, 2006.

[144] K. Tanaka, C. Saito, Y. Fukada, and M. Moriya ``Integration of form, texture, and color information in the inferotemporal cortex of the macaque,'' *Vision, Memory and the Temporal Lobe*, pp. 101--109, 1990.

[145] K. P. Kording, C. Kayser, W. Einhauser, and P. Konig, ``How are complex cell properties adapted to the statistics of natural stimuli ?,'' *Journal of Neurophysiology*, vol. 91, no. 1, pp. 206--212, 2004.

[146] J. J. Atick, ``Could information theory provide an ecological theory of sensory processing,'' *Network*, no. 3, pp. 213--251, 1992.

[147] J. J. Atick and A. N. Redlich, ``What does the retina know about the natural scene ?,'' *Neural Computation*, vol. 4, pp. 196--210, 1992.

[148] E. T. Rolls and A. Treves, ``The relative advantages of sparse vesrus distributed encoding for associative neural networks in the brain,'' *Network*, no. 1, pp. 407--421, 1990.

[149] I. Buciu and I. Nafornita, ``Feature extraction through phase congruency for facial expression analysis'', *International Journal of Pattern Recognition and Artificial Intelligence*, accepted for publication, 2009.

[150] A. V. Oppenheim and J. S. Lim, ``The importance of phase in signals,'' *Proceedings of the IEEE*, vol. 69, pp. 529--541, 1981.

[151] M. G. A. Thomson, ``Visual coding and the phase structure of natural scenes,'' *Network: Comput. Neural Syst.*, vol. 10, pp. 123--132, 1999.

[152] P. Kovesi, ``Image features from phase congruency,'' *Videre : Journal of Computer Vision Research*, vol. 1, no. 3, pp. 1--27, 1999.

[153] P. Kovesi, ``Phase congruency: A low-level image invariant,'' *Psych. Research*, vol. 64, pp. 136--148, 2000.

[154] P. Kovesi, ``Phase congruency detects corners and edges,'' *in DICTA*, Sydney, December, 2003.

[155] M. C. Morrone, J. R. Ross, D. C. Burr, and R. A. Owens, ``Mach bands are phase dependent,'' *Nature*, vol. 324, pp. 250--253, 1986.

[156] S. Venkatesh and R. A. Owens, ``An energy feature detection scheme,'' *International Conference on Image Processing*, pp. 553--557, 1989.

[157] D. J. Field, ``Relations between the statistics of natural images and the response properties of cortical cells,'' *Journal of the Optical Society of America A*, vol. 4, no. 12, pp. 2379--2394, 1987.

[158] D. J. Field, ``What the statistics of natural images tell us about visual coding,'' *SPIE: Human Vision, Visual Processing, and Digital Display*, vol. 1077, no. 12, pp. 269--276, 1989.

Titluri recent publicate în colecția „TEZE DE DOCTORAT"
seria 7: Inginerie Electronică și Telecomunicații

1. **Adrian Lazăr Șchiop** – *Contribuții la studiul convertoarelor utilizate la acționarea motoarelor sincrone, ISBN 978-973-625-409-3,* 2007;
2. **Ioan Gavriluț** – *Contribuții la navigația roboților mobili autonomi utilizând rețelele neuronale celulare, ISBN 9789-973-625-417-8,* 2007;
3. **Marian Constantin Bucos** – *Dezvoltarea sistemelor informatice pentru e-learning și realizarea de organizații educaționale virtuale, ISBN 978-973-625-560-1,* 2007;
4. **Horia – Gheorghe Baltă** – *Contribuții la dezvoltarea și proiectarea turbo-codurilor binare și nebinare, ISBN 978-973-625-601-1,* 2008;
5. **Marin Titus Tomșe** – *Contribuții la studiul teoretic și experimental al surselor de alimentare pentru cuptoarele de încălzire inductivă, ISBN 978-973-625-608-0,* 2008
6. **Radu Dan Mihăescu** – *Concepția unor surse de curent de referință pentru circuite integrate CMOS, ISBN 978-973-625-707-0,* (2008);
7. **Raul Ciprian Ionel** – *Contribuții la localizarea surselor de zgomot utilizând instrumentație virtuală, ISBN 978-973-625-746-9,* (2008).

EDITURA POLITEHNICA