Universitat
Autònoma
de Barcelona

Statistical Independence for Classification of High Dimensional Data

A dissertation submitted by **Marco J.M. Bressan**
at the Universitat Autònoma de Barcelona to fulfill
the degree of **Doctor en Informática**.

Bellaterra, March 6, 2003

Director: **Dr. Jordi Vitrià i Marca**
Universitat Autònoma de Barcelona
Dept. Informàtica & Computer Vision Center

Centre de Visió
per Computador

This document was typeset by the author using LaTeX 2$_\varepsilon$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

ISBN 84–932156–5–1

Printed by Ediciones Gráficas Rey, S.L.

A mi familia.

En sus remotas páginas está escrito que los animales se dividen en (a) pertenecientes al Emperador, (b) embalsamados, (c) amaestrados, (d) lechones, (e) sirenas, (f) fabulosos, (g) perros sueltos, (h) incluidos en esta clasificación, (i) que se agitan como locos, (j) innumerables, (k) dibujados con un pincel finísimo de pelo de camello, (l) etcétera, (m) que acaban de romper el jarrón, (n) que de lejos parecen moscas.

(**Jorge Luis Borges**, Otras Inquisiciones)

Agradecimientos

Esta tesis fue realizada con la beca de Formación de Personal Universitario de la Secretaría de Estado de Educación y Universidades del Ministerio de Educación, Cultura y Deporte de España.

En primer lugar quiero agradecer al director de esta tesis, el Dr. Jordi Vitrià. Al doctor, por su confianza, su disponibilidad, sus conocimientos generosos y sus consejos. A Jordi, por su apoyo permanente y entereza. De él he aprendido mucho más de lo que cabe en estas páginas.

Al Dr. Juan José Villanueva quien, como director del *Centre de Visió per Computador*, me ha brindado la oportunidad de investigar en este entorno privilegiado, aparte de proporcionarme su ayuda personal en todo momento.

Los trabajos que aquí se presentan están en gran parte basados en el análisis de componentes independientes y el algoritmo de estimación de preferencia ha sido el FastICA, desarrollado por el Dr. Aapo Hyvärinen y el grupo de investigación en ICA de la Universidad Tecnológica de Helsinki. Agradezco al Dr. Hyvärinen haberme dado la oportunidad de realizar una breve estancia en su laboratorio y, en ese lapso, haberme aclarado dudas acumuladas a lo largo de casi tres años respecto de su método y aplicaciones.

La aplicación del algoritmo CC-ICA al problema de reconocimiento de objetos, así como el desarrollo del método WNMF son fruto del trabajo con David Guillamet, un gran profesional de una eficiencia inhibidora y, sobre todo, una gran persona con los dos sentidos básicos bien puestos: el común y el del humor.

La aplicación al análisis de datos multiespectrales de los algoritmos que aquí se presentan debe mucho a la colaboración e intermediación de la Dra. Petia Radeva. Lo mismo sobre los experimentos de clasificación de tapones de corchos, donde también ha colaborado Antoni Tovar. Le agradezco a Petia toda su colaboración y el entusiasmo puesto en cada uno de los proyectos que emprendimos.

No sé si esta tesis no estaría aquí, pero seguro que no estaría aquí *hoy* de no ser por Albert Pujol, quien ha hecho suyas mis obligaciones en más de una ocasión. En Albert he encontrado reunidos un profesional excelente, un colega crítico, un compañero de trabajo indispensable y, sobre todo, un gran amigo.

Mucha de la teoría que aparece en esta tesis es el resultado de discusiones, consultas y sugerencias con colegas del CVC. En particular, me han aportado mucho las discusiones con Xevi Orriols y el Dr. Xavi Binefa.

Gracias a Anna, que ha estado en las buenas y en las malas. Me cuesta imaginarme un despacho sin Anna como compañera. En ocasiones Anna ha aliviado el peso de

soportarme en el despacho sobre otras personas. Unos italianos que me hacían sentir que todavía no me había ido de la Argentina aunque nunca tanto como las charlas con el Dr. Toledo, Jeff Berens, Castor, Fernando Vilariño, Anders Ericsson, Jordi Arnabat y David. A todos ellos Anna les queria agradecer por su ayuda. A Arnabatman gracias por todo.

Gracias a aquellos colegas con quienes he compartido momentos maravillosos más allá del trabajo, a Javito, a Alex, Poal, Xavi R. y Paco. Y gracias a toda la gente del CVC, la que está y la que se fue, siempre dispuestos a compartir un momento y a dar una mano: [davidm, juanma, rfelip, botey, felipe, maria, ramon, joanm, enric, josep, oriol, robert, cristina, daniel, agnesba, debora, vicente, raquel, judit, misael, selma]@cvc.uab.es. Muchas (pero muchas) gracias a los que hacen que esto funcione: [mjose, pilar, montse, anacelia, mcmerino, pedro]@cvc.uab.es. Al resto de la gente de la unidad y de transferencia tecnológica, tambien muchas gracias.

Gracias a mis amigos. Los que están acá al norte y los que están allá al sur. A los que están acá por siempre estar cerca y a los que están allá también.

Sobre todo quiero agradecer a mi familia. Mis tíos, los Pacos y su pasión por la ciencia. Edda, Silvina, Bicho, Felipe y Maga y su pasión por la pasión. A María y Joe, por su interés. Mis hermanas, Nadia y Ailén y Tomy, su cariño ayuda más que cualquier subsidio. Sobre todo a mis padres, a quienes agradezco su incondicionalidad. El simple hecho que a lo largo de este trabajo, mis alegrías han sido sus alegrías y mis tristezas han sido sus tristezas hace que, con justicia, esta tesis sea tan de ellos como mía.

Y último lo primero. A Joaquín y Lucas, quienes cada día me enseñan que en la vida hay cosas infinitamente más importantes que un buen clasificador. Es que hay algo considerablemente más duro que hacer una tesis y es soportar a una persona que está haciendo una tesis. Por eso a Yanina, quien me ha regalado paciencia, comprensión, amor, impulso, energía y, sobre todo, tiempo. Gracias bonita y espero saber devolverte a diario todo lo que me has dado y seguís dando.

Resum

El rendiment de la classificació estadística està directament relacionat amb una estimació acurada de les probabilitats condicionades a la classe. Aquesta estimació es degrada críticament a mida que creix la dimensionalitat. Es poden fer diferents assumpcions per tal de simplificar l'estimació, essent la parametrització la més freqüent. Un altre apropament al problema són bàsicament tècniques de reducció de soroll. En aquest cas, qualsevol cosa que no contribueixi positivament a la classificació és considerat com a soroll. Les tècniques d'extracció i selecció de característiques s'utilitzen per a aïllar aquest soroll, preservant només les característiques rellevants. En l'espai de dimensions reduïdes és possible una estimació més robusta de les probabilitats. Les tècniques de components principals, d'anàlisi discriminant o d'escalat multidimensional fan això.

Una tercera alternativa consisteix en assumir dependències no estadístiques entre les característiques, reduint l'estimació a problemes unidimensionals. Aquest tesi aprofundeix en aquesta tercera alternativa i la connecta amb la segona des de la perspectiva següent: si la independència estadística prova una assumpció massa restrictiva, utilitza una tècnica d'extracció de característiques on aquesta hipòtesi pot ser mantinguda amb més força. Aquest enfocament també es justifica des de la perspectiva de l'error de Bayes. Des d'una perspectiva estrictament teòrica l'extracció de característiques lineal només pot incrementar aquest error, aleshores la única bona raó per aplicar una transformació lineal a les dades és la certesa que aquesta beneficiarà la precisió de l'estimació de la densitat. Els resultats en el camp de la neurociència també donen suport a aquest mètode. El cervell necessita una representació amb una redundància oculta tan petita com sigui possible. Les probabilitats d'ocurrència dels objectes i dels esdeveniments haurien de tenir una dependència estadística mínima entre ells . S'ha observat que, per a imatges naturals, la representació obtinguda mitjançant Anàlisi de Components Independents s'assembla a la representació present en el còrtex visual primari dels mamífers (V1).Aquest resultat, un cop interpretat com a una reducció de redundància natural, pot ser vista des de la nostra perspectiva com una representació que simplifica l'estimació de probabilitats.

L'Anàlisi de Components Independents és una tècnica lineal d'extracció de característiques que busca minimitzar les dependències d'alt ordre. Quan les seves assumpcions es compleixen, es poden obtenir característiques estadísticament independents a partir de les mesures originals. Adaptem l'Anàlisi de Components Independents al problema de reconeixement estadístic de patrons de dades d'alta dimensionalitat. Això s'aconsegueix utilitzant representacions condicionals a la classe i un esquema de decisió de Bayes adaptat específicament. Degut a l'assumpció d'independència aquest esquema resulta en un modificació del classificador ingenu de Bayes. Els experiments es realitzen en un rang ampli de problemes clàssics de classificació en visió per computador, com ara el reconeixement d'objectes, aplicació de sensors remots i control de qualitat, etc.

L'inconvenient principal de les representacions condicionades a la classe és la seva incapacitat per aprendre les relacions entre les classes. Per tant també tractem el problema de la selecció de característiques dins d'aquesta representació. Un cop s'han extret les característiques s'introdueix un criteri per a seleccionar un subconjunt de característiques adequades per a la classificació. Adaptem el criteri de selecció de característiques de divergència a la nostra representació i les nostres assumpcions. L'Anàlisi de Components Independents obté components independents no Gaussianes. Això fa que la divergència sigui una elecció com a mesura de la separabilitat de classes. Aquesta mesura no fa cap assumpció sobre la distribució de les dades i ademés es simplifica considerablement sota l'assumpció d'independència. A més a més, mostrem no es requereix una cerca exhaustiva de característiques per a obtenir el subconjunt òptim de característiques

La solució que proposem pot ser entesa des d'una perspectiva completament diferent. L'extracció de característiques independents també pot ser interpretada com a l'extracció i selecció de característiques que millor serveixen al classificador ingenu de Bayes. Extenem aquesta línia de raonament a un context no paramètric. Donat un classificador mètric com pot ser el de k veïns més propers mostrem que buscar una tècnica lineal d'extracció de característiques òptima per aquest classificador resulta en una lleugera modificació de l'anàlisi no paramètric discriminant. Aquesta connexió està justificada per resultats teòrics i pràctics. Experiments clàssics de visió per computador com ara el reconeixement de cares i de gèneres il·lustren el potencial d'aquesta tècnica.

En tots dos casos, l'enfocament estadístic paramètric i el no paramètric tractem el disseny d'un classificador de patrons com un procés integrat. Un cop les assumpcions inicials s'han complets, els diferents passos com ara l'extracció de característiques, selecció i classificació es relacionen naturalment entre ells. Tots i cadascun dels passos estan justificats teòricament i íntimament lligats amb els altres.

Abstract

Statistical classification performance is directly related with an accurate estimation of the class-conditional probabilities. For fixed sample sets, this estimate severely degrades as dimensionality increases. Different assumptions can be made such that estimation is simplified. The most frequent assumptions are those that allow the parametrization of the problem. A second approach basically consists in noise reduction techniques. Anything that does not positively contribute to classification is regarded as noise. Feature extraction and selection techniques are used to isolate this noise, preserving only relevant features. In the reduced dimensional space, a more robust parametric or nonparametric estimation of the probabilities is possible. Principal components, discriminant analysis or multidimensional scaling techniques take this approach.

A third alternative is to assume no statistical dependencies between the features, estimation being reduced to unidimensional problems. This thesis will deepen into this last approach and connect it with the second from the following perspective: if statistical independence proves a too restrictive assumption, make use of a feature extraction technique where this hypothesis can be held on stronger grounds. This approach is also justified from a Bayesian error perspective. Since, from a strictly theoretical perspective, linear feature extraction can only increase this error, the only good reason for applying a linear transform to the data is the belief that this will benefit accuracy of density estimation. Results in the field in neuroscience also support this method. The brain needs a representation with as little hidden redundancy as possible. Probabilities of occurrence of objects and events should have minimum statistical dependencies between them. Results once interpreted as natural redundancy reduction, can be seen from our perspective as a representation that simplifies probability estimation.

Independent Component Analysis (ICA) is a linear feature extraction technique that aims to minimize higher-order dependencies in the extracted features. When its assumptions are met, statistically independent features can be obtained from the original measurements. A first insight and intuitive understanding of independent component analysis is presented using this technique as a feasible model for non-rigid shape variation. We then adapt independent component analysis to the particular problem of statistical pattern recognition of high-dimensional data. This is done by means of class-conditional representations and a specifically adapted Bayesian decision scheme. Due to the independence assumption this scheme results in a modification of the naive Bayes classifier. Experiments are made on a wide range of classical computer vision classification problems such as object recognition, remote sensing applications, quality control, etc.

The main disadvantage of class-conditional representations is that they fail to learn the relationship among classes. Therefore, we also treat the problem of feature selection within this representation. Once features are extracted a criterion is introduced for selecting a feature subset that best serves classification. We adapt the feature selection criterion of divergence to our class-conditional representation and assumptions. ICA obtains nongaussian independent components. This makes divergence an adequate choice since this class separability measure makes no assumption on the data distribution and is greatly simplified under the assumption of independence. Moreover, we show that no exhaustive search is required to obtain the optimal feature subset under our premises.

The solution we provide can be seen from a whole different perspective. The extraction of independent features can also be seen as finding a representation and then selecting those features that enhance the performance of the naive Bayes classifier. We extend this line of reasoning to a nonparametric context. Given a purely metric classifier such as the K-nearest neighbours (K-NN) we show that searching for a linear feature extraction technique optimal for this classifier results in a slight modification of nonparametric discriminant analysis. Theoretical and practical results justify this connection. Experiments in classical computer vision problems such as face and gender recognition illustrate the potential of this technique.

In both cases, the parametric statistical approach and the nonparametric metric approach, we treat the design of a pattern classifier as an integrated process. Once the initial assumptions are made the different stages such as feature extraction, selection and classification are naturally associated among each other. Each of the stages is theoretically justified and intimately linked with all the others.

Contents

List of Tables

List of Figures

Chapter 1

Statistical Classification of High Dimensional Data

1.1 Introduction

It is a widely spread notion that decision-making processes of a human being are related to the recognition of more or less complex patterns. For instance, when climbing a mountain the conditions of the environment in combination with our physical response determine the path to take, when buying a gift we might consider the addressee's taste or the budget on which we count, buying or selling stocks is decided by a complex pattern of information. Such pattern recognition acts, such as recognizing the face of a friend, searching the car keys in our suitcase, or naming a song from a few tones might seem simple to us, but are actually the result of astoundingly complex processes. This complexity becomes evident when we attempt to make machines emulate these procedures. The main goal of pattern recognition or machine perception is to clarify these complicated recognition mechanisms in decision-making processes and to automate these functions, using computers. It is needless to say that reliable pattern recognition by machines is an immensely useful task which, to give a few examples, includes speech recognition, optical character recognition, DNA sequence identification or waveform classification. In this thesis we focus on the classification [1] of visual patterns: the raw data defining the object we wish to classify is present in the form of a digital image. Examples of this problem can be, military target detection on aerial images, quality control (good-bad classification) from images taken on a production line, face recognition, land use classification on satellite images, etc.

For the classification stage, we will restrict ourselves to the statistical approach to the problem. This approach is derived from the assumption that the features used

[1] In general we will refer to the problem of classification as equivalent to the problem of recognition, preferring the first expression. Other related problems such as detection, identification or authentication can be seen as particular cases of classification: object detection in an image can be understood as deciding upon the class "object" versus the class "background".

to describe the pattern are random variables, so the behaviour of the random vector formed by all features can be described probabilistically. In visual recognition, the number of features is usually high, for instance the number of pixels in an image. So, for this case, the problem of statistical pattern classification can be considered as a problem of estimating density functions in a high-dimensional space and dividing the space into regions of categories or classes. For instance, imagine we wish to decide whether a 32×32 dimensional intensity image of a face corresponds to a male or a female subject. If we work with the raw intensity measurements, our feature vector has 1024 dimensions. The face is classified as belonging to a certain gender depending on which region of this high dimensional space fall the feature values. The regions have been defined from the distributions of both random vectors representing male and female subjects. This example is illustrated in Figure 1.1.

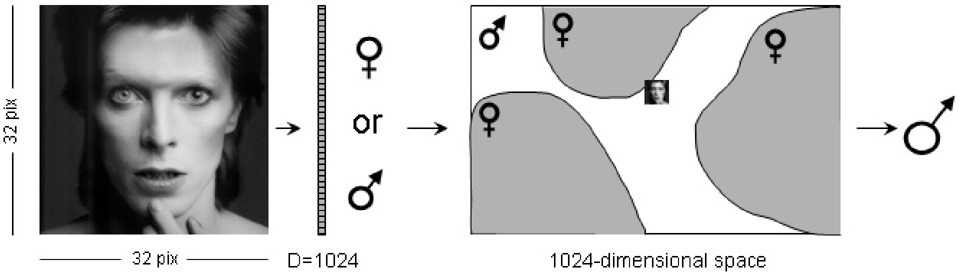


Figure 1.1: An example of visual recognition. In the problem of gender classification from 32×32 face images, samples live in a 1024D space. Gender regions in the space are determined from the distributions of male and female sample vectors. Regions determine class belonging.

So the first question we must answer is which is the best classifier, assuming the distribution of the random vectors is given. The answer to this question, in terms of probability of misclassification is the Bayes classifier or Bayesian decision theory. In this chapter we first introduce this theory as well as related concepts such as prior and posterior probabilities, likelihood and Bayes error. Since we focus on high dimensional data, the way in which dimensionality affects density estimation and consequently, Bayes classifier performance, is treated. A modification of the classifier, known as the naive Bayes classifier, is presented as a way to overcome this problem. We also explore the theory underlying naive Bayes in order to understand its capabilities and limitations. Finally we develop an example illustrating the theory introduced throughout the chapter, which consists in applying the naive Bayes classifier to the classical pattern recognition problem of face detection.

1.2 Bayesian Decision Theory

In general the problem can be stated as follows: Given a random feature vector $\boldsymbol{x}$, to classify it as belonging to one of K classes $C^1, C^2, \ldots, C^K$ [2] . To assume the distribution is known is equivalent to assume we know the probability of a class, given the outcome $\boldsymbol{x}$. For a class C^k this distribution, noted by $P(C^k|\boldsymbol{x})$, is named the *posterior* or *a posteriori* probability of the class. Although it is not the only alternative it seems natural to assign to $\boldsymbol{x}$ the class with maximum posterior probability. This assignment is known as the Bayes or Maximum A Posteriori (MAP) decision rule [38]:

$$C_{MAP} = \arg \max_{k=1\ldots K} P(C^k|\boldsymbol{x}). \tag{1.1}$$

Bayes' theorem provides an alternative expression for the posterior probability in terms of quantities which are often easier to estimate. Remember that this theorem for random variables x and y states that,

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}, \tag{1.2}$$

providing the following alternative expression for equation (1.1)

$$C_{MAP} = \arg \max_{k=1\ldots K} P(\boldsymbol{x}|C^k)P(C^k). \tag{1.3}$$

Notice that we dropped the expression $P(\boldsymbol{x})$ from the denominator since it does not depend on the class and therefore has no effect on the maximization. Actually, the purpose of this denominator in (1.2) is to ensure the left-hand term is actually a probability function, integrating to unity. $P(C^k)$ is called the *prior* probability and $P(\boldsymbol{x}|C^k)$ is referred to as the *class-conditional probability* or, when seen as a function of the parameters, the *likelihood* or *class-conditional likelihood.* In practice, the class-conditional probabilities are modelled, as we will see, parametrically or non-parametrically from our training set. The priors are usually based on previous knowledge we might have on class frequencies. For instance, if we were to classify face images according to subject gender, the priors should be based on the probabilities of finding a male or a female on the population of study. If no knowledge on the priors is available, equiprobable classes should be assumed and the second term in equation (1.3) can be dropped. In this case, the MAP decision rule is called the Maximum Likelihood decision rule or ML solution. It helps to read the objective function in (1.3) as: "The probability of $\boldsymbol{x}$ belonging to class C^k (or posterior probability) is proportional to the probability of generating x within C^k (or likelihood) weighted with the probability of class C^k (or prior)."

[2] The general notation and variable names convention used along this thesis is summarized in table (A.1) in Appendix A.

1.2.1 Example: Multivariate Gaussian Likelihoods

Probably no density function has received more attention than the Gaussian or normal density. An explicit expression of the multivariate Gaussian distribution is

$$N_{\boldsymbol{x}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\} \tag{1.4}$$

where $N_{\boldsymbol{x}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a shorthand notation, $\boldsymbol{\mu}$ the expected value or mean and $\boldsymbol{\Sigma}$ the covariance matrix, respectively. The positive definite quadratic function on $\boldsymbol{x}$ in the exponent is a very commonly used distance, in this case from a sample to the mean, called the Mahalanobis distance (d_{MAH}). The main property of this distance can be observed directly in the equation: the fact its geodesics are constant density contours for our feature vector.

The Gaussian distribution is an appropriate model for the situation in which feature vectors $\boldsymbol{x}$ for a given class C^k are continuous-valued, randomly corrupted versions of a single typical or prototype vector $\boldsymbol{\mu}^k$ [38], since this corruption is frequently Gaussian itself (Gaussian noise). Subscripted means ($\boldsymbol{\mu}^k$) and covariances ($\boldsymbol{\Sigma}^k$) will be used when class correspondence exists: the class-conditional likelihoods are assumed Gaussian. This density also arises when the observed random variables are a linear combination of many variables and the central limit theorem can be applied. Moreover, the normal density is analytically tractable and has a number of interesting properties. Firstly, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are sufficient to characterize the distribution uniquely. The marginal densities of a multivariate normal are all normal and the characteristic functions are also normal. Under any nonsingular linear transformation, the normal distribution becomes another normal distribution with different parameters. Parameters are usually estimated using maximum likelihood, obtaining the following estimates from samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n \tag{1.5}$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{n=1}^{N} (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})^T. \tag{1.6}$$

These estimates are called *sample mean* and *sample covariance matrix.*

It is important to notice that, despite its many advantages, normality should not be assumed without a good justification. More often than not this leads to meaningless conclusions [48].

Now imagine we have a two-class problem where normality can be safely assumed for each of the class-conditional likelihoods: $p(\boldsymbol{x}|C^1) \sim N_{\boldsymbol{x}}(\boldsymbol{\mu}^1, \boldsymbol{\Sigma}^1)$ and $p(\boldsymbol{x}|C^2) \sim N_{\boldsymbol{x}}(\boldsymbol{\mu}^2, \boldsymbol{\Sigma}^2)$. If we have equiprobable priors, the ML solution for this problem is said to assign a feature vector $\boldsymbol{x}$ to class C^1 if $N_{\boldsymbol{x}}(\boldsymbol{\mu}^1, \boldsymbol{\Sigma}^1) > N_{\boldsymbol{x}}(\boldsymbol{\mu}^2, \boldsymbol{\Sigma}^2)$, and to class C^2 otherwise. After replacing by (1.4), applying logarithms and performing a few algebraic operations we have that this inequality is equivalent to

$$\log|\boldsymbol{\Sigma}^1| + (\boldsymbol{x}-\boldsymbol{\mu}^1)^T {\boldsymbol{\Sigma}^1}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}^1) < \log|\boldsymbol{\Sigma}^2| + (\boldsymbol{x}-\boldsymbol{\mu}^2)^T {\boldsymbol{\Sigma}^2}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}^2). \tag{1.7}$$

The second term on both hands of the inequality are the Mahalanobis distances to the respective class means. If we neglect the first term, what this classifier suggests is to assign $\boldsymbol{x}$ to class C^1 if the mean of this class is closer to $\boldsymbol{x}$ than the mean of C^2 using the Mahalanobis distance. So our rule becomes to assign $\boldsymbol{x}$ to the class with the closest class mean using the Mahalanobis metric. We observe from this that the ML (or MAP) Gaussian classifier is very closely related to a metric and also quadratic classifier: the one that uses the Mahalanobis distance as a metric. Actually, if we would have assumed identity covariances, we have that our classifier is equivalent to the quadratic classifier that assigns to our sample the closest mean, using the Euclidean distance.

As was mentioned in the introduction, this decision rule divides our feature space into regions of categories and we should classify a sample according to where its feature vector falls. In Fig. (1.2) we illustrate these regions for an artificial bivariate two-class problem. We generate the samples for each class from a Gaussian distribution. In this case the parameters for the normal densities modeling each likelihood are known beforehand so we do not need to estimate them. They were chosen as: $\boldsymbol{\mu}^1 = [-0.6, 0]$, $\boldsymbol{\mu}^1 = [0.6, 0]$, $\boldsymbol{\Sigma}^1 = [[0.04, 0]; [0, 0.64]]$ and $\boldsymbol{\Sigma}^2 = [[0.34, 0.3]; [0.3, 0.34]]$.

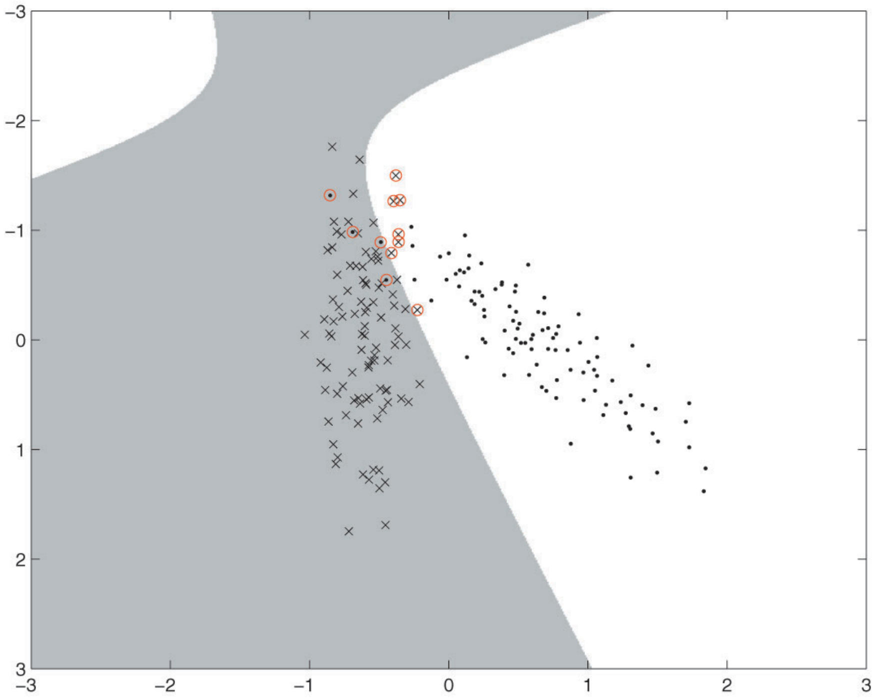


Figure 1.2: Decision regions as determined by the ML Gaussian classifier on an artificial Gaussian two-class problem (dimensionality=2). Misclassified samples are indicated with a circle.

1.2.2 The Bayes Error

The error of a classification rule is the probability of assigning a certain sample to the wrong class. For the two-class case and the Bayes Rule this error is the minimum posterior probability value of the sample. The Bayes error is the expected value of this minimum over all the samples. Of course many other choices of discriminant functions, instead of the posterior probability can be made, yielding different classifiers. But it can be seen that the Bayesian choice is the best in the sense it minimizes the probability of misclassification. Of course, if the Bayes error is zero and the distributions known, all samples should be classified correctly. The worst case occurs when posterior probabilities are equal for all classes. Since posterior probabilities add to one, this means that, for the two class case, the worst situation occurs when the Bayes error is equal to 0.5. If L^1 and L^2 correspond to the regions where a sample is classified as belonging to classes C^1 and C^2, respectively, then the Bayes error can be expressed as

$$\varepsilon = E\{\min p(C^1|\boldsymbol{x}), p(C^2|\boldsymbol{x})\} = p(C^1)\int_{L^2} P(\boldsymbol{x}|C^1)d\boldsymbol{x} + p(C^2)\int_{L^1} P(\boldsymbol{x}|C^2)d\boldsymbol{x}. \quad (1.8)$$

Extension to the multiclass case is straightforward. For the Gaussian case, the Bayes error can be obtained analytically. For instance, for the artificial problem if Fig. 1.2 its value is 0.059. This means that for this example, the most we can hope for, is to classify correctly roughly 94 out of every 100 samples. In general, the computation of the Bayes error is a very complex problem. This is due to the fact that ε is obtained by integrating high-dimensional density functions in complex regions. Nevertheless, its theoretical implications make it a powerful tool. For instance, if we design a classifier, relating its error with the Bayes error can give us a highly reliable perspective on its expected performance.

1.2.3 Feature Extraction

A necessary stage within the pattern classification scheme is to decide upon the way we represent our data, that is, the features we use. Basic features are those as given by the sensor itself, for instance in the case of images, the features are the intensity values captured on each cell of a charged couple device (CCD) or pixel values. Since these basic features frequently come from sensor measures, they are also named *measurements*. The space in which the measurements lie, is usually called *domain space*. Given the measurements, it is also common to further combine them in order to make them more *effective* for a specific purpose. Effectiveness can be understood as a tradeoff between accuracy and simplicity. Simple yet useful examples of this step can be the discretization of continuous feature values when the purpose might be storage, the elimination of redundant features when the purpose is to obtain a reduced representation in terms of dimensionality, the normalization of feature values when the purpose is to combine them on a single classifier.

Since the features are extracted or directly selected from the original ones, these techniques are called *feature extraction* or *feature selection* techniques. Feature extraction techniques are particular types of *learning techniques* since they learn new features from existing data. When these techniques take into account class labels of the samples, they are also referred to as *supervised* learning techniques since proper assignment of these labels requires supervision. When this prior knowledge is not included, they are called *unsupervised* learning techniques. The most common objectives for feature extraction are representation (features might provide a reduced representation with precise reconstruction) and classification (features are appropriate for representing and discriminating among classes) [48].

We can illustrate the importance of feature extraction through the example of a visual quality control system with the task of separating bad from good objects. In this case the measurements taken should depend on the type of image we consider appropriate for representing the objects. If it is going to be a range image, X-ray image, color or grayscale image, etc. If the choice were to be a 256×256 RGB color image with 256 levels per color, each object would be represented by a $256 \times 256 \times 3 = 196608$-dimensional vector, each vector component taking values between 0 and 255. This would be our domain space. Now imagine the goodness of a certain object had only to do with its color, or texture, some consideration on the shape, etc. In the first case, and having color consistency ensured, a histogram might be enough. If we, for instance, work with 8-bin histograms per color spectrum, we are now representing our objects with $8^3 = 512$-dimensional vectors. This new representation, might lose accuracy but is clearly simpler and more manageable. Additional prior information will surely provide even more effective features. If certain colors were known never to occur, or more subtly, if some colors were independent of the quality of an object, why include them in the histogram? By knowing these colors in advance or by, mathematically speaking, having learnt a subspace containing only those colors relevant to object quality, we can project the histograms in this subspace surely obtaining a considerable reduction of dimensionality. Also, since we are dealing with a classification problem, this last subspace can be mapped into another space where these relevant colors enhance their differences, making classification more robust and effective. In this coarse example we have extracted and chosen the feature *discriminated relevant colors*, among the features (or measurements) *pixel values*, *colors* and *relevant colors* as an appropriate representation for our objects. All the feature extraction process, mainly acquisition, histogram calculation, projection and mapping, can be done in a pre-processing stage, which leaves the data ready for the classification algorithm. We observe here how feature extraction can completely change the characteristics and complexity of a problem. We also observe how feature extraction obeys to generic postulates such as simplicity and dimensionality considerations, but is basically a problem-oriented process.

A very important family of feature extraction techniques arises when we restrict ourselves to linear transformations of our features, $\boldsymbol{y} = \boldsymbol{W}(\boldsymbol{x} + \boldsymbol{b})$, with $\boldsymbol{W}$ an $M \times D$ matrix and $\boldsymbol{b}$ a D dimensional vector. Some authors restrict linear transformations to the case in which $M = D$, otherwise referring to linear mappings or projections. Along this thesis, we will respect the strictly mathematical term where a linear transform can involve different dimensionalities. Many well known feature extraction techniques fall under this type and we will explore some of them along this thesis. On chapter 2 we will present three linear feature extraction techniques that focus on representation (and their application to classification). We will see how one of these techniques (Independent Component Analysis) can be used to enhance pattern classification in the case of high domain space dimensionality for a particular family of classifiers: those that assume statistical independence on the features. On chapter 5 we will present techniques that focus directly on discrimination and consequently on classification.

Since we deal mainly with the search of a linear transformation adapted to a given statistical classifier, two points should be considered. First, and if the transform is invertible, the distribution g of the (linearly) transformed data obeys the *change of variables theorem*,

$$g(\boldsymbol{y}) = \frac{1}{|\boldsymbol{W}|} f(\boldsymbol{W}^{-1} y) \tag{1.9}$$

where f is the distribution of the original features. In second place, it will be very useful to understand what happens to the Bayes error when any linear transformation is applied to our data. The result is quite simple: **the transformed data $\boldsymbol{y}$ will have a Bayes error greater or equal than the original data**. So in general information is lost and we have that $\varepsilon_{\boldsymbol{x}} \leq \varepsilon_{\boldsymbol{y}}$ [83]. Equality is achieved if $\boldsymbol{W}$ is an invertible transformation. This can be seen by substituting (1.9) in (1.8). At a first glance, this is a very disencouraging result, no matter how hard we search in the linear transform space we will never be able to transform our data and achieve a better discrimination than in domain space. This is true while we know the data distribution in domain space. A very unfrequent situation. In general, we are forced to estimate this distribution from a finite set of samples. These samples do not necessarily provide an accurate estimation, particularly as we shall see, when we are dealing with high dimensional data. It is from this perspective that we can understand the benefit of an adequate transformation. We can unsupervisedly choose a transformation that reduces dimensionality at low cost in representation such as Principal Component Analysis (in section (2.2) with the certainty that a more reliable density estimation can be achieved in the lower dimensional space. Or we can supervisedly choose a representation that also reduces dimensionality while seeking for class discriminability and not caring about the representation error such as Linear Discriminant Analysis (in section (5.2)). A third option can be to choose a representation that does not focus on dimensionality but, instead, attempts to simplify the estimation by restricting the plausible densities to a particular density family. This third case is the option we explore along this thesis.

1.2.4 Likelihood Estimation

In practice, the class-conditional distributions that appear in (1.3) are seldom known in advance and estimation is performed from a finite set of samples $\boldsymbol{X} = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N^k}$, with N^k the total number of samples in class C^k with $k = 1, \ldots, K$. A spread out taxonomy of approaches to likelihood estimation is to classify techniques according to their parametric or nonparametric nature [16]. In this section we briefly outline the main properties of these approaches.

Parametric Techniques

A simple method for density estimation consists in restricting the probability density to a specific functional form which contains a number of functional parameters. As we mentioned, the simplest, and most widely used parametric model is the Gaussian distribution, but many other parametrical models apply. Once we have determined the parameters necessary for modelling our distribution, the values of the parameters are optimized to give the best fit to the data using techniques such as *maximum likelihood* or *Bayesian Inference*. For instance, in (1.6) we expose the maximum likelihood solution for estimating mean and covariance which are, as we said, all the parameters needed for uniquely identifying a Gaussian distribution. Since maximum likelihood estimation will be encountered more than once along this thesis, we will now state its main characteristics [1].

Suppose we consider a density function $p(\boldsymbol{x})$ which depends on a set of parameters $\boldsymbol{\theta} = \theta_1, \dots, \theta_L$ (class-conditional densities being a particular case of this problem). To make dependence on the parameters explicit, we can write the density function as $p(\boldsymbol{x}|\boldsymbol{\theta})$. If our dataset $\boldsymbol{X}$ consists in N samples drawn independently from the distribution $p(\boldsymbol{x}|\boldsymbol{\theta})$, then the joint probability of the whole dataset is given by

$$p(\boldsymbol{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) \tag{1.10}$$

where $\mathcal{L}(\boldsymbol{\theta})$ is referred to as the *likelihood* of $\boldsymbol{\theta}$ for dataset $\boldsymbol{X}$. Maximum likelihood then sets the parameter values by maximizing $\mathcal{L}(\boldsymbol{\theta})$. For most choices of density function, the optimum has to be found by an iterative numerical procedure such as gradient descent, Newton's method, the expectation maximization (EM) algorithm , etc. [95, 33]. In practice, it is generally more convenient to consider minimization of the negative logarithm of $\mathcal{L}$ for the optimization. Since the negative logarithm is a monotonically decreasing function, results are equivalent. In addition, the product in (1.10) becomes a sum and the negative log-likelihood can be regarded as an error function.

The main drawback of parametric estimation techniques relies on the fact that the particular form of parametric function chosen might fail to account for the eventual complexity of the true density. So we might want to consider estimation techniques that prevent us from imposing incorrect prior assumptions such as those used for parametric estimation.

Nonparametric Techniques

Nonparametric density functions are those for which the functional form is not specified in advance (no prior knowledge is assumed), but which depend on the data itself. The most simple and intuitive of these techniques are histogram-based methods, which make use of frequential probability techniques. The histogram is obtained by dividing the feature space into a number of bins and assigning a probability to each bin, depending on the number of data points that fall within. Once the histogram has been constructed, data can be discarded and only information on bin sizes, location and frequencies retained. In general, if V_R is the volume of a certain bin R, N_R the number of samples falling within the bin and N the total number of samples, by using histograms, we approach the probability of any value $\boldsymbol{x}$ that falls in R by,

$$\tilde{p}(\boldsymbol{x}) = \frac{N_R}{NV_R}. \tag{1.11}$$

The volume of the bins (bin-width in the case of unidimensional data) acts as a very sensitive smoothing parameter so special care should be taken when performing this choice. Several solutions have been proposed to this problem, by expressing the optimal volume in terms of the size of the dataset and dimensionality. Solutions should consider a desirable behaviour in the limit case, such that $\tilde{p}(\boldsymbol{x})$ converges to the true distribution $p(\boldsymbol{x})$ when an unlimited amount of data is available. This happens if, for instance, the volume is specified as a function of N. A common choice for the unidimensional case is to set $V_R = 1/\sqrt{N}$.

It can be seen from (1.11) that the main drawback from this technique is that the estimated density function is not smooth but has discontinuities at the boundaries of the histogram bins. This fact has direct relationship with the number of available samples and the space dimensionality. In a D-dimensional space, if each variable is divided into J intervals then the whole space will be divided into J^D bins. This exponential growth with D also called, the *curse of dimensionality*, will be discussed later in this chapter.

A second approach is the *kernel function* or *Parzen window* approach [112]. This approach as well as its extensions consists, instead of arbitrarily binning the feature space, in summing the contribution of a predefined kernel (or window) functions $H(\boldsymbol{x}, \boldsymbol{x}_n, h)$ on each available sample $\boldsymbol{x}_n$. If the functions satisfy positivity and integrate to one, then it can be seen that the function

$$\tilde{p}(\boldsymbol{x}) = \frac{1}{N}\sum_{n=1}^{N} \frac{1}{V(h,D)} H(\boldsymbol{x}, \boldsymbol{x}_n, h) \tag{1.12}$$

is actually a density function, where h defines the width of the neighborhood surrounding the sample, and $V(h,D)$ is the volume of this neighborhood. The shape of the kernel is predefined and the two most common options are to choose a hypercube kernel (Parzen window), or a Gaussian kernel. The problem of selecting the width (h) is the same as with the histogram approach. Along this thesis, all density estimations using kernel methods are performed with Gaussian kernels. In this case

$$H(\boldsymbol{x}, \boldsymbol{x}_n, h) = \exp\left(-\frac{1}{2h^2}(\boldsymbol{x} - \boldsymbol{x}_n)^T(\boldsymbol{x} - \boldsymbol{x}_n)\right) \tag{1.13}$$

and

$$V(h,D) = (2\pi h^2)^{D/2} \tag{1.14}$$

is the volume. Unless stated otherwise, when working with the kernel approach, we also select the kernel width as $h = [\frac{4}{N}(D+2)]^{\frac{1}{(D+4)}}$ as suggested in [136].

The kernel approach presents the same problems as the histogram approach in the presence of high dimensionality, though the smoothing effect imposed by the presence of the kernel function attenuates discontinuities.

The last nonparametric approach we will mention results from relaxing the fixed width parameter for all data points. Remember that in (1.11) we suggested an adequate volume based on the amount of available data. Instead, we can fix the number of data points falling within a bin (K) and allow the volume V to vary in order to contain K samples. This is called the *K-nearest neighbours* (K-NN) approach [42]. The main advantage of this approach is that it can be used both for density estimation as well as for providing a direct classification rule. Suppose our dataset consists in N^k samples per class, so that $\sum_k N^k = N$ and that we choose a neighborhood with volume V around a sample $\boldsymbol{x}$ that contains K points, K^k of these points belonging to class C^k. Then, from (1.11) we have that the class-conditional density can be approximated by [30]

$$\tilde{p}(\boldsymbol{x}|C^k) = \frac{K^k}{N^k V}, \tag{1.15}$$

while the unconditional density can be similarly estimated from

$$\tilde{p}(\boldsymbol{x}) = \frac{K}{NV} \tag{1.16}$$

and the priors from

$$\tilde{p}(C^k) = \frac{N^k}{N}. \tag{1.17}$$

Using Bayes' theorem we can obtain the posterior density (1.2)

$$\tilde{p}(C^k|\boldsymbol{x}) = \frac{K^k}{K}. \tag{1.18}$$

Thus, if we want to minimize the probability of misclassification using these approximated densities we should assign $\boldsymbol{x}$ to the class for which the ratio K^k/K is largest. This traduces in assigning to a sample the class label majoritary among its K-nearest neighbors. The distance of choice will affect the shape of the neighborhood, for instance, choosing the Euclidean metric is equivalent to choosing hyperspheric neighborhoods, the L^∞ metric is equivalent to choosing hypercubic neighborhoods, etc.

K-Nearest Neighbors (K-NN) [42] has for long been probably the most intuitive classification rule one can think of. I know many people who sort their CD collection using K-NN. Not that they are aware of it when they say, here is a couple of jazz records so let us place this other one, which is also jazz, in the same shelf. When, in 1967, Cover and Hart [29] showed that this simple technique has an asymptotic error rate on the number of samples of at most twice the Bayes rate, the K-NN rule received its major boost, its popularity spread and many modifications of the rule arose since then [30]. Additionally, despite we can no longer ensure its theoretical properties, this rule can be applied to datasets with small sample size and high dimensionality, often impassable obstacles for most pattern classification techniques.

Nonparametric models have another disadvantage, besides their strong dependence on dimensionality. For these models, the number of variables involved grows directly with the number of samples in the training dataset, while this number was fixed in the case of parametric estimation. This also causes a strong decrease in the speed of evaluation of new input vectors.

Semiparametric Techniques

This third approach arises as an attempt to overcome the main drawbacks of the techniques presented in the two previous sections through the combination of the advantages of parametric and nonparametric methods. The price to pay is a computationally intensive process of setting up the model using the training dataset. We focus on the semiparametric approach given by one particular form of density function, called a *mixture model* [149]. A mixture model is a density function formed from a linear combination of basis functions. Actually, the kernel approach (1.12) can be seen as an extreme form of mixture model where the number of basis functions is equal to the number of points in the dataset. The difference is made by choosing J basis functions, with $J << N$ usually. We can therefore write our model for the density as a linear combination of parametric component densities $p(\boldsymbol{x}|j, \boldsymbol{\theta}_j)$ in the form

$$p(\boldsymbol{x}) = \sum_{j=1}^{J} p(\boldsymbol{x}|j, \boldsymbol{\theta}_j)p(j). \tag{1.19}$$

The coefficients $p(j)$ are called the mixing parameters or prior probabilities, since they state the probability of generating any data point from the j-th component of the mixture. These priors should be zero or positive and add to one. The component densities should actually be a distribution, being positive and integrating to one, in order to ensure the final linear combination is also a density. A very important property of these mixture models is that, for many choices of component densities, they can approximate any continuous density to arbitrary accuracy, provided the parameters (J and $\boldsymbol{\theta}_j$) are correctly chosen. Such is the case, when the most common choice of component densities is made: that they are given by Gaussian distributions functions. Many other choices apply, for instance, later in this thesis we will explain and work with unidimensional mixtures of double-exponential functions.

For the case of mixture models, maximum likelihood estimation of the parameters proves complex and suggests the need of an iterative scheme for finding the solution [16]. This iterative scheme can be found and justified by adapting the more general procedure known as the expectation-maximization or EM algorithm [33]. The objective of this algorithm is to produce maximum-likelihood estimates of parameters. This is achieved through a two step algorithm. An expectation step respect to the unknown distribution using the current estimate of the parameters and conditioned on the observations. And a maximization step which provides a new estimate of the parameters.

Finally, we will illustrate the different approaches on density estimation through the simple problem of estimating a unidimensional uniform distribution in the interval, assuming we have 100 samples in our training set. Figure (1.3) plots the results of applying the following methods: (a) nonparametric histogram-based approach with ten bins; (b) parametric Gaussian approximation; (c) and (d) nonparametric nearest neighbors approach considering 5 and 20 neighbors for the estimation, respectively; (e) nonparametric Gaussian kernels using $h = 0.1$ as the scale factor (equivalent to the standard deviation) on each Gaussian; (f) A mixture model of two Gaussians estimated using the expectation maximization algorithm. We can roughly observe the differences between the different approaches. Of course, each of these approaches can greatly vary from the illustrated results, simply by choosing other parameters, i.e. more components in the mixture or less bins in the histogram. So it is not advisable to extrapolate these results to a general case.

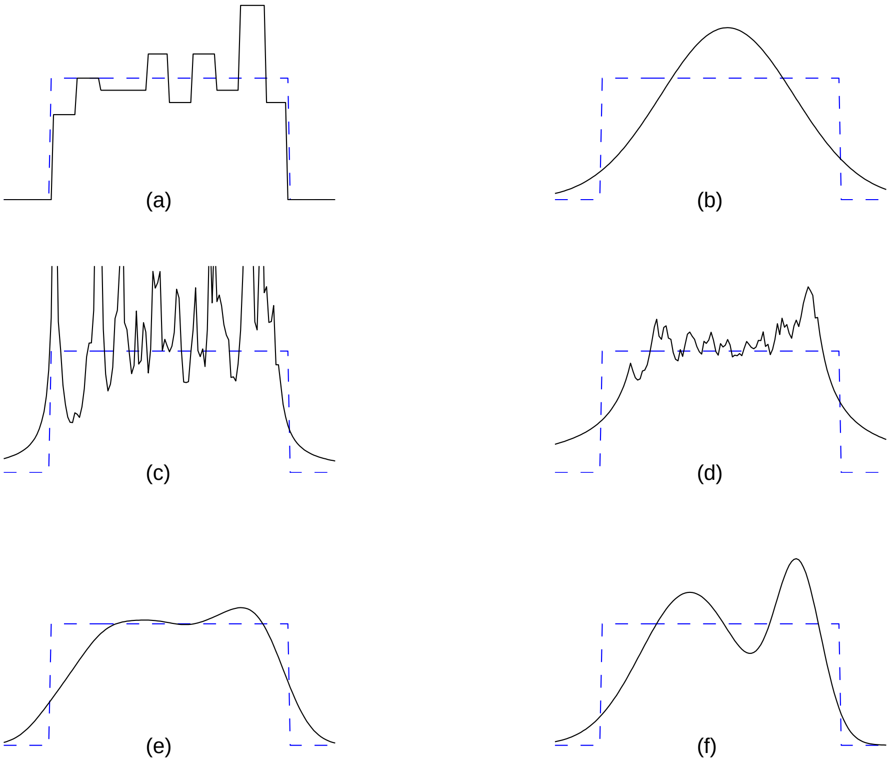


Figure 1.3: Different density estimation approaches applied to the estimation of a uniform distribution: (a) 10-bin histogram, (b) parametric Gaussian, (c) 5-nearest neighbors, (d) 20-nearest neighbors, (e) Gaussian kernels, ($h = 0.1$), (f) mixture of two Gaussians.

1.2.5 High Dimensional Data

While discussing nonparametric density estimation techniques we ran into the following situation, also known as *curse of dimensionality* [15]: Given a D dimensional space and dividing each feature value into J possible intervals, the resulting number of cells in the feature space is J^D, i.e. the number of cells grows exponentially with the dimensionality. This fact makes nonparametric approaches such as the histogram approach practically unfeasible on spaces with dimensionality larger than 3. Of course, dividing feature space into such cells is a particularly inefficient way of representing our multivariate density functions. Nevertheless, this extreme example helps to understand how high dimensionality affects negatively density estimation.

The properties of high dimensional space often appear counterintuitive because of our experience with the physical world in a low-dimensional space. The main problem is that objects in high dimensional spaces have a larger amount of surface area for a given volume than objects in low-dimensional spaces [23]. We will now enumerate four properties of high dimensional distributions which clearly illustrate the atypical qualities that contribute to this problem [46]:

- *Sample sizes yielding the same density increase exponentially with dimension.* As mentioned in the previous paragraph.

- *A large radius is needed to enclose a fraction of the data points.* Very large neighborhoods are required to capture even small portions of the data, making it difficult to make local estimates for high-dimensional samples.

- *Almost every point is closer to an edge than to another point.* Almost all points lie on the distribution boundary with the consequence this might have on classification.

- *Almost every point is an outlier in its own projection.* From almost every point, the rest of the points, though they belong to the same distribution, look like a faraway cluster.

The inconvenients with high dimensional data are also present in parametric estimation. It has been proved [48] that the required number of training samples is linearly related to the dimensionality for a linear classifier, and to the square of the dimensionality for a quadratic classifier. On the other hand, we mentioned in section (1.2.2) that any feature selection (which can be seen as a particular linear feature transform) causes an increase in the Bayes Error. Of course this last fact affects exclusively the infinite sample case. So on one hand we have that, for a fixed training size, estimation accuracy decreases as dimensionality increases. On the other hand, class separability increases together with the number of features. The combination of these two results in what is known as the *peaking* or Hughes phenomenon [61] which states that classification accuracy first grows and then declines, as dimensionality increases.

Solutions to this problem are mainly focused on data preprocessing (for instance, filtering) or particular feature extraction/selection techniques which attempt to reduce dimensionality. In the next section we will see an alternative solution which takes whole a different path by imposing restrictions on the family of estimated densities.

1.3 The Naive Bayes Classifier

The naive Bayes classifier [38] results from introducing the assumption that features are statistically independent given the class in (1.3). Statistical independence is a tricky subject which we will examine in depth in the next chapter. For our purpose, which is understanding the naive or *simple* Bayes classifier, the definition of conditional independence will be sufficient. This definition states that features can be considered independent given the class if the joint (conditional) density can be marginalized into the product of the unidimensional densities corresponding to each feature variable. That is

$$p(\boldsymbol{x}|C^k) = \prod_{d=1}^{D} p(x_d|C^k) \forall k = 1 \dots K \tag{1.20}$$

By replacing this equation into (1.3), and assuming equiprobable classes, we can state the naive Bayes solution to the problem of classification,

$$C_{NB} = \arg \max_{k=1\dots K} \prod_{d=1}^{D} p(x_d|C^k) \tag{1.21}$$

In practice, attributes (features) are seldom independent given the class, which is why this assumption is termed as "naive". Despite this unrealistic hypothesis, the naive Bayes classifier is frequently used in practice. Not only are its results comparable to much more complex classifiers but it also has serious advantages in terms of learning speed, classification speed (computational efficiency), storage space and incrementality. Moreover, its statistical nature implies interesting theoretic properties in terms of modeling and predictability.

As an illustrative example let us, consider the multivariate Gaussian with uncorrelated variables. Uncorrelatedness traduces into a diagonal covariance matrix, so for this particular case, $\boldsymbol{\Sigma} = diag[\sigma_1^2, \dots, \sigma_D^2]$ with σ_d the standard deviation of the random variable x_d. Introducing this assumption into (1.4) we have, through simple algebraic operations, that the multivariate distribution of a random and uncorrelated Gaussian vector is:

$$N_{\boldsymbol{x}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\prod_d \sigma_d|^{1/2}} \exp\{-\frac{1}{2}\sum_{d=1}^{D}\frac{1}{\sigma_d}(x_d - \mu_d)^2\} = \prod_{d=1}^{D} N_{x_d}(\mu_d, \sigma_d^2) \tag{1.22}$$

This is precisely the necessary and sufficient condition for independence, so we have shown that, at least for the Gaussian case, uncorrelatedness implies statistical independence. Actually, the Gaussian is the only distribution where this statement holds so extending this result to any other distribution is generally wrong. Nevertheless, the reverse is true always: independence implies uncorrelatedness. This is not a very useful implication since uncorrelatedness is, as we will see in the next chapter, much easier to achieve than independence. The Gaussian case also illustrates how the independence assumption can simplify the parametric approach to density estimation. If full (correlated) Gaussians are assumed on each class, then the total number of parameters we have to estimate is $K(D(D+1)/2+D)$, quadratic on the dimensionality. If instead, variable independence is assumed, the total number of parameters needed reduces to $2KD$, linear on the dimensionality.

From the perspective of pattern classification we have just seen that imposing the parametric prior of assuming conditionally uncorrelated Gaussian distributions is equivalent to using the naive Bayes classifier (1.21). We can see the effect this has on the artificial example shown in Fig. (1.2). For this example imposing uncorrelatedness implies that the means remain the same and, for C^1, also does the covariance matrix since it is already diagonal. For C^2 the new, diagonal, covariance matrix is $\mathbf{\Sigma}^2 = [[0.34, 0]; [0, 0.34]]$. Once again and quite exceptionally, the classification error can be calculated analytically yielding a value of 0.16, which is logically higher than the Bayes error (0.059). Figure (1.4) illustrates the decision regions of this rule together with the misclassified samples for a set of 100 samples drawn per class. The effect of assuming class separability can be observed in this new and less accurate decision region: Decision regions given by the naive Bayes classifier, result from the Cartesian product of unidimensional decision regions.

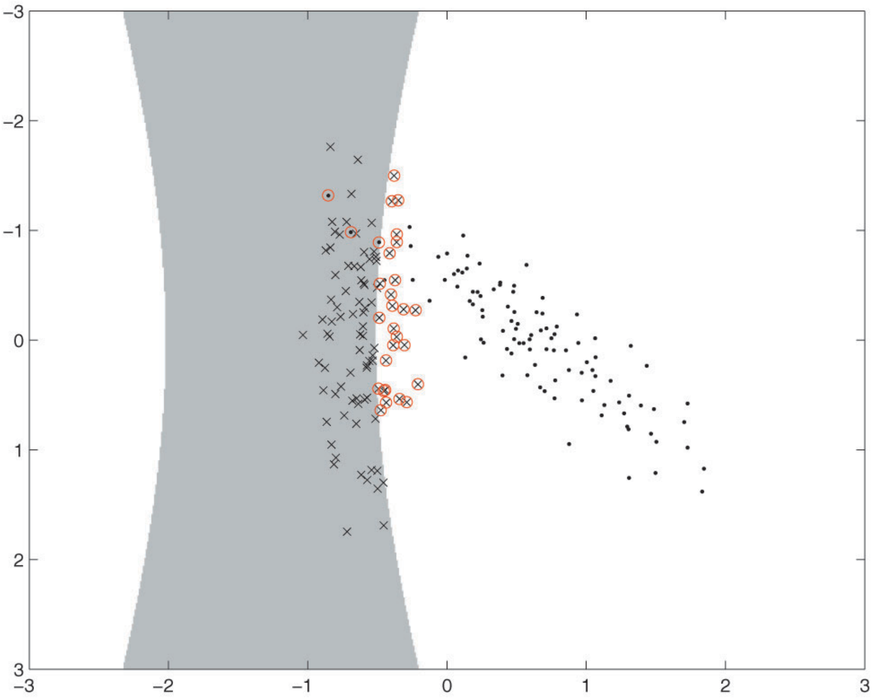


Figure 1.4: Decision regions as determined by the naive Bayes classifier which results of assuming uncorrelated Gaussians as conditional likelihoods, on an artificial Gaussian two-class problem (dimensionality=2). Misclassified samples are indicated with a circle. The data is exactly the same as in Fig. (1.2).

1.3.1 Remarks on Naive Bayes

To this point, we can conclude that the naive Bayes classifier, besides its simplicity and computational efficiency, can be a useful classifier for high dimensional data since it transforms a D-dimensional density estimation, subject to the inconvenients exposed in section (1.2.5), into the estimation of D unidimensional densities. We have also seen that, for the conditionally Gaussian case, application of naive Bayes is equivalent to assuming diagonal class-conditional covariance matrices. But the most important question still remains. Is, in general, naive Bayes a good classifier? If the independence assumption holds: it is the best classifier one can get. But, if this assumption is not met, is it still competitive?. And also, if we can force by any way this assumption, for instance, by projecting the data into a space where it is actually met, can we expect to improve its performance? While the answer to this last question will be delayed to chapter 3, we will now refer to the results in several works that have addressed the general efficiency of naive Bayes.

For years, the most common use of the naive Bayes classifier has been to appear in classification benchmarks outperformed by other, generally more recent, methods. Despite this fate, in the past few years this simple technique has emerged once again, basically due to its results both in performance and speed in the area of information retrieval and document categorization [105, 91, 161]. Since then, it has been successfully applied to several classification problems coming from real-world data such as EEG signal classification [81], information filtering [114], performance management [82], statistical diagnosis [58] or relevance feedback within the area of information retrieval [127]. Naive Bayes has been also applied on artificial datasets and compared with a wide range of approaches [85, 101, 24, 47].

Probably, the most significative and methodic research performed in this sense is the one published by Domingos and Pazzani [36]. In this work, the authors first solve this question heuristically, by setting up a large experiment composed of 28 datasets taken from the UCI repository [18] and comparing naive Bayes performance with other usual approaches for classification learning: the Bayesian classifier with full Gaussian distribution on the class-conditional densities, decision tree induction, instance-based learning and rule induction. Results show that naive Bayes is significantly more accurate than all other classifiers, despite there is no prior knowledge on the independence of the features. This result should be carefully considered since a restricted scope of classifiers is considered. Moreover, for most of the databases in this experiment there exists a classifier, other than those evaluated, that yields considerably better results. From this that the importance of this result should be understood in terms of overall performance. Also in this work, a statistic of feature dependence is proposed and the same datasets are analysed in order to determine the relation between this statistic and naive Bayes performance. Results show that naive Bayes performs well even when a high dependence among features is observed. Results are taken further to artificial cases showing that, at least under a zero-one loss function, naive Bayes clearly does not require attribute independence. An important observation is that optimality in terms of this classification error is not necessarily related to the quality of the fit to a probability distribution but to the chances the actual and estimated distribution agree on the most probable class.

Rish takes this article a step further [126] generalizing some of the concepts introduced by Domingos and focusing on concrete data characteristics that affect naive Bayes performance. They focus on the *bias* of the classifier instead of its *variance*: the bias examines the classifier performance assuming an infinite amount of data and compares this with the optimal performance given by the Bayesian classifier; the variance, instead, deals with the robustness of the classifier with respect to training sample set size (discrete). Most importantly, they show that naive Bayes reaches optimal performance in two extreme cases: completely independent features, as expected, and, more surprisingly, functionally dependent features. They also show that the product of marginal distributions approximate better the joint distribution in the case of low feature entropy. A result relevant to us, as we will see in chapter 3.

Both Rish and Domingos aim to underestimate the independence assumption and thus extend the applicability of naive Bayes. Another approach to overcome the restriction imposed by the independence assumption consists in relaxing this hypothesis via a modification of the classifier [153] through dependence models. Even though results are interesting, they are seldom used in practice. A third approach is yet possible. To act over the representation in order to allow the independence assumption to hold on stronger grounds. This thesis follows this last approach.

1.3.2 Example: Naive Bayes on High Dimensional Discrete Data

This example will try to illustrate the theory exposed throughout this first chapter. We will refer to a typical pattern recognition problem which the human visual system has solved quite acceptably but still remains challenging in the field of machine vision. We will decide over a statistical classifier adequate for the features we choose to represent our patterns. We will estimate the conditional densities, and we will finally see how it performs.

We now examine the performance of the naive Bayes classifier on the classical problem of *face detection*. The main goal of face detection is to determine whether there are any faces in an image and, if present, to return the location and scale of the face within the image. For a complete and actual survey on face detection, its challenges and solutions, refer to Yang [160]. One of the main characteristics of most detection problems, and this one is no exception, is the fact that we wish to detect a single object or family of objects from a possibly very wide variety of counterexamples. In our case, these counterexamples are any image that is not a face. That is to say, almost any image. The naive Bayes classifier has already been (successfully) tested in this problem and more generally in the problem of object detection, but from a completely different approach, at least in the selection of features representing our pattern (faces) [133].

From the classification perspective we have two classes, faces (C_F) and nonfaces (C_{NF}), and the fact this second class has such a huge number of examples makes it a delicate issue to determine statistically significant training and test sets and to fix a proper classifier evaluation procedure. On the first inconvenient, we will consider using quite simple datasets which, even though they could be considered as toy datasets from within the face detection community, will do for our purpose of giving an example. We will choose 7000 faces for training, 5000 faces for evaluation, and 100000 images not containing faces. The number of faces is acceptable, but if we take into account that state of the art detectors achieve an order of one false positive (a nonface mistakenly considered a face) every 100000 [160] images, in order to detect around 94% of the evaluation faces, we will understand the limitations of this example. Simple as it results, guidelines for extending our classifier to become a robust detector are given at the end of the example. We could have simplified even more our approach by directly neglecting the nonfaces in the training stage, and modeling only the face class. Actually, this was the approach taken by the first object detectors, and their results quite unsuccessful in practice. Face detectors are usually evaluated not only in terms of accuracy, since this problem is a clear example where, depending on the application, a false positive should have a different weight than a false negative. So all results will be given in terms of true positives and false positives.

All images are scaled to 32×32 pixels so $D = 1024$, so our samples are indeed high dimensional data. Two of the major challenges in face detection are robustness to pose and illumination. We will restrict ourselves to frontal images, with slight rotations (never more than 10 degrees) and small variations in scale (5% at the most). Instead, we will confront the problem of strong variations in the illumination through the proper choice of a feature set: instead of using grayscale values of the images, we will use the result of extracting the image's ridges and valleys. This process can be understood as a preprocessing filter on every image. Ridges and valleys have proved robust to illumination as well as resulting in interesting features for recognizing, detecting and matching faces [122]. The method we will use for detecting such features is the MLSEC-ST operator as presented in [94]. Features are finally obtained by thresholding this response, and assigning a -1 to the pixel where a valley is present, a 1 to a ridge, and 0 if there is neither a ridge nor a valley. The fourth case, both ridge and valley present, never occurs. So our final feature values are actually ternary: $-1, 0$ or 1. In the experiments, parameters of the detector were not tuned at all and set as 1.5 for integration and derivation scales. We also need to decide over the threshold so we decided that any value below -0.95 would be a valley and any value above 0.95 a ridge. In Fig. (1.5) we can observe 4 faces used in the dataset together with 4 nonfaces, as well as their response to this operator.

We can still simplify our scheme by using binary instead of ternary features. This can be easily done separating the responses into two images, one corresponding to the ridges and another to the valleys (1 if there is a ridge/valley, 0 otherwise). Images are vectorized and concatenated so actually our feature vector has double dimensionality ($D = 2048$), a yet even higher dimensional space. Notice, that the ternary representation is a linear combination of reduced dimensionality but without loss of information, of these binary representations, since it is the result of subtracting the valleys from the creases.

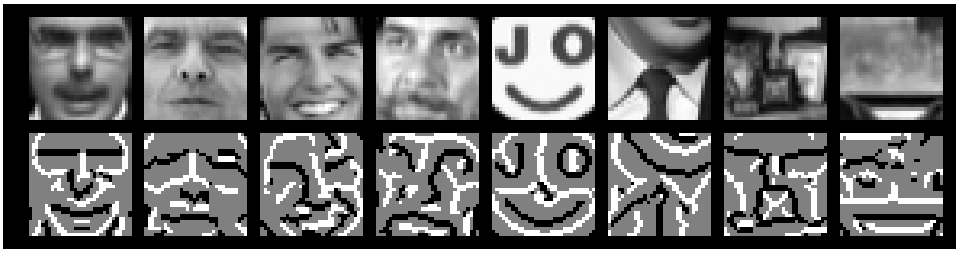


Figure 1.5: Sample 32 × 32 faces and nonfaces used to train and evaluate our classifier. In the bottom row the ridges (white) and valleys (black). Nonfaces shown are particularly confusing to illustrate the difficulty of the problem.

We now adapt the naive Bayesian classifier to binary data. This particular case of naive Bayes has been extensively used in the field of text categorization to the extent it receives its own name: Binary Independence Model or BIM [91]. Remember that for naive Bayes we need only to model unidimensional densities, and since data is binary the *Bernoulli* distribution is a reasonable parametric assumption on the data. This distribution is defined as followed:

$$p_{Ber}(x) = p^x (1-p)^{1-x} \tag{1.23}$$

Notice that, for the Bernoulli $p_{Ber}(1) = p$ and $p_{Ber}(0) = 1 - p$, so p is actually the probability of x being 1 and can be easily estimated from the dataset just by finding the frequencies of this value. In our case, it will be the frequency with which a valley or a ridge are found at a certain pixel position in our dataset. We note as p_d with $d = 1 \ldots 2048$ this probability for every pixel (every feature) of both concatenated images obtained from the face training set and $\tilde{p}_d$ the same probability for nonfaces. We can still simplify the notation by defining $q_d = 1 - p_d$ and $\tilde{q}_d = 1 - \tilde{p}_d$. By introducing (1.23) into (1.20), we have that combining the marginal probabilities for the faces

$$p_{Ber}(\boldsymbol{x}|C_F) = \prod_{d=1}^{D} p_d^{x_d} q_d^{1-x_d} \tag{1.24}$$

and for the nonfaces,

$$p_{Ber}(\boldsymbol{x}|C_{NF}) = \prod_{d=1}^{D} \tilde{p}_d^{x_d} \tilde{q}_d^{1-x_d}. \tag{1.25}$$

Now that we have defined the conditional likelihoods, we can move on to the classifier. Remember from (1.21) that we decide that sample $\boldsymbol{x}$ is a face if $p_{Ber}(\boldsymbol{x}|C_F) > p_{Ber}(\boldsymbol{x}|C_{NF})$. Since the logarithm is a monotonic increasing function we can rewrite this inequality by applying log-likelihoods instead of likelihoods. So we will classify $\boldsymbol{x}$ as a face if,

$$\sum_{d=1}^{D} x_d \log p_d + (1 - x_d) \log q_d > \sum_{d=1}^{D} x_d \log \tilde{p}_d + (1 - x_d) \log \tilde{q}_d. \tag{1.26}$$

This equation can be further simplified through some algebraic operations. For instance, we can separate the terms that affect the random variable from those who don't,

$$\sum_{d=1}^{D} x_d \log \frac{p_d \tilde{q}_d}{\tilde{p}_d q_d} + \sum_{d=1}^{D} \log \frac{\tilde{p}_d}{\tilde{q}_d} > 0, \tag{1.27}$$

to find out that if we define $w_d = \log(p_d \tilde{q}_d / \tilde{p}_d q_d)$ and $b_d = \sqrt{(\log \tilde{p}_d / \tilde{q}_d)}$ our Binary Independence Model is nothing more than a linear classifier since we can express (1.27) as

$$\boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}^T \boldsymbol{b} > 0 \tag{1.28}$$

Using this classifier as it is does not allow us to fix the number of detected faces to 95% as we wished. This can be done by determining a constant λ from the training set, such that 95% of the training faces satisfy that $\boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}^T \boldsymbol{b} > \lambda$. For the strictly Bayesians, this constant can be easily derived from the naive Bayes rule if we would have included class priors and would exactly be the logarithm of the ratio between such priors.

The final results we achieve is a detection rate of 94.7% on the evaluation faces and 1.282% of false positives. This false positive rate was obtained from the evaluation of 1560600 nonface images different from those in the training. This is not impressive when compared with state of the art detectors but we should consider that we are using a linear classifier on binary high dimensional data. But we can do a little bit to improve this result, through careful observation of the estimated likelihoods. Since there is one marginal probability for each pixel, we have a probability image for ridges, and another for valleys. Values close to one or close to zero on each of these images mean a high/low presence of ridges (valleys) at a certain pixel and a consequently the presence of a pixel more capable of describing a face. For instance, a pixel x_d with probability $p_d = 0.9$ says that 90% of the faces present a ridge (valley) in that position, so it is a highly discriminative pixel. In Fig. (1.6) we illustrate the obtained probability images for creases and valleys, estimated from the training faces and nonfaces, respectively. The faces probabilities are what we might have expected from the location of creases and valleys in any given face. For the nonfaces, since these are practically any image, the probability of a ridge or a valley appearing in a position can be compared to the probability of finding a ridge or valley in any point

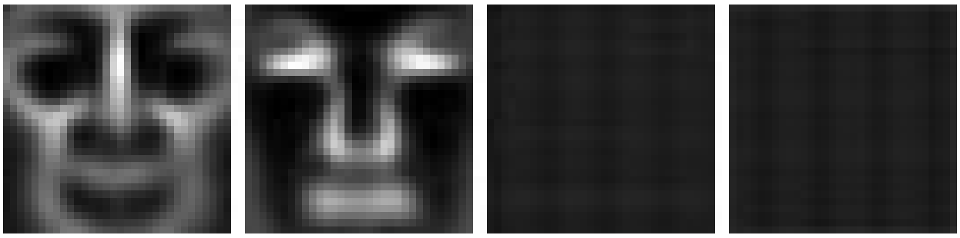


Figure 1.6: Probabilities for ridges and valleys, estimated from the training faces (first two images) and for training nonfaces (last two images).

of an image. This depends more on the parameters of our filter than on anything else. This is confirmed when we observe the probabilities images for our nonfaces: flat. The conclusion is that it would have been exactly the same to forget about the nonface information and that the obtained classifier is too general. We would like to classify correctly those more complicated nonfaces that resemble faces.

Boosting is a process that takes advantage of wrongly classified samples to improve the performance of a classifier [75]. In order to illustrate the importance of a nonface sample set, we decided to incorporate one step of a very simple boosting scheme. What we do, is re-train our classifier with the false positives obtained. From the 1560600 nonfaces we evaluated, our BIM said that 20009 of them were faces (the ratio of these two numbers is our mentioned false positive rate). These are the samples we use in this new stage. Final classification is performed in a cascaded manner: If the first step says an image is a nonface we discard it, if it says its a face, we test it on the second step. Results of applying this second step are a detection rate of 92.24% and 91.88% on training and evaluation sets, respectively. Final false positive rate was 0.282%. So we err in approximately 3 out of every 1000 images which is much better but still far from good. We can now observe the estimated probabilities of these misclassified nonfaces. These probabilities are shown in Fig. (1.7) where we can observe a more defined pattern, closer to the mean response for a face. Also, in fig. (1.8) we can observe a sample of 256 false positives encountered by the boosted detector. These false positives are not confusing patterns themselves, but still our boosted detector cannot tell the difference. The reason for this is that our BIM uses only the mean to train the classifier. We can hope that further boosting can solve this problem to a certain extent.

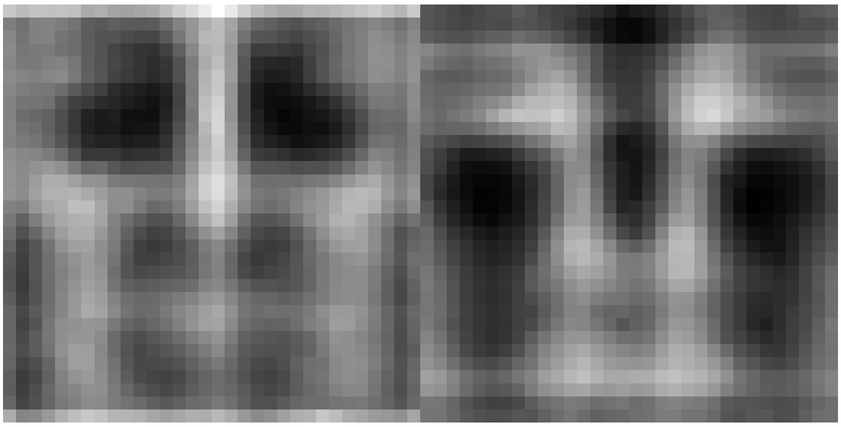


Figure 1.7: Probabilities for ridges (left) and for valleys, estimated from the misclassified nonfaces (false positives).

Of course, results are not sufficient for a complete face detector. Actually the main point, which is to assume that pixels in faces have statistically independent responses is altogether wrong. But there are also some advantages in this method. We

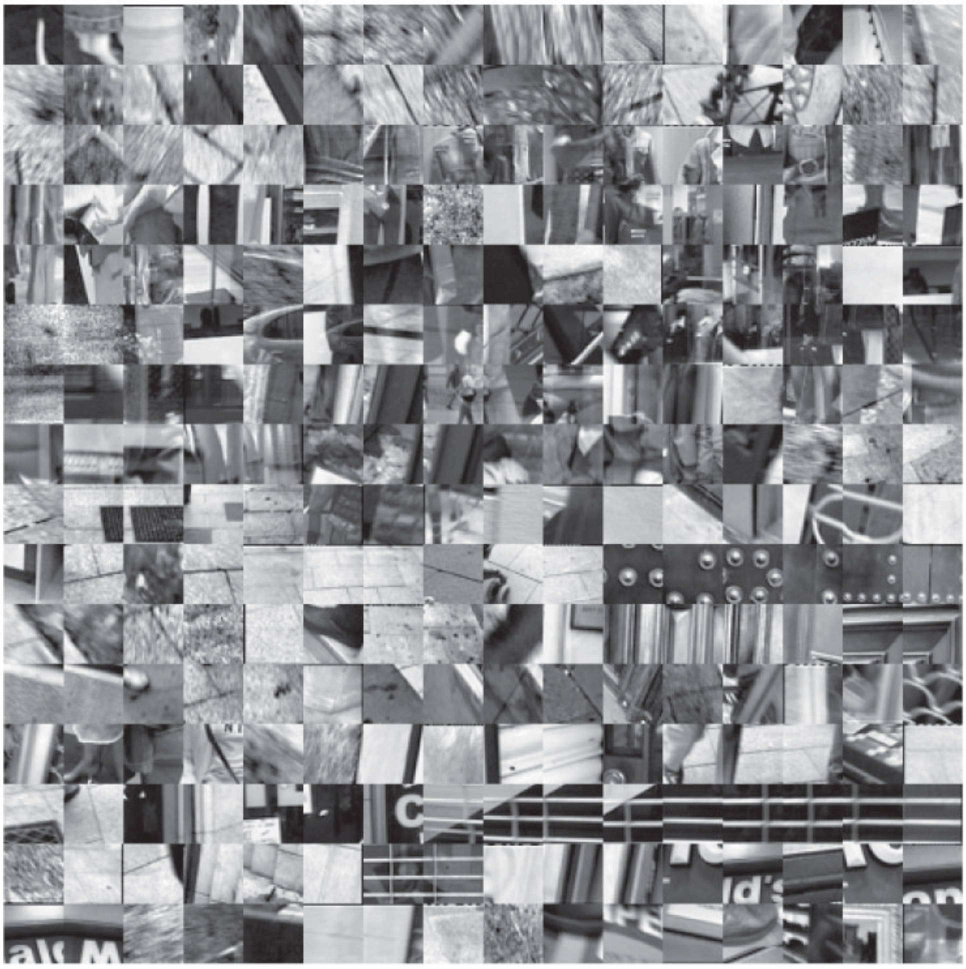


Figure 1.8: False positives for the boosted binary naive Bayes detector.

have developed a face detector that allows very simple and accurate (unidimensional) density estimations, which is scalable, can be learnt in an iterative procedure, and, above all, is very fast computationally: projecting a binary vector on a hyperplane is fast. Gathering the 12000 faces from famous people as you might have noticed in Fig. (1.5) can be a very long process, unless you actually have a face detector. The detector used for this task was quite more complex, but strongly based on the classifier we just developed as an illustrative example of the theory introduced in this chapter.

1.4 Conclusions

In this introductory chapter we first outlined the main concepts underlying statistical pattern classification. The scheme followed by this approach is called the Bayesian decision scheme. Given features that represent patterns and classes that group these patterns under different labels, we have seen that statistical classification is based on the distribution of these features for each class. We have overviewed some of the usual approaches for estimating these distributions, or class-conditional densities and observed that, if this estimation is exact, the Bayesian classifier is the best classifier, provided we wish to minimize the probability of misclassification. In consequence the error associated with this classifier, called the Bayes error, is a lower bound for the error made by any classifier, statistical or not. Direct density estimation from the measurements has inconvenients due to factors such as the noise present in the measurements or the eventual high-dimensionality. This is a frequent situation when dealing with visual data.

Measurements are usually combined into features which result more adequate for our purpose. Feature extraction techniques can be used for reducing noise or dimensions, introducing invariants, preserving discriminative information, etc. We have focused in the particular case in which the extracted features are a linear combination of the measurements. Linear feature extraction can be seen as the result of applying a linear transformation to the measurements.

An experiment on the problem of face detection from images attempts to illustrate statistical classification interacting with feature extraction. In this case the measurements are the pixels corresponding to greyvalue images and the extracted features are binary descriptors. Features are extracted not due to the dimensionality reduction they provide (actually, there are more features than measurements), but because of their invariance, in this case, to illumination changes. Another characteristic of the chosen features is that they have binary values. In this case statistical classification can be simplified. For instance, if we assume feature independence, naive Bayes can be applied and the result is a very simple linear classifier.

We have observed that the classification error of data after any linear transformation increases the Bayes error. From this fact, a highly relevant premise that drives this whole thesis is derived: **a statistical classifier will only benefit from a linear transformation if the conditional densities of the resulting features can be more accurately estimated than the original density**. This motivates the search of linear representations exclusively from the scope of the simplification they provide on the densities. In the next chapter we will analyse three common unsupervised linear feature extraction techniques from this perspective. A well known simplification is derived from the assumption of independence. In this case, computation of an N-dimensional density is reduced to the computation of N unidimensional densities. When the Bayes classifier makes this assumption, the result is the well known naive Bayes classifier which has been presented with detail. One of the transformations analysed in the next chapter is independent component analysis (ICA), whose objective is to provide statistically independent projections. If the objective is met, ICA can greatly simplify density estimation and, in consequence, statistical classification. Chapter 3 will see how independent component analysis, when applied from within a class-conditional framework, results in a class-conditional representation naturally associated with the naive Bayes classifier. We will also see that one of the disadvantages of class-conditional representations is that they fail to model intra-class relationships. Chapter 4 attempts to solve this problem by introducing a measure of class separability adequate for the obtained representation. This measure can act as a robust criterion for selecting discriminative features.

Our intention, along these three next chapters is to provide a unified framework for the design of a statistical pattern classifier, opposed to the alternative where each stage in a pattern classification process is considered independently from the other stages. In our case, once the initial assumptions are made, feature extraction, selection and finally classification are naturally associated among each other. Chapter 5 finally extends this line of reasoning beyond statistical and parametric techniques.

Chapter 2

Unsupervised Linear Transformations

2.1 Introduction

Generalizing the information obtained from training data to non-training situations is the aim of learning. Unsupervised learning algorithms such as those we will study in this chapter are helpful for modeling low dimensional pattern structures present in high dimensional data. Linear transformations account for most of the feature extraction performed in practice. Examples of spread out multivariate linear transformations are the Discrete Fourier transform, signal convolutions (linear filters), wavelets, Principal Component Analysis, Factor Analysis, Independent Component Analysis, Nonnegative Matrix Factorization, Canonical Correlation Analysis, Linear Discriminant Analysis, etc. There are many reasons to restrict ourselves to linear mappings, mainly their simplicity. Linear mappings are, among the family of multivariate transforms, those that involve less commitment in terms of prior assumptions. Also, if the mapping is linear, the mapping function is well defined and our task is to find the coefficients of the linear function so as to optimize a criterion. The problem of optimality of a representation under a certain criterion is computationally and conceptually simpler for linear transformations. Moreover, most linear techniques can be extended to the nonlinear case but these extensions are generally problem-specific and computationally expensive. Along this thesis we will restrict ourselves to linear transformations for feature extraction, focusing on the Independent Component Analysis.

As mentioned in section (1.2.3) any linear transformation on a D-dimensional random vector $\boldsymbol{x}$ yielding an M-dimensional random vector $\boldsymbol{y}$ can be expressed as

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} \tag{2.1}$$

with $\boldsymbol{W}$ an $M \times D$ matrix and, in general, $M \leq D$. In this case, since each component of $\boldsymbol{y}$ is obtained from a sum of every component of $\boldsymbol{x}$ weighted with a row of $\boldsymbol{W}$, we can see any transformation as linear filter of the data, $\boldsymbol{W}$ can be called the *filter matrix* and its rows are the filters. If the transformation is invertible with $\boldsymbol{W}^{-1} = \boldsymbol{A}$, we can reexpress (2.1) as

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{y} = \sum_{m=1}^{M} \boldsymbol{A_m} y_m \tag{2.2}$$

with $\boldsymbol{A_m}$ the columns of $\boldsymbol{A}$. This last equality can be seen as expressing each sample in the basis given by the columns of $\boldsymbol{A}$, where the $\boldsymbol{y}$ contains the coefficients in projected space. Therefore $\boldsymbol{A}$ is called the *basis* matrix and the coefficients of $\boldsymbol{y}$ *components*.

The main objective of this chapter is to introduce Independent Component Analysis (ICA), a linear unsupervised learning technique. This technique will be adapted for its supervised use in chapter 3 as a way of improving naive Bayes performance. We will first introduce Principal Component Analysis (PCA), a classical technique in statistical data analysis, closely related in its goal to ICA to the extent it is often used as a preprocessing step of the latter. We then focus on Nonnegative Matrix Factorization (NMF) which has a similar objective function to PCA and also includes the hypothesis of feature nonnegativity. This provides local representations and gives us an insight on sparsity, a concept closely linked with ICA. We also introduce a weighted variation of this transformation in an attempt to eliminate redundant basis, a problem not encountered with PCA. Finally, we focus on Independent Component Analysis. After introducing the ICA model we detail two methods for estimating the transformation: maximum likelihood and negentropy maximization, related between each other through the concept of mutual information. The first estimation procedure gives us an insight on the nature of ICA from the perspective we are interested in this thesis: as the representation in which the product of the marginal probabilities of the projected features best approximates the probability of the original features. The second approach, achieves the same objective indirectly resulting in a much more efficient algorithm. We then compare ICA with PCA and NMF and detail the scope of applicability of this nonsupervised learning technique. We finally apply ICA to model shape variations, a process usually done with PCA. This experiment allows us to compare both techniques and also helps to understand the capability of ICA as a representation and its nature.

2.2 Principal Component Analysis

The roots of Principal Component Analysis (PCA) can be found in the early work of Pearson [116] and its first application shortly after in a classic paper by Spearman [143]. In the latter, the author considered data that consisted of school performance rankings corresponding to schoolchildren and determined a single linear combination of the data such that it explained for the maximum amount of variation in the results, claiming to have found a general factor of intelligence. PCA is discussed at length from historical, theoretical and applicability perspectives by Jollife in [67].

We have mentioned that effective features are those which represent our data both accurately and in a *simple* way. Simplicity and accuracy are closely related to dimensionality and representation error. This error can be measured, for instance, in terms of the mean square distance between the original data points and their projections on the subspace. PCA can be seen as the transformation that finds the orthogonal vectors spanning the subspace that minimizes this mean square error (MSE criterion). We will now assume that the data is centered, that is $E\{\boldsymbol{x}\} = \bar{\boldsymbol{x}} = 0$. If this were not the case we translate the origin to the mean obtaining centered data. If the dimension of the data is D, we are seeking for orthogonal vectors spanning the subspace $\boldsymbol{w}_m$ with $m = 1, \dots, M$ which minimize the following criterion

$$J^{PCA} = E\{\|\boldsymbol{x} - \sum_{m=1}^{M} (\boldsymbol{w}_m^T \boldsymbol{x}) \boldsymbol{w}_m\|^2\} \tag{2.3}$$

since the projection of $\boldsymbol{x}$ into the subspace spanned by $\boldsymbol{w}_m$ is precisely $\sum_m (\boldsymbol{w}_m^T \boldsymbol{x}) \boldsymbol{w}_m$. Due to the orthogonality of the vectors $\boldsymbol{w}_m$, this criterion can be further written as,

$$\begin{aligned} J^{PCA} &= E\{\|\boldsymbol{x}\|^2\} - E\{\sum_{m=1}^{M} (\boldsymbol{w}_m^T \boldsymbol{x})^2\} \\ &= \text{trace}\{\boldsymbol{\Sigma}_x\} - \sum_{m=1}^{M} \boldsymbol{w}_m^T \boldsymbol{\Sigma}_x \boldsymbol{w}_m \end{aligned} \tag{2.4}$$

with $\boldsymbol{\Sigma}_x$ the covariance matrix of $\boldsymbol{x}$. The first term does not affect the optimization so it can be dropped and minimization of J^{PCA} is equivalent to maximization of the last term in (2.4). It can be seen through standard linear algebra [76, 16] that the solution to this problem is given by considering $\boldsymbol{w}_m = \boldsymbol{v}_m$ where $\boldsymbol{v}_m$ is the m-th eigenvector of the covariance matrix in descending order of eigenvalues. So the PCA filter and basis matrix can be found analytically by directly estimating the eigenvectors of the covariance matrix of the data and setting them as rows and columns, respectively. If $\boldsymbol{V}$ is the $M \times D$ matrix that contains these eigenvectors as rows, then the PCA solution can be expressed as,

$$\boldsymbol{V}\boldsymbol{x} = \boldsymbol{y}. \tag{2.5}$$

For this choice of filter matrix, the projected data or components $\boldsymbol{y}$ are called *principal components*. Notice that the covariance matrix is real and symmetric so the eigenvalues $\lambda_1, \dots, \lambda_D$ are real and nonnegative. By replacing $\boldsymbol{w}_m$ by $\boldsymbol{v}_m$ in (2.4) we have that the value of the minimum square error objective function is

$$J^{PCA} = \sum_{i=m+1}^{D} \lambda_i \tag{2.6}$$

PCA is probably the most widely used data-adaptive transformation and it is of great practical significance. Its success is largely due to the fact that the solution can be found in an analytical way. And when the type of application (speed requirements, dataset size, high dimensionality) does not allow standard covariance matrix or eigenvector estimation, several on-line procedures based on artificial neural networks or gradient ascent methods are available [35, 107].

2.2.1 PCA Properties

PCA is closely related to both the Karhunen-Loève and Hotelling transforms [67] and it arises as the solution to many closely related problems which contribute to the understanding of this powerful technique. It is observed from the preceding section that PCA obtains the subspace that best represents our data in the minimum squared error sense. Also, from the solution found we observe that the PCA directions are those orthogonal directions corresponding to the maximum variance of the data. Actually if PCA is proposed as a variance maximization problem, the solution is exactly the same as in (2.5). Additionally we find out that the variance of the data on each direction is the value of the corresponding eigenvalue. This follows from the equality

$$E\{y_m^2\} = E\{\boldsymbol{v}_m^T \boldsymbol{x}\boldsymbol{x}^T \boldsymbol{v}_m\} = \boldsymbol{v}_m^T \boldsymbol{\Sigma}_x \boldsymbol{v}_m = \lambda_m. \tag{2.7}$$

The fact PCA can be also understood as a projection into orthogonal directions of maximum variance is illustrated in Fig. (2.1) where the PCA basis vectors learnt from a bivariate Gaussian distribution are shown to effectively span the maximum variance directions. The basis vectors are scaled to three times their corresponding eigenvalues to illustrate their relation to the variance.

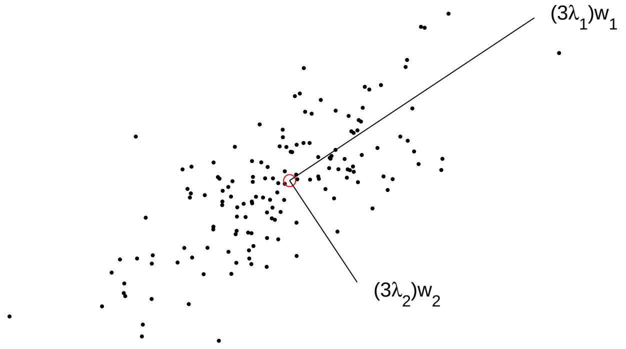


Figure 2.1: The PCA basis for a bivariate artificial Gaussian dataset.

The interpretation in terms of variance also leads to a widely spread out method for choosing the number of principal components in terms of variance preservation. We can define the total variance as the trace of the covariance matrix. Since the trace is preserved under orthonormal linear transformations, the total variance of a full PCA transformed dataset is the same as the total variance of the original data and is equivalent to the sum of the eigenvalues of the covariance matrix. If only the first M components are chosen, then the total variance of the resulting data is obtained by adding the first M eigenvalues. The normalized ratio between this resulting variance and the original variance of the data,

$$PV = 100 \times \frac{\sum_{m=1}^{M} \lambda_m}{\text{trace}(\boldsymbol{\Sigma}_x)} \% \tag{2.8}$$

gives us a measure of the percentage of data variation we are preserving under the transformation. Notice that $J^{PCA} = \text{tr}(\boldsymbol{\Sigma}_x)(1 - PV/100)$. A frequently chosen criterion to decide over the dimensionality consists in thresholding this total variation in order to preserve a fixed percentage of the variation. Typical percentages are above

90%. Notice from (2.6) that this is equivalent to choosing a fixed amount of error in the mean squared sense. Small variances in the data are often associated to noise. So, under certain simple assumptions, we can state that PCA reduces noise as well as dimension. Also notice what happens when there exist eigenvalues taking zero values. Dropping their corresponding eigenvectors from the PCA Basis has no effect on the mean square error or, what is equivalent, preserves 100% of the data variation, meaning that dimensionality is reduced and no information lost.

The PCA model, when seen as arising from the minimum squared error criterion or from variance maximization, assumes no underlying statistical model on the data. Nevertheless, PCA can also be derived from a generative latent variable model using the techniques of factor analysis [55]. In this case the model is expressed as

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{n} \tag{2.9}$$

where $\boldsymbol{y}$ and $\boldsymbol{n}$ are a zero-mean uncorrelated Gaussians, the first with identity covariance matrix. The likelihood function can be formulated using the fact that the density of $\boldsymbol{x}$ given $\boldsymbol{y}$ is Gaussian and the maximum likelihood solution for obtaining $\boldsymbol{A}$ under this model, when the noise tends to zero, is the PCA solution. In this case the obtained basis are scaled with the square root of the inverse of the eigenvalues, since we have assumed that $E\{\boldsymbol{y}\boldsymbol{y}^T\} = \boldsymbol{I}$. The factor analysis solution corresponds to leaving the observation noise matrix diagonal and not making its trace tend to zero.

Another application of PCA is data *whitening.* To *whiten* or *sphere* the data is to transform the data linearly so the components of the transformed vector are uncorrelated and have unit variance. The term "white" comes from the fact that the power spectrum of white noise is constant over all frequencies, resembling the spectrum of white light which contains all colors. Whitening is frequently used as a preprocessing stage, since it can provide invariance to displacement and scale changes. Since PCA uncorrelates the data, one of its applications is to perform whitening. We define $\boldsymbol{D}$ as the diagonal matrix containing the square root of the first (the largest) M eigenvalues so $\boldsymbol{D} = diag\{\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_M}\}$. In this case, if $\boldsymbol{V}$ is the PCA filter matrix the following transform whitens the data:

$$\boldsymbol{z} = \boldsymbol{D}^{-1/2}\boldsymbol{V}(\boldsymbol{x} - \bar{\boldsymbol{x}}) \tag{2.10}$$

where $E\{\boldsymbol{x}\} = \bar{\boldsymbol{x}}$. The advantage of PCA whitening is that, if $M < D$ (2.10) also reduces the dimension minimizing the mean square error. Regardless of the eigenvalue normalization, PCA always uncorrelates the data. We have mentioned that statistical independence implies uncorrelation, so PCA can be also understood as one step towards a representation that yields statistically independent components. If the data is Gaussian, the resulting components are in effect independent with unidimensional Gaussian distributions.

Nevertheless PCA is a powerful and simple technique we should not forget the natural limitations derived from its definition, mainly the fact that PCA fails to distinguish high order relationships between the data. This is illustrated in Fig. (2.2) [38] where four very different data sets with identical statistics up to second-order are shown. Particularly interesting is the second distribution on the top row where we could associate each of the two clusters with a different class. Consider classifying this

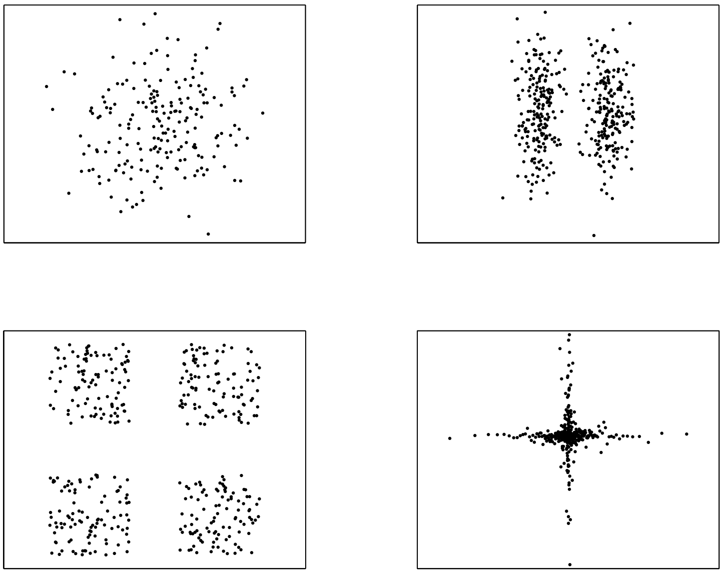


Figure 2.2: These four different datasets with identical mean and covariance, are indistinguishable to PCA.

data previously reducing the dimensionality to one. If dimensionality reduction was performed with PCA, since the direction of maximum variance is nearly vertical, the projection on the first principal eigenvector would remove all ability to discriminate among both classes. Instead, a projection on the second principal eigenvector would reduce dimensionality with no loss of discriminative information. This fact should be considered when using PCA as a dimensionality reduction technique previous to classification. Nevertheless, PCA performs successfully in several such problems. Two reasons account for this achievement: the first is that in practice it is not unfrequent to find noise in the directions with small variance, the second is that classes are frequently found to be distributed along the main directions of variance. In both of these examples, no discriminative information is lost when choosing the proper amount of principal components.

2.2.2 Example: Eigenfaces

A particularly successful application of PCA in the field of computer vision is learning an accurate low dimensional representation of face images. An interesting fact when the domain space samples are images is that, under a linear representation, the resulting basis can be also viewed as points in this space and consequently are also images. In the case of PCA applied to face images these eigenvectors receive their own name: *eigenfaces* [152]. Figure (2.3) shows the ordered set of the first 64 eigenfaces computed from the same 7000 sample set of faces used in the face detection example in (1.3.2). Once again faces are represented by 1024-dimensional vectors corresponding to 32×32 dimensional images, so the eigenfaces have also 32×32 pixels. In this case, we use the grayvalued image instead of use ridges and valleys. Nevertheless, certain type of attenuation of the illumination is always advisable so each sample is normalized in their own mean and variance, previous to the PCA. From (2.8) we have that using only these 64 eigenfaces preserves 88.71% of the total variance, while as few as 131 (approximately 10% of the initial number of features) components are required to account for more than 95% of the variance. In the bottom row of fig. (2.3), a sample face is shown, with an image representing the component activation for this particular face. Positive values are illustrated with red pixels and negative values with black pixels. Components and basis vectors have geometrical correspondence in the images. Also, in fig. (2.3.c) we show the result of retroprojecting the components into domain space. In this case, we can observe that the approximation is quite accurate and smoother than the original image, coherent with PCA assumptions.

Some interesting facts can be observed in the set of eigenfaces in fig. (2.3). At least the first two eigenfaces are directions of illumination variation, meaning that we have not been completely successful normalizing illumination with our mean-variance correction, since it still accounts for the largest variations in our database. Other eigenfaces can be also directly associated with particular variations, for instance the fourth eigenface deals with the position of the eyes, the fifth with the position of the mouth, etc. We can also observe the presence of higher frequencies as we advance in the eigenfaces. The fact PCA is a linear transformation makes it simple to interpret this basis. If we project a face image into the PCA space, each principal component indicates the weight of the corresponding basis: any face image can be written as a weighted sum of these eigenfaces. If a proper training set is chosen (a training set that generalizes correctly) the error of this reconstruction is upper bounded.

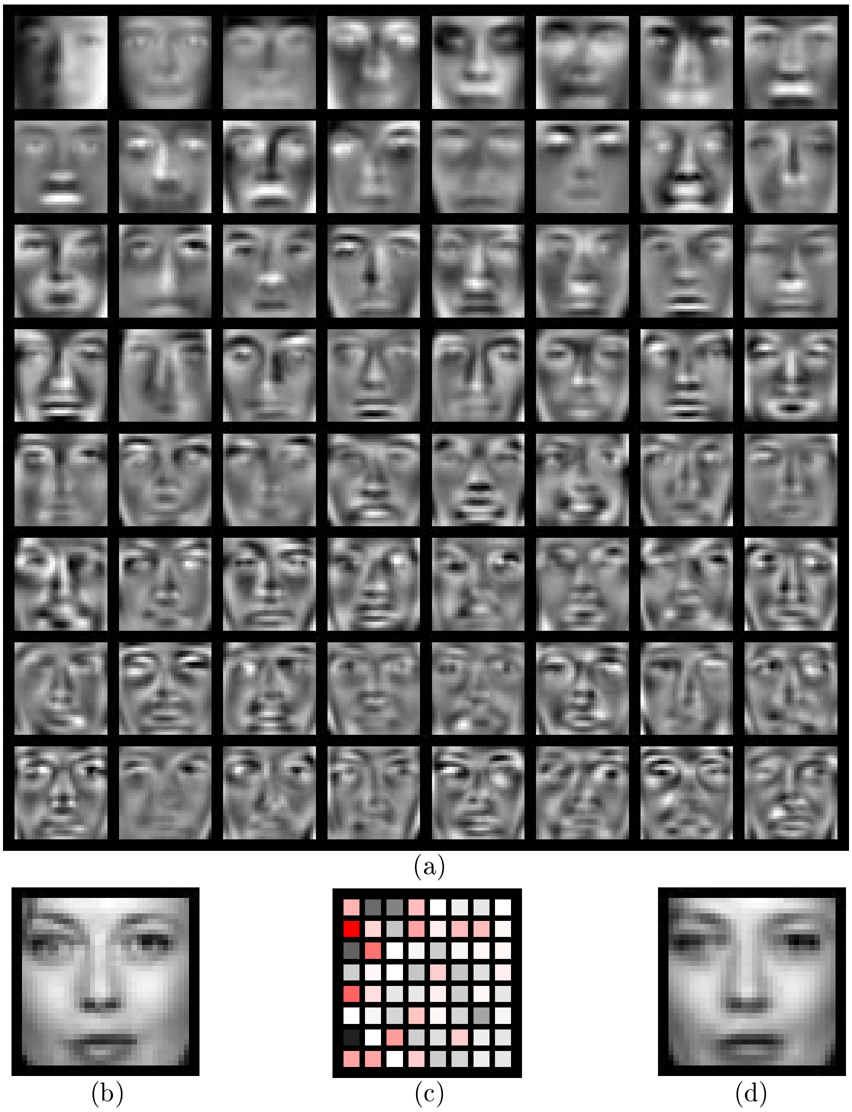
Additionally, Turk and Pentland [152], suggest using a vector's distance to this subspace to decide whether an image is a face or not. The *distance from face space* of a sample vector $\boldsymbol{x}$ is the error made by projecting it to the learnt model,

$$d(\boldsymbol{x}) = \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \|\boldsymbol{V}(\boldsymbol{x} - \bar{\boldsymbol{x}})\|^2 \tag{2.11}$$

Since PCA itself does not provide an estimate of the density, Moghaddam and Pentland adapted PCA to define a density estimation in order to give a generative model of the face class [102]. Considering the generative model of PCA, the densities used are generally Gaussian. Under this model, the likelihood of a new image can be evaluated, yielding low values for an image far from the training set even if it is near the principal subspace.

2.3 Nonnegative Matrix Factorization

Each learning algorithm has its own assumptions that bias generalization over particular problems and thus restrict the scope of applications. For instance, PCA assumes an interesting subspace should orthogonally span maximum variance directions or, equivalently, have low mean reconstruction error. Via these assumptions, PCA results in a popular technique useful for general dimensionality reduction, data compression, etc. Another consequence of these assumptions is that, through a global treatment of the input, PCA provides a holistic representation. This can be observed in the



(a)

(b)

(c)

(d)

Figure 2.3: The first 64 eigenfaces computed from a training set of 7000 frontal view face images (a). All basis images are normalized between 0 and 255 such that pixels corresponding to zero basis values have an intensity of 128. In the bottom row a sample face (b) with its component values (c) and the result of retroprojecting the components (d). Positive values in the components in (c) are represented in red, negative values in black.

eigenfaces example where the low dimensional pattern structures are the result of linearly combining a set of bases that take into account the pattern (the face) as a whole. Restricting (or relaxing) the assumptions should affect its performance and possibly modify the scope of applications.

A particularly interesting example of this situation occurs when a linear representation that also minimizes the mean squared error (MSE criterion) is learnt and non-negativity constraints included in the model. The model enclosing this situation is called Nonnegative Matrix Factorization (NMF) [86], and the main consequence of the nonnegativity constraints is that only non-subtractive combinations are allowed. This ensures that the components are combined to form a whole in a non-subtractive manner. For this reason, NMF yields a parts-based representation opposed to the holistic representation obtained through other methods such as PCA. Localized features offer several advantages in the context of object recognition, including stability to local deformations, lighting variations and partial occlusions. In addition there exists psychological and physiological evidence favouring a parts-based representation in the brain [110, 157, 93]. Notice also, that NMF is of straight application in several problems related with visual recognition where the features are naturally nonnegative (pixel intensities, histogram values, etc.) and where nonnegative components have a direct interpretation. For instance, if PCA were applied to a number of sample histograms, the PCA basis would contain negative values, making it impossible to interpret the basis itself as a histogram. This drawback is carefully eluded in the case of the eigenfaces where the negative valued basis is presented as (rescaled) intensity images which often lack of intuitive meaning.

The NMF model is simplified if matrix instead of vector notation is used. This change can be done by simply representing the set of N (in this case nonnegative) samples in the a D dimensional space with the $D \times N$ matrix $\boldsymbol{X}$ that contains each sample $\boldsymbol{x}_n$ as a column; a nonnegative $D \times M$ basis matrix $\boldsymbol{A}$; and a nonnegative $M \times N$ components matrix $\boldsymbol{Y}$ that contains the components $\boldsymbol{y}_n$ that correspond to the n-th sample as columns. In the matrix factorization framework, the sample matrix is approximated as a linear combination of the basis. So the NMF model can be stated as [86]

$$\boldsymbol{X} \approx \boldsymbol{A}\boldsymbol{Y} \tag{2.12}$$

The way the model is estimated depends on the definition of a proper cost function that quantifies the quality of this approximation. Such a cost function can be constructed using some measure of distance between two nonnegative matrices, in our case $\boldsymbol{X}$ and $\boldsymbol{AY}$. One possible cost function is simply the Euclidean distance between the matrices [109],

$$||\boldsymbol{X} - \boldsymbol{AY}|| = \sum_{n=1}^{N} \sum_{d=1}^{D} (\boldsymbol{X}_{dn} - (\boldsymbol{AY})_{dn})^2. \tag{2.13}$$

Notice that minimization under this cost function is equivalent to minimizing the (frequential) mean squared error between the data and the recovered data. Another objective function, but with an underlying statistical model, is introduced in [86],

$$\Gamma(\boldsymbol{X}, \boldsymbol{AY}) = \sum_{n=1}^{N} \sum_{d=1}^{D} [\boldsymbol{X}_{dn} \log{(\boldsymbol{AY})_{dn}} - (\boldsymbol{AY})_{dn}] \tag{2.14}$$

In this case, NMF can also be stated as a problem of likelihood maximization, since it is equivalent to blind Richardson-Lucy restoration with nonnegativity constraints and the assumption that $\boldsymbol{X}$ is drawn from a Poisson distribution with mean $\boldsymbol{AY}$ [87].

Unlike PCA, maximization of (2.13) or (2.14) subject to the nonnegativity constraints does not have an analytical solution. In [87], rescaled gradient ascent is used to derive a set of multiplicative update rules for estimating the basis matrix and components. Convergence of the resulting algorithm is proved using an auxiliary function analogous to that used for proving convergence of the Expectation-Maximization algorithm [33]. We now present the update rules associated to the cost function given in (2.14). Refer to Lee and Seung [86] for convergence details and the update rule associated to (2.13). Considering that the gradients for (2.14) take the form

$$\frac{\partial \Gamma}{\partial Y_{\mu\nu}} = \sum_{d=1}^{D} \left(A_{d\mu} \frac{X_{d\nu}}{(\boldsymbol{AY})_{d\nu}} - A_{d\mu} \right) \tag{2.15}$$

$$\frac{\partial \Gamma}{\partial A_{j\mu}} = \sum_{n=1}^{N} \left(Y_{\mu n} \frac{X_{jn}}{(\boldsymbol{AY})_{jn}} - Y_{\mu n} \right), \tag{2.16}$$

the resulting update rules are

$$Y_{\mu\nu} \leftarrow Y_{\mu\nu} \sum_{d=1}^{D} A_{d\mu} \frac{X_{d\nu}}{(\boldsymbol{AY})_{d\nu}} \tag{2.17}$$

$$A_{j\mu} \leftarrow A_{j\mu} \sum_{n=1}^{N} Y_{\mu n} \frac{X_{jn}}{(\boldsymbol{AY})_{jn}} \tag{2.18}$$

$$A_{j\mu} \leftarrow \frac{A_{j\mu}}{\sum_{d=1}^{D} A_{d\mu}}. \tag{2.19}$$

As an example we have applied the resulting algorithm to the same dataset used in (2.2.2). In Fig. (2.4) the resulting NMF basis is exposed. Locality can be observed in features: NMF has learnt how to represent faces with a non-subtractive linear combination of parts of faces such as eyes, mouth, eyebrows, etc. We can also observe that there is no apparent hierarchy on the basis factors and that, since no orthogonality constraint is imposed on the basis vectors, they can be redundant. In the bottom row of fig. (2.4), a sample face is shown, with an image representing the component activation for this particular face. Positive values are illustrated with red pixels and negative values in with black pixels. In this case, there are no negative values. We also observe that almost all components are activated, meaning that most of the basis take part in the representation of a sample image. Components and basis vectors have geometrical correspondence in the images. Also, in fig. (2.4.c) we show the result of retroprojecting the components into domain space. The approximation is also quite accurate since NMF seeks to minimize reconstruction error, though less accurate than the one provided by PCA due to the nonnegativity constraints included in the objective function.

2.3.1 Weighted NMF

One of the main disadvantages of NMF arises when applied to local representations. In the field of visual object recognition, the earlier systems were focused on holistic object representations, the object as a whole. This approach has been successfully used in different applications such as face recognition or robot positioning, being its main advantage the ability to perform fast and reliably at low spatial resolutions and without any kind of prior knowledge. Still, there are some problems that prove difficult to solve for these kind of representations, like partial object occlusions and severe lighting changes. Recently, several approaches have proposed the use of local window representations as a reliable solution to occlusions, complex background [131, 130], scale changes [124], illumination changes [32], and different viewpoints or orientations [130, 32]. This approach relies on domain specific knowledge to address the problem such as what to look for in an object and where that feature should be. This allows for a richer class representation and, consequently, a model that can be used to focus on more complex situations. Local models, representing a more specific and possibly less complex part of the object usually have high levels of redundancy among classes. Nevertheless this redundancy is dealt when the combination of parts is performed, it might present a problem to certain unsupervised learning techniques. In the case of NMF, redundant samples generate redundant basis. This situation does not arise with PCA due to the assumption of basis orthogonality. Such assumption is not possible within the NMF context since it would clearly break the nonnegativity constraints. We now present a slight modification of NMF that, through the introduction of a sample weight matrix, solves this problem. We call this approach Weighted Nonnegative Matrix Factorization (WNMF) and perform an experiment to illustrate its advantages over the classical approach.

The main reason NMF finds redundant basis in local representation is the fact this approach extracts several feature vectors from each object instead of a single global representation. Feature vectors can be redundant and strong similarities can exist among them. These similarities are not taken into account in the classical NMF approach: several identical feature vectors have more weight in the cost function than a few strongly different vectors and possibly relevant for the representation samples. A possible solution to this problem is to introduce a weight on each the training vectors, giving more weight to those vectors with low probability of appearing in the training set. This weighted model can be seen as the result of right-multiplying both sides of the factorization with a $N \times N$ diagonal weight matrix $\boldsymbol{Q}$ and to estimate the basis and encodings for the new factorization model,

$$\boldsymbol{XQ} \approx \boldsymbol{AYQ} \tag{2.20}$$

where the diagonal element q_n corresponds to the weight of training vector $\boldsymbol{x}_n$, with $1 \leq n \leq N$. It is also assumed that all the weights sum to unity. The modified

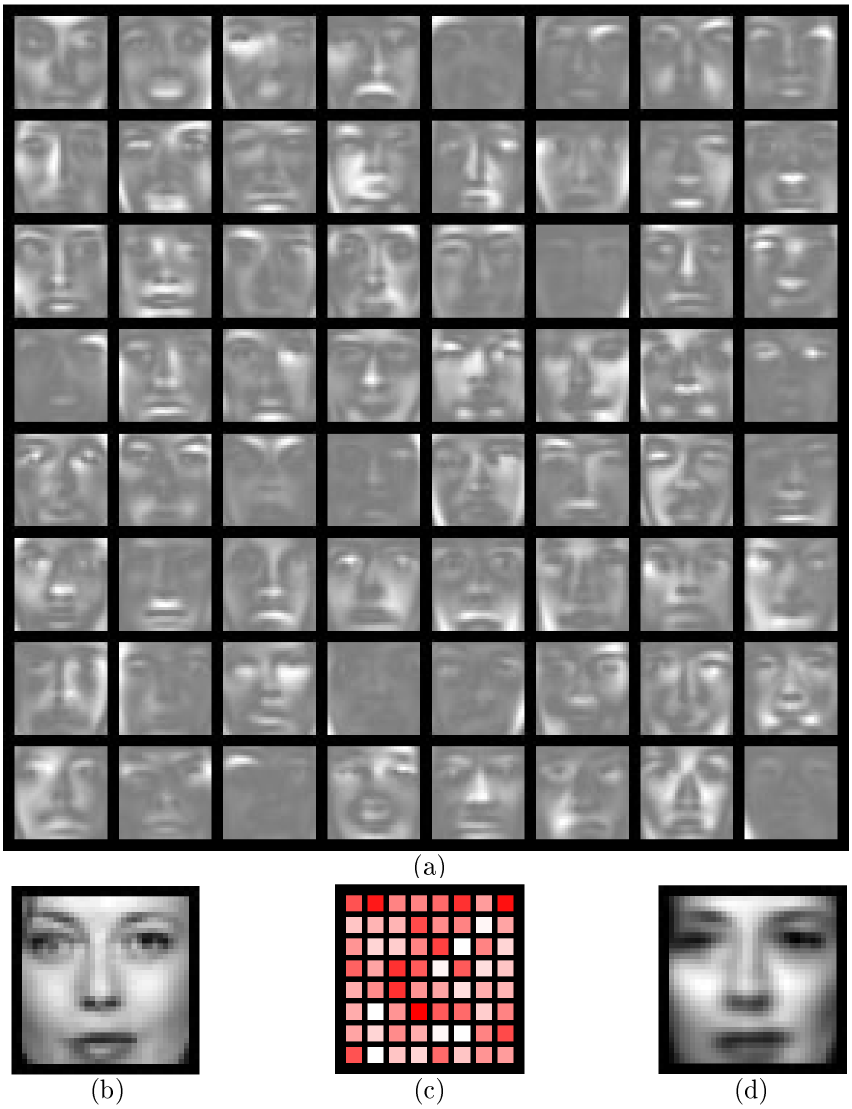


Figure 2.4: 64 NMF basis computed from a training set of 7000 frontal view face images (a). All basis images are normalized between 0 and 255 such that pixels corresponding to zero basis values have an intensity of 128 (in this case no intensity levels lower than this value are present). In the bottom row a sample face (b) with its component values (c) and the result of retroprojecting the components (d). Positive values in the components in (c) are represented in red, negative values in black (in this case no negative components exist).

objective function in this case takes the form

$$\Gamma_Q(\boldsymbol{X}, \boldsymbol{AY}) = \sum_{n=1}^{N} q_n \sum_{d=1}^{D} [\boldsymbol{X}_{dn} \log{(\boldsymbol{AY})_{dn}} q_n - (\boldsymbol{AY})_{dn}] \tag{2.21}$$

Using gradient descent, as with classical NMF, we can deduce the iterative rules for maximizing this objective function. The iterative rule for both basis and encodings turns out to be

$$Y_{\mu\nu} \leftarrow Y_{\mu\nu} \sum_{d=1}^{D} A_{d\mu} \frac{X_{d\nu}}{(\boldsymbol{AY})_{d\nu}} \tag{2.22}$$

$$A_{j\mu} \leftarrow \frac{A_{j\mu}}{\sum_{n=1}^{N} q_n Y_{\mu n}} \sum_{n=1}^{N} Y_{\mu n} \frac{q_n X_{jn}}{(\boldsymbol{AY})_{jn}} \tag{2.23}$$

$$A_{j\mu} \leftarrow \frac{A_{j\mu}}{\sum_{d=1}^{D} A_{d\mu}} \tag{2.24}$$

The iterative rule for the encodings (2.22) is not altered. Derivation of this multiplicative rule from the gradient descent algorithm requires a change in the gradient step to take into account the weights.

As global NMF finds the redundant basis corresponding to the most frequent training vectors when applied to local data, a good choice to define the weighted matrix $\boldsymbol{Q}$ is giving more weight to the less frequent training vectors. By obtaining the probability of each training vector with respect to the training database and assuming that this probability will hold on the test stage, we invert these probabilities and we take them as the q_n coefficients. In this way, we emphasize those training vectors with less importance. Thus, the obtained basis will contain a wide variety of behaviours: from the most to the least frequent ones, improving the global NMF capacity of representation that only takes into account the most frequent ones.

2.3.2 Weighted versus Classical Approach

A first experiment compares the representation capacity of NMF and WNMF. We have selected 5 different color newspapers (see Fig. (2.5)) each of them containing particular local characteristics. Having 10 instances of each newspaper, we have divided all of them using a predefined grid obtaining a large amount of local regions (200 per image). Each local region is represented using the combination of one 8-bin histogram per spectral band (resulting dimensionality is $D = 512$) and all of them are used as input in the learning process of the NMF matrices ($\boldsymbol{A}$ and $\boldsymbol{Y}$). Because of the initial random conditions of the algorithm, we have initialized both approaches with the same random matrices, for accurate comparison. The selection of the weights (matrix $\boldsymbol{Q}$) in the WNMF approach is performed using a leave one out technique that tests the probability of finding the training histogram in the whole training database. Once we have estimated matrix $\boldsymbol{A}$ for both approaches, other local color histograms were projected. [1] These additional histograms were extracted from

[1] Notice that projection in NMF cannot be done by simply using the (pseudo)inverse since this would break the nonnegativity constraint. One alternative for projection is to use exactly the same

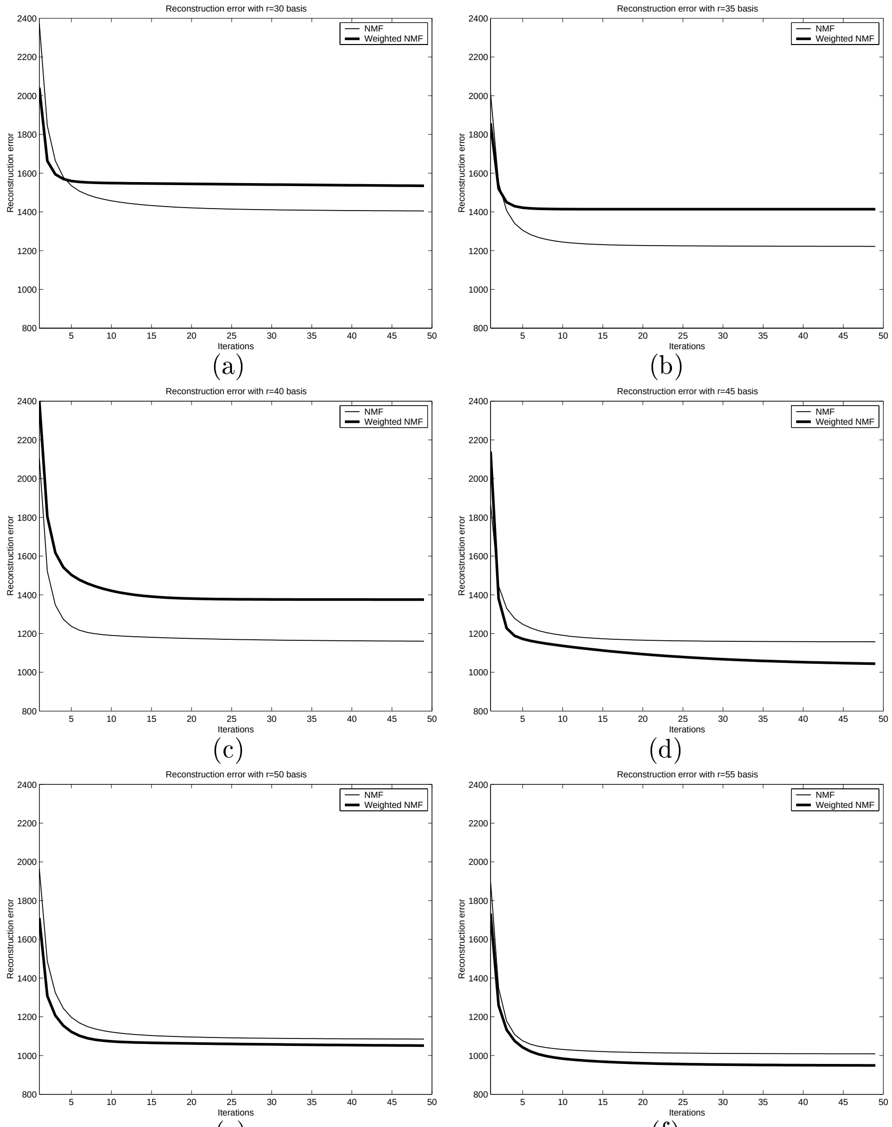
the same objects but using a different grid. Finally, the reconstruction error of both approaches was compared using (2.13). Notice that, since NMF lacks restrictions, reconstruction error on the training set should be less than for WNMF. What we are actually measuring here is how well the reconstruction generalizes to a test set. This experiment was performed for a different number of basis $M = 30, 35, 40, 45, 50, 55$. Results are shown in fig. (2.6) where we can see the evolution of the reconstruction error of the testing set against the number of iterations. We have to note that for the training stage, we learned our models using approximately 400 iterations of the algorithm, until it stabilized. For the test stage (the projection) we only made 50 iterations. Observing the figure we can observe that much less iterations were needed until stabilization.

Figure 2.5: Five different color newspapers each of them containing some characteristic local regions.

Lee and Seung [86] note that the number of desired basis M is generally chosen so that $(D + N)M < DN$ where D is the dimension (512 in our case) and N the size of the dataset (10.000 in our case). With this rule, we can choose up to $M = 487$ basis. In Fig. (2.6), we obtain that WNMF cannot outperform NMF if it does not have a sufficient number of basis to represent specific tonalities (remember that our features are color histograms). If we consider $M \geq 40$ basis we have the opposite behaviour. To illustrate this situation we trained both models with a variable number of basis from 20 up to 200 in steps of 5 and a fixed number of iterations (50). Results are shown in Fig. (2.7) in terms of the number of basis against the difference between reconstruction errors. The WNMF error was subtracted to the NMF error so a positive value indicates a superior performance of the first.

Figure (2.7) confirms that with few basis, NMF generalizes better than its weighted version. Having a large amount of training vectors, NMF seeks the best basis that represents all this space and will unlikely generate redundant basis. Instead, WNMF gives more weight to specific color regions which might not be relevant in terms of reconstruction error, but that would surely avoid redundancy with an increased number of basis. This is also confirmed. We observe that, depending on the variability of the input data NMF will start to generate redundant basis at some point in which simultaneously WNMF will improve its performance. In our specific problem, this point is found when $M = 40$ and it is useful to know this number if we have to choose the best representation for a given problem. This situation, in which WNMF outperforms NMF in reconstruction is apparently in contradiction with the fact that

algorithm, previously fixing the basis in order to estimate the encodings of the new samples



(a) (b) (c) (d) (e) (f)

Figure 2.6: NMF (light line) and WNMF (solid line) reconstruction error against iterations on the test data. Using a small number of basis $M = 30, 35, 40$ from (a) to (c), NMF outperforms WNMF. When $M = 45, 50, 55$ from (d) to (e), WNMF outperforms NMF.

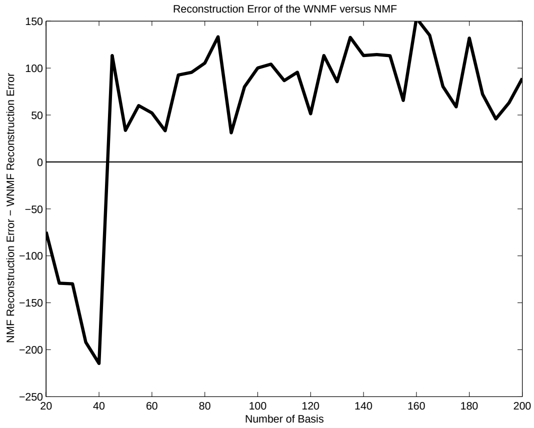


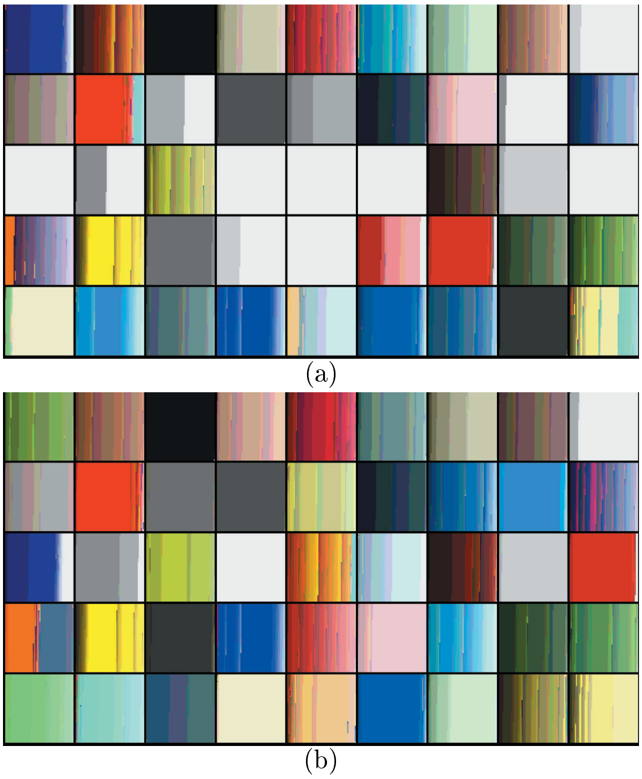
Figure 2.7: WNMF error subtracted to the NMF error against M, the number of bases.

NMF is a direct attempt to minimize the reconstruction error. Reasons for this result should be found in the fact that the optimization landscape for our problem is nonconvex, so both methods are likely to fall in local minima of the objective function. Since, from a theoretical standpoint, the only difference between both techniques is in the gradient directions and step, we can affirm that WNMF provides a better initialization for gradient descent, and consequently a local minimum nearer to the optimal solution.

Taking advantage of the nonnegativity of the solution we can also visualize the results: the basis vectors are themselves color histograms. From these histograms we can generate artificial images with the adequate proportion of colors. We chose the number of basis as 45 and generated such images for both approaches. The results are shown in Fig. (2.8). In both approaches we can already see the main characteristic of NMF: a specialized representation. This translates in sparse histograms: each basis vector accounts for a few colors, generally only one. The differences between both approaches can also be observed. In the NMF basis there exist several repeated histograms. Notably white which is present alone, as well as mixed with other colors. This is understandable since white is the predominant color in the newspapers. WNMF avoids this problem, localizing white in a single basis and diminishing its presence in other basis. Also, WNMF contains 5 different green tonalities against 2 in the classical approach. This kind of specialization is more likely to provide a better discrimination among objects, i.e. if a newspaper (or ad, maybe) is associated with a particular green tonality, it will be simpler to discriminate it from other newspapers (ads).

2.4 Independent Component Analysis

If we assume our data is the result of linearly combining nongaussian and mutually independent latent variables with an unknown mixing matrix, independent compo-



(a)

(b)

Figure 2.8: Histogram basis obtained by NMF (a) and WNMF (b). Note that the weighted NMF contains 5 different histogram basis with green tonalities against 2 green tonalities of the classical NMF.

nent analysis (ICA) is the statistical technique which reveals these hidden factors by defining a generative model on the observed data. In this case, the latent variables are called *independent components* or *sources* of the observed data.

Blind Source Separation (BSS, also known as Blind Signal Separation) is the classical application of the ICA model, and one of the main motors for all initial research on ICA [71]. BSS consists in recovering unobserved signals or *sources* from several observed mixtures. The *cocktail-party problem*, a paradigmatic BSS situation, provides a clarifying picture of the ICA context. Imagine a room with D people talking simultaneously and M microphones placed in different room locations. In this case, the original speech signals, or sources, can be represented by an M-dimensional random vector (one per person in the room) and the recorded sound signals which actually is our observed data is represented by a D-dimensional random vector (one per microphone). The problem is to estimate the original speech signals from the recorded signals. If we omit time delays, noise and other extra factors to simplify our model, we can linearly approximate the mixing function. Also, it is not unrealistic to assume that the speech signals are statistically independent. This is equivalent to assuming that speech waveforms corresponding to different persons are statistically independent signals. Since this waveform is nongaussian (speech waveforms are generally nongaussian), we are under the assumptions of the linear ICA model, and ICA provides a solution for this problem.

Figure (2.9) illustrates the situation through an example with 4 artificial sources. A 4×4 random mixing matrix was generated, and these sources mixed. The resulting signals are displayed in fig. (2.9.b). Figure (2.9.c) shows the estimated sources through ICA. As can be seen, if sign and order are not considered, the estimated sources are very close to the original source signals. Several interesting examples of ICA applied to the cocktail party problem on real world data (real speech waveforms) can be found on the Internet.

We will restrict ourselves to what is known as the *basic* ICA model [64]. This is the linear instantaneous noise-free mixing model classic in ICA, opposed to several extensions which consider nonlinear mixing, inclusion of explicit observational noise or time dependency. As with PCA and unless stated otherwise, we will assume our data is zero-centered. In practice, this situation can be always achieved by previously subtracting the global mean to the working dataset. Given a set of observations represented by the D dimensional random vector $\boldsymbol{x}$, assume the following generative model

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} \tag{2.25}$$

where the latent variables or *independent components* s_m in vector $\boldsymbol{s} = (s_1, \dots, s_M)^T$ are assumed independent and the $D \times M$ basis matrix $\boldsymbol{A}$ is unknown. Independent component analysis consists on estimating both matrix and independent components, when we only observe $\boldsymbol{x}$. Following the BSS application, the independent components are also known as sources and the basis matrix, usually called mixing matrix. The pseudoinverse of $\boldsymbol{A}$ which we will represent as $\boldsymbol{W}$, is called the filter or projection matrix and provides an alternative expression for ICA,

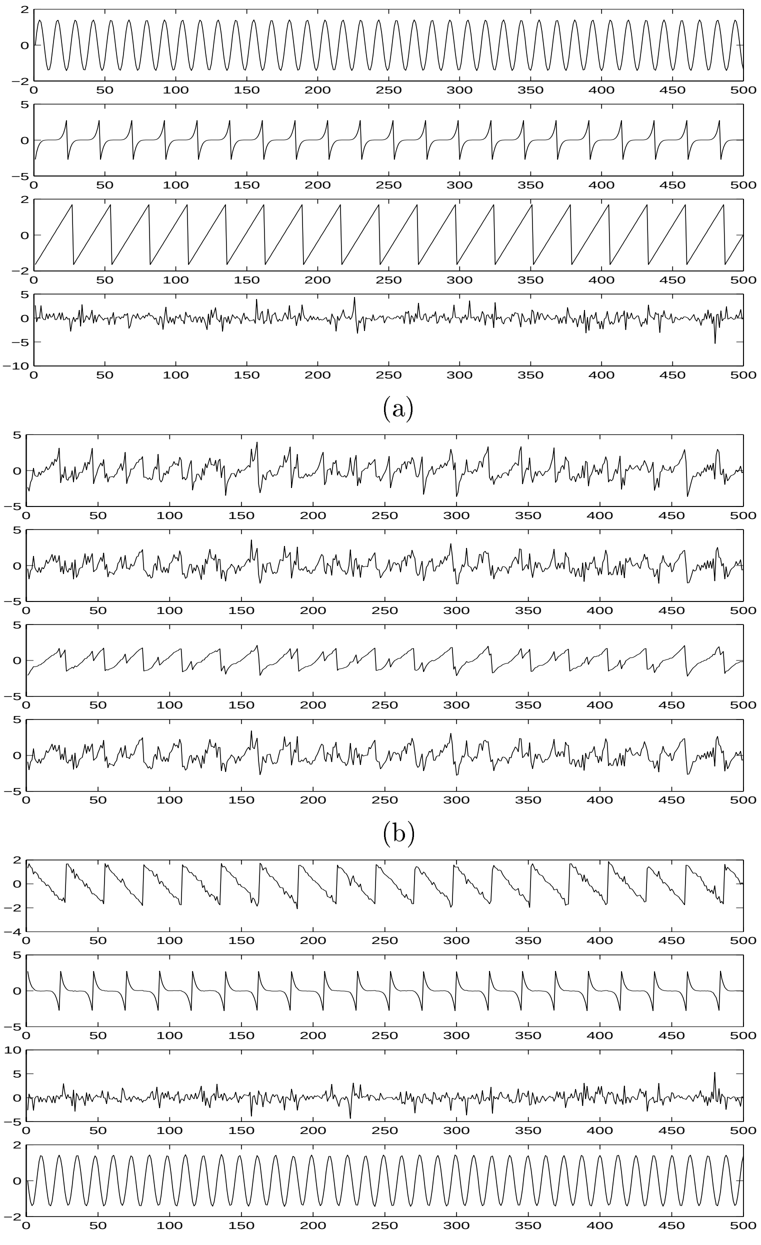
$$\boldsymbol{W}\boldsymbol{x} = \boldsymbol{s} \tag{2.26}$$

This expression provides an alternative definition for ICA, less rigorous but far more illustrative. Given a dataset $\boldsymbol{x}$, ICA searches for the linear transformation of the data $\boldsymbol{W}$, such that the projected variables are as independent as possible.

It has been shown [25] that if certain assumptions hold, the ICA model is completely identifiable, i.e. there exists a solution to the problem of estimating the mixing matrix and components. These assumptions are,

- *The independent components are assumed statistically independent.* Knowledge of one component gives us no information on the value of any other component. From a strictly probabilistic perspective, according to the definition of independence (1.20) this means that the distribution of the independent components $p(\boldsymbol{s})$ is factorizable in the product of the unidimensional marginal distributions. Among many other implications independence causes that cumulants of order larger or equal than 2 are all zero.

- *At least $M-1$ independent components are nongaussian.* The main inconvenient with Gaussian data is that cumulants of order higher than 2 are cancelled. As we mentioned, Gaussian data is completely identifiable given its statistics up to



(a)

(b)

(c)

Figure 2.9: (a) The original signals (sources). (b) The sources are mixed using a random mixing matrix. (c) ICA estimation of the sources, using only the mixed signals.

order 2 and thus, uncorrelatedness implies independence. On the other side, it can be easily seen that orthogonal transformations preserve uncorrelatedness. This would mean that for Gaussian data, the ICA model could eventually be estimated up to an orthogonal transformation, making the mixing matrix not identifiable.

- *The mixing matrix is square.* This situation, in which $M = D$, is called the *complete* case. In this case $\boldsymbol{W} = \boldsymbol{A}^{-1}$ and estimation greatly simplified. This assumption is not a necessary condition and can sometimes be relaxed. When this is done, two situations can arise. If we consider less sources than observations ($M < D$) we have that, though $\boldsymbol{W}$ can be completely determined, $\boldsymbol{A}$ contains uncertainty. In this case, the common approach is to previously perform dimensionality reduction using for instance PCA in a preprocessing stage, and then restrict the problem to the complete case. The second situation is when there are more independent components than dimensions in the data ($M > D$) and it is referred to as ICA with *overcomplete* bases. In this case, estimation is much more complicated and estimation methods less developed [90].

The ICA model also contains some ambiguities. From Eq. (2.25) we know that the independent components are zero-centered but we cannot determine the variances of the independent components. This is due to the fact that both $\boldsymbol{s}$ and $\boldsymbol{A}$ are unknown so any scalar multiplier on one of the sources can be cancelled by dividing the corresponding column of $\boldsymbol{A}$ by the same scalar. To overcome this situation, the magnitudes of the independent components are considered fixed. For instance, considering that each independent component s_m has unit variance: $E\{s_m^2\} = 1$. Notice that this restriction still leaves an ambiguity in the sign, because the multiplication of an independent component by -1 does not affect the model. Fortunately, this is insignificant in most situations.

Another important ambiguity in the model is that the order in the components cannot be determined. Given any permutation matrix $\boldsymbol{P}$ the model given in Eq. (2.25) is equivalent to $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{P}^{\boldsymbol{-1}}\boldsymbol{P}\boldsymbol{s}$. In many applications an order for the sources is necessary, so different ordering criterions can be used. The norm of the columns $\boldsymbol{A}$ can be understood as the contributions of the different sources to the variance of $\boldsymbol{x}$, so an order reminiscent to that of PCA would be to number the independent components in decreasing order of the norm of the columns of mixing matrix A. As we shall see, measures of nongaussianity play a significative role in ICA estimation. So another possibility is to order the sources according to their nongaussianity. The order here obtained would be related with that order given by projection pursuit. Nevertheless, none of these approaches is definitive and ordering the independent components is absolutely problem-dependent. Actually, imposing a hierarchy on the independent components would break the nature of ICA: if no source gives information on another source, does it have any sense to sort them?

2.4.1 Maximum likelihood ICA estimation

Given the model and its restriction we now turn to the estimation of the parameters in (2.26). One of the first approaches for ICA estimation is the classical maximum likelihood (ML) method detailed in section (1.2.4). This algorithm has been extensively tested and improved. Its main drawbacks are its computational load and the heuristic fact it does not generalize properly to high dimensions. Throughout this thesis we mainly work on the FastICA algorithm [65], introduced in the next section. Nevertheless, maximum likelihood is introduced since it is a natural approach to the statistical parameter estimation problem we are faced with (the parameters are the components of the filter matrix) and it illustrates clearly a basic point we want to make: **ICA is the representation in which the product of the marginal probabilities of the projected features best approximates the probability of the original features, times a constant value**. This section, as well as the following are very much based on chapters 8 and 9 of the book on ICA [64] by Hyvärinen. So we first derive the likelihood of the ICA model.

Based on the independence assumption on the sources and the change of variables theorem (1.9) we have that,

$$p(\boldsymbol{x}) = |\det \boldsymbol{W}| p(\boldsymbol{s}) = |\det \boldsymbol{W}| \prod_{m=1}^{M} p_m(s_m) = |\det \boldsymbol{W}| \prod_{m=1}^{M} p_m(\boldsymbol{w}_m^T \boldsymbol{x}) \tag{2.27}$$

where p_m is the unidimensional marginal distribution of the m-th independent component and, if we restrict ourselves to the complete model, $\boldsymbol{W} = \boldsymbol{A}^{-1}$. The last equality uses that if $\boldsymbol{w}_m$ is the vector corresponding to the m-th row of matrix $\boldsymbol{W}$, then $s_m = \boldsymbol{w}_m^T \boldsymbol{x}$. Remember that maximum likelihood searches for the parameter values that give highest probabilities to the observations so we now assume we have N observations of feature vector $\boldsymbol{x}$ which we note by $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N$, then, assuming sample independence, the likelihood $\mathcal{L}(\boldsymbol{W})$ is obtained as the product of (2.27),

$$\mathcal{L}(\boldsymbol{W}) = \prod_{n=1}^{N} |\det \boldsymbol{W}| \prod_{m=1}^{M} p_m(\boldsymbol{w}_m^T \boldsymbol{x}^n), \tag{2.28}$$

and the algebraically simpler log-likelihood,

$$\log \mathcal{L}(\boldsymbol{W}) = \sum_{n=1}^{N} \sum_{m=1}^{M} \log p_m(\boldsymbol{w}_m^T \boldsymbol{x}^n) + N |\det \boldsymbol{W}|. \tag{2.29}$$

Notice that this sum can be replaced by the (frequential) expectation operator if both sides of the equation are divided by the number of samples, yielding the following equivalent expression,

$$\frac{1}{N} \log \mathcal{L}(\boldsymbol{W}) = E\{\sum_{m=1}^{M} \log p_m(\boldsymbol{w}_m^T \boldsymbol{x})\} + |\det \boldsymbol{W}|. \tag{2.30}$$

There are several algorithms for maximizing this expression such as gradient methods, natural gradient methods, fixed-point algorithms or even an application of the expectation-maximization (EM) algorithm. By differentiating (2.30) with respect to $\boldsymbol{W}$ we obtain the following expression for the gradient,

$$\frac{1}{N} \frac{\partial \log \mathcal{L}}{\partial \boldsymbol{W}} = E\{\boldsymbol{g}(\boldsymbol{W}\boldsymbol{x})\boldsymbol{x}^T\} + \left(\boldsymbol{W}^T\right)^{-1}, \tag{2.31}$$

where g is the component-wise vector function whose components $1 \leq m \leq M$ are defined as,

$$g_m(s) = \frac{\partial \log p_m(s)}{\partial s} = \frac{1}{p_m(s)} \frac{\partial p_m(s)}{\partial s} \tag{2.32}$$

These functions are also called *score* functions of the distribution p. Equation (2.31) yields the following gradient descent iteration for ML estimation,

$$\Delta \boldsymbol{W} \propto E\{\boldsymbol{g}(\boldsymbol{W}\boldsymbol{x})\boldsymbol{x}^T\} + \{\boldsymbol{W}^T\}^{-1}. \tag{2.33}$$

This algorithm was first derived by Bell and Sejnowski [12] from an information theoretic approach that yields the same results, and further studied by several authors [119, 20, 115]. The main drawback of this algorithm is its slow convergence, both theoretically (as with most gradient descent procedures) and computationally (inversion of $\boldsymbol{W}$ is required on each step). This situation can be attenuated by using the natural or relative gradient, which amounts to multiplying the right hand side by $\boldsymbol{W}^T\boldsymbol{W}$. This algorithm makes use of the fact that, for our problem, the parameter space (nonsingular matrices) has a Riemannian instead of an Euclidean metric structure [3]. The natural gradient descent iteration for ML estimation is better conditioned than its gradient version,

$$\Delta \boldsymbol{W} \propto (\boldsymbol{I} + E\{\boldsymbol{g}(\boldsymbol{W}\boldsymbol{x})(\boldsymbol{W}\boldsymbol{x})^T\})\boldsymbol{W} \tag{2.34}$$

We still have not treated a basic and necessary condition for the implementation of any ML estimation procedure: the choice of the distributions. Since the independent components are themselves unknown, so are their distributions. The most common approach in this case is to restrict the densities to a particular family (taking a parametric approach). Of course, not any family since this choice affects the consistency of the estimator. Results for stability analysis and local consistency of ML for ICA estimation have been derived [4, 64] notably showing that accurate density estimation is not absolutely necessary so simple models can be applied. Based on these results several choices for the distributions have been proposed [51, 89, 134, 64].

Remember from the assumptions made that we should restrict ourselves to nongaussian densities. These densities can be further divided into two groups, *subgaussian* and *supergaussian*, based on the value of the fourth order statistic known as *kurtosis* defined as,

$$\mathcal{K}(s) = E\{s^4\} - 3 \tag{2.35}$$

for a zero mean, unit variance random variable (true for the independent components). Its value is proportional to the concentration of the variable around zero. It can be seen that $\mathcal{K}(s)$ is zero if s is Gaussian. From here that negative kurtotic variables are said to have subgaussian (or platykurtotic) distributions, and positive kurtotic supergaussian (or leptokurtotic) distributions. Kurtosis measures the *peakiness* of a distribution. In the range of unimodal distributions the uniform distribution can be considered the least "peaky", the Dirac delta its opposite. Having made this distinction, the simplest form of effective family densities for maximum likelihood ICA estimation have a single parameter [12, 64]: a binary parameter which decides whether the distribution of the m-th independent component is sub or supergaussian and, having decided this, assigns a predefined (sub or supergaussian) fixed density to s_m. The most extended moderately supergaussian density is the *Laplacian* or *double-exponential* density,

$$p_m(s_m|\alpha) = \frac{1}{\sqrt{2}\alpha} e^{-\frac{\sqrt{2}|s_m|}{\alpha}} \tag{2.36}$$

This density has the undesirable property of not being differentiable in the origin, so a smooth approximation of the logarithm of this density is introduced,

$$p_m^+(s_m) = \alpha^+ - 2 \log\cosh(s_m) \tag{2.37}$$

where α^+ is fixed in order to make the function the logarithm of a probability density. Replacing (2.37) in (2.32) we have that the score function for this choice is,

$$g_m^+(s_m) = -2\tanh(s_m) \tag{2.38}$$

For the subgaussian case, the following log-density is proposed,

$$p_m^-(s_m) = \alpha^- - \big(\frac{s_m^2}{2} - \log\cosh(s_m)\big) \tag{2.39}$$

where α^- is also a normalizing constant and the corresponding score density is given by,

$$g_m^+(s_m) = \tanh(s_m) - s_m \tag{2.40}$$

Of course, more precise parametric models can be studied for the component densities. A highly flexible model with the interesting property of joining the sub and supergaussian cases in a single parametrization is the result of using the *generalized Gaussian* distribution [134]. In this case, the component is assumed to belong to the following family of distributions,

$$p_m^G(s_m, \alpha_m) = \frac{\alpha_m}{2\lambda\Gamma(\frac{1}{\alpha_m})} e^{-\frac{|s_m|}{\lambda_m}{}^{\alpha}_{m}} \tag{2.41}$$

where the real positive number α_m controls the "peakiness" and is often referred to as the Gaussian exponent of the distribution, λ_m is depends on α_m and the variance (here fixed as 1), and Γ is the Gamma function given by

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt \tag{2.42}$$

The only parameter we need to estimate is the value of α_m for each component. The score function for (2.41)

$$g_m^G(s_m, \alpha_m) = |(s_m)|^{\alpha_m - 1} \mathrm{sign}(s_m) \tag{2.43}$$

In practice estimation of α_m can be done using the kurtosis of the corresponding component $\mathcal{K}(s_m)$ [134]. Figure (2.10) illustrates the distribution $p^G(s, \alpha)$ for different choices of $\alpha = (0.7, 1, 2, 4)$. Notice that $\alpha = 1$ corresponds to the double exponential

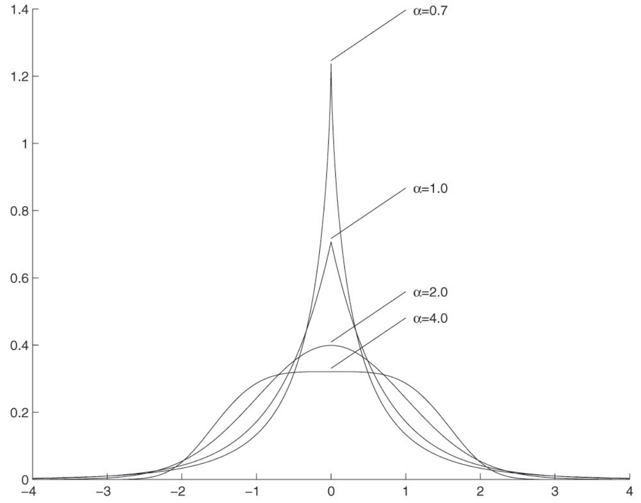


Figure 2.10: The generalized Gaussian distribution for different Gaussian exponents.

distribution and $\alpha = 2$ to the Gaussian. Also notice the aspect of the above introduced supergaussian ($\alpha = 0.7, 1$) and subgaussian ($\alpha = 4$) distributions.

Regardless if we choose g_m^+ and g_m^- or g_m^G, the stochastic natural gradient algorithm is very similar in both cases. For the first and simpler case, this algorithm is exposed in Table (2.1). Step 4 of the algorithm involves the selection of the sub or supergaussian score function for each component. We have mentioned that this selection is done based on the kurtosis value. Due to the choice of score functions, it is advisable to choose a more robust nonpolynomial moment of the component instead of using the polynomial moment based on s_m^4 (kurtosis). This will be further justified in the next section, when we discard kurtosis as a measure of nongaussianity, due to its sensibility.

2.4.2 Maximum nongaussianity ICA estimation

We now introduce an alternative algorithm for estimating ICA, based on maximizing the nongaussianity of the independent components [64]. ICA seeks a statistically independent representation. As we have seen in the last section, maximum likelihood estimation was directly based on the definition of independence. In this sense, the approach we will now present is indirect. We first need to show that nongaussianity maximization leads to independence, then introduce ways of measuring nongaussianity, and finally include these measures as the objective function of an optimization algorithm for estimating ICA.

In order to understand the relationship between nongaussianity and independence we first introduce the information theoretic concept of *mutual information*. The *differential entropy* of random vector $\boldsymbol{s} = (s_1, \ldots, s_M)^T \sim p(\boldsymbol{s})$ is defined as,

$$\mathcal{H}(\boldsymbol{s}) = -\int_{-\infty}^{\infty} p(\boldsymbol{s}) \log(p(\boldsymbol{s})) d\boldsymbol{s}. \tag{2.44}$$

```
1. Center the data $\boldsymbol{x}$ and randomly select the initial values for the parameters $\boldsymbol{W}$ and
   $\gamma_m$ for $m = 1, \ldots, M$. Also select the learning rates $\mu$ and $\mu_m$.
2. Compute $\boldsymbol{s} = \boldsymbol{W}\boldsymbol{x}$.
3. for all $m = 1, \ldots, M$
   (a) $\gamma_m \leftarrow (1 - \mu_\gamma)\gamma_m + \mu_\gamma E\{-\tanh(s_m)s_m + (1 - \tanh(s_m)^2)\}$
   (b) if $\gamma_m > 0$, use $g_m^+$, else use $g_m^-$
4. $\boldsymbol{W} \leftarrow \boldsymbol{W} + \mu\big(\boldsymbol{I} + \boldsymbol{g}(\boldsymbol{s}\boldsymbol{s}^T)\big)\boldsymbol{W}$
5. If not converged, go back to 2.
```

Table 2.1: The natural gradient algorithm for ICA maximum likelihood estimation

The entropy of a random variable can be interpreted as the degree of information that the observation of the variable gives. The more unpredictable and unstructured the variable is, the larger its entropy. We can now define

$$\mathcal{I}(s_1, \ldots, s_M) = \sum_{m=1}^{M} \mathcal{H}(s_m) - \mathcal{H}(\boldsymbol{s}) \tag{2.45}$$

as the mutual information. The Kullback-Leibler divergence provides an insight on this concept. For probability densities p_1 and p_2 it is defined as

$$\mathcal{KL}(p_1 \| p_2) = \int_{-\infty}^{\infty} p_1(\boldsymbol{s}) \log \frac{p_1(\boldsymbol{s})}{p_2(\boldsymbol{s})} d\boldsymbol{s} \tag{2.46}$$

It is an (asymmetric) distance between probability densities. If the frequential distribution is used, Kullback-Leibler can be understood as the likelihood under p_1 of the log-likelihood ratio between both distributions. Using Eq. (2.44) it can be seen that the mutual information is actually the Kullback-Leibler divergence between the joint and the marginal densities, resulting in a natural measure for independence. Mutual information is always positive and is zero if and only if the components of random vector $\boldsymbol{s}$ are independent. An important property for mutual information is the way it behaves under a linear transformation such as $\boldsymbol{s} = \boldsymbol{W}\boldsymbol{x}$

$$\mathcal{I}(s_1, \ldots, s_M) = \sum_{m=1}^{M} H(s_m) - H(\boldsymbol{x}) + log|\boldsymbol{W}| \tag{2.47}$$

Observing this formula one might ask, since it is such a fine measure of independence, why not use it directly as a cost function for estimating ICA. This is an alternative approach for estimating ICA, closely related to maximum likelihood. Its main drawback is that as with maximum likelihood, density estimation is required and mutual information is very sensitive to outliers. Nevertheless, if the logarithm term is neglected, this equation already suggests that the independent components correspond to directions in which differential entropy is minimized. The logarithm term is still an inconvenient, that arises from the fact that differential entropy is not invariant under affine transformations. The concept of *negentropy* can be used instead,

$$\mathcal{J}(\boldsymbol{s}) = \mathcal{H}(\boldsymbol{s}_{Gauss}) - \mathcal{H}(\boldsymbol{s}) \tag{2.48}$$

since it provides an invariant version of entropy [25]. Here, $\boldsymbol{s}_{Gauss}$ refers to the Gaussian random vector with the same mean and covariance as $\boldsymbol{s}$. It can be seen that the negentropy of $\boldsymbol{s}$ is the Kullback-Leibler divergence between $\boldsymbol{s}$ and $\boldsymbol{s}_{Gauss}$ and that, by replacing and using the change of variables theorem that negentropy results invariant for nonsingular linear transformations. Since the normal distribution has the maximum entropy of all distributions having a given mean and variance, negentropy is always positive and only zero if $\boldsymbol{s}$ has a Gaussian distribution: negentropy is a natural measure of nongaussianity. We will now relate nongaussianity with independence. If $\boldsymbol{s}$ is restricted to be uncorrelated, we have the following expression for mutual information in terms of negentropy,

$$\mathcal{I}(s_1, \ldots, s_n) = \mathcal{J}(\boldsymbol{s}) - \sum_{m=1}^{M} J(s_m) \tag{2.49}$$

We can now conclude, due to the invariance property, that finding maximum negentropy directions is equivalent to finding a representation in which mutual information is minimized. This can also be read as *independence is found in the directions of maximum nongaussianity.* And we have finally related independence with nongaussianity. In addition, we have an effective measure for determining the nongaussianity of a given Gaussian variable. For example, for all four distribution functions plotted in fig. (2.10) we have the following approximate negentropy values on increasing value of the Gaussian exponent α: $(0.20, 0.07, 0, 0.03)$. The second value is the negentropy of the Laplace density and the zero value corresponds to $\alpha = 2$, the Gaussian density.

In order to calculate negentropy, the densities of the independent components have to be estimated so the problems with mutual information hold for negentropy. Several approximations of entropy, negentropy or mutual information, such as Edgeworth and Hermite polynomial expansions, or cumulant based expressions can be used in this stage [2, 25, 68, 77]. In [62] it is argued that these approximations result quite inaccurate and are very sensitive to outliers so the following approximations for the unidimensional case were introduced and their efficiency tested,

$$\mathcal{J}(\boldsymbol{s}) \approx \|E\{G(\boldsymbol{s})\} - E\{G(\boldsymbol{\nu})\}\|^p \tag{2.50}$$

where G is a non-quadratic function, ν is a standarized Gaussian variable and p is usually chosen such that $1 \leq p \leq 2$, being 2 the most frequent choice and the one made from here on. The following prove to be efficient problem-dependent choices for

G

$$\begin{aligned} G_k(s) &= y^4 \\ G_t(s) &= \frac{1}{a}\ln(\cosh(as)) \\ G_e(s) &= \exp(-\frac{s^2}{2}) \end{aligned}$$

If G_k and $p = 1$ are used in Eq. (2.50) what we obtain is the modulus of kurtosis as can be seen from (2.35) . A direct approximation of negentropy by the modulus of kurtosis, although inaccurate, makes sense since kurtosis is also a widely used measure for nongaussianity. An additional result that we informally expose here makes kurtosis interesting for the estimation of the independent components. Assuming unit variance independent components $s = \boldsymbol{w}^T\boldsymbol{z}$ obtained from whitened data $\boldsymbol{z}$, and using the properties of kurtosis and (2.25), we have that

$$\mathcal{K}(\boldsymbol{w}^T\boldsymbol{x}) = \mathcal{K}(\boldsymbol{w}^T\boldsymbol{A}\boldsymbol{s}) = \mathcal{K}(\boldsymbol{z}^T\boldsymbol{s}) = \sum_{i=1}^{n} z_i^4\mathcal{K}(s_i) \tag{2.51}$$

Using this, we have that the optimization landscape for the problem

$$\max_{\|w\|=1} |kurt(\boldsymbol{w}^T\boldsymbol{z})| \tag{2.52}$$

has $2M$ local maxima, corresponding to the values $\boldsymbol{z} = \pm\boldsymbol{e}_m$, where $\boldsymbol{e}_m$ notes the m-th canonical vector. So one effectively obtains $\boldsymbol{w}^T\boldsymbol{x} = \pm s$ which is one of the independent components. The ambiguity of the model with respect to the sign is also observed in this result.

Kurtosis has been widely used in ICA, but it provides a poor estimator due to its sensitivity and asymptotic variance [64]. So, usually, approximations provided by functions other than G_k are the common choice.

Now that we have a measure of nongaussianity which is robust and easy to calculate from the observations since it is the expectation of a nonquadratic function, we can derive the algorithm that maximizes (2.50). All these algorithms are greatly simplified if whitened data is used, instead of domain space features. Besides the usual advantages of using white data, it can be seen that in this case the filter matrix $\boldsymbol{W}$ we seek to estimate has to be orthogonal since

$$\boldsymbol{I} = E\{\boldsymbol{z}\boldsymbol{z}^T\} = E\{\boldsymbol{A}\boldsymbol{s}\boldsymbol{A}\boldsymbol{s}^T\} = E\{\boldsymbol{A}\boldsymbol{s}\boldsymbol{s}^T\boldsymbol{A}^T\} = \boldsymbol{A}\boldsymbol{A}^T \tag{2.53}$$

so $\boldsymbol{A}$ is orthogonal and in consequence so is $\boldsymbol{W}$, its inverse.

In what follows we use vector notation instead of matrix notation for simplicity so we will actually show how to estimate a single component. We will turn back to matrix notation when detailing the final algorithm in Table (2.2) Vector $\boldsymbol{w}$ notes one of the M rows in filter matrix $\boldsymbol{W}$. A first approach could be to apply gradient descent directly in (2.50). If g is the derivative of the function G used in the approximation of negentropy, then it can be seen using the gradient of (2.50) that the following algorithm is obtained,

$$\Delta\boldsymbol{w} \propto \gamma E\{\boldsymbol{z}g(\boldsymbol{w}^T\boldsymbol{z})\} \tag{2.54}$$

1. Let M be the number of components to estimate, and g (G^t with $a = 1.5$ is a standard choice). Center the data $\boldsymbol{x}$ and whiten to obtain $\boldsymbol{z}$.

2. Randomly select the initial values for $\boldsymbol{W}$. Normalize its rows such that $\|w\| = 1$ and orthogonalize using step 4 below.

3. for all $m = 1, \ldots, M$

$$\boldsymbol{w}_m \leftarrow E\{\boldsymbol{z}g(\boldsymbol{w}_m^T\boldsymbol{z})\} - E\{g\prime(\boldsymbol{w}_m^T\boldsymbol{z}\}\boldsymbol{w}$$

4. Orthogonalize matrix $\boldsymbol{W}$ using, for instance, Gram-Schmidt,

$$\boldsymbol{W} \leftarrow (\boldsymbol{W}\boldsymbol{W}^T)^{-1/2}\boldsymbol{W}$$

5. If not converged, go back to 3.

Table 2.2: The FastICA algorithm, using symmetric orthogonalization.

with $\gamma = E\{G(\boldsymbol{s})\} - E\{G(\boldsymbol{\nu})\}$. Though it can be seen that, for particular choices of G and step value this algorithm is stable [64], it still has the inconvenients that can be found on any gradient descent procedure: the choice of the step on each iteration and its speed, generally very slow. Actually, this algorithm usually performs as slowly as the maximum likelihood approach introduced in the previous section.

In [65], Hyvärinen introduced the FastICA algorithm [63], which is a fixed-point iteration scheme for finding maximum nongaussianity directions using (2.50). This fixed-point algorithm is derived using an approximative Newton iteration. In this way, FastICA combines the computational efficiency of the fixed-point iteration with the desirable properties of negentropy. Details on the derivation of this algorithm can be found on [64]. In table (2.2) this algorithm is detailed in its symmetric and much more efficient form. For our particular purpose, this algorithm was a frequent choice since it proves fast, robust and is not affected by high dimensional data. When the dataset was yet too big for this algorithm the deflationary approach, also presented in [63], was taken. This approach, chooses to estimate independent components one by one. Though this simplifies the algorithm, it also causes accumulation of error along the iterations.

Before entering into the scope of applications of ICA we should make clear that there are several more algorithms that perform ICA estimation from a range of approaches. Among the most widely spread out algorithms we should mention the infomax principle algorithm, based on maximizing the output entropy and closely related to maximum likelihood [12]; JADE, which stands for Joint Approximate Diagonalization of Eigenmatrices, introduced by Cardoso and Souloumiac [21] based on properties of the cumulant tensor eigenvalues, is a very precise algorithm but inadequate for high dimensional data; the early efforts on nonlinear decorrelation [71]; Bayesian inference has also been used for estimation independent components using

Gaussian mixtures as conditional likelihoods [17]; or finally, ICA through nonlinear PCA which obtains independent components through the proper choice of a nonlinearity in the PCA minimum squared error criterion. Much work has been done on nonlinear PCA first used in the field of signal separation by [74]. An extensive review of this approach can be found in [52]. ICA estimation is by no means a closed object of research. By the time this thesis is being written, new estimation techniques are being proposed (for instance, product density estimation) or old techniques refined (for instance, variational Bayesian learning of ICA).

2.4.3 ICA Applications

Independent component analysis first irrupted as a solution to the problem of Blind Source Separation (BSS) and many of its practical applications are found within this field. BSS applied to medical engineering is mainly oriented to the problem of artifact identification. ICA has been successfully applied to this problem using signals coming from magnetoencephalographic and electroencephalographic recordings [70, 156], from cardiographic signals [10] and magnetoneurographic signals [162]. BSS through ICA has been applied to the field of telecommunications to solve the problem of multiuser detection in CDMA (code division multiple access) downlink [69] or to financial applications for detecting hidden factors in stock portfolio data [6]. ICA has not been exclusively applied to the problem of blind signal separation. Other applications include text document analysis [79], image denoising [59] and clustering [72].

Without doubt, the most influential application of ICA has been as a feature extraction technique. These results, though more theoretical than practical, have proven to be a major impulse for spreading ICA. The main interest of the features extracted by ICA is found on the close relationship between this technique and the representation principle known as *sparse coding*. Such coding of a given dataset should satisfy that only a small number of basis vectors are activated at the same time, or equivalently, most components of the coded data are zero, or close to zero, and only a few are significantly nonzero. In a neural network interpretation this means that, if each basis vector corresponds to a single neuron and its coefficient the corresponding activation, then we have that a given neuron is rarely activated in the network. It is said that such data has a sparse distribution. Notice that this distribution should have a strong peak in the value zero and heavy tails, so sparseness can be equated to supergaussianity (or, also equivalently, leptokurtosity). As we have seen, a suitable estimation of the independent components can be done by searching for nongaussian projections of the data. So, in those cases in which supergaussian projections are obtained, ICA provides a sparse coding of the data: ICA can be understood as a sparse coding technique. Of course, these nongaussian projections can derive into subgaussian components and no sparse coding is achieved, but this is not the case for a large set of interesting problems: those problems in which large amounts of redundancy is observed in the data.

Though sparse coding has practical utility in signal processing such as data compression and denoising, it was first developed as a model for image representation in the primary visual cortex of mammals (V1) [8, 39, 108]. In the late nighties, a highly successful experiment with natural images finally related sparse coding with V1 response through independent component analysis [155, 14]. This experiment consisted in randomly extracting patches of fixed size from natural images and using this data for ICA estimation. Results showed that the obtained basis filters have the three principal properties of simple cells in V1: they are localized, oriented and bandpass. We have reproduced this result by extracting 13000 patches from natural images. The patches were normalized on mean and variance and had the mean (an approximately flat image) subtracted. Dimensionality was reduced to 144 using PCA in a preprocessing stage to remove noise. Results of estimating ICA on this dataset using the algorithm in table (2.2) with the nonlinearity given by G^t are shown in fig. (2.11). In this figure we have sorted the basis vectors by their norm. Similar experiments to this of natural image patches have been performed on color and stereo images [60], video data [154], audio data [13] and hyperspectral data [111].

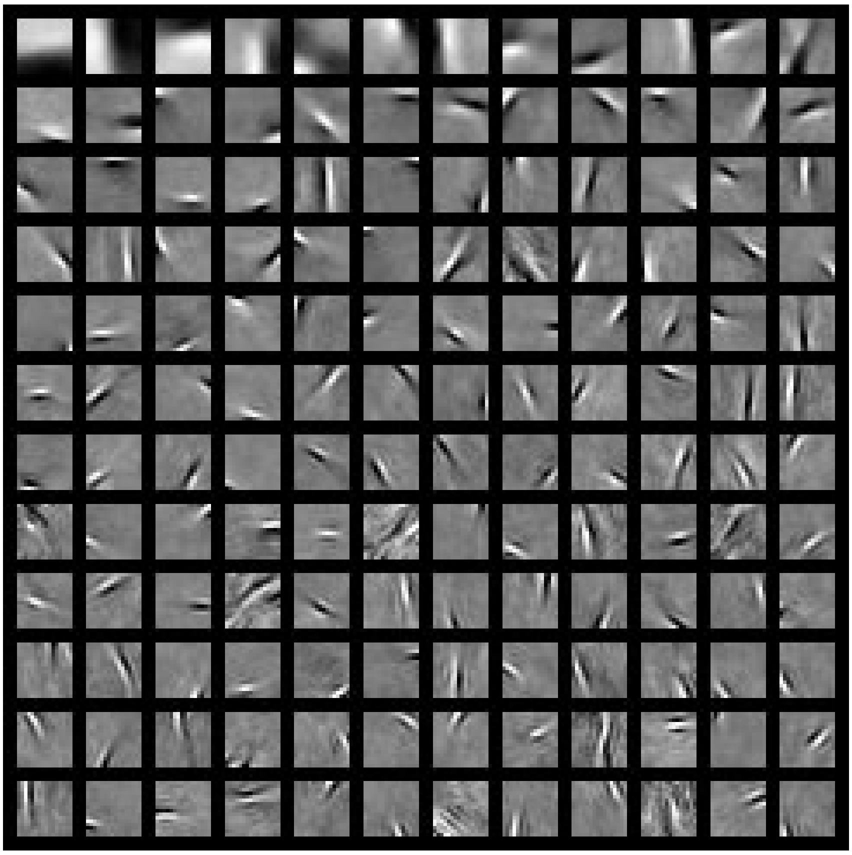


Figure 2.11: The ICA basis vectors for natural image patches are localized, oriented and bandpass.

ICA is not the only unsupervised learning technique that obtains a basis localized in space, frequency and orientation. It can be seen that the obtained basis are closely related to Gabor functions or to certain types of wavelets [37].

Traditionally, the sparse coding principle was associated to the information theoretic concept of redundancy reduction for compressive coding. On the other side, it has for long being defended that the coding strategy followed by the early visual system is adapted to the input statistics through combined evolution and neural learning [5, 7]. The emphasis of this learning process was biased towards the assumption that the cortex seeks to represent highly redundant sensory data efficiently, so sparse coding seemed a natural solution to the problem. A recent review on the evolution of these statements through the study of the statistics of natural images can be found in the paper by Simoncelli and Olshausen [137]. Quite recently Barlow [9] has strongly contributed to redirect the understanding of the mechanisms linking perception and redundancy, arguing that importance should be shifted from economy and efficiency to be set upon its contribution for building an accurate estimation of a probabilistic model for the environment. From this perspective, a compressed representation of sensory experience is doubtfully useful for the brain due to the unreliability of the estimates it can produce. Yet a representation with as little redundancy as possible is desirable since it allows easy access to the event probabilities and statistical dependencies among the responses. This change of emphasis favours the following reasoning: ICA provides a response similar to that present in the visual cortex not thanks to its sparse coding nature but thanks to the fact it reduces statistical dependencies providing a framework where probability estimation is greatly simplified. For the researcher involved in pattern recognition the conclusion is straightforward: why not use these simple yet accurate density estimates within a statistical classification framework? In the next chapter such a framework is introduced and validated on artificial and real world problems. Applications of ICA to classification others than ours are also detailed in this chapter.

2.4.4 ICA and PCA

The objective of this section is to illustrate through two short examples the main differences between ICA and PCA. For the first case we build an artificial dataset by mixing a 2-dimensional signal, with uniform distribution on each dimension $(x_1, x_2) \sim U[-1,1] \times U[-1,1]$. The mixing matrix can be randomly chosen. We apply ICA and PCA to this data and illustrate the results in fig. (2.12), where we can observe the original data, the principal and the independent components. ICA recovered the original, unmixed data since our problem satisfies all ICA assumptions: we have a linear mixture of statistically independent nongaussian data. On the other side, PCA did what it is expected to do, project the data in the directions of maximum variance. The point that concerns us is that, from this example, it is clear that density estimation within the ICA representation is much simpler than within the PCA representation, since we can accurately model the data using two unidimensional uniform distributions.

For the second example we will perform an ICA estimation on the face dataset used in section (2.2.2). ICA has been applied to face representation with a success in face recognition and detection similar to that of PCA [11]. This is an application of ICA to classification so the details will be delayed to the next chapter. Our main

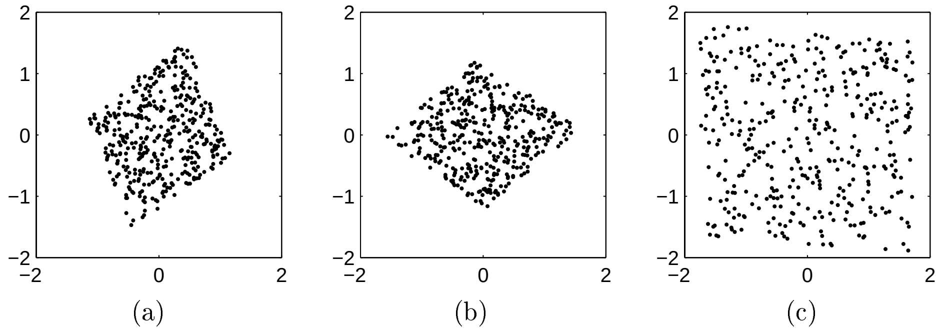


Figure 2.12: (a) Mixed 2-dimensional uniform distribution. (b) Principal Components. (c) Independent Components.

interest now is to understand the differences of ICA with PCA and NMF solely as a representation technique. In fig. (2.13) we show the basis vectors for the ICA representation. In this case, we have first whitened the data using PCA and reduced dimensionality to 128, preserving 94.9% of the data variation. Basis vectors were ordered by their norm (from higher to lower) resembling the order given by PCA and the first 64 vectors are shown. In the bottom row of fig. (2.13), a sample face is shown, with an image representing the component activation for this particular face. Positive values are illustrated with red pixels and negative values in with black pixels. We notice here the sparsity of the representation: a certain sample only activates a few components since they are zero most of the time. Components and basis vectors have geometrical correspondence in the images. Also, in fig. (2.13(c)) we show the result of retroprojecting the components into domain space. In this case, we can observe that the approximation is less accurate than the one obtained with a PCA space of similar dimensionality.

The resulting basis vectors provide an holistic representation similar to the eigenfaces. Still, some differences with the eigenfaces can be observed. Firstly, faces are more prototypical, as we will see this is connected to the sparse nature of the independent components. There are several components which account for apparently unrelated (statistically independent) illuminant changes. Also, certain bases are observed to capture variations not isolated by PCA. For example, the first basis in the second and sixth rows involve geometric transformations of the face. This also shows that frontality in the original samples is not always true. We can go on extracting heuristic conclusions from the exposed basis but practical use of these results is by no means straightforward. Instead we would rather focus on differentiating ICA from PCA by the distribution of the obtained components. In fig. (2.14) the histograms of a typical independent component (a) and a typical principal component are shown (b). The first histogram is clearly supergaussian while the latter is almost Gaussian. Another way of measuring sparsity of the whole representation is through the estimated mean kurtosis value of the components. For the independent components this value is 8.47 and for the principal components 0.49. As expected, the independent components are "more" nongaussian than the principal components.

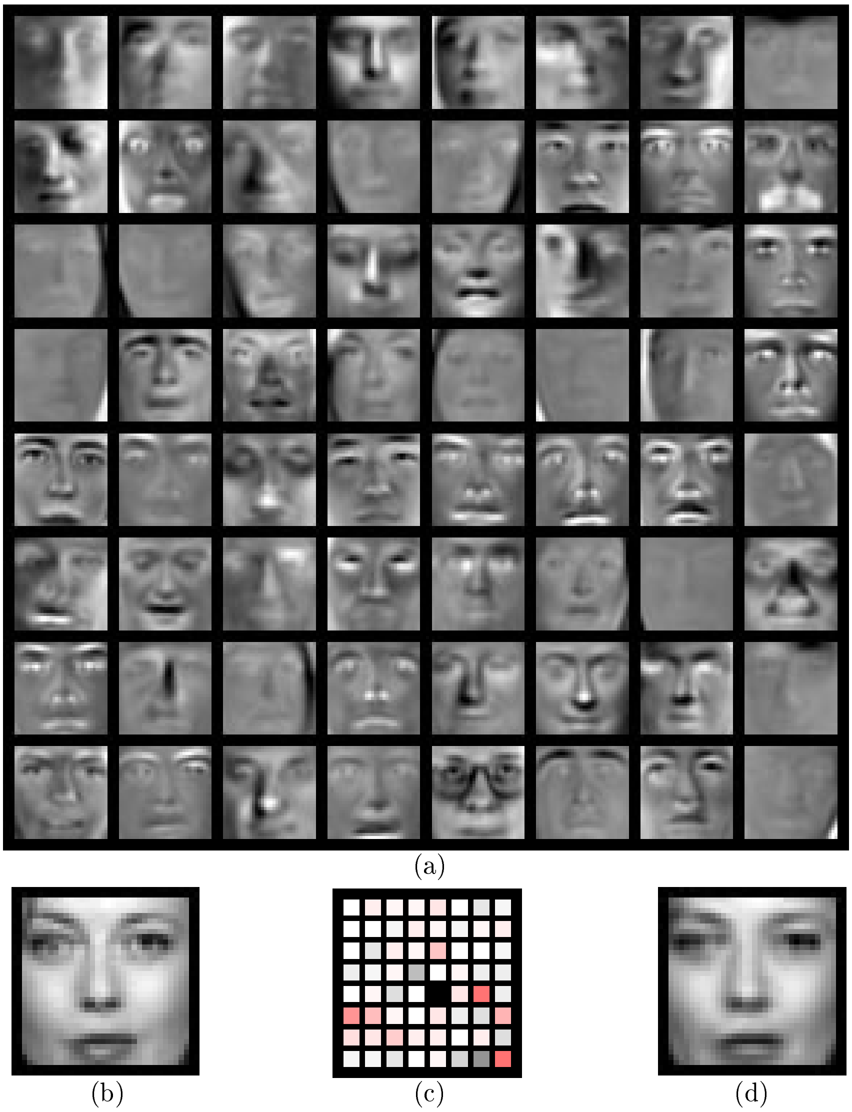
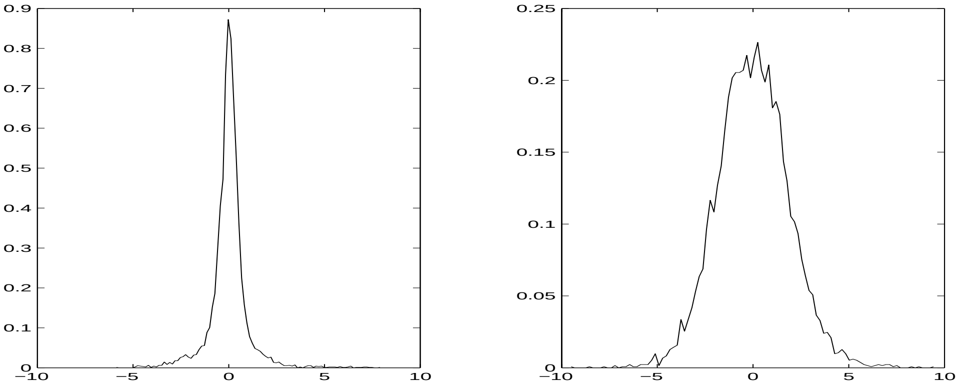


Figure 2.13: The first 64 ICA basis vectors computed from a training set of 7000 frontal view face images (a). All basis images are normalized between 0 and 255 such that pixels corresponding to zero basis values have an intensity of 128. In the bottom row a sample face (b) with its component values (c) and the result of retroprojecting the components (d). Positive values in the components in (c) are represented in red, negative values in black (notice the sparsity of the components).



(a) (b)

Figure 2.14: Typical distributions of: (a) the independent components of faces, (b) the principal components of faces.

2.4.5 ICA and NMF

Though efforts have been made towards including the nonnegativity constraints within the ICA framework [120], these efforts have been mostly theoretical and beyond the scope of this thesis. Nevertheless, ICA and NMF can be connected, not through nonnegativity, but rather through the sparsity of the basis. Remember that NMF results in nonnegative sparse basis vectors: their values are mostly zero except for a few components. In the case of ICA this occurs with the components. A workaround through this dual situation in order to obtain a sparse basis can be achieved by reversing the roles of samples and dimensionality. In the faces example from the preceding section, each sample corresponded to a face image such that independence could be assumed on the components and the basis vectors could be considered images themselves. This is also called a factorial code architecture. If instead, we consider each sample as the value of all face images at a certain pixel, and apply ICA to this new representation, we would obtain a representation for each face in terms of sparse independent basis images. In practice this change of architecture turns out to be straightforward, instead of using dataset $D \times N$ matrix $\boldsymbol{X}$, use $\boldsymbol{X}^T$ to estimate the ICA model. In this new situation we have a D N-dimensional samples. Since in general $N >> D$ this is not good for ICA estimation, or practically any learning technique. This problem is overcomed by performing ICA on a set of M linear combinations of the N images. Since the model assumes that the images are a linear combination of unknown statistically independent sources, it should not be affected by this change. A frequent strategy for deciding over these M linear combinations is to choose the M principal component vectors of the image set [100]. If $\boldsymbol{V}$ is the $D \times M$ matrix containing the first M PCA eigenvectors, then $\boldsymbol{X} \approx \boldsymbol{V}\boldsymbol{Y}$. In our new ICA model, we have estimated a mixing matrix and independent components such that $\boldsymbol{V}^T = \boldsymbol{A}\boldsymbol{S}$, so we have that $\boldsymbol{X} \approx (\boldsymbol{A}\boldsymbol{S})^T\boldsymbol{Y} = \boldsymbol{S}^T(\boldsymbol{A}^T\boldsymbol{Y})$. Notice that our components are the basis vector in this equation, and the components of the original images in this basis are given by $\boldsymbol{A}^T\boldsymbol{Y}$, with $\boldsymbol{A}$ the mixing matrix for our model and $\boldsymbol{Y}$ the principal components. We have obtained a sparse and statistically independent basis for our model at the cost of losing independence in the components. Actually they are not even uncorrelated. Mean correlation can be used for imposing a certain ordering in the basis images: we are interested in preserving those components with low absolute correlation among each other.

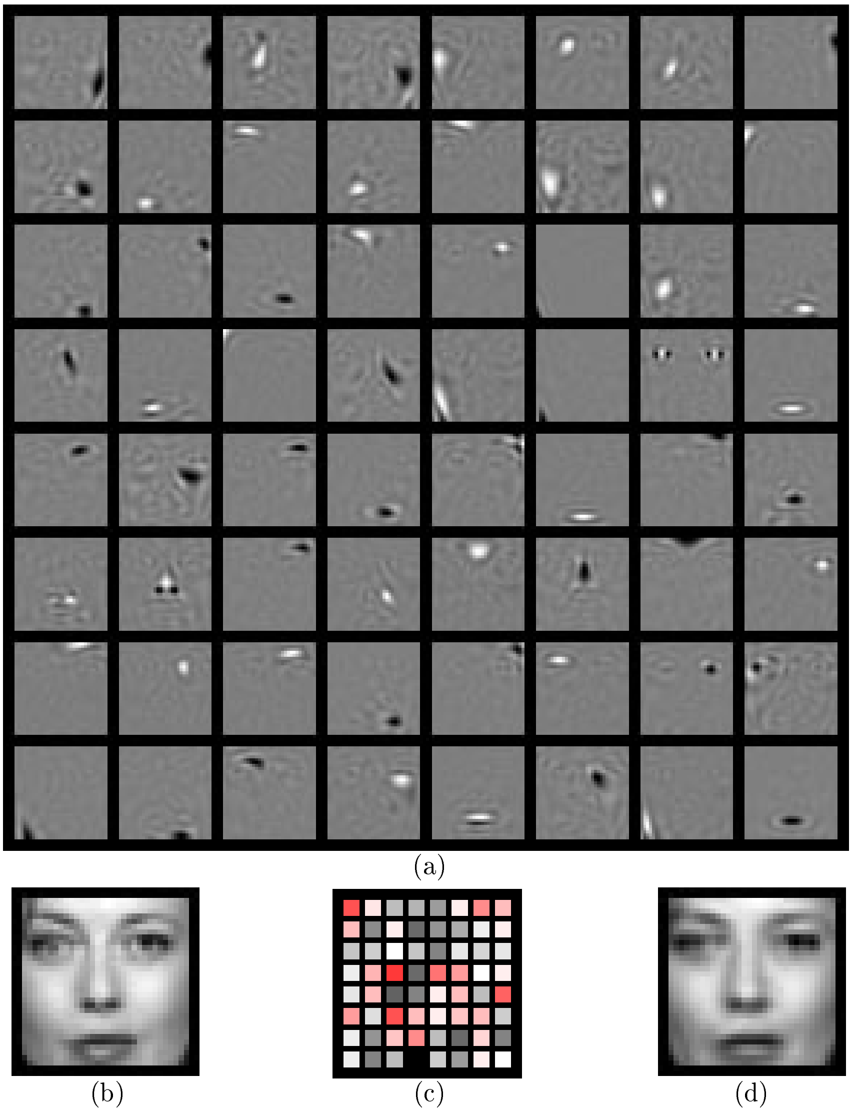
Both approaches have been examined in [11] for the task of face recognition using a K-NN classifier with angle distance, with similar results when compared among themselves and with PCA. Our interest here is focused on the fact we can obtain a sparse basis for representing faces: a basis which is localized in its nature, with all the advantages of a local representation. Results presented in [11] confirm these advantages: when attempting to recognize people, this new approach performed similarly to the more classical approach, but when the people from training and test had a different expression in their faces, the new approach outperformed the classical approach. Locality isolates this expression in few components, diminishing its effect on the distance measure.

Fig. (2.15) illustrates 64 independent basis images from the same dataset used in all our face experiments. These images were estimated from the first 128 eigenvectors. The order was chosen in terms of mean correlation of the components. In terms of density simplification this representation is useless to us since we have no reasons to assume independence on the resulting components. In the bottom row of fig. (2.15), a sample face is shown, with an image representing the component activation for this particular face. Positive values are illustrated with red pixels and negative values with black pixels. In this case, having obtained a sparse basis, nearly all of them are required in order to represent a certain sample. Components and basis vectors have geometrical correspondence in the images. Also, in fig. (2.15.c) we show the result of retroprojecting the components into domain space. In this case, reconstruction is accurate since it should preserve approximately the same error as PCA.

2.4.6 Experiment: Independent Modes of Variation

In this section we will illustrate the representational differences between PCA and ICA through their application to shape modeling using Point Distribution Models. The Point Distribution Model (PDM) [27] is a shape description technique based on the vectorized representation of shapes to estimate a statistical model for non-rigid shape variation. By modeling this distribution, we can generate new examples, similar to those in the original training set, and we can also examine the plausibility of new shapes. It has been seen that this model succeeds in the treatment of non-rigid shapes, their analysis and synthesis. The statistical modeling for shape variation, and its combination with several image processing techniques has generated an important number of applications in the last years. These applications include tracking, recognition, biomedical imaging, special effects for film and television and registration among others [56, 144].

The construction of an appropriate PDM for a certain type of shape we wish to learn, requires both the selection of a good representation and of an appropriate density estimation method for the distribution of the shapes within this representation. For the representation we can use linear or nonlinear models. As usual, if



(a)

(b)

(c)

(d)

Figure 2.15: 64 ICA independent basis images computed from the first 128 eigenvectors of the covariance matrix of 7000 frontal view face images (a). All basis images are normalized between 0 and 255 such that pixels corresponding to zero basis values have an intensity of 128. In the bottom row a sample face (b) with its component values (c) and the result of retroprojecting the components (d). Positive values in the components in (c) are represented in red, negative values in black (notice the non-sparsity in the components).

we use a nonlinear model we can relax the hypotheses and obtain higher reliability. This precision has a high cost in the training stage due to the undeterministic characteristic of nonlinear models and also in the application stage, due to the fact that nonlinear algorithms are generally computationally expensive. This is justifiable for a large number of problems and several nonlinear models have been proposed [141, 142, 57]. Even though, a linear representation is still a common choice for their speed and straightforward interpretability. As a matter of fact, most of the nonlinear representations are applied over a linear representation which previously performs the dimensionality reduction. On the other side, even when the training set generates complex distributions, a linear representation can be used and complexity charged to the statistical model [26]. The most successful linear representation so far, for its simplicity and straightforward interpretation, is the one obtained through Principal Component Analysis (PCA). By projecting a shape in a previously learnt PCA space, we have a set of coefficients or parameters (the principal components) which control the variation along the maximum variance directions. So we can naturally associate each principal component to a mode of variation of the shape. Nevertheless, the assumption that these projections are optimal for modeling shape deformations is not necessarily correct. In certain cases (the fingers of the hand can prove to be a good example) higher order relationships such as independence can be important for better modeling. In this section we propose an alternative linear representation of the PDMs using Independent Component Analysis (ICA). This representation can be particularly interesting for those non-rigid shapes whose modes of variation are supposed statistically independent. We will refer to such modes as *independent modes of variation*. Independence also simplifies any kind of statistical estimation, providing direct solutions for the problems related with shape feasibility. These problems arise in several PDM applications, notably in Active Shape Models. Within this context, independence also has a more profound meaning. In a hand, the position of the thumb gives us no information on the position of the other fingers. This direct relationship between the independent components and shape deformations would allow robust tagging and tracking. It is also observed that ICA provides a sparse representation for the data, so we will make use of this fact together with the independence assumption for illustrating how the ICA representation can also provide, in an unsupervised way, a set of shape prototypes useful for searching shapes by similarity to conditional combinations of these prototypes. We will first outline the basics of point distribution models and the way ICA can be introduced as a representation of these models. We then state the solution to the most common problems when dealing with shapes, assuming they are represented by their independent modes of variation. We finally present some experiments. A first experiment, over an artificial dataset, illustrates the difference between uncorrelated and independent shape deformations, and compares the accuracy of the density model obtained from the PCA representation using classical techniques against our proposed methods. A second experiment, over a low dimensional PDM for hands will confirm what we mentioned on the thumb. A final experiment was performed, this time on a PDM extracted from the Squid System database for fish boundaries [104]. The results of an ICA representation for this PDM is exposed and it is also shown how the independent modes of variation can be used for performing queries on the fish database.

Point Distribution Models

If we use J points to describe a certain shape in R dimensions, we can represent this shape by a $D = JR$ dimensional vector by simply concatenating the point position values. Given N samples of a certain shape, we choose certain locations as key points or "landmarks points", and obtain N vectors representing each shape of the training set. In the choice of the key points three important issues should be considered: sampling, correspondence and alignment. The number of keypoints should be sufficiently high as to capture the shape with its full complexity. From here that complex shapes need to be studied in possibly high dimensional spaces. Correspondence means that, in different shape samples the same keypoint should correspond to approximately the same part of the shape. This is essential for the modeling of the deformations. Alignment seeks to relax the restriction on the training set, allowing to a certain extent affine transformations between different samples.

While sampling and correspondence are considered when gathering the data, alignment is performed on the training set. Procrustes method [53] or modifications are frequently used in this stage. The selection of a correct criterion for alignment should not be underestimated since these operations will greatly affect the final distribution by introducing or avoiding nonlinearities. For two dimensional PDMs to be used in Active Shape Models, Cootes [27] suggests aligning by minimizing a sum of squared distances. Mathematically, the expression $T_{\boldsymbol{t},\theta,\sigma}(\boldsymbol{y})$ represents the application to shape $\boldsymbol{y}$ of a translation by $\boldsymbol{t}$, a rotation by θ and a scaling by σ. Shape $\boldsymbol{y}$ is said to be aligned with reference shape $\boldsymbol{y}_{ref}$, if T minimizes the expression

$$|T_{\boldsymbol{t},\theta,\sigma}(\boldsymbol{y}) - \boldsymbol{y}_{ref}|_{\boldsymbol{Q}} \tag{2.55}$$

Where $|\boldsymbol{y}|_{\boldsymbol{Q}} = \boldsymbol{y}^T\boldsymbol{Q}\boldsymbol{y}$ and $\boldsymbol{Q}$ is a diagonal matrix of weights for each point. An iterative method for aligning a set of shapes is also provided in the cited article. For the rest of this paper we will assume that our aligned training set is a sample of random vector $\boldsymbol{x}$.

The next step is to find a proper representation for $\boldsymbol{x}$. In the choice of the representation simplicity, dimensionality reduction, statistical properties and interpretability should be considered. As mentioned, a frequent approach that fulfills all these requirements is PCA. Notice that the covariance matrix of our data, from a simplified point of view, tells us the way each landmark tends to move, as the others move. PCA would give us then uncorrelated directions of maximum variance, and using this technique a set of parameters for the shape can be defined then by

$$\boldsymbol{y} = \boldsymbol{V}(\boldsymbol{x} - \bar{\boldsymbol{x}}) \tag{2.56}$$

where $\bar{\boldsymbol{x}}$ is the data mean, $\boldsymbol{V}$ the $M \times D$ PCA basis matrix, M a chosen dimensionality and $\boldsymbol{y}$ the principal components. The choice of M can be done, for instance, using (2.8).

We now choose a proper statistical density model for our shape representation and address the problem of examining the plausibility of new shapes or equivalently, generating new examples within the model. For each model we also expose the solution for the problem of, given a certain shape, finding the nearest feasible shape within our model.

A reasonable definition for shape plausibility is presented by Cootes in [27]. If we have estimated, from the training set, the distribution of the parameters $\boldsymbol{y} \sim p(\boldsymbol{y})$, a shape with parameters $\boldsymbol{y}$ is said to be feasible if

$$p(\boldsymbol{y}) > p_t \tag{2.57}$$

where p_t is a certain threshold we consider appropriate. Since a threshold value based on the likelihood is not recommendable, it is usually chosen so that some proportion of the training set passes the threshold. If the parameters $\boldsymbol{y}$ are assumed Gaussian and independent, a natural choice under the PCA representation, we have that

$$\log p(\boldsymbol{y}) = -\frac{1}{2}\sum_{m=1}^{M}\frac{y_m^2}{\lambda_m} + C \tag{2.58}$$

where λ_m corresponds to the m-th eigenvalue of the covariance matrix and C is a constant. In this case, the threshold represents a likelihood which constrains feasible shapes to a hyperellipsoid. The size of the hyperellipsoid can be obtained considering that the sum of squared Gaussian variables has a chi-squared distribution,

$$\sum_{m=1}^{M}\frac{y_m^2}{\lambda_m} \sim \chi^2(M) \tag{2.59}$$

From (2.59) we can, given a certain probability value, obtain the desired threshold p_t. In finding the nearest feasible shape we first check the likelihood. If it is lower than our threshold, then our current shape is not feasible. The nearest feasible shape is the closest shape belonging to the hyperellipsoid boundary.

Another approach, is to choose hard limits on each direction, also proposed in [27]. This is related with the idea of statistical independence of the components, and is equivalent to constraining feasible shapes to a hypercube. A good heuristic value for the threshold on each direction is 3 times the standard deviation on that direction. If we assume a Gaussian distribution on each direction, this choice of limit values means that a shape is plausible if it belongs to the symmetrical mean-centered interval which has a marginal probability of 0.997. In this case, for each $m = 1, \ldots, M$, the feasibility of a shape is checked by $|y_m| < 3\sqrt{\lambda_m}$, and the nearest feasible shape is obtained by

$$b_m^F = \text{sign}(y_m) * \min(3\sqrt{\lambda_m}, |y_m|) \tag{2.60}$$

If a simple Gaussian estimation is not enough we can use more complex models such as Gaussian Mixture Models (GMM), a particular case of (1.19). In this case, the plausibility of a shape is a more complex problem. A simple and frequent solution consists on deciding that a shape is plausible if its likelihood is above the likelihood of a certain percentage of shapes in the training set. The percentage value is generally above 80%. When using a GMM, a general solution for the problem of finding the nearest feasible shape is not available so Monte Carlo and gradient descent methods are employed. Moreover, estimating the GMM parameters on high dimensions is a highly unstable problem.

ICA for PDMs

Assuming we have learnt the mixing and filter matrix for ICA, the independent components $\boldsymbol{s}$ are obtained as for PCA

$$\boldsymbol{s} = \boldsymbol{W}(\boldsymbol{x} - \bar{\boldsymbol{x}}) \tag{2.61}$$

We will call $\boldsymbol{s}$ the *independent modes of variation*, and assume that the components s_m, $i = m, \ldots, M$ are statistically independent. We also assume, as usual, that they have unit variance. The choice of dimension is not as straightforward as in PCA, where a natural hierarchy arises from the corresponding eigenvalues. In the shape problem it is logical to assign small variances to errors in the labelling process so, when required we will reduce dimensionality by first performing a PCA, respecting a certain percentage of the variance, and then ICA. Since our ICA model estimation algorithm needs whitened data, and PCA succeeds in whitening the data, this seems a natural choice. Concerning the order of the independent components, we have mentioned that it is contradictory to privilege any single component over the rest due to their independence. So the hierarchy should be decided in terms of the task to be performed, in our case the shape deformation associated with the independent mode of variation.

Regarding the density model to be employed, some considerations on shape distributions can be made. When preparing a training set for the generation of a PDM there is a tendency towards generating uniform distributions of the modes of variation. This bias favours subgaussian distributions. On the other side, a shape with a preferred state and seldom deformations will have a sparse or supergaussian distribution. Symmetrical and gradual deformations of a part of the shape correspond to continuous symmetrical distributions. When a shape can be found in different states, and seldom in the intermediate positions, multimodality appears. Multimodality can be also introduced by the incorrect identification of landmark points. We conclude that our density model should be open to both sub and supergaussian distributions, but particularly the first. Symmetry is very frequent, and it should also include multimodal densities, unless we have other prior knowledge. When testing feasibility, the likelihood of the shape is not relevant and geometrical considerations are more important than statistical ones. Still, the statistical model is useful for answering, for instance, how feasible a shape is. In our case, unidimensional density estimation was performed using GMMs when possible, and RBF kernel-based methods otherwise.

As in PCA, the plausibility of a shape can be decided by evaluating its likelihood against a threshold value. In this section, we introduce a simple density-independent method of feasibility intervals for selecting the threshold which also provides a way for efficiently analysing shape plausibility and solves the problem of finding the nearest feasible shape. Suppose we have estimated the density for each of the independent modes of variation with one of the methods suggested above so that $s_m \sim p^m(s)$.

Given a certain probability value P_t between 0 and 1, let $p_t = P_t^{\frac{1}{M}}$. For each component there exists a union of disjoint intervals

$$I^m = [a_1^m, a_2^m] \bigcup [a_3^m, a_4^m] \bigcup \cdots \bigcup [a_{2t_m-1}^m, a_{2t_m}^m]$$

such that for all $m = 1 \dots M$ I^m satisfies

$$\int_{I^m} p^m(s)ds = p_t$$

and

$$p(a_i^m) = l^m, \qquad\qquad \forall i = 1, \dots, 2t_m$$

Once we have the set of intervals for each component, using the assumption of independence, it can be seen that

$$\int_{I^1 \otimes \dots \otimes I^M} p(\boldsymbol{s})d\boldsymbol{s} = \prod_{m=1}^{M} \int_{I^m} p^m(s_m)ds_m = \prod_{m=1}^{M} p_t = P_t \tag{2.62}$$

The existence of these intervals is observed constructively. We first assume the probability density is continuous in $\mathbb{R}$. Given the likelihood value $\mathcal{L}$, it can be seen that if the line $y = \mathcal{L}$ intersects the function $y = p^m(s)$ it has to be in an even number of points. These points determine the interval borders. If the intersection is empty, we define I^m also as the empty set. The method consists in starting at a likelihood above the maximum and decreasing the likelihood value, thus increasing the probability, until the threshold is reached. Of course, this solution is not unique, but this construction ensures the inclusion of global maxima.

For certain parametric and semi-parametric models the interval borders can be obtained analytically, but still, the constructive approach, through bisection and similar techniques is fast and accurate. In practice, this is performed in the training stage. Any algorithm working on new shapes will need only the interval information for plausibility tests. Dividing each direction in intervals divides the whole space into "hyperboxes" which have a geometric distribution reminiscent of that which arises from separable functions. In fig. (2.16) the joint distribution of two independent directions obtained in experiments are plotted. Each marginal density (clearly bimodal) was estimated with a kernel-based method. The product of the marginal densities is plotted with gray levels and contour lines. The rectangular boxes represent the Cartesian product of the intervals estimated for each direction for $P_t = 0.95$.

Working with these intervals has several advantages. Since the intervals are obtained in the training stage, all complex algebraic operations are removed from the working algorithms. This is because there is no need for the calculation of likelihoods once we have the interval limits. This interval structure also provides precision. It can be seen in fig. (2.16) that, if we decided to use a GMM, the only way to improve the estimation would have been using more than four components in the mixture. This is not problematic in a two dimensional context but can worsen as dimensions increase and we have no prior knowledge of the structure. Additionaly a GMM would need complex calculations throughout the algorithm.

In the interval context, plausibility is easily checked by first projecting the shape in the parameter space and then by verifying if $s^m \in I^m$ for all $m = 1, \dots, M$.

Using the feasibility interval notation, given the intervals I^m, and a shape with independent modes of variation s_T^m, the nearest feasible shape $\boldsymbol{s}_F$, with components s_F^m is

$$s_F^m = \begin{cases} s_T^m & \text{if } s_T^m \in I^m; \\ \arg\min_{1 \leq i \leq 2t_m} |s_T^m - a_i^m| & \text{otherwise.} \end{cases} \tag{2.63}$$

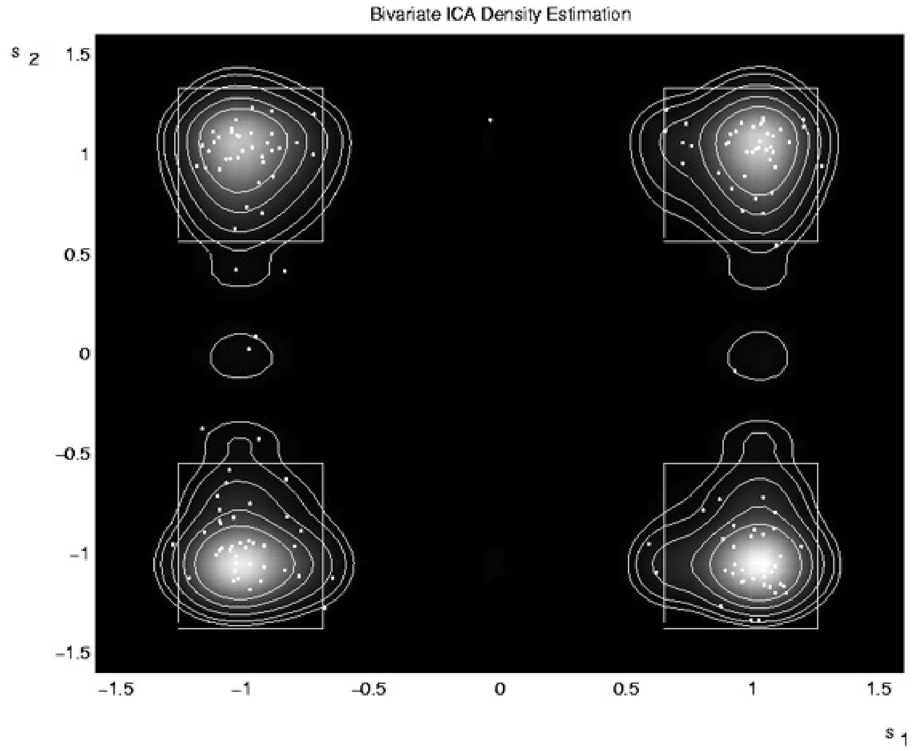


Figure 2.16: ICA intervals for testing shape feasibility. The curves represent contour lines of the density estimation (obtained with a kernel method), and the rectangles represent the Cartesian product of the intervals.

We can also see that the sparsity of the independent components results in useful shape prototypes. Sparsity means that when projected in the ICA space, a shape responds positively or negatively to only a few components and has almost zero values for the remaining components. On the other side, experiments confirm that the independent modes of variation perform a clearer and much more intuitive separation than PCA of the shape deformations. These two facts suggest that, if any shape can be described with few components and, the effect of these components is clearly distinguishable, then a shape query can be performed by the selection of a few prototypes from the set of M prototypes that represent the upper and lower values of each independent component. In the experiments a heuristic rule was employed to transform the prototype selection into an actual query. This rule searches for the shapes that have high response to all selected prototypes without combining their values in an attempt to avoid the problems arising from an incorrect normalization. Suppose we have selected prototypes $P = p_1, ..., p_L$, where $p_l = \pm m$ with $1 \leq l \leq L, 1 \leq m \leq M$ and the sign of p_l depends on which of the two prototypes provided by each mode of variation was chosen. We then sort the values of the independent modes in our database, so let $R_{|p_l|}$ be the permutation of $1...K$ that sorts modes $s_{|p_l|}$ in descending on ascending order, depending on $sign(p_l)$. We then define $R^j_{|p_l|}$ as the first j elements in the permutation. Finding the NUM shapes closest to our prototypes consists in finding the minimum j, such that

$$\#(\bigcap_{l=1}^{L} R^j_{|p_l|}) = NUM \tag{2.64}$$

In this case, our NUM shapes are those whose indexes belong to the expressed intersection. What we are doing is finding the shapes that have the highest value in *all* the modes selected for prototypes, regardless of this value. We have also tested the combination of the modes for generating a single mathematical expression in order to perform the query (i.e. L_1 distance to the origin). Results were poor, when compared to this approach, probably due to the strong difference in the values of the outliers of a mode (the outliers is precisely what we are looking for). In our third experiment, there is a practical application of this technique.

Experiments

In order to make clear the differences between the PCA and the ICA representation, an artificial set of shapes was created. In each shape we used 19 points to describe a fixed base and three deformable extensions of fixed length (see figure (2.17)). Each extension can be found rotated in an angle between $-\frac{\pi}{4}$ and $\frac{\pi}{4}$. We generated a training set of 400 shapes, by randomly choosing the angle corresponding to each extension. We have then, three independent degrees of freedom for each shape. The alignment step was skipped since the shapes were already aligned when created. Only centering and appropriate rescaling was necessary. Figure (2.17.a) shows the three principal modes of variation as we deviate from the mean along each component. PCA decorrelates the movements but does not take in account statistics of higher order. The decorrelated movements have no relationship with the degrees of freedom chosen in the creation of the shapes. Instead, Figure (2.17.b) shows the three independent modes of variation. We observe how ICA separated the deformations corresponding to each of the extensions. If the original problem would have been to classify a certain shape according to its deformations, it is observed how ICA would have successfully solved this problem. Also, the generation of prototypes that represent, the deformation of a single extension is obtained in a completely unsupervised way.

Experiments were also performed on a set of shapes representing extended hands. These hands were described by 11 points each and were obtained from an image dataset not specifically generated for the shape problem. Only hands with extended fingers were considered and these were found in many positions, distances and planes from the camera, totalling 180 hands. The small amount of chosen points results in a naive hand descriptor since $D = 22$, but a more complete set of landmark points would result in a higher dimension and the ICA model would no longer be trustable due to the small number of samples. This can be solved by increasing the number of samples. In this case correspondence was exact since landmark points were chosen in exactly the same position for all hands. After alignment, the PCA representation captures 95% of the data variation in a 5-dimensional subspace. In fig. (2.18.a) we

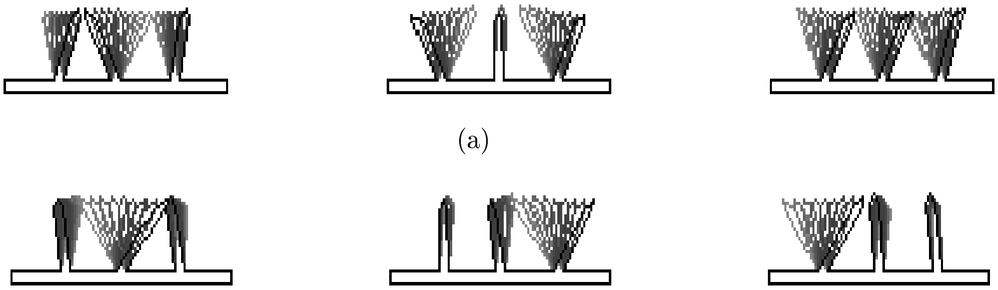


Figure 2.17: Three first modes of variation for artificial shapes using (a) PCA, and (b) ICA.

observe the five principal modes of variation. The first two modes capture practically all the hand movement, mixing the finger deformations, except for the index in the first mode. The remaining three components capture the uncorrelated variation of groups of one or two fingers. In all, except maybe for the second mode, the variations here presented do not resemble any kind of realistic han movement. Figure (2.18.b) shows five independent modes of variation, corresponding to the ICA representation of the shapes, when performed over the previous representation. It has been observed [121] that an excessive dimensionality reduction can bring up corrupted independent components. Even though this seems to be the case, we observe some interesting differences with the principal modes of variation. ICA has isolated the thumb, which clearly is the only single finger we can independently displace in our hand. Second and fourth modes illustrate the dependence of the ring and little finger, and of the index on the thumb, respectively. Third and fifth modes correspond to second and third principal modes respectively.

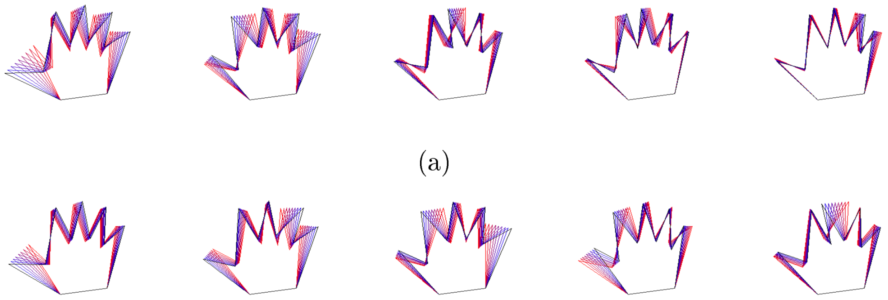


Figure 2.18: (a) The five principal modes of variation for set of hand shapes. (b) The five independent modes of variation, for the same set.

A third experiment was performed using the online available fish contour database developed for the Squid System [104]. This database contains the boundaries of 1100 different fish from diverse ecosystems. We first selected a subset of 758 elements, leaving out those outliers that would most seriously affect our correspondence by lacking clear landmark points for fins, tail, head, fins, etc. These outliers were basically eels, morays, sea horses or sting rays. Sixty nine landmark points for generating the PDM were sampled from each boundary by selecting twelve representative points corresponding to head, dorsal fin, tail and lower fin and interpolating the remaining points. This reduced number of keypoints results in a loss of complexity but is a compromise between the resulting dimensionality and the available number of samples in order to obtain a trustable ICA model. We are also conscious that interpolation can cause problems with correspondence. Considering our lack of ichtyologic expertise, any other choice of keypoints would have probably had this problem so our approach was chosen for its speed and satisfactory results. We then aligned the shapes, and performed both a PCA and an ICA on the aligned shapes. As expected, the two principal modes of variation captured the length and width of the fish, respectively. The rest of the modes captured deformations on the fins, head, tail, gills, etc. In most of the cases, a single mode of variation would capture simultaneously several deformations.

The ICA model was estimated over a 33-dimensional PCA representation that preserves 99% of the shape variation. In this case, the deformations were much more located since a single independent mode of variation captures the displacement of the dorsal fin, another its size, or the shape of the tail, and so on. Figure 2.19 shows the effect of varying five representative independent modes along their feasibility intervals. Each mode corresponds to a single ray, numbered 1 through 5, while the center corresponds to the mean shape of our PDM. It can be observed, for instance, that ray 1 isolates tail asymmetries and ray 2 varies the shape of symmetric tails, that ray 3 remarks the presence of gills, ray 4 deforms exclusively the dorsal fin, or that ray 5 captures the position and existence of a secondary set of both, upper and lower fins.

These extracted features have interesting applications beyond those classical in PDMs, and we have experimented with database queries. Noticing that the feature that corresponds to the first ray in Figure (2.19) affects the asymmetry in the tail, we can search for those fish with strongly asymmetric tails by simply selecting the ones with a maximum value on this single independent component. Notice that this is equivalent of selecting the extremes of the first ray as prototypes and defining closeness to a prototype as the response in the corresponding component. Figure (2.20) shows the original boundaries of the five fish that have the highest response to this component.

More surprisingly, we have searched exhaustively in the database for all the fish with notably asymmetric tails, totalling 80. On the other side, we sorted the database by the absolute response to the corresponding component. Of the 80 fish with higher response, 84% belonged to the original group and the remaining 16% still presented some kind of asymmetry in the tail.

As an example of a combined search, fig. (2.21) shows the result of querying the database for fish with a protuberant dorsal fin (corresponding to large positive values

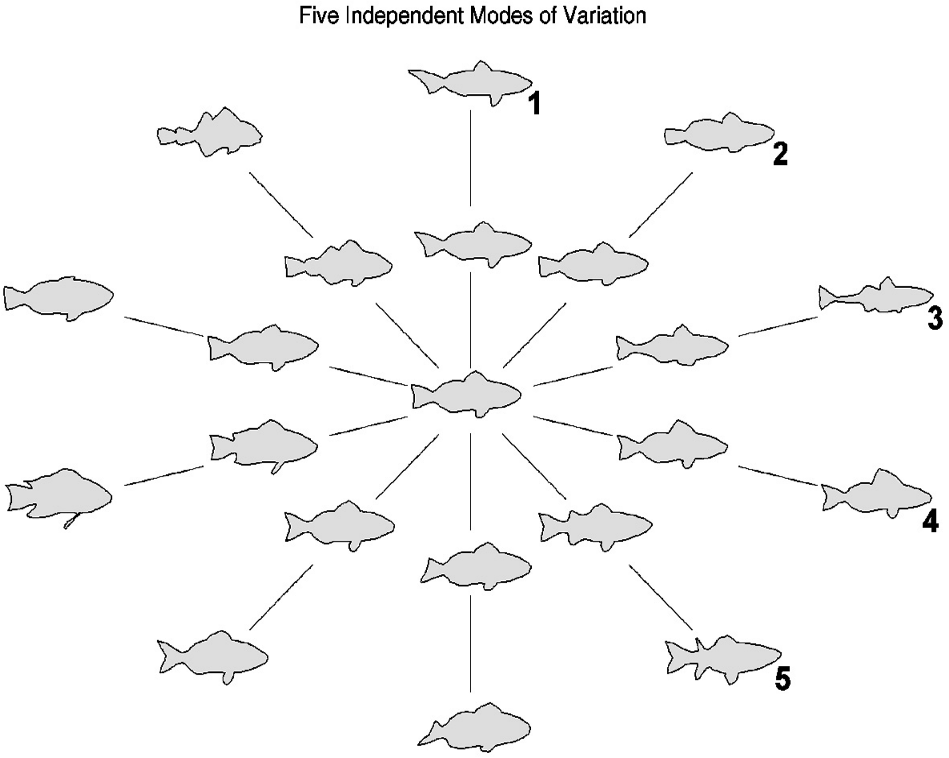


Figure 2.19: Five representative independent modes of variation for the fish database. Each ray contains the five shapes obtained through the variation of a single mode along its feasibility interval. The central shape is the mean shape of the working database.

on the independent component represented by ray 4 in fig. (2.20)) and with observable secondary fins (large negative values on ray 5). The results show that the ICA representation for shapes, besides the improved accuracy of the density estimation, can also provide in an unsupervised way, a set of prototypes useful for querying shape databases by similarity to conditional combinations of these prototypes.

From the results we conclude that the ICA representation can be successfully used when there are reasons to think that different shape deformations correspond to independent factors, and the shapes we observe are linear mixtures of these deformations. In this case, ICA can not only separate the deformations allowing control, classification and database queries, but can also provide a robust and simple density estimation framework which contributes in solving the problems of shape plausibility and, given a certain shape, finding the nearest feasible shape.

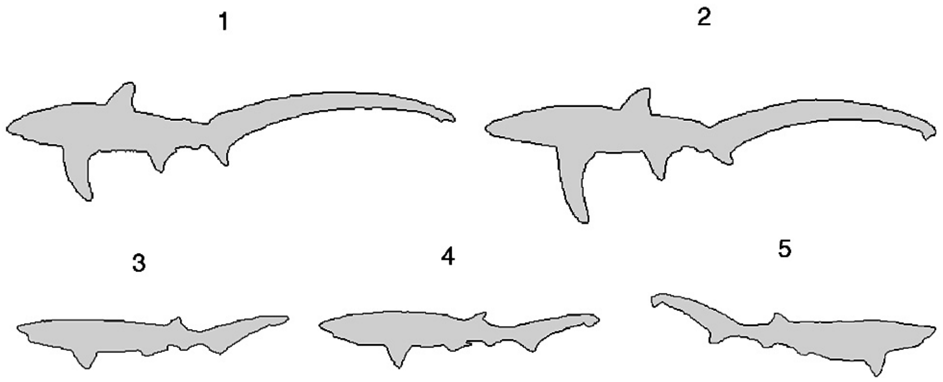


Figure 2.20: Result of querying the database for fish with asymmetric tails through the usage of a single unsupervisedly generated ICA prototype.

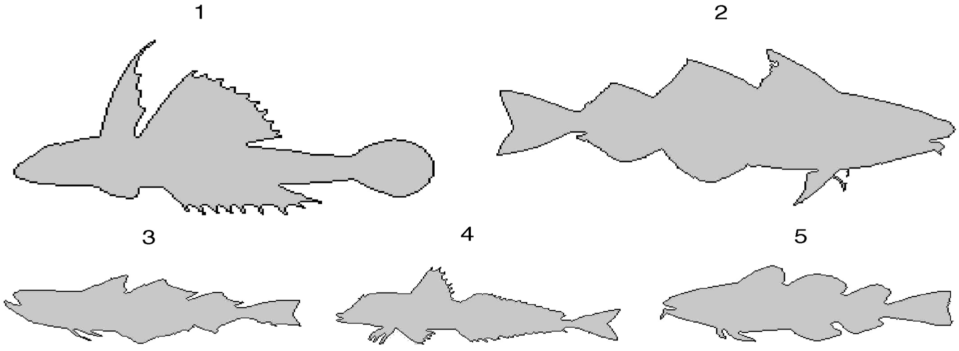


Figure 2.21: Result of querying the database for fish with a large dorsal fin and secondary fins through the usage of two unsupervisedly generated ICA prototypes.

2.5 Conclusions

This thesis is focused on the problem of linear feature extraction for the task of statistical classification of visual data, usually a particular case of classification of high-dimensional data. It is well known that the choice of features used to represent data affects classification performance, and not all classifiers are affected in the same way. For instance, an orthogonal linear transformation has no effect on Euclidean nearest neighbor classification since the Euclidean distance is preserved under this type of transforms, but instead can benefit other classifiers such as decision trees or statistical classifiers. Reversely, Fischer discriminant analysis is well justified as a feature extraction technique for quadratic classifiers and Gaussian classes, but we cannot anticipate its results if the assumptions are not met or other classifiers used. From a purely statistical perspective all linear feature extraction techniques affect negatively the Bayes error so, from this perspective, they are valuable only if they contribute to reduce noise or simplify density estimation.

In this chapter we have exposed three different nonsupervised linear transformations and examined their connections. At this point we should be able to state, at least theoretically, if classification can benefit from each of these methods. In the case of PCA, though no assumptions are made on the latent variables, we observe that this technique is "blind" beyond statistics of second order, restricting its capacity to learn complex data. The main advantage of PCA is its ability to reduce dimensionality, preserving the reconstruction error. We can conclude that statistical classification can benefit from PCA since density estimation can be more accurate on low dimensions and, if low variance directions correspond to noise, discarding them does not affect classification. In addition classes are frequently found distributed along the main directions of variance. For these cases PCA preserves the discriminative information of the dataset. The main problem with PCA appears when precisely the low variance directions contain useful discriminative information. In this case, information is lost and classification performance reduced.

The way NMF might benefit classification comes from a whole different standpoint. Conclusions are less general, since nothing can be said about reconstruction and noise reduction, and more related with the nature of the problem, since NMF will only benefit problems where advantage can be taken from its localized nature. For example, object detection robust to occlusion or illumination changes. Also, when negative data values do not have a straightforward interpretation, NMF results in a representation where visualization of the results is simplified. From a statistical perspective, there are no reasons to assume that densities are simplified, discriminability enhanced or noise reduced, so we can conclude that statistical classifiers have, a priori, no reasons for improving when NMF components are used as representation.

The relationship of ICA with density estimation is straightforward. When the ICA assumptions are met, density estimation of the M-dimensional extracted features is reduced to M unidimensional estimations. It is difficult to think of more simplification than this. A drawback of this technique is that not all problems satisfy the ICA assumptions, and the complexity of the statistical concept of independence makes it difficult to know beforehand if they do. For these cases, all we can hope for is that the approximation in terms of marginal densities of the extracted features is, if not exact, at least better than using the same approach on domain space. In addition, we have seen that the representation principle of sparsity is present in several real world problems. Statistical estimation benefits from sparsity not only from its close relationship to independence, but also from the fact that parametric density estimation techniques can be used. Another important disadvantage with ICA is that accurate model estimation, generally requires a large number of samples, in particular when dealing with high dimensional data. This is due to the fact that higher order relationships are only robustly observed when the size of the dataset sufficiently large.

Having posed the main characteristics of ICA and the advantages they can provide for classification, the next two chapters will focus on this approach detailing the two main contributions of our research. The way in which ICA can be used to obtain a class-conditional representation improving naive Bayes performance, and a theoretically sound technique for extracting discriminative features from this novel representation. Notice that the scope of possible classifiers is and will be restricted to those of statistical nature. Nevertheless, the underlying philosophy is applicable in different contexts: linking the representation to the classifier in the sense that once we have decided which classifier to use, extract those features that will enhance its performance. The fifth chapter shows how this reasoning can also be applied to the nonparametric nearest neighbors classifiers. As we have mentioned, this classifier has a number of advantages in addition to its well known simplicity. We show, that considering the metric nature of nearest neighbors, the nonparametric discriminant representation known as Nonparametric Discriminant Analysis (NDA), is able to provide features that enhance its performance.

Chapter 3

Class-Conditional Independent Component Analysis

3.1 Introduction

In the last chapter we have seen that when the conditions for performing Independent Component Analysis (ICA) on a certain dataset hold, density estimation of the projected data is strongly simplified. From a statistical perspective, the maximum likelihood approach states that the product of the marginal densities of the projected data (independent components) best fits the global distribution for the observations. From an information theoretic approach, assuming both algorithms converge to the same solution, we have a representation where the mutual information of the independent components is minimized, i.e. the Kullback-Leibler distance between the multivariate distribution and the product of the marginalized distributions is minimized. Considering the change of variables theorem (1.9), and assuming we are in the context of the basic ICA model (2.26) this means that, if the independent components satisfy that $p(\boldsymbol{s}) \approxeq \prod_m p(s_m)$, estimation of the D-dimensional density of our features ($\boldsymbol{x}$) in domain space can be approximated estimating D unidimensional densities since

$$p(\boldsymbol{x}) = |\det \boldsymbol{W}| p(\boldsymbol{s}) \approxeq |\det \boldsymbol{W}| \prod_{m=1}^{M} p(s_m) \tag{3.1}$$

with $\boldsymbol{W}$ the ICA filter matrix and assuming random vector $\boldsymbol{x}$ is zero-centered ($E\{\boldsymbol{x}\} = 0$). Though tempting, straightforward application of ICA for classification, i.e. unsupervisedly learning ICA from the dataset and working with the marginal densities of the independent components is incorrect. The Bayesian classification scheme makes use of the class-conditional densities and, as we will see, global independence does not imply class-conditional independence. This is why a representation that takes into account class belonging is required. We will call this representation class-conditional ICA (CC-ICA).

Before exposing our technique, we will shortly mention some previous approaches which use the ICA representation for classification. We then explore the concept of independence more in depth, directing our attention to the counterintuitive relationship between independence and conditional independence. We then expose what we understand as a class-conditional linear representation and fit it within a Bayesian classification scheme which we later adapt to the ICA representation. Results are summarized in two algorithms, one for learning (training) CC-ICA, and one for statistical classification using CC-ICA.

Experiments are performed on artificial, benchmark and real world data. The experiments with toy data sets attempt to illustrate the essence of our approach. The tests with benchmark data allow comparison of our technique with different approaches. Since our theory results in a modified naive Bayes classifier, comparing our approach with naive Bayes applied to domain space or other representations is particularly interesting. Experiments with real world data analyses the applicability of our criterion beyond theory. We here include the result of applying CC-ICA to the recognition of pharmaceutical products, a problem that arises from a real industrial vision situation. CC-ICA is also applied to the problem of land use classification from multispectral data, an application in the field of remote sensing.

3.2 ICA and classification

Independent Component Analysis, traditionally understood as a signal processing technique related with blind source separation and sparse coding has not received much attention from within the Pattern Recognition community. Therefore it is not until quite recently that we can find some results which apply ICA to the problem of classification. In this section we will mention two, which we consider specially interesting for their treatment and impact.

One of the pioneer works on applying ICA to classification was performed by Bartlett et al [11] on the problem of face recognition. In this work, ICA is used as a global unsupervised feature extraction technique: the representation is learnt from a set of face images without taking into account class membership. Classification is performed by first projecting the test samples in the learnt representation and then applying the chosen classifier. Two different independent component representations are proposed: a factorial code in which the components are statistically independent (this is the usual approach), or to make use of ICA to obtain statistically independent basis images. Both of these approaches are detailed in the previous chapter. While the first approach can be considered as high-order eigenfaces, the second approach is closely related to the method of Local Feature Analysis (LFA) [117] which is a topographic representation based on PCA that produces local filters resembling the sparse basis shown in (2.15). The face recognition performance of these two representations and the classical PCA is compared. Recognition is performed over faces obtained at different times or with changes in their expression. The original images are large for what is usual in face recognition experiments (50 × 60 pixels) and the classifier of choice is the 1-nearest neighbor rule with the angle distance,

$$d = \frac{\boldsymbol{s}_{test} \cdot \boldsymbol{s}_{train}}{||\boldsymbol{s}_{test}|| \cdot ||\boldsymbol{s}_{train}||} \tag{3.2}$$

where $\boldsymbol{s}$ corresponds to the independent or principal components of a projected sample. Results show that the three techniques perform similarly. In the article, discriminative features are also selected through a simple inter-intra class variance criterion, improving the performance. An interesting result, not pointed out in the original article is that, when changes in expression are considered, the independent bases achieve better results than all other techniques. This can be justified by the local nature of the filters, since the expressions are usually localized within the face so a change in the expression should only affect the value of few components.

In Bartlett's work, the potential of ICA for classification is limited to its properties as a representation capable of learning higher-order relationships in the data. Once this is stated, no further use is made of this information. For instance the same nearest neighbor classifier is used for all tested representations. There are no reasons to priorly assume that this metric classifier benefits from the discriminability provided by the ICA representation. Further studies on ICA for face recognition [92] show that ICA is highly sensible to dimensionality for a fixed number of samples so working in a compressed and whitened space is recommendable. Though this work also tests different distances for nearest neighbors, once again, the choice of representation has no relationship with the choice of classifier. The main difference between our work and these applications is that we do take into account the ICA objective function when designing our classifier. Reversely, in chapter 5 we will also show the way to provide the nearest neighbor classifier with a representation suited to its nature.

Another important work that addresses the problem of using ICA for classification is that by Lee [88]. From a simplified point of view, if the sources are considered as classes, ICA can be understood as an unsupervised classifier of unlabeled classes. The ICA model restricts these sources to be independent which seriously limits the applicability of ICA as an unsupervised classifier. In [88] this assumption is relaxed by using linear mixture models of independent component analysers following equation (1.19). In this case, the parameters to be estimated in the mixture model are the ICA parameters, and a maximum likelihood approach to estimation of these parameters is also presented in the paper. The resulting algorithm is applied to several problems where nongaussianity is present in the data. Though this approach is more related to clustering than to classification, it is closer to us in the sense it makes use of ICA for the obtainment of classes. Moreover, ICA mixture models could quite naturally be included within our framework. **Our main assumption is that the samples for each class are the result of linearly mixing independent sources.** ICA mixture models could be used to relax the independence assumptions providing more accurate class-conditional probabilities, in the same way a Gaussian mixture model can be used to improve the Gaussian (quadratic) classifier to a more general Bayesian classifier. This integration is beyond the scope of this thesis.

3.3 On Conditional Independence

Let x and y be random variables taking values in Ω. And let $p(x,y)$, $p(x)$, $p(y)$ and $p(x|y)$ be, respectively, the joint density of (x,y), the marginal densities of x and y, and the conditional density of x given the value of y. We say that x and y are independent if any of the following two equivalent definitions hold [31]:

$$p(x,y) = p(x)p(y) \tag{3.3}$$

$$p(x|y) = p(x) \tag{3.4}$$

It proves useful to understand independence from the following statement derived from (3.4): Two variables are independent when the value one variable takes gives us no knowledge on the value of the other variable. For the multivariate case $\boldsymbol{x} = (x_1, ..., x_N)$, independence can be defined by extending (3.3) as $p(\boldsymbol{x}) = p(x_1)...p(x_N)$. Conditional independence is defined as a natural extension of (3.3) and (3.4) through the incorporation of the conditional operator: $p(x,y|z) = p(x|z)p(y|z)$ and, equivalently, $p(x|y,z) = p(x|z)$. A frequent mistake is to think that global independence implies conditional independence, being Simpson's paradox [138] probably the most well known counterexample. The falseness of assuming that independence implies conditional independence can also be visualized considering random variables (x, y) with uniform distribution in the square $\Omega = [0,1] \times [0,1]$ and z, the random variable defined as 1 for the set $\{(x,y) \in \Omega, x > y\}$ and 0 otherwise. It is clear in this case that given z, knowledge on the value of, for instance, x provides information on y: it should be greater or less than x, depending on the value of z.

In the context of statistical classification, given K classes in $\Omega = \{C^1, ...C^K\}$ and a set of features represented by an N-dimensional random vector $\boldsymbol{x} = (x_1, ..., x_N)$, the Maximum A Posteriori (MAP) and the Maximum Likelihood (ML) solutions both make use of the class-conditional densities $p(\boldsymbol{x}|C^k)$. The case in which class-conditional independence is encountered has interesting consequences in the field of pattern classification. Particularly in Bayesian classification and feature subset selection.

The following example transmits the essence of Simpson's paradox through a toy pattern recognition example. We have a 2-dimensional binary classification problem with the space $\{0,1\} \times \{0,1\}$ divided in two classes C^1 and C^2. Both classes are equiprobable, meaning that $p(x,y) = 0.5p(x,y|C^1) + 0.5p(x,y|C^2)$. The global and class-conditional probabilities are given by the contingency tables shown in fig. (3.1). Notice that the unconditional probability is statistically independent but the class-conditional probabilities are not. For instance, we have that

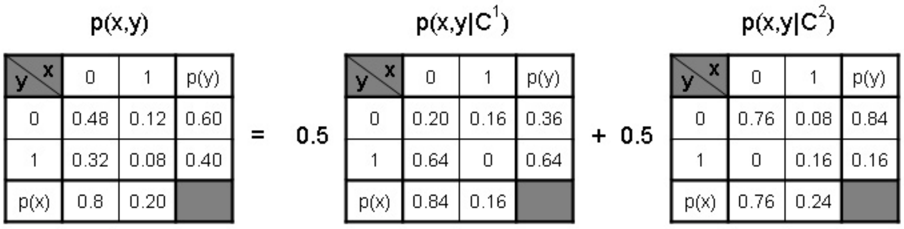
$$p(x=0, y=1) = 0.32 = 0.8 \times 0.4 = p(x=0)p(y=1)$$

but

$$p(x=0, y=1|C^2) = 0 \neq 0.76 \times 0.16 = p(x=0|C^2)p(y=1|C^2).$$

3.4 Class-Conditional ICA

We will refer to a learning technique as class-conditional when its parameters depend on any given class, as opposed to a *global* representation, usually estimated from all available samples regardless of their labels. In the case of a nonsingular linear representation, and for a certain class C^k, what we have is that filter and basis matrices are



p(x,y)

y \ x	0	1	p(y)
0	0.48	0.12	0.60
1	0.32	0.08	0.40
p(x)	0.8	0.20	▨

= 0.5

p(x,y|C^1)

y \ x	0	1	p(y)
0	0.20	0.16	0.36
1	0.64	0	0.64
p(x)	0.84	0.16	▨

+ 0.5

p(x,y|C^2)

y \ x	0	1	p(y)
0	0.76	0.08	0.84
1	0	0.16	0.16
p(x)	0.76	0.24	▨

Figure 3.1: Contingency tables for a toy binary classification problem in which independence of the unconditional density does not imply class-conditional independence.

class-dependent $\boldsymbol{W} = \boldsymbol{W}^k$ and $\boldsymbol{A} = \boldsymbol{A}^k$, with $\boldsymbol{A}^{k^{-1}} = \boldsymbol{W}^k$. Since class-conditional representations are adapted to the class they are able to learn patterns that otherwise would be lost. For instance, for a given reconstruction error class-conditional PCA used for dimensionality reduction would surely allow a more compact set of features for the description of a certain class than global PCA. This is because global PCA takes into account extra-class variances as well as intra-class. The counterpart of this choice is that, instead of a single representation, we have to learn as many representations as classes there are. More importantly, class-conditional representations fail to model the relationship among classes, for instance discriminability. If necessary, these relations have to be learnt using further techniques such as feature selection, which is not straightforward considering that different classes are represented with different features.

An important characteristic of class-conditional linear feature extractors is that they can be simply included within a Bayesian classification scheme. If we apply the change of variables theorem (1.9) to each class-conditional probability we have that,

$$p(\boldsymbol{x}|C^k) = |\det \boldsymbol{W}^k| p(\boldsymbol{s^k}|C^k) \overset{def}{=} |\det \boldsymbol{W}^k| p^k(\boldsymbol{s}) \tag{3.5}$$

if $\boldsymbol{s}^k = \boldsymbol{W}^k \boldsymbol{x}$. For this case, the MAP solution (1.3) takes the following form,

$$C_{MAP} = \arg \max_{k=1\dots K} |\det \boldsymbol{W}^k| p^k(\boldsymbol{s}) P(C^k). \tag{3.6}$$

This simplicity is not true with other classifiers such as the nearest neighbor classifier. The distance of a test sample to members of different classes is performed in different features spaces so it is very complex to compare them in order to choose the label that corresponds to the sample with the nearest distance. In this case, making the distance invariant to the particular representations makes us lose whatever we have gained through the choice of representation.

We now turn to the way ICA can be used within a class-conditional representation and call this approach CC-ICA. Remember that decision for using a linear transform as a feature extraction technique should always be supported by the belief that density estimation is simplified for the extracted features. And that what ICA can contribute in this sense is the possibility of assuming that the extracted features are statistically independent. From what we have seen in the last section, the interesting point with

ICA is that, if we wish to make use of the independence assumption for the (class-) conditional probabilities, we are then obliged to use class-conditional representations (CC-ICA). The basic CC-ICA model is estimated from the training set for each class. If $\boldsymbol{W}^k$ and $\boldsymbol{s}^k$ are the ICA filter matrix and the independent components for class C^k, then from (2.26)

$$\boldsymbol{s}^k = \boldsymbol{W}^k(\boldsymbol{x} - \overline{\boldsymbol{x}}^k) \tag{3.7}$$

where $\boldsymbol{x} \in C^k$ and $\overline{\boldsymbol{x}}^k$ is the class mean, estimated from the training set. Most ICA methods require, or at least advise, data whitening as preprocessing. Since some simple denoising is also recommended, dimensionality reduction and whitening through PCA is very common practice as a preprocessing stage for ICA. In this case, using (2.10), $\boldsymbol{W}^k$ can be decomposed as

$$\boldsymbol{W}^k = \boldsymbol{B}^k \boldsymbol{D}^{k^{-1/2}} \boldsymbol{V}^k,$$

where $\boldsymbol{V}^k$ and $\boldsymbol{D}^k$ are the matrices composed by the eigenvectors and eigenvalues of the class covariance matrix, and $\boldsymbol{B}^k$ the ICA unmixing matrix. We have also seen that, for this case, $\boldsymbol{B}$ results in an orthogonal matrix (2.53). Through CC-ICA, we have a space where all class-conditional probabilities can be assumed independent or at least, where the error involved with working with the marginal probabilities instead of the whole distribution is minimized. We call this kind of space a CC-ICA space.

Most learning techniques make very light or no assumptions on the classifier to be used with the extracted features. We could also use any Bayesian classifier on the extracted features through equation (3.6). But this is pointless: taking into account the independence assumption we observe that a modified naive Bayes is the natural choice when working in a CC-ICA space: CC-ICA has in naive Bayes a naturally associated classifier. We can see this by replacing the change of variables theorem for an ICA representation (3.1) into the MAP solution(3.6),

$$C_{MAP} = \arg \max_{k=1 \ldots K} |\det \boldsymbol{W}^k| \prod_{m=1}^{M^k} p^k(s_m) P(C^k). \tag{3.8}$$

If dimensionality is sufficiently high the product on the right-hand side of this equation, generally made up of values lower than 1, will be very close to zero, so the logarithm of the likelihoods will be used whenever possible. Also, unless stated otherwise, classes will be considered equiprobable so the MAP solution becomes the maximum likelihood (ML) solution,

$$C_{ML} = \arg \max_{k=1 \ldots K} \sum_{m=1}^{M^k} \log p^k(s_m) + \log |\det \boldsymbol{W}^k| \tag{3.9}$$

The constant in the right-hand side of the equation can be regarded as a normalizing constant, necessary to compare conditional probabilities calculated in different representations. We can further estimate this constant considering that

$$|\det \boldsymbol{W}^k| = |\det \boldsymbol{B}^k| |\det \boldsymbol{D}^{k^{-1/2}}| |\det \boldsymbol{V}^k| = \prod_{m=1}^{M^k} \frac{1}{\sqrt{\lambda_m^k}} \tag{3.10}$$

where λ_m^k are the eigenvalues of the covariance matrix of class C^k. This equality can be proved using the fact that the absolute value of the determinant of an orthogonal matrix is 1 and that the determinant of a diagonal matrix is the product of the terms in the diagonal. To this point, we have assumed that the dimensionality of the original data (D) which will usually be a large value, is equal to the dimensionality in the CC-ICA representations (M^k). This is not always the case since PCA whitening generally conveys some kind of dimensionality reduction making it impossible to directly calculate the determinant of a no longer square matrix. In this case, we can assume that the conditional distribution of the measurements is approximated by the distribution of the principal components. This approximation can be arbitrarily good if a sufficient number of components are considered since the reconstruction error is bounded by the sum of the eigenvalues and tends to zero. Notice that for this case in (3.10) the third term in the second equality no longer is present and the estimation of the constant remains the same. By replacing (3.10) into (3.9) we have that samples will be classified using the version of naive Bayes,

$$C_{ML} = \arg \max_{k=1\dots K} \sum_{m=1}^{M^k} \log p^k(s_m) - \frac{1}{2} \sum_{m=1}^{M^k} \log \lambda_m^k \tag{3.11}$$

which can be written as

$$C_{ML} = \arg \max_{k=1\dots K} \sum_{m=1}^{M^k} \log p^k(s_m) + \nu^k \tag{3.12}$$

with $\nu^k = -0.5 \sum_m^{M^k} \log \lambda_m^k$.

We arrived to this modified naive Bayes classifier first by stating the advantages ICA poses for density estimation, then motivating the need for a class-conditional approach and finally by replacing the conditional independence assumptions on the conditional densities within the Bayesian classification scheme. If we reverse the reasoning, results are identical but maybe, easier to understand. Let us suppose we want to improve naive Bayes. One of the possible ways for doing this might be reinforcing the class-conditional independence assumption made by this classifier. This can be done using ICA on each of the classes, and we are back to where we started. This reversed reasoning is useful for understanding some of the experiments and clearly observed in the experiment with artificial data. The goodness of our method should be measured, besides in terms of absolute performance when compared with other classifiers, in terms of the improvement it represents for naive Bayes. Even though naive Bayes can perform fairly good with an unmet independence assumption as we said in chapter 1, we will see that trying to meet this assumption using CC-ICA is of great benefit for the classifier.

3.4.1 Estimation of the Marginal Densities

Using our classifier (3.11) still requires the estimation of the densities $p^k(s_m)$, where in this case $s_m = \boldsymbol{w}_m^{k\,T}(\boldsymbol{x}-\overline{\boldsymbol{x}}^k)$ with $\boldsymbol{w}_m^k$ the m-th row of the filter matrix $\boldsymbol{W}^k$, $\boldsymbol{x} \in C^k$ and $\overline{\boldsymbol{x}}^k$ the class mean. This estimation is simplified not only by being unidimensional but also by prior information we have on the independent component. We know that the independent components have zero mean and unit variance. We also know they are highly nongaussian, and we can easily find out if they are sub or super Gaussian. All this knowledge can restrict density estimation to particular families. In addition, some ICA algorithms such as maximum likelihood with the generalized Gaussian estimate these density simultaneously with the ICA parameters. Since in most cases we will use the maximum negentropy approach for ICA estimation which restricts the densities to very general families, we will now consider the problem of estimating $p(s)$ where s is a zero mean, unit variance nongaussian random variable.

As mentioned in section (1.2.4) density estimation techniques can be categorized depending on their nonparametric, parametric and semiparametric nature. Of the first group, we should discard the histogram approach for its limitations. The Gaussian kernel approach (1.13), adapted to our situation results in

$$p(s) = \frac{1}{N}\sum_{n=1}^{N} \frac{1}{\sqrt{(2\pi)}\sigma} \exp\left(-\frac{1}{2}\frac{(s-s_n)^2}{\sigma}\right), \tag{3.13}$$

where s_n are each of the N samples in the training set (in CC-ICA, the components of the training set with a certain class label). The kernel width can be selected as $\sigma = [\frac{12}{N}]^{\frac{1}{5}}$ as suggested in [136]. In the case of sparse data, this kernel method can cause the probability to drop to zero. This can be solved by increasing the value of σ at the cost of eventually over-smoothing the estimate.

Parametric models can be also used. We have mentioned a few of these models in section (2.4.1). The Laplace or double-exponential distribution (2.36) works fine with moderately sparse independent components. The fact it has a single parameter makes it a simple choice but highly unflexible as well. When the nonderivability of the Laplace distribution results inconvenient, its smoothened version (2.37) can also be used. If the independent components are very sparse, a quite robust parametrization which provides an accurate approximation of very sparse data was introduced by Hyvärinen in [65],

$$p(s) = \frac{1}{2}\frac{(\alpha+2)[\alpha(\alpha+1)/2]^{(\alpha/2+1)}}{\sqrt{\alpha(\alpha+1)/2} + |s|^{(\alpha+3)}} \tag{3.14}$$

As $\alpha \to \infty$ this approaches the Laplace density. The parameters are estimated as follows:

$$\alpha = \frac{2-k+\sqrt{k(k+4)}}{2k-1}$$

where $k = p(0)^2$ and $p(0)$ can be estimated using a suitable kernel. These two parametric approaches, the Laplace and (3.14), both assume supergaussianity of the independent component. A more generic approach, flexible enough to account for any kind of unimodal nongaussian data is the generalized Gaussian (2.41). The drawback of this approach is that no method which is both robust and simple is available for the estimation of the Gaussian exponent.

Semiparametric models, as the generalized Gaussian and kernel approaches, have the advantage of being applicable to any kind of nongaussianity, even highly sparse data if the number of mixture components is properly chosen. In addition, the high cost of training semiparametric models is practically inexistent in our case due to the unidimensionality of the data. Gaussian mixture models can be used, the model estimated through the expectation-maximization (EM) algorithm ([33]). In our case,

$$p(s) = \sum_{j=1}^{J} p(j) \frac{1}{\sqrt{(2\pi)}\sigma_j} \exp\left(-\frac{1}{2}\frac{(s-\mu_j)^2}{\sigma_j}\right), \tag{3.15}$$

Gaussian mixture models might require far too many mixture components in order to accurately estimate a highly supergaussian component. So other, more appropriate distributions can be used within the mixture. The EM algorithm can also be used to estimate a *mixture of Laplacians.* A mixture of two zero-mean Laplace distributions proves easy to estimate and highly adaptive to strong variations in the level of sparsity. These would be modelled as,

$$p(s) = \sum_{j=1}^{2} p(j) \frac{1}{\sqrt{2}\alpha_j} e^{-\frac{\sqrt{2}|s|}{\alpha_j}} \tag{3.16}$$

Of course, these approaches can be combined. For instance, by sorting the independent components by their kurtosis value a different approach can be used for different levels of kurtosis. All these approaches applied to a moderately sparse independent component obtained from one of the experiments are illustrated in fig. (3.2). In all cases the histogram is plotted in the background with a dotted line for comparison. The nonparametric Gaussian kernel approach in (a), with $\sigma = 0.1$ is good at approximating the histogram but has zero valued densities where no samples were present. The parametric approaches ((b) (c) and (d)) are quite inaccurate. The Laplace (a) fails because it is visibly less sparse than the component. Two reasons can account for the inaccuracy of the density proposed by Hyvärinen (d). In the first place, an incorrect estimation of $p(0)$ can propagate to all parameters. In second place, this density is adequate for highly supergaussian variables, and this is not the case of the distribution of the sample component. The generalized Gaussian (c) shows the problem of correctly estimating the Gaussian exponent. Both mixture models are quite accurate in their estimation.

In practice, all these approaches, if adequately estimated, yield approximate results. In cases where strong variations of nongaussianity are present, the parametric approaches were greatly affected. Also, performance of the kernel approach was very sensible to the choice of an adequate kernel width. So generally the semiparametric approach was taken. Most of the time a mixture of 2 or 3 Gaussians was used, and when kurtosis was too high, the mixture of 2 zero mean Laplacians proved accurate. Since sparsity is found in most of the cases, it is interesting to understand the meaning of classifying data under a class-conditional sparse representation. For this situation, the sparse coding for a given class will efficiently code any sample belonging to the class and the projected vector will be mostly zero, except for a few strongly activated components. The characteristic strong peak in zero of sparse distributions will give this sample a high probability. The heavy tails will avoid the activated components from causing the probability to drop to zero. On the other hand, when a sample not belonging to the class is projected, the sparse coding will fail to efficiently represent

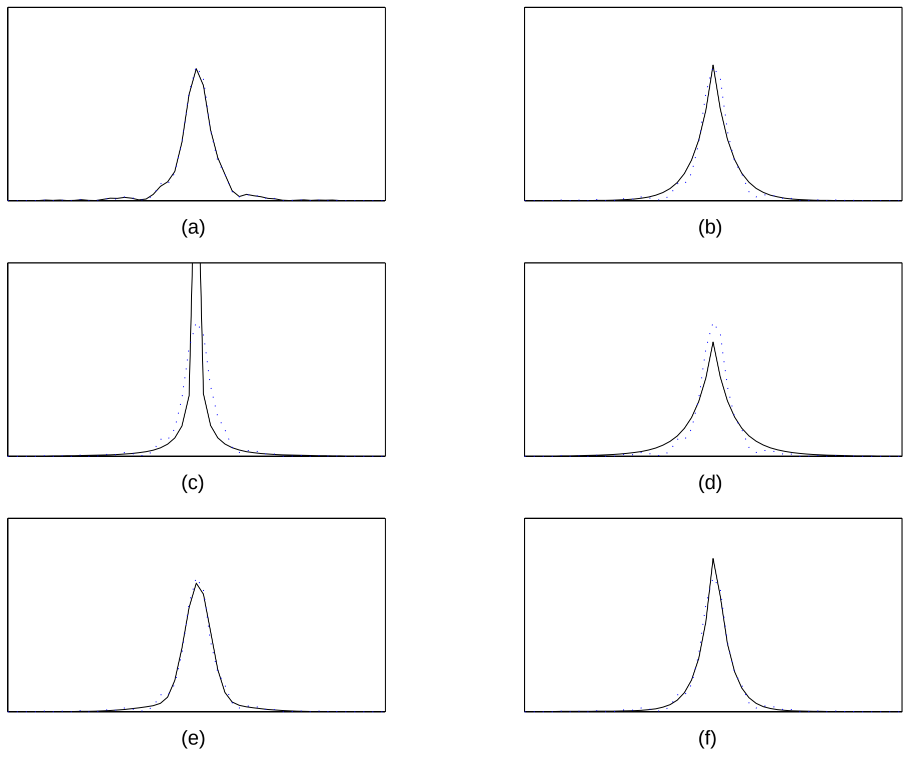


Figure 3.2: Estimating the distribution of the independent components: (a) kernel estimate; (b) Laplace estimate; (c) generalized Gaussian estimate, (d) estimate as proposed by Hyvärinen (e) mixture of 3 Gaussians (f) mixture of 2 Laplacians. The histogram of the component (dotted line) can be used as a reference.

this unlearnt sample. This is usually reflected in random activations for all the components, resulting in a low probability of class belonging. More on this will be said on the experiments.

3.4.2 CC-ICA Algorithms

All these results can be summarized in two algorithms which we call CC-ICA-Train and CC-ICA-Test. The first algorithm, exposed in table (3.1), illustrates the learning procedure to be followed when working with CC-ICA. In few words, for each class, estimate an ICA model and the unidimensional densities of each independent component. We also outline what information should be preserved for the evaluation instance, from this training stage. The test algorithm is detailed in table (3.2). Given a test sample, we project it on each of the learnt representations and calculate the probability under that representation. We then use (3.11) to determine the most probable class. Notice that, except for the dimensionality of the ICA projections, none of the algorithms require any artificial parameters or thresholds.

```
1. for each class $k = 1 \dots K$
    (a) $\boldsymbol{x}^k$ represents the $D$-dimensional samples in class $C^k$. Whiten and possibly reduce dimensionality of $\boldsymbol{x}^k$ with PCA, obtaining $M^k$-dimensional whitened data $\boldsymbol{z}^k$. Preserve the normalizing constant,
            $\nu^k = -\frac{1}{2}\sum_m \log \lambda_m^k.$
        and the class mean $\overline{\boldsymbol{x}}^k$ for evaluation.
    (b) Using any basic ICA algorithm, for instance (2.2), learn the ICA filter matrix $\boldsymbol{B^k}$. Right-hand multiply by the whitening matrix to obtain the ICA model for the class: $\boldsymbol{W}^k(\boldsymbol{x}^k - \overline{\boldsymbol{x}}^k) = \boldsymbol{s}^k$, where $\boldsymbol{s}^k$ are the independent components for class $C^k$. Preserve filter matrix $\boldsymbol{W}^k$ for evaluation
    (c) for each $m = 1, \dots, M$
        i. Estimate unidimensional zero-mean, unit-variance density $p^k(s_m)$ using one of the proposed techniques, or directly using the output of the ICA algorithm, if the estimate is available. Preserve density parameters for evaluation.
    (d) end loop
2. end loop
```

Table 3.1: Class-conditional ICA training algorithm (CC-ICA-Train).

```
1. $k_{Test} = -1$, $\mathcal{L}_{Test} = -\infty$
2. for each class $k = 1 \dots K$
    (a) Project test sample in CC-ICA space
            $\boldsymbol{W}^k(\boldsymbol{x}_{Test} - \overline{\boldsymbol{x}}^k) = \boldsymbol{s}$
    (b) Calculate class conditional log-likelihood using the estimated densities in (3.11),
            $\mathcal{L}^k = \sum_{m=1}^{M^k} \log p^k(s_m) + \nu^k$
    (c) if ($\mathcal{L}^k > L_{Test}$) then $k_{Test} = k$, $\mathcal{L}_{Test} = \mathcal{L}^k$
3. end loop
4. Assign $\boldsymbol{x}_{Test}$ to class $k_{Test}$ with log-likelihood $\mathcal{L}_{Test}$.
```

Table 3.2: Class-conditional ICA classification algorithm (CC-ICA-Test).

These algorithms also provide an idea of the computational load involved in the technique. CC-ICA-Train is difficult to evaluate computationally since it depends exclusively on the chosen ICA and density estimation methods. ICA on high-dimensional data is by itself computationally expensive, more considering it has to be calculated for each class. The density estimation, even if semiparametric methods are used, is quite fast thanks to the unidimensional nature of the samples. Still we can calculate the total number of parameters we should preserve from the training stage: K $M \times D$ filter matrices; K D-dimensional class means; K normalization constants; and the density parameters. If, for instance, the mixture of three Gaussians is used per variable, then each component density requires 9 parameters corresponding to the means, standard deviation and weights of the mixture components. For this typical case and assuming all ICA spaces have a dimensionality of M, the total number of parameters that CC-ICA-Train outputs is $K(M(D+9)+D+1)$. For the CC-ICA-Test algorithm we can directly measure the number of arithmetic operations involved for a single test sample. These operations are K D-dimensional vector subtractions (KD operations) and K matrix by vector multiplications (KMD operations). The evaluation of the likelihood depends on the chosen parametrization and if we once again assume a mixture of 3 Gaussians is used, we have approximately $6KM$ operations, totalling $K(M(D+6)+D)$ operations. Since M depends on the original data dimensionality, the number of operations is approximately quadratic on the dimensionality.

3.5 Experiments

Though more than one ICA estimation method was used in some of the experiments, all the results presented in this section were obtained using the symmetric FastICA [63] algorithm. For those experiments in which the other algorithms converged, classification performance was not significatively different. We should note that several of the algorithms mentioned in chapter 2 have serious problems when dealing with high dimensional data as is the case with most of the real world data we deal with. For reducing noise in the whitening stage, different approaches were taken, but the most frequent approach was preserving approximately 97.5% of the variance of the data. When this variance is preserved with approximately the same dimensionality in every class we preferred to choose a fixed M: though the normalization constant takes care of this, differences in dimensionality might negatively affect the magnitude of the class-conditional probabilities. Unless stated otherwise, a mixture of 3 Gaussians was chosen to estimate the distribution of the independent components. Though in some cases other choices such as nonparametric Gaussian kernels might improve the performance, the Gaussian mixtures had very good overall performance and were quite robust throughout the different experiments.

3.5.1 Artificial Data

We generated a toy dataset in order to illustrate the exact nature of our approach. We took care that the generated data fulfilled all the CC-ICA assumptions: for each class, the observed data is a linear mixture of statistically independent sources and these sources are all nongaussian except, possibly, for a single source. We generated two classes, C^1 and C^2, the samples on each class were a linear mixture of two zero-mean, unit-variance Laplacian distributions (supergaussian components), so the dimensionality of our dataset is 2. The mixture matrices for each class were randomly chosen. In the exposed results $A^1 = [[-0.47, -0.37]; [-0.05, -0.22]]$ and $A^2 = [[0.49, 0.10]; [-0.24, 0.38]]$ are the mixture matrices for classes C^1 and C^2, respectively. The class means were chosen as $\boldsymbol{\mu}^1 = [0.25, 0.25]$ and $\boldsymbol{\mu}^2 = -\boldsymbol{\mu}^1$. In fig. (3.3) we observe 200 randomly generated samples for each of the classes. The overlap between the classes can also be observed on this figure.

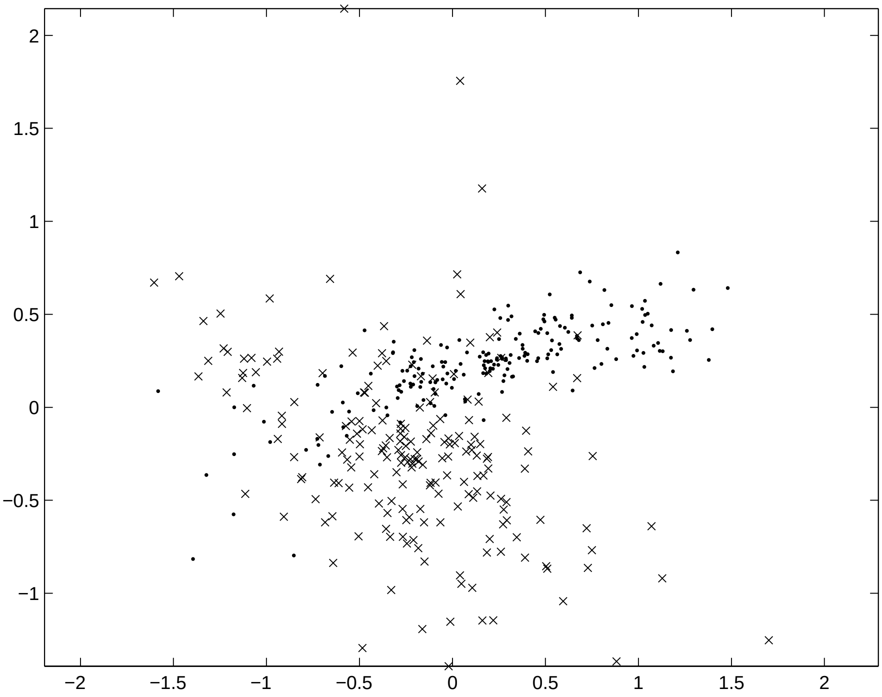


Figure 3.3: Artificial two-class problem (dimensionality=2).

Since all the assumptions are met, if the mixtures are perfectly separated and the densities correctly estimated, the CC-ICA error in this problem should be exactly equal to the Bayes Error. So we first heuristically estimated the Bayes Error of our problem using the exact filter matrices and class means. For the error estimation, we generated 100000 samples for each class and obtained the empirical Bayes error. We repeated this experiment 200 times in order to approximate the true Bayes error with the mean empirical Bayes error. Almost no important changes were noticed after the second iteration so convergence was ensured. The obtained estimate for the Bayes error was 0.1257. We then proceeded to the evaluation and comparison of our algorithm. This was done randomly generating 400 training samples and 200 evaluation samples. The CC-ICA algorithm was applied to this dataset and the error calculated. This process was repeated 2000 times in order to obtain a robust measure for the error. Other classifiers were applied to the same framework for comparison. A numbering and coding for the different classifiers we tested, along with their main characteristics are summarized in table (3.3) [1].

Number	Abbreviation	Description
1	MLGAU	Maximum likelihood Gaussian classifier (quadratic classifier).
2	1-NN	(One) Nearest Neighbor classifier ($L2$ distance).
3	NBGAU	Naive Bayes with a Gaussian on each variable
4	NBKER	Naive Bayes with a Gaussian kernel estimator on each variable
5	NBPCAGAU	Naive Bayes on PCA representation with a Gaussian on each variable
6	NBICALAP	Naive Bayes on ICA representation with a Laplacian on each variable
7	NBCCPCAGAU	Naive Bayes on class-conditional PCA with a Gaussian on each variable
8	NBCCPCALAP	Naive Bayes on class-conditional PCA with a Laplacian on each variable
9	NBCCICALAP	Naive Bayes on class-conditional ICA with a Laplacian on each variable

Table 3.3: Classifiers and codes used in the artificial CC-ICA experiment.

The results of applying these classifiers to the toy dataset are exposed through boxplots in (3.4), where the classifier numbering corresponds to the numbering in the table. The box plots have lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers, illustrated with a + sign, are data with values beyond the ends of the whiskers. Some interesting conclusions can be extracted from this figure. In the first place, class-conditional representations (7, 8 and 9) yield much better results than global representations. As expected CC-ICA (9) achieves the Bayes Error for this problem: ICA succeeded in recovering the sources. The choice of marginal densities for CC-PCA (7 and 8), Gaussian or Laplacian, did not greatly affect the performance. The quadratic classifier (1) and the nearest neighbor classifier (2), performed similarly, and better than any global naive Bayes classifier except for the one in which marginal probabilities were estimated using a kernel approach (4). Notice that applying ICA to a global representation that probably does not meet the ICA assumptions and then assuming class-conditional independence with a naive Bayes classifier can be disastrous (6). In the first place, we can conclude that CC-ICA does what it should when the appropriate conditions are given. In second place we observe the poor performance of naive Bayes when class-conditional independence is not met. This is regardless of the density estimation since in (4) a nonparametric density estimation method was used on domain space, and results were very similar to that of assuming a Gaussian on each variable. The experiment confirms that class-conditional representations are more appropriate then global representations for achieving class-conditional independence.

[1] The only unexplained classification method present in this table is naive Bayes on class-conditional PCA (NBCCPCA). This method is exactly the same as CC-ICA except that the representations used are those given by PCA. Any difference in performance between NBCCPCA and NBCCICA can be exclusively blamed on the accuracy of the density estimate and in consequence on the importance naive Bayes gives to the independence assumption.

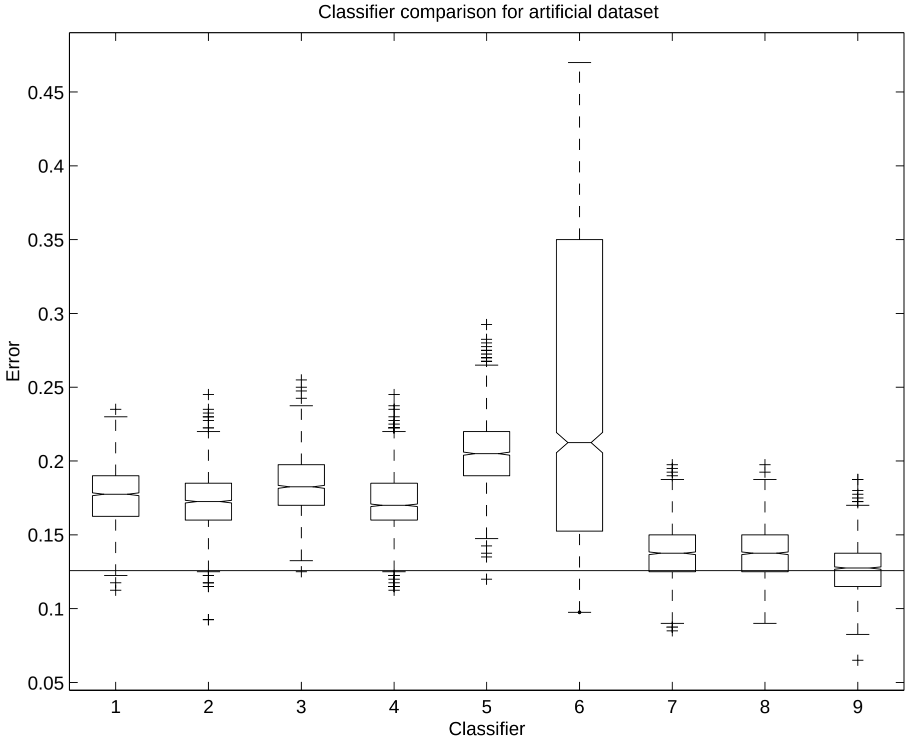


Figure 3.4: Comparison of classification error for different classifiers applied on the toy dataset. CC-ICA is number 9. The horizontal line on error 0.1257 represents the Bayes error of the dataset.

3.5.2 Benchmark Data

As we will see in further experiments, the main advantage of our technique is its applicability to high-dimensional data by reducing any problem to unidimensional density estimations. This section illustrates that, even in the low-dimensional case, CC-ICA performs comparably to standard classification techniques and considerably improves naive Bayes when this classifier is applied to other representations. From the artificial experiment we can already anticipate that the main drawback of our technique is that, when the assumptions are not met, ICA does not necessarily always converge to the correct solution and this seriously affects classifier performance. One sure cause of incorrect ICA estimation is a low sample number. The benchmark datasets on which experiments are performed in this section were chosen such that a large number of samples was available. This limitation on the sample number is restrictive but, as we will see in the next section, there are several real world experiments in which a sufficiently large number of samples is available.

The benchmark databases analysed in this section all belong to the UCI Machine Learning Repository [18]. From all the databases present in this repository we selected only those that fulfill the following conditions: numeric attributes are present, there exists a large number of available samples per class (typically above 10 times the dimensionality), and there are no missing feature values. For the selected databases we test the following classification techniques: Maximum Likelihood under the assumption of Gaussian classes for domain space and a PCA representation (MLGAU, MLGAUPCA); naive Bayes also assuming Gaussian variables on each class for domain space, PCA and an ICA representation (NBGAU, NBGAUPCA and NBGAUICA); naive Bayes assuming a mixture of 3 Gaussians per variable and class for the same three representations (NBGAUMIX, NBGAUMIXPCA and NBGAUMIXICA); and finally naive Bayes under class-conditional PCA and ICA representations using a mixture of 3 Gaussians on each variable and representation (CCPCAGAUMIX and CCICAGAUMIX). Results of classifying with the nearest neighbor technique using the Euclidean distance are also included as a referential nonparametric classifier. Notice that we do not test ICA for the maximum likelihood quadratic classifier (MLGAUICA). This is because this classifier is invariant to linear transformations, so the results given by MLGAUICA should be equal to the results obtained by MLGAUPCA. Actually, only if some kind of dimensionality reduction is present, MLGAUPCA should differ from MLGAU. The maximum published accuracy for these databases is also mentioned on each case.

For proper comparison PCA dimensionality reduction always followed the same scheme. For global PCA and ICA, no noise reduction was introduced: dimensionality was chosen such that PCA preserves 100% of the data variation. For the class-conditional representations, we chose for all classes the same dimensionality: the one corresponding to the class that required less features in order to preserve 100% of the data variation. Since databases were chosen such that a large number of samples were available per class, this number usually agreed with the dimensionality obtained for the global case. A precision of 0 or very close to 0 usually means that the classifier could not be applied under that choice of dimensionality. For instance, the maximum likelihood Gaussian classifier in domain space is not always applicable since the covariance matrix might result singular for a given class.

The following databases and their main characteristics are considered.

- LETTER: The Letter Image Recognition Data consists in 20000 instances, each representing a capital typewritten letter in one of twenty fonts. Each letter is represented using 16 integer valued features corresponding to statistical moments and edge counts. Training is usually done on the first 16000 instances, and test on the final 4000. There are approximately 615 samples per class in the training set. The first results with this database, using a Holland-style genetic classifier [45], gives at its best an average predictive accuracy of 82.7%. Later, a minor variation of the nearest neighbors introduced in [43] yielded a predictive accuracy of 95.7%. Actually, 1-NN with the Euclidean distance also yields a very similar accuracy.

- IMAGE: The image segmentation data consists in 19 features of an image processing database used to predict whether the image is brick face, sky, foliage, grass, cement, window or path. So there are 7 classes in this database, formed by a total of 2310 samples, 330 samples per class. In this case, we used cross-validation considering 20% randomly chosen samples for evaluation and the remaining for training. The experiment was repeated 10 times and the results are the average of the obtained accuracies. There is no maximum reported accuracy for this database.

- PENDIGITS: The pen-based recognition of hand-written digits consists in 7494 training cases and 3498 evaluation cases, each consisting in 16 attributes which are integers from 0 to 100. The number of classes is 10, one per digit. The best reported accuracy for this database is 98.1% using error correcting output codes *ECOC* for deciding the optimal variance of Gaussian kernels applied in a maximum likelihood scheme [113].

- PIMA: The Pima Indians diabetes database from National Institute of Diabetes and Digestive and Kidney Diseases has 768 instances corresponding to 2 classes and is represented by 8 numeric attributes. Each instance represents the record of a female patient tested for diabetes. The best reported accuracy for this database is 76% [139] using their ADAP algorithm, an adaptive learning routine that generates and executes digital analogs of perceptron-like devices. Usual accuracy results on this database for other tested algorithms range from 67% to 76%.

Results of applying the mentioned approaches to these four databases are summarized in table (3.5.2). The main objective of this experiment is to illustrate the differences among the statistical classifiers, and show that the attempt to fulfill the independence assumption through a class-conditional representation has an important effect on the naive Bayes classifier. We also intend to show that CC-ICA, even when applied to low-dimensional data performs similarly to state of the art classifiers. Probably the first observation that can be made from this table is the superiority of the NN classifier on all datasets except for PIMA. Actually, in the LETTER database, NN achieves the highest reported accuracy while in PENDIGITS almost. Maximum likelihood using a Gaussian or a mixture of Gaussians performs poorly when any given covariance matrix is close to singular. In these cases (PENDIGITS), the unidimensional estimation given by naive Bayes outperforms the multidimensional approach. The limitations of using a fixed number of mixtures for the Gaussian Mixture also become evident when we observe that naive Bayes performance using mixture models and a global representation is usually outperformed using a single Gaussian per variable. As with the artificial dataset, in general class-conditional representations greatly enhance naive Bayes performance. In this case, the decorrelation introduced by CC-PCA can be seen as one step towards independence. Nevertheless, the unmixing of higher order relationships achieved by CC-ICA has a stronger effect: In all cases the best results for naive Bayes were obtained using this representation. In PENDIGITS and PIMA the results were close or even better to the best reported accuracies for these database. Naive Bayes, as its name indicates is assumed limited by its simplicity and most of the publications show it outperformed by other, more elaborate, classifiers. Our experiment shows that, though there other ways of enhancing naive Bayes, the logical approach of trying to hold its assumptions with more strength is effective and turns this classifier into a powerful tool. This capability will be particularly evident in the next chapter, where high dimensional data does not allow straightforward multidimensional estimation, and simplification to the unidimensional case becomes necessary.

	LETTER	IMAGE	PENDIGITS	PIMA
NN	95.7	95.8	97.7	66.4
MLGAU	87.5	0.0	10.4	72.7
MLGAUPCA	87.5	73.7	10.4	72.7
NBGAU	62.4	14.0	72.2	73.1
NBGAUPCA	67.4	73.7	85.3	73.5
NBGAUICA	66.2	22.6	87.2	72.7
NBGAUMIX	14.6	14.3	17.3	58.9
NBGAUMIXPCA	47.3	41.4	54.9	64.9
NBGAUMIXICA	44.4	61.5	62.4	65.7
CCPCAGAUMIX	87.2	89.6	96.3	71.6
CCICAGAUMIX	91.1	95.1	97.1	76.2

Table 3.4: Comparison of predictive average accuracies on benchmark databases.

3.5.3 Real-world Data

Local Color Histograms for Object Recognition

As mentioned in section (2.3.1) several approaches have recently proposed the use of local window representations as a reliable solution to occlusions, complex background, scale changes, illumination changes, and different viewpoints or orientations [124, 131, 130, 32] within the problem of object recognition. When lighting conditions do not change severely, or color normalization methods can be used, the color distribution of an object is a simple kind of sensor data that have demonstrated to be an efficient signature for object-recognition in the appearance-based framework. Swain and Ballard [145] were the first to show the usefulness of color histograms for indexing large object databases independently of the object's pose. Other approaches that use color as a relevant signature are exposed in [98, 50, 40, 123, 22].

One of the main drawbacks of using color distributions as salient features is the difficulty of constructing a good model, due to their high dimensionality. If we consider a single 8-bin histogram per color spectrum resulting feature dimensionality is $8^3 = 512$. A first approach is to classify data using metric techniques such as nearest neighbor techniques. If we consider probabilistic classifiers, high dimensionality forces nonparametric density estimators such as Gaussian kernels or naive Bayes. For the latter, CC-ICA has demonstrated, through the benchmark experiments, to provide a clear improvement. But first we should choose the way for extracting the representative local color distributions. In our case color distributions are obtained from circular masks in neighborhoods of perceptually salient keypoints, allowing invariance to rotation in the objects [54]. We now briefly outline this approach and how it can be used for the general task of object recognition.

A complete reference work for keypoint detectors has been developed by Schmid, Mohr and Bauckhage [132] where they evaluate the most important interest point detectors by considering two measures: repeatability rate and information content. One of the algorithms mentioned in this work that correctly handles the problem of finding a set of representative keypoints is Tomasi and Kanade‘s algorithm [150]. This feature detector is based on defining an initial matrix $\boldsymbol{A}$ that averages the Gaussian derivatives in a window W around a point (x, y) as

$$\boldsymbol{A}(x,y) = \begin{bmatrix} \sum\limits_W (I_x(x_j,y_j))^2 & \sum\limits_W I_x(x_j,y_j) I_y(x_j,y_j) \\ \sum\limits_W I_x(x_j,y_j) I_y(x_j,y_j) & \sum\limits_W (I_y(x_j,y_j))^2 \end{bmatrix} \tag{3.17}$$

where $I(x, y)$ is the image, (x_j, y_j) are the points in the window W surrounding (x, y), and I_x, I_y are the first Gaussian derivatives of the image $I(x, y)$. As noted in [150], (x, y) is a good point (it contains salient features) if the eigenvalues of matrix $\boldsymbol{A}$ are significant. By thresholding these eigenvalues, we can obtain different numbers of points. This keypoint detector is very useful in various tasks: corner and T-junction detection, flow-like texture analysis, shape cues, etc.

In Figure (3.5) all the interesting points (shown as grayvalues) of a pharmaceutical product that is taken in different poses are shown. In this case, we detect 4538 local interesting points whose eigenvalues are greater than a predefined threshold (λ_{thr}). By extracting the local maxima of the eigenvalues of expression (3.17), the number of keypoints is reduced significantly to 22. These local maxima are shown in fig. (3.5.b) as white crosses. The other instances of the same product, figures from (3.5.c) to (3.5.f), show that the local maxima are nearly the same.

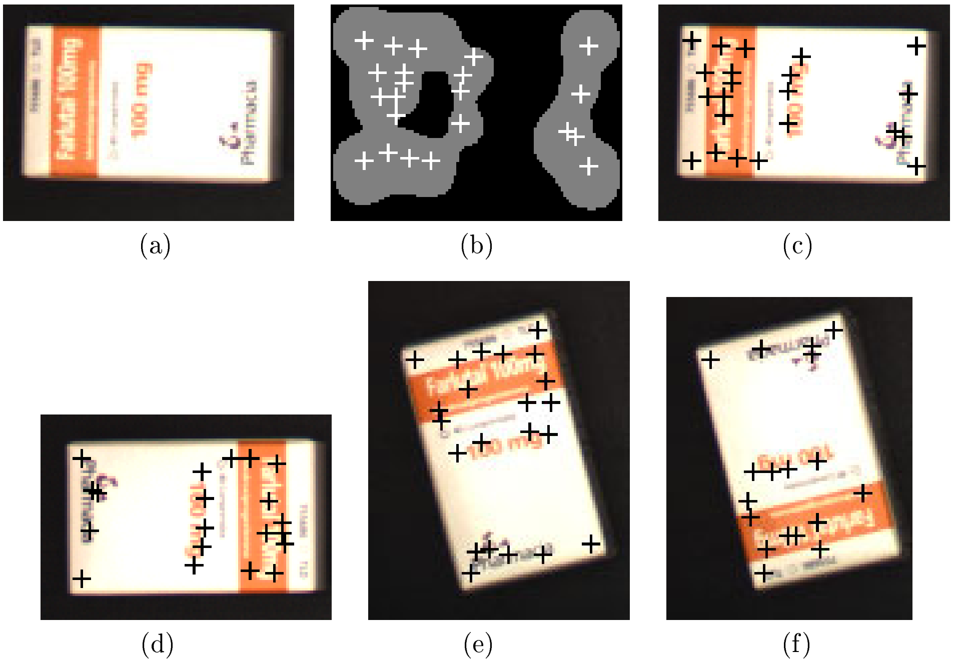
Once we have selected a set of keypoints we extract the salient features, in our case local color histograms. Since we find desirable to be able to capture the objects at any rotation angle, the extracted local histograms should be invariant to this feature. Extracting the histogram from a circular mask in the neighborhood of a keypoint minimizes the rotational effects that the image can suffer. Figure (3.6) shows which region is taken, given a set of detected keypoints. As mentioned, a robust probability density estimation requires a considerable number of data vectors. An excessive reduction of the eigenvalue threshold is not advisable since noninteresting points can be included. Instead, we can increase the amount of data vectors by taking into account the neighborhoods of the detected keypoints, and extracting more sample histograms from these neighborhoods.

So finally we are able to represent an object $\boldsymbol{H}$ (i.e. an image) belonging to one of K possible classes C^k, through the local histograms extracted from its L detected keypoints and neighbors $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_L]$. The training histograms for a certain class consist of the representative histograms for all the training objects of the class. For instance, given 2 images of a given object, with maybe 100 and 150 selected local histograms, respectively. We are representing this image with 250 512-dimensional samples. We can now turn to classification.

The χ^2 distance has been extensively used for histogram comparisons [129],

$$d_{\chi^2}(\boldsymbol{h}_1, \boldsymbol{h}_2) = \sum_d^D \frac{(h_{1d} - h_{2d})^2}{h_{1d} + h_{2d}} \tag{3.18}$$

So it seems natural to use this distance within the nearest neighbor approach, which can be adapted to our case making use of a voting scheme: given an image of an object



(a) (b) (c)

(d) (e) (f)

Figure 3.5: (a) A sample object image. (b) Gray values represent all the interesting points. White crosses represent local maxima. (c,d,e,f) Different instances of the product taken in different poses and their interesting points.

$\boldsymbol{H}_{Test}$, with L representative histogram, calculate the distances of these histograms to all histograms in the training set and assign the most voted class label to this object. In the experiments (3.18) was used as well as the $L2$ distance.

Within the Bayesian context, if the local histograms are assumed independent and the priors equiprobable, then the Maximum A Posteriori rule (1.3) for this particular problem results in,

$$C^{ML} = \arg \max_{k=1\ldots K} p(\boldsymbol{H}|C^k) = \arg \max_{k=1\ldots K} \prod_{l=1}^{L} p(\boldsymbol{h}_l|C^k). \qquad (3.19)$$

The class-conditional probabilities $p(\boldsymbol{h}_l|C^k)$ can be estimated using any of the methods exposed in chapter 1, always considering the limitations imposed to the estimation by the high dimensionality of $\boldsymbol{h}_l$. In general, these high-dimensional situations restrict this estimation to nonparametric kernel methods, because other approaches usually impose too many restrictions on the data, for instance Gaussianity. Also, semiparametric methods and their estimation algorithms become increasingly nonstable with dimensionality. But the precision in the estimation of the class-conditional probabilities is decisive on the performance of the classifier.

Two experiments were performed, the first one illustrates the properties of the CC-ICA representation for color distributions, mainly independence and sparsity. In

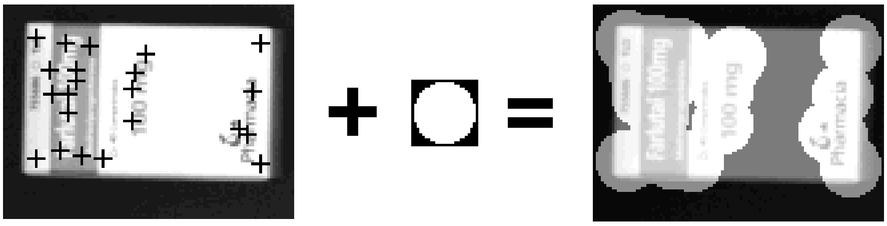


Figure 3.6: Given a set of detected keypoints, a circular region around each keypoint is used to extract its local histogram representation achieving rotational invariance. It can be seen that the non homogeneous regions are used to identify an object.

the second experiment we test CC-ICA classification for a large set of pharmaceutical products and compare this scheme with other nonparametric and probabilistic approaches.

For the first experiment we used two images of similar objects except for the color distribution, as can be seen in fig. (3.7). The images correspond to two milk boxes of the same brand. The full cream milk has predominant rose tones and we will refer to it as f-milk. The semi skimmed milk box has mainly green tones and we will refer to it as ss-milk. We have chosen these two objects because though they have the same color in a large portion of the image, their design is almost identical, and each of them contains a specific color tonality. Ideally and for a particular object, the ICA representation will provide a sparse coding for the color distribution of this object. As mentioned, this means that when a test histogram is recognized as belonging to this object it will have values close to zero in most of the components, and consequently have a high probability. For this experiment, a dataset of 144 representative 8-bin color histograms ($D = 512$) was extracted from both images using a predefined grid. Dimension was reduced from the original color histogram space of 512 dimensions to 35 using PCA and preserving a 99.9% of the total variation of the original data. We obtained the ICA representation for the f-milk and estimated the one dimensional densities corresponding to each independent component using a mixture of 2 zero-mean Laplacians.

We then manually selected a component from the f-milk ICA representation to illustrate independence and sparsity. Manual selection has to be performed because the ICA representation does not provide a natural hierarchy on its components. In this case we chose independent component number 19 due to the fact it represents a connected color distribution inside the object. This makes visualization more clear, but any other component would do. Figure (3.8) illustrates the activations of this component. The straight line in fig. (3.8.a) shows the value of component 19 for the representative histograms of the f-milk. These values correspond to a sparse distribution (concentrated around zero), and from these values, the density of the component was estimated. The dotted line in the same figure shows the value of component 19 when the representative histograms of the ss-milk are projected into the f-milk ICA representation. The ss-milk histograms randomly activate this component, yielding a low probability in the sparse distribution learnt from the f-milk histograms.

From Figure (3.8.a) we can deduce that the projection of histogram 120 of the

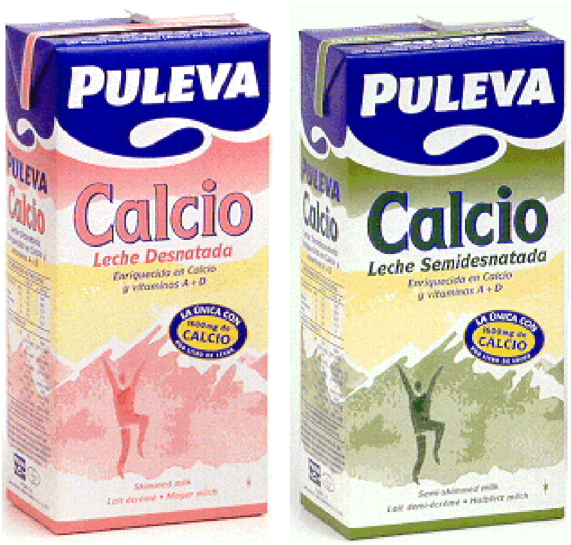


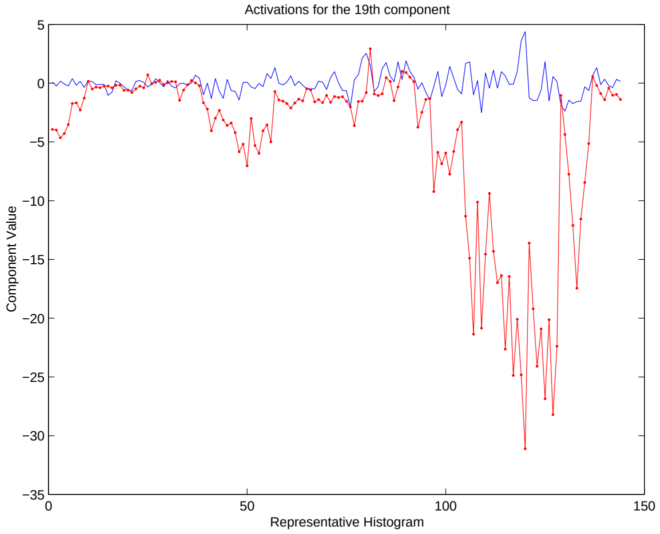
Figure 3.7: Full cream milk box with predominant rose tones (f-milk) and semi skimmed milk box with predominant green tones (ss-milk)

f-milk has the highest absolute value on component 19. Figure (3.8.b) plots the projection values of all the other components of this histogram. Since most of them are near zero, the probability for this histogram will be high. Histogram 120 is then a highly representative histogram for the f-milk. This is confirmed when we find out that histogram 120 corresponds to a neighborhood inside the human figure of the f-milk. ICA is gathering in the 19^{th} component the dark pink that corresponds to the color distribution of the human figure in the f-milk image.

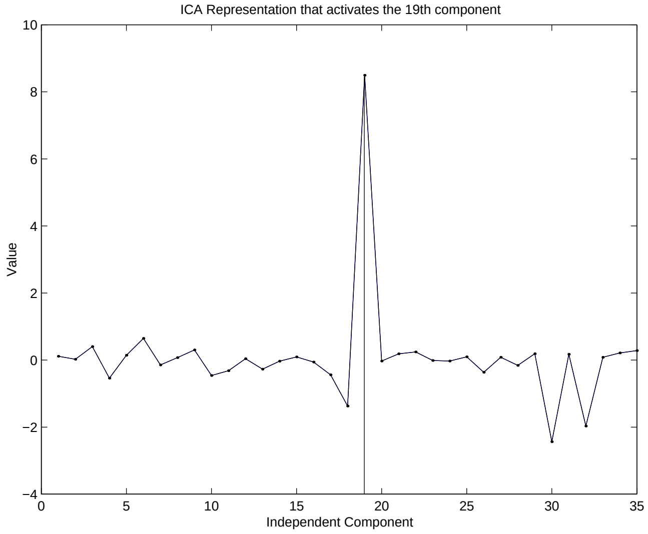
In figures (3.9.a) and (3.9.b) the probability map for components 19 and 21 of the f-milk ICA representation are shown. These maps were calculated projecting the color histogram in a neighborhood of every point of the image in this representation and then calculating the probability of this projection. So low probability values correspond to color distributions that activate the components. It can be seen how component 19 effectively captures the color distribution surrounding the human figure, while component 21 captures the color distribution around an ellipsoidal blue and yellow tag shared by both images. Figures (3.9.c) and (3.9.d) are the result of multiplying the image of the f-milk with a threshold of the probability map.

With this experiment it is observed the way the ICA representation separates color distributions in the input data, providing a sparse coding for each of these separated components. When we try to code an unlearnt color distribution with this coding, sparsity is lost so probabilities drop. It is also observed how this sparse coding can be effectively used for distinguishing the most dissimilar and unique regions between objects.

The second experiment CC-ICA performance for object recognition using local color histograms. For this experiment we counted with 2400 images of 400 different pharmaceutical products (the classes) with dark background. There a total of six



(a)



(b)

Figure 3.8: Shows the value of the 19^{th} independent component for the representative histograms of the full milk image (continuous line) and for the representative histograms of the semi skimmed milk image (line with dots). (b) Shows how this component is the only one activated upon the appearance of a certain color distribution in the full milk image.

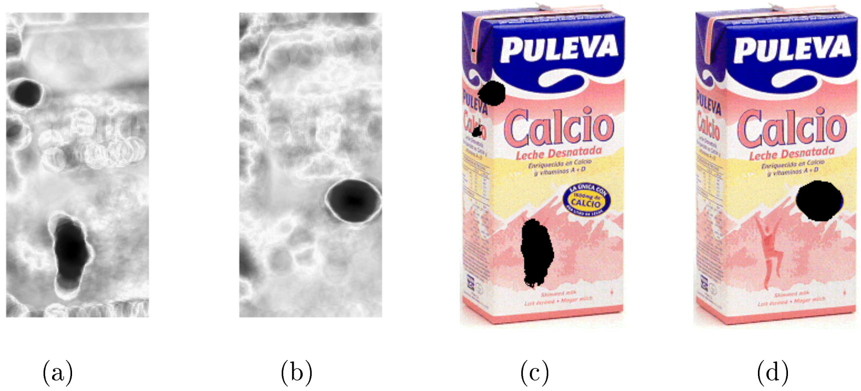


Figure 3.9: (a,b) Probability maps for component 19, activated by the color distribution of the human figure, and component 21 activated by the color distribution of the "Calcio" tag. (c,d) Thresholded probability map shown over original image.

images per class. These products present several color ambiguities so there is a lot of class overlap (some products are very similar only differing in reduced regions) and, in all the images, the background color is black an the illumination controlled. Figure (3.10) shows a subset of the pharmaceutical products used in the experiments.

From the six instances, five were used to train our statistical models in order to increase the accuracy of the probability estimation. From each image, we extracted a large amount of representative 8-bin local color histograms (around 150 interesting local regions per image) with the explained keypoint detector. So 400 ICA models were estimated in CC-ICA-Train from approximately 750 samples per class. The results of the classification are presented in table (3.5) where we can check that CC-ICA outperforms all the other statistical classification methods. The results are presented in terms of ranks where, for example, rank five means that the test product was correctly classified within the top five probabilities. For instance, ICA classified correctly 99.0% of the test images, if the top four positions are considered (rank 4). Particularly interesting is comparison with the Gaussian kernel approach to density estimation: any difference between this classifier and the CC-ICA approach can only be blamed on the level of accuracy of the density estimation. We also compare the local approach with the global approach in order to point out the advantages of searching for local cues. A nearest neighbor technique with Euclidean distance was used for classification (NNL2). In our problem, the background can be easily subtracted so both cases, with and without background, were considered. As expected, not considering the background performed better. But neither performed comparably to the local approaches. Actually, the recognition rates of the global approach is what led us to focus this particular problem with local strategies. Table (3.5) also includes these results.

For our object recognition experiment, the estimation of the 400 projection matrices took around 20 hours on a dual Pentium III with 850 Mhz, and the density estimation only fifteen minutes. Testing is straightforward since for each test object,

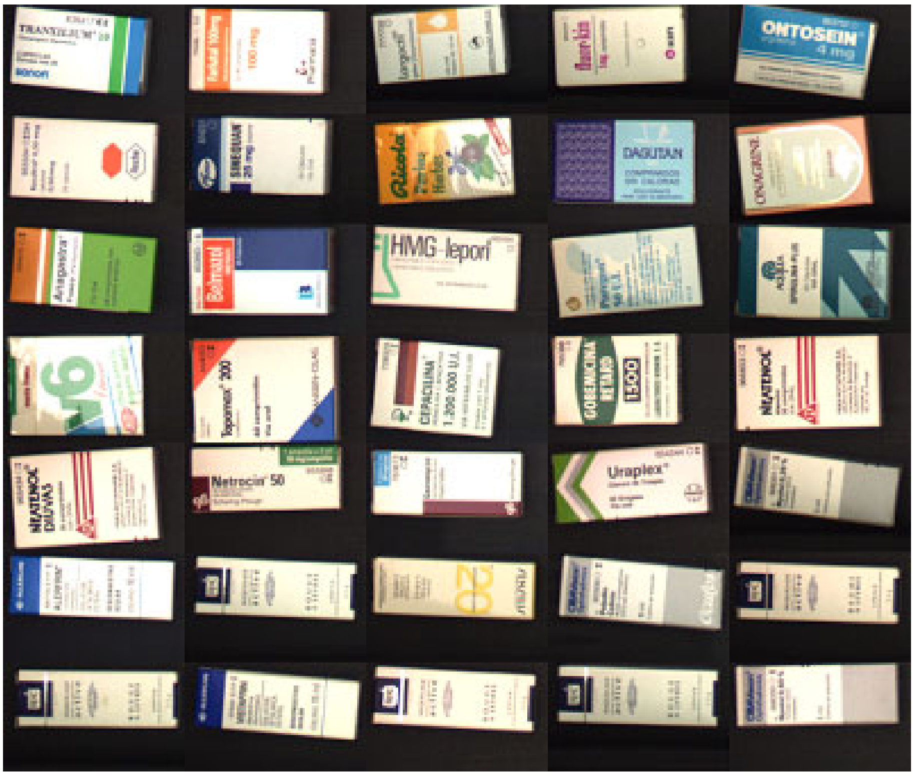


Figure 3.10: Subset of 35 pharmaceutical products used in our experiments.

K projections are needed and a simple algebraic operation obtains the probability on each component of the projected data. The probabilities are then added and compared for classification.

Land Use Classification from Multispectral Data

Within the remote sensing field, pattern classification finds a challenging problem in what is known as land use classification. Though a wide range of data can be used, this problem can be understood as: Given an image obtained from some kind of geographical sensor (multispectral satellite data, hyperspectral aereal imagery, radar data, etc.) the objective is to associate each pixel with a particular label indicating the use given to the land in that place [125]. Typical land use categories are seawater, urban area, wheat, conifers, fresh water, sand, rocks, etc. The categories in which the land use is divided are related with biological, social or even economical characteristics. By no means they are related with the spectral response of the different soils. This lack of relationship between the response and class labels generates what is known as spectral confusion.

This experiment is a partial result of the feasibility study that the *Centre de Visió per Computador (CVC)* did for the *Institut Cartogràfic de Catalunya (ICC)*

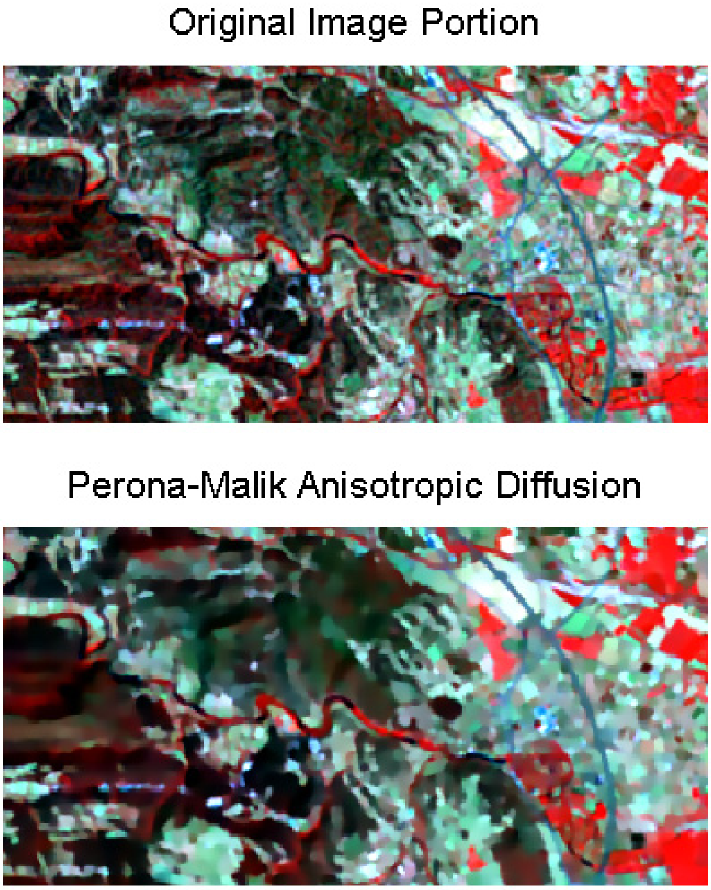
Method	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5
10 Nearest Neighbour $L2$ based technique	90.25 %	91.75 %	95.50 %	96.25 %	97.25 %
10 Nearest Neighbour χ^2 based technique	92.25 %	94.25 %	96.75 %	97.25 %	98.50 %
Gaussian Kernel based technique	89.75 %	91.25 %	93.50 %	96.00 %	96.50 %
Class-conditional ICA	**96.00** %	**98.25** %	**98.75** %	**99.00** %	**99.00** %
Global Histograms (NN$L2$ - with background)	80.50 %	83.50 %	85.25 %	86.00 %	86.0 %
Global Histograms (NN$L2$ - no background)	82.75 %	85.00 %	86.00 %	86.50 %	86.50 %

Table 3.5: Recognition percentages in the first five places for the six compared classification methods in the object recognition experiment.

along year 2001. The main objective of the study was the feasibility analysis of state of the art classification methods for land use classification from multispectral and multitemporal images. A region of approximately $668km^2$ was considered within the province of Girona in the Alt Empordà region, Catalonia. The available data for this region consisted in a temporal series of five images taken by the Enhanced Thematic Mapper (ETM) sensor from the Landsat 7 satellite. This sensor captures six spectral bands with an instantaneous field of view of 25 (IFOV, indicating meters per pixel), one thermal band with an IFOV of 60 and a high resolution panchromatic band with an IFOV of 15. If all 8 bands and 5 images are considered simultaneously this means we have $8 \times 5 = 40$ different values per pixel, so domain space dimensionality is 40. The images were taken along one year, with a 3 month interval between each other.

A total of 20 different classes were considered for labelling: road infrastructures, open spaces with scarce vegetation, deciduous forest, coniferous forest, rain forest, urban zones, parks and gardens, sea, fresh water, dry farming, dry herbs, irrigated farming, wet zones, housing schemes, vineyards, burnt land, industrial zones, woody zones, fruit plantations and sand. Training and test plots with ground truth data (the *actual* land use of that land portion at the time the image was taken) was made available. These training and evaluation plots accounted for 3.16% and 2.84% of the total surface to be analyzed. The region covered by the images, along with the ground truth, can be visualized in Appendix B, figure (B.1). This color image was obtained by selecting three clear bands (typically bands 7, 5 and 4) such that, arranged together, they provide a false color illusion that resembles the true image colors.

As with several pattern recognition problems, in the multispectral image case, preprocessing is a common and very convenient stage. The object of preprocessing is generally plain visualization, dimensionality reduction or enhancing discriminability. A very simple dimensionality reduction technique frequently encountered when working with multispectral ETM images is discarding the low-resolution spectral band. It can be seen, using any discriminant analysis technique that this band, besides its high IFOV, has very low if any discrimination capability. This leaves us with a dimensionality $D = 35$ since we are working with 5 images together. Another technique that has no effect on dimensionality but might positively affect visualization and discriminability is multispectral filtering. We have tested several anisotropic and isotropic diffusion techniques on the images such as Gaussian smoothing, coherenceenhancement anisotropic diffusion [159], etc. Finally, best results were obtained using the scheme known as Perona-Malik non-linear isotropic diffusion [118] which we briefly expose.



Original Image Portion

Perona-Malik Anisotropic Diffusion

Figure 3.11: Portion of the original image and diffusion filtering of the same portion. Three bands were chosen in this false color composite.

The final objective of image filtering is usually to suppress small details, standing out important and noticeable structures present within the image. From the classification perspective, detail suppression contributes with homogeneous regions, diminishing the class dispersion in domain space. The main risk of these techniques is that, by making an image homogeneous we might mix different classes augmenting spectral confusion. A way of avoiding this inconvenient is through edge-preserving filters. This type of filters attempt to respect edges within an image, smoothing only within regions where small variations are present. Still, they might present problems in preserving structures of objects present only in high resolution scales. From a theoretical point of view, diffusion techniques are the result of translating the physical theory of diffusion to the image context. This equation postulates the following evolution over time

$$\frac{\partial u}{\partial t} = div(D.\Delta u)u(x,0) = f(x) \tag{3.20}$$

where *div* stands for the divergence[2], depending on D, known as ***diffusion tensor*** and the spatial gradient of the image. So, given an image $f(x)$ we wish to filter, we place it as the initial condition of our system and we update it in time following (3.20). Finite differences are used to approximate the derivatives in this equation. The diffusion tensor indicates the directions and intensities of the diffusion. Diffusion will be isotropic if there are no privileged diffusion directions. In this case, the diffusion tensor is equivalent to a diffusion function. Diffusion will be non linear if we choose the tensor to depend on the local structure of the image. For instance, choosing $D = 1$ we have a linear isotropic diffusion model. In this case the diffusion equation results in the heat equation. It can be seen that, for this particular case, the solution of (3.20) is the result of successively convolving with Gaussians whose variance depends on the temporal step of the diffusion process. So Gaussian filtering is actually linear, isotropic diffusion. Perona, Shiota and Malik suggest [118] to make the diffusion tensor depend inversely on the gradient,

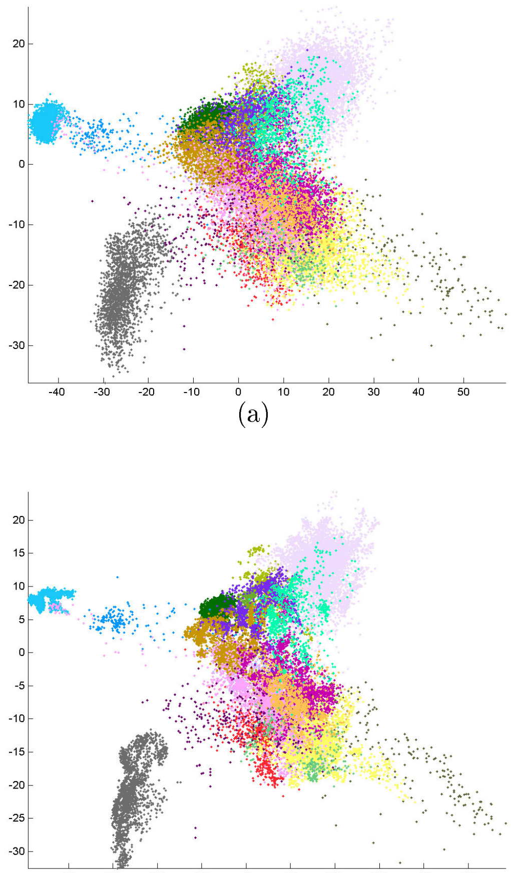
$$D = \frac{1}{1+(\frac{\|\Delta u\|}{\lambda})}^{2} \tag{3.21}$$

In this way, diffusion is low in those directions in which there are gradients with high norm: diffusion will respect edges. The choice of λ determines what we consider as an edge by minimizing or enhancing the effect of the image gradient. According to our definitions, this scheme is nonlinear and isotropic, nevertheless in literature it is frequent to encounter it as *Perona-Malik anisotropic diffusion*.

Diffusion filters generally have versions for multispectral images. For instance, we could choose to calculate the gradient in (3.21) from a weighted mean of all bands. We take a more simple approach, which proves effective in practice: apply the filter to each of the spectral bands independently. Figure (3.11) illustrates the effect of isotropic nonlinear filtering using (3.21) on a portion of the working image in fig.(B.1). Three bands are displayed and composite colors used.

But our main objective is classification, so this preprocessing can only be justified if it contributes in any way in improving this task. This is true if, for instance within class variation is reduced, while between class variation (separability) augmented. In fig. (3.12) we observe the distribution of the classes before and after diffusion. For visualization we considered projecting the original vectors in the 2 dimensional space given by the two principal discriminant directions obtained through Fischer discriminant analysis (FDA) [38]. This representation enhances the differences among classes so, the effect of the filtering can be readily observed. Even though FDA facilitates visualization, we will see in the experiments that it is not the best representation for classification. Among other inconvenients such as very approximate class means and similar covariance matrices, FDA is limited in its dimension by the total number of

[2] This vector field divergence, should not be confused with the statistical and information theoretic notion of divergence introduced in the next chapter



(a)

(b)

Figure 3.12: Portion of the original image and diffusion filtering of the same portion. Three bands were chosen in this false color composite.

classes. Figure (3.12.a) illustrates the class distributions for the original image, while in (3.12.b) the same distribution is observed for the filtered image. In the latter, more compact classes are observed and their separability is in effect enhanced. Consider the gray dots correspond to the class *forest fires*. While, at least for these two dimensions, certain confusion between this class and the others was present, in the filtered image this class is linearly separable from all others.

In remote sensing a statistic based on the confusion matrix other than accuracy is usually used to measure classification performance. Accuracy represents the number of correctly classified samples (in our case, pixels) with respect to the total number of samples. This definition only contemplates misclassification but does not take into account the type of error. Imagine a situation in which two classes C^1 and C^2 are present, 80% of the test samples belong to C^1, while the remaining 20% belong to

κ value	Interpretation
< 0.0	poor
$0.00 - 0.2$	slight
$0.21 - 0.4$	acceptable
$0.41 - 0.6$	moderate
$0.61 - 0.8$	substantial
$0.81 - 1.0$	almost perfect

Table 3.6: Interpretation of different values of the κ statistic.

C^2. Imagine two classifiers, one (CL1) that assigns all samples to C^1 and the other (CL2), that labels correctly all samples in C^2 but fails in 25% of the samples in C^1. The accuracy of these two classifiers is exactly the same (80%) but their performance differs and, depending on the real situation one of the two might be far more preferable than the other. For instance, if we know that samples from any of the two classes have equal probabilities of appearance, we can expect the performance of CL1 to drop to 50% while the performance of CL2 will reach 87.5%. The kappa statistic (κ) attempts to account for this difference basing its value on the $K \times K$ *confusion matrix* where K is the number of classes. The value of this matrix in position i, j is $x_{ij} = \{$pixel of class C^j classified as belonging to class $C^i\}$. According to this definition, the i-th row of the confusion matrix indicates the assignment of all pixels actually belonging to C^i. The sum of this row, represented as x_{i+} indicates how we performed on the C^i test pixels. The j-th column indicates the pixels assigned to C^j regardless of their true label, and the sum of this column x_{j+} measures how many pixels were classified as belonging to C^j. Notice also that the sum of the diagonal of the confusion matrix, divided by the total number of test samples results in the accuracy as we have used it so far. Using these measures, κ is defined as,

$$\kappa = \frac{N \sum_{k=1}^{K} x_{kk} - \sum_{k=1}^{K} x_{k+} x_{+k}}{N^2 - \sum_{k=1}^{K} x_{k+} x_{+k}}. \tag{3.22}$$

In table (3.6) a reference for different κ values and their interpretation in terms of evaluation performance is exposed [84].

In table (3.7) we expose a comparison of the final results using different representation approaches and classifiers. We also compare, in this table, the difference between applying the classifiers to the original and filtered multispectral data. Classification performance is exposed in terms of the κ statistic for both training and evaluation tests. Classifier notation follows that introduced in the previous section. From this table we can conclude that the simple maximum likelihood classifier with Gaussian mixtures per class, together with the CC-ICA approach have the best overall performance and that the latter very much benefits from the diffusion filtering. Possibly this preprocessing stabilizes the ICA estimation. Mixtures of Gaussians prove adequate for this problem since, even though dimensionality is moderately high, the number of samples for this problem is also high enough. Once we have chosen a classifier, for instance CC-ICA, and trained it, we can use CC-ICA-Test to assign a class label to every pixel in the test region to obtain the final land use map. In Appendix B,

Representation	Classifier	Filter	κ_{tr}	κ_{te}
original	1NN - L2 distance	no	-	.798
original	10NN - L2 distance	no	-	.833
original	10NN - L2 distance	yes	-	.841
Fischer D.A.	MLGAU	no	.771	.736
original	MLGAU	no	.923	.846
original	MLGAU	yes	.952	.851
original	NBGAU	no	.690	.681
original	MLGAUMIX	no	.968	.862
original	MLGAUMIX	yes	.988	.872
CC-ICA	NBGAUMIX	no	.915	.800
CC-ICA	NBGAUMIX	yes	.993	.872

Table 3.7: Comparative table of classification methods for the land use classification problem.

this land use map is shown together with a land use map manually made in the year 1998 to be used as a reference. One annoying characteristic of the obtained map is the pixelization observed in all the image. This has more relationship with the sensor data than with the classifier itself and is usually caused by subpixel structures that strongly affect the spectral response, generating outliers. It can be overcomed by several postprocessing techniques, being 3 × 3 median filtering the most widely used. The median, besides its robustness, is a natural statistic for our class labels since it is itself a class label.

A further insight in the results is provided by the confusion matrix obtained for each of the classifiers. In this confusion matrix, we can observe particular class confusions and try to improve this confusion using specific techniques. The confusion matrix resulting from the CC-ICA-NBGAUMIX classifier is shown, together with the land use map images in fig. (B.4) in appendix B. In this matrix we can observe strong differences in the classification performance for different classes. For instance, classes 8 and 15 have good overall classification (most of their test samples were correctly classified, and no other classes were confused with them). These classes correspond to sea water and forest fires, respectively. This could have been anticipated since the spectral response for these two classes is considerably different from the other responses. Instead, urban areas presented high levels of confusion among each other since all these classes have high levels of reflection for every band. These classes and their corresponding codes are, urban zones (6), housing schemes (13) and industrial zones (16). Note, in the confusion matrix, that these classes are mostly confused among each other or with sand (19) a class with similar spectral properties. Sand instead is confused with these three classes and with sea water. This suggests either an incorrect labeling or shallow zones, where the reflectance from the sandy sea-bed causes the confusion.

An additional advantage of using the CC-ICA scheme is that, after all, we are working within a strictly Bayesian framework, where it is straightforward to contemplate and include any kind of prior information. In this particular problem, prior information might arrive from the concrete knowledge of areas which have suffered relevant perturbation such as coastal lines, forest fires, river dams, etc., texture analysis of high resolution aereal images, digital elevation models, etc. The old land use map also results in a natural source of prior information. We will now see, how the prior information of any previous classification results, together with a very simple spatial model for the data can be included to improve classification.

Given a pixel $\rho = (x, y, \boldsymbol{f}(x, y))$ given by its position and D dimensional intensity vector $\boldsymbol{f}(x, y)$ we have that the likelihood $P(\rho|C^k)$ is the class-conditional likelihood for pixel ρ. Now consider a neighborhood of ρ which we will note as ω_ρ which includes the J pixels ρ_j. Then, the MAP solution to our classification problem is given by,

$$C^{MAP} = \arg \max_{k=1,\dots,K} P(\omega_\rho|C^k, \rho) P(C^k, \rho) \tag{3.23}$$

Direct estimation of the likelihood term $P(\omega_\rho|C^k, \rho)$ is not feasible due to its high dimensionality $(D \times J)$ and the highly probable fact we have far less training neighborhoods than training pixels. On the other side, assuming spatial pixel independence seems contradictory since it is from their interdependencies that we wish to take advantage of in our model. A feasible approximation is to use the weighted sum of the pixelwise probabilities as the probability of the whole neighborhood. This is equivalent to approximate the probability of the neighborhood event with the union of the probabilities of the pixels within the neighborhood, instead of the intersection as would be the case if independence were assumed. So we have that

$$P(\omega_\rho|C^k, \rho) \overset{def}{=} \sum_{j=1}^{J} \lambda_j P(\rho_j|C^k, \rho) \tag{3.24}$$

with $\sum \lambda_j = 1$. In practice, the weights were chosen as the values of Gaussian centered in pixel ρ. What we are actually doing is defining the class-conditional probability of the class for a certain neighborhood as the result of convolving the pixelwise class-conditional probabilities with a mask the size of the neighborhood.

The prior term $P(C^k, \rho)$ can be understood as the degree of confidence we have in finding, simultaneously pixel ρ and class C^k. If we count with a previous land use map which makes the mapping to a certain label $LU(\rho)$, a very simple approximation can be to use

$$P(C^k, \rho) = \begin{cases} \alpha & \text{if } C^k = LU(\rho), \\ \frac{1-\alpha}{K-1} & \text{otherwise.} \end{cases} \tag{3.25}$$

with $1/K \leq \alpha < 1$. With this choice we are giving the old map a degree of confidence α. If α is chosen as $1/K$, equiprobable classes are assumed and if alpha is chosen as 1 we risk ourselves to obtaining null posterior probabilities for every class. In practice, the value of α can be obtained from the training set.

We have applied this scheme together with CC-ICA classification to our problem. The size of the neighborhoods was chosen as 5×5 and α was conservatively chosen as 0.2. In our problem of 20 classes this means that classes differing from the old map labelling are given approximately 5 times less probability than those which agree. Results are considerably improved and it can be seen that differences are preserved in the sense we are not actually making a copy of the old land use map. When a pixel is classified as belonging to a new class with high confidence, the probability of the old label is so low that the prior weights have no effect in changing the choice. A good example of this situation can be the forest fires (see the top right corner of the land use maps in the appendix, grey color code). The fires occurred after the 1998 land use map was prepared so this map misclassifies all the pixels belonging to regions affected by the fire. In the CC-ICA land use map, approximately 99% of these pixels were correctly classified. After the inclusion of the prior information, less than 0.5% of these correctly classified pixels lost their true label. Final results give, for CC-ICA with a MAP approach using the 1998 land use map as prior information, $\kappa_{tr} = 0.981$ and $\kappa_{te} = 0.936$.

Visual Inspection of Cork-Stoppers

The last example with real-world data we give in this chapter arises from the industrial vision problem of visual inspection of cork stoppers. Cork inspection is the least automated task in the production cycle of the cork stopper. Due to the inspection difficulty of the natural cork material and the high production rates even the most experienced quality inspection operators frequently make mistakes. In addition, it is increasingly difficult to find labor willing and able to do a job that is at the same time both skilled and highly repetitive. On the other hand, human inspection leads to a lack of objectivity and uniform rules applied by different people at different time. As a result, there is a urgent need to modernize the cork industry in this direction. In this section, we consider a real industrial computer vision application of classification of natural (cork) products. Cork products in the manufacture are inspected for different faults like small holes due to insect attacks, channels due to imprecise stopper cutting, stopper breaking, cracks and woody surfaces. Although it does not seem difficult for human beings to detect different faults in the cork material, it turns out difficult to precisely formulate the features of the cork faults due to the porosity of the natural material. It is difficult even for the cork quality experts to exactly define all cork features that they take into account in the process of stopper inspection, the feature values and ranges in order to define whether there is a fault in the cork stopper or if the stopper is of poor quality. There have been different attempts to develop vision cork inspection systems in the manufacture where the people working in the manufacture should define the values and ranges of the image features and elaborate the decision rules in the process of the stopper inspection. Given that people in the manufacture work with rather qualitative than quantitative information to classify the quality of a stopper, managing such vision cork inspection systems represent a tedious and time-consuming task. The problem of the classification of the cork product in different (in this case, five) quality groups additionally difficult the problem. This fact prevents cork stopper industry from defining and assuring the quality of the products in front of the providers.

In this experiment we analyse the performance of the CC-ICA classifier to the problem of classification of cork stopper images into one of five classes, defined according to the quality of the stopper. These quality groups are illustrated in fig. (3.13). To this purpose the operators have provided training examples. In order

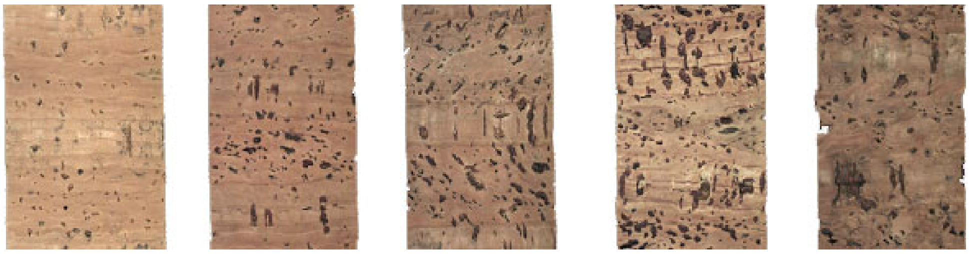


Figure 3.13: Cork stoppers of 5 quality groups ordered from best to worst quality (from left to right).

Classifier	dimensionality	accuracy
NN	43	40.7%
NN-PCA	11	46.1%
NN-FDA	4	49.6%
NN-NDA	14	58.5%
MLGAU	32	90.3%
CCICA-GAUMIX	32	98.1%

Table 3.8: Comparative table of classification methods for the cork stoppers classification problem.

to classify the cork stoppers we extract 43 image features. A blob analysis is done and blob features are considered as follows: stopper area, number of blobs, average blob area, average blob elongation, average blob grey-level, average compactness, average roughness, features of the blob with largest area (area, length, perimeter, convex perimeter, compactness, roughness, elongation, length, width, average blob gray-level, position with respect to the center of the stopper), features of the longest blob, etc. Approximately 400 samples were used for evaluating the classifier using a leave-one-out technique.

As with the other experiments, our approach has been compared with other classifiers and representation techniques. In this case, comparison was performed with the nearest neighbor classifier applied directly on the measurements (NN), the same classifier after PCA dimensionality reduction (NN-PCA), after Fisher discriminant analysis (NN-FDA) [41], and after nonparametric discriminant analysis (NN-NDA) [49] (see chapter 5), the maximum likelihood classifier using a Gaussian distribution per class (ML-GAU) and CC-ICA as described in this chapter (CCICA-GAUMIX). Results are exposed in table (3.8). In the nearest neighbor case, different choices for the number of nearest neighbors were considered and the best results were achieved using a single nearest neighbor. For PCA, different dimensionalities were also considered and the best results achieved after reducing the dimensionality to 11. With NDA this optimal dimensionality was 14. For the statistical classifiers a dimensionality of 32 was considered, by direct mean Bhattacharyya feature selection (see next chapter). The results in the table correspond to these optimal configurations.

It is quite clear from this table that, for this particular experiment, parametric methods outperform nonparametric techniques. In the case of CC-ICA, the resulting classification rate can only be blamed on the improved accuracy in the density estimation provided by working in the CC-ICA framework.

3.6 Conclusions

We have mentioned that the only good reason for applying a linear transform to data is the belief that this will benefit accuracy of density estimation. For the high-dimensional case, it is frequent to assume variable independence in order to factorize the multidimensional distribution in terms of the marginal distributions. In this case, the accuracy of the estimation will be improved as long the independence assumption can be strongly held. We observe that global independence does not imply class-conditional independence making it difficult to use a single linear transform that allows factorization of all the class-conditional densities and motivating the need for a class-conditional framework. We then introduce class-conditional independent component analysis as a solution to this problem showing how this choice of representation can be used within a Bayesian scheme. The algorithms for learning our representation and classifier from data and evaluating new data samples in order to classify them are presented. We finally test these algorithms on artificial, benchmark and real-world data. The experiments show that our scheme performs comparably or better than commonly used classifiers when dimensionality is small, and is robust to high dimensionality. In particular, when comparing with other Bayesian classifiers this improvement in performance can only be blamed on the accuracy of the density estimation. The experiments also confirm that naive Bayes performance is considerably improved when correct independence assumptions can be made on the class-conditional densities.