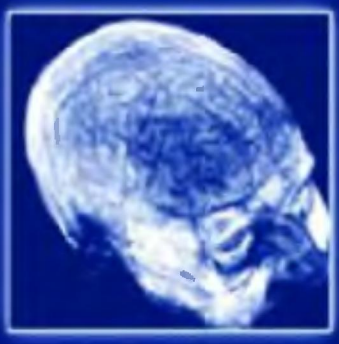
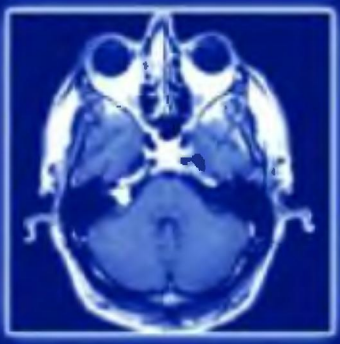
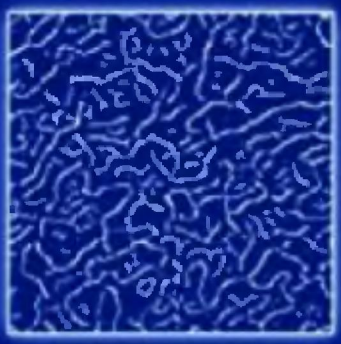
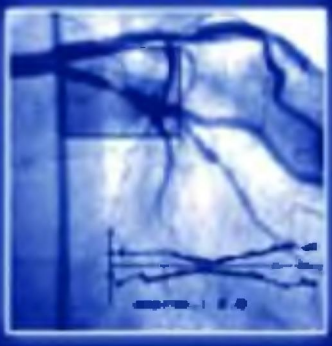
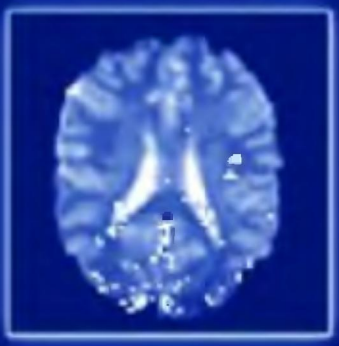
Pattern Recognition for Medical Imaging

Anke Meyer-Bäse

Elsevier Academic Press
525 B Street, Suite 1900, San Diego, California 92101-4495, USA
84 Theobald's Road, London WC1X 8RR, UK

This book is printed on acid-free paper. ♾

Copyright 2004, Elsevier, Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher.

Permissions may be sought directly from Elsevier's Science & Technology Rights Department in Oxford, UK: phone: (+44) 1865 843830, fax: (+44) 1865 853333, e-mail: permissions@elsevier.com.uk. You may also complete your request on-line via the Elsevier Science homepage (http://elsevier.com), by selecting "Customer Support" and then "Obtaining Permissions."

Library of Congress Cataloging-in-Publication Data

Meyer-Bäse, Anke.
Pattern recognition in medical imaging / Anke Meyer-Bäse.
p. ; cm.
Includes bibliographical references and index.
ISBN 0-12-493290-8
1. Imaging systems in medicine. 2. Diagnostic imaging--Mathematics. 3. Image processing--Mathematics. 4. Pattern recognition systems. I. Title.
[DNLM: 1. Diagnostic Imaging--methods. 2. Pattern Recognition. 3. Image Processing, Computer-Assisted. 4. Mathematical Computing. WN 26.5 M612p 2003]
R857.06M495 2003
616.07'54--dc22

2003058316

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN: 0-12-493290-8

For all information on all Academic Press publications
visit our website at www.academicpress.com

Printed in the United States of America
03 04 05 06 07 08 9 8 7 6 5 4 3 2 1

CONTENTS

DEDICATION v

PREFACE xvii

ACKNOWLEDGMENTS xxi

I
INTRODUCTION

1.1. Microscopic Image Analysis 2
1.2. Macroscopic Image Analysis 4

II
FEATURE SELECTION AND EXTRACTION

2.1. Introduction 14
2.2. Role of Feature Selection and Extraction 15
2.3. Preliminary Notations 17
2.4. Feature Extraction Methods 17
2.5. Feature Selection Methods 41

III
SUBBAND CODING AND WAVELET TRANSFORM

3.1. Introduction 50
3.2. The Theory of Subband Coding 51
3.3. The Wavelet Transform 65
3.4. The Discrete Wavelet Transformation 73
3.5. Multiscale Signal Decomposition 75
3.6. Overview: Types of Wavelet Transforms 91

IV
THE WAVELET TRANSFORM IN MEDICAL IMAGING

4.1. Introduction 97
4.2. The Two-Dimensional Discrete Wavelet Transform 98
4.3. Biorthogonal Wavelets and Filter Banks 100
4.4. Applications 105

V
GENETIC ALGORITHMS

5.1. Introduction 121
5.2. Encoding and Optimization Problems 122
5.3. The Canonical Genetic Algorithm 123
5.4. Optimization of a Simple Function 127
5.5. Theoretical Aspects of Genetic Algorithms 130
5.6. Feature Selection Based on Genetic Algorithms 133

VI
STATISTICAL AND SYNTACTIC PATTERN RECOGNITION

6.1. Introduction 136
6.2. Learning Paradigms in Statistical Pattern Recognition 137
6.3. Parametric Estimation Methods 139
6.4. Nonparametric Estimation Methods 148
6.5. Binary Decision Trees 154
6.6. Syntactic Pattern Recognition 156

6.7. Diagnostic Accuracy Measured by ROC-Curves 167
6.8. Application of Statistical Classification Methods 171
6.9. Application of Syntactic Pattern Recognition 177

VII
FOUNDATIONS OF NEURAL NETWORKS

7.1. Introduction 186
7.2. Multilayer Perceptron (MLP) 189
7.3. Self-Organizing Neural Networks 197
7.4. Radial Basis Neural Networks (RBNN) 204
7.5. Transformation Radial Basis Neural Networks 215
7.6. Hopfield Neural Networks 219
7.7. Comparing Pattern Recognition Methods 221
7.8. Pixel Labeling Using Neural Networks 223
7.9. Classification Strategies for Medical Images 225
7.10. Classifier Evaluation Techniques 228

VIII
TRANSFORMATION AND SIGNAL-SEPARATION NEURAL NETWORKS

8.1. Introduction 234
8.2. Neurodynamical Aspects of Neural Networks 235
8.3. PCA-Type Neural Networks 241
8.4. ICA-Type Neural Networks 254

IX
NEURO-FUZZY CLASSIFICATION

9.1. Introduction 282
9.2. Fuzzy Sets 283
9.3. Neuro-Fuzzy Integration 285
9.4. Fuzzy Neural Network 287
9.5. Fuzzy Clustering 288
9.6. Comparison of Fuzzy Clustering versus PCA for fMRI 309
9.7. Fuzzy Algorithms for LVQ 311

X
SPECIALIZED NEURAL NETWORKS RELEVANT TO BIOIMAGING

10.1. Introduction 318
10.2. Basic Aspects 319
10.3. Convolution Neural Networks (CNNs) 322
10.4. Hierarchical Pyramid Neural Networks 325
10.5. Problem Factorization 327
10.6. Modified Hopfield Neural Network 328
10.7. Hopfield Neural Network Using a Priori Information 332
10.8. Tumor Boundary Detection 336
10.9. Cascaded Self-Organized Neural Network 341

XI
NEURAL-BASED COMPUTER-AIDED DIAGNOSIS SYSTEMS IN BREAST CANCER DETECTION

11.1. Introduction 346
11.2. Segmentation of Mass and Normal Breast Tissue 349
11.3. Classification of Mass and Normal Breast Tissue 354
11.4. CAD System for Mass Detection 357
11.5. Microcalcification Analysis System 360
11.6. CAD System for Microcalcification Detection 363

REFERENCES 367

INDEX 383

GLOSSARY

AFNM	Adaptive fuzzy n-means
AFNS	Adaptive fuzzy n-shells
AO	Alternating optimization
APEX	Adaptive principal components analysis using lateral inhibition
AR	Autoregressive
AUC	Area under the ROC curve
CAD	Computer-aided design
CNN	Convolution neural network
CNR	Contrast-to-noise ratio
CT	Computer tomography
CWT	Continuous wavelet transform
DCT	Discrete cosine transform
DNA	Desoxyribonucleic acid
DST	Discrete sine transform
DWT	Discrete wavelet transform
fMRI	Functional magnetic resonance imaging
fn	False negative
fp	False positive
FALVQ	Fuzzy algorithm for learning vector quantization
FCA	Fuzzy clustering analysis
FIR	Finite impulse response
FNM	Fuzzy n-means

FT	Fourier transform
GA	Genetic algorithm
GAFNM	Generalized adaptive fuzzy n-means
GAFNS	Generalized adaptive fuzzy n-shells
GFNM	Generalized fuzzy n-means
GFNS	Generalized fuzzy n-shells
HPNN	Hierarchical pyramid neural network
ICA	Independent component analysis
Infomax	Information maximization
JPEG	Joint photographic experts group
KNN	K-nearest neighbor
LC	Linear classifier
LBG	Linde-Buzo-Gray
LMS	Least mean squares
LPC	Linear predictive coding
LTM	Long term memory
LVQ	Learning vector quantization
MEG	Magnetoenchephalographic
MIAS	Mammographic Image Analysis Society
MLP	Multilayer perceptron
MPEG	Motion Photographic Experts Group
MRI	Magnetic resonance imaging
NMSE	Normalized mean-squared error
NSNR	Normalized signal-to-noise ratio
PCA	Principal component analysis
PET	Positron emission tomography
PR-QMF	Perfect reconstruction quadrature mirror filter
PSNR	Peak signal-to-noise ratio
QC	Quadratic classifier
RCE	Restricted Coulomb energy
ROC	Receiver operating characteristics
ROI	Region of interest
SPECT	Single-photon-emission-computer-tomography
STM	Short term memory
tn	True negative
tp	True positive
US	Ultrasound
WT	Wavelet transform

LIST OF SYMBOLS

a	Scale parameter
$a_t(i)$	Activity of neuron i at time t
a_{ij}	Connection between the ith and the jth neuron of the same layer
$\mathbf{a}_i$	Vector of lateral connections of the ith neuron
$\mathbf{A}$	Mixing matrix
b	Shift parameter
$c_{m,n}$	Scaling coefficient at scale m and translation n
C	Fuzzy set
$d_{m,n}$	Wavelet coefficient at scale m and translation n
$d(\mathbf{x}, \mathbf{y})$	Distance between data vectors $\mathbf{x}$ and $\mathbf{y}$
$d(\mathbf{x}, \mathbf{m^i}, \mathbf{K^i})$	Mahalanobis distance of the ith receptive field
$D(\mathbf{x_j}, \mathbf{L_i})$	Dissimilarity between a data point $\mathbf{x_j}$ and a prototype $\mathbf{L_i}$
E_p	Error function due to the pth feature vector
f_i	Evaluation associated with the ith string
f_v	Approximation signal
f_w	Detail signal
$\bar{f}$	Average evaluation of all strings in the population
$f_i(x_i)$	Output function of the ith neuron for the current activity level x_i
$\mathbf{f}(\mathbf{x}(t))$	Vector of output functions
$\mathbf{F_X}$	Input field of a neural network

$\mathbf{F_Y}$	Output field of a neural network
$g_i(n)$	Filter coefficients in the synthesis part
$\mathbf{G}$	Grammar
$\mathbf{G}(\|\|\|\|)$	Radial basis function
$h_i(n)$	Filter coefficients in the analysis part
$H(\mathbf{y})$	(Differential) entropy of stochastic signal $\mathbf{y}$
$H(y_s, \mathbf{d})$	Probability that a given difference y_s occurs at distance $\mathbf{d}$
$I(y_1, \cdots, y_m)$	mutual information between m random variables y_i
$I(A_i, \mathbf{L_i})$	Inadequacy between a fuzzy class A_i and its prototype $\mathbf{L}_i$
J	Criterion function
$J(\mathbf{P}, \mathbf{L})$	Criterion function (inadequacy) for partition $\mathbf{P}$ and representation $\mathbf{L}$ for GFNM algorithm
$J(\mathbf{P}, \mathbf{L}, \mathbf{M})$	Criterion function for partition $\mathbf{P}$ and representation $\mathbf{L}$ and shape matrix $\mathbf{M}$ for GAFNM algorithm
$J(\mathbf{P}, \mathbf{V}, R)$	Criterion function for partition $\mathbf{P}$ and hypersphere prototypes $\mathbf{L}$ described by set of radii R and centers $\mathbf{V}$ for GFNS algorithm
$J(\mathbf{P}, \mathbf{V}, R, \mathbf{M})$	Criterion function for partition $\mathbf{P}$ and hyperellipsoidal prototypes $\mathbf{L}$ described by set of radii R and centers $\mathbf{V}$ and shape matrix $\mathbf{M}$ for GAFNS algorithm
$J(\mathbf{y})$	Negentropy of stochastic signal $\mathbf{y}$
$\mathbf{K^i}$	Inverse covariance matrix of the ith receptive field
$K(f\|g)$	Kullback divergence between the pdfs f and g
L	Language or log–likelihood function in ICA
$L(X)$	Family of all fuzzy sets on universe X
$\mathbf{L}_i$	Prototype of fuzzy class A_i
$\mathbf{L}_i(\mathbf{v}_i, r_i)$	Hyperspherical shell prototype of the fuzzy class A_i
$\mathbf{L}_i(\mathbf{v}_i, r_i, \mathbf{M}_i)$	Hyperellipsoidal shell prototype of the fuzzy class A_i
$\mathbf{m}_i$	Centroid of the ith receptive field
M	Factor of the sampling rate change (either decimation or interpolation)
$\mathbf{M}$	Feedforward weight matrix between field $\mathbf{F_X}$ and field $\mathbf{F_Y}$ or shape matrix in fuzzy logic
$\mathbf{N}$	Feedback weight matrix between field $\mathbf{F_Y}$ and field $\mathbf{F_X}$
$p(\mathbf{x})$	Probability density function of $\mathbf{x}$
$p(\mathbf{x}\|\omega_i)$	Likelihood function of ω_i with respect to $\mathbf{x}$
$\mathbf{p}$	Feature vector
P	Set of production rules or fuzzy partition in fuzzy logic
$\mathbf{P}$	Intraconnection matrix in field $\mathbf{F_X}$
$P(\omega_i)$	A-priori probability of feature class ω_i
$P(\omega_i \mathbf{x})$	A-posteriori probability for a pattern

	$\mathbf{x}$ to belong to class ω_i
$P_m f$	Lowpass content of f in the subspace V_m
$\mathbf{q}_i$	ith eigenvector
$Q_m f$	Highpass content of f in the subspace W_m
$\mathbf{Q}$	Eigenvector matrix
r_i	Radius of hypersphere of prototype $\mathbf{L}_i$
R	Set of radii for prototypes $\mathbf{L}$
$\mathbf{R}$	Correlation matrix
$\mathbf{s}$	Source signal
$\mathbf{S}_i$	Within-class scatter matrix of fuzzy class A_i
S	Start variable
T_1	Time constant for the molecules to recover their orientation
T_2	Decay time constant
u_{ij}	Membership value of jth input vector to ith fuzzy class
$\mathbf{U}$	Membership matrix
$\mathbf{v}$	Chromosome (binary vector)
$\mathbf{v}_i$	Center of the hypersphere of the prototype $\mathbf{L}_i$
V	Auxiliary alphabet or set of centers in fuzzy logic
V_m	Subspace of the approximation signals
w_{ij}	Feedforward connection between the ith and the jth neuron
$\mathbf{w}_i$	Feedforward weight vector for the ith neuron
$\mathbf{W}$	Demixing matrix
W_m	Orthogonal complement of V_m
$x(n)$	Input signal
$x^{'}(n)$	Intermediate signal
$\mathbf{x}$	Input vector (either feature vector or neural activity vector) or sensor signal
X	Remaining feature set or universe of discourse in fuzzy logic
$y(n)$	Output signal
$\mathbf{y}$	Output vector
Y	Initial available feature set
Z	Alphabet
η	Learning rate
$\phi[\mathbf{y}]$	Contrast function of output $\mathbf{y}$
σ_j^i	jth standard deviation of the ith receptive field
$\psi(t)$	Wavelet function
$\phi(t)$	Scaling function
$\gamma(\mathbf{x}, \mathbf{m})$	Potential function
λ_i	ith eigenvalue

ω_i	ith feature class
ϱ	Proton density
$\oplus$	Orthogonal sum
$\perp$	Orthogonal complement

PREFACE

Over the years, medical imaging has become a vital part of the early detection, diagnosis, and treatment of cancer. In some cases, medical imaging is the first step in preventing the spread of cancer through early detection, and in many cases makes it possible to cure or eliminate the cancer altogether. Computed tomography (CT) imaging, magnetic resonance imaging (MRI), mammography, nuclear medicine (NM) imaging, ultrasound (US) imaging, and X-ray imaging are all very important tools in the fight against cancer.

Images in analog format do not facilitate simple distribution to multiple sites, either in-hospital or off-site; besides, there are increased costs for storage, archiving, and retrieval of films. On the other hand, digital image processing provides the possibility of easy image retrieval, efficient storage, rapid image transmission for off-site diagnoses, and the maintenance of large image banks for both educational and research purposes. Its basic operations, such as filtering, enhancement, classification, and combination of images obtained from different modalities, can assist screening, diagnosis, research, and treatment. Pattern recognition techniques play a critical role when applied to medical imaging by fully automating the process of abnormality detection and thus supporting the development of computer-aided diagnosis (CAD) systems. Often this involves identifying structures such as tumors or lesions, but it can also include monitoring present structures such as the size of the heart in chest X-rays. In most cases, CAD systems are designed to be used for screening purposes, in which large numbers

of images need to be examined, but relatively few would be expected to show abnormalities. They are adopted as an alternative "second opinion" that can assist a radiologist in detecting lesions and in making diagnostic decisions. Mostly used in digital mammography, CAD systems detect objectively, accurately, and reproducibly, and analyze mammographic lesions to reduce the number of false-negative mammograms (missed detections) and the number of false-positives (diagnoses of malignancy). The computerized schemes combine detection and classification of malignancies. The importance of these CAD systems in almost all telemedicine applications is evident and is expected to increase dramatically over the coming years. For example, the FDA has approved fully digital mammography systems for the screening and diagnosis of breast cancer. Also, integrated medical systems are expected to support all telemedicine solutions of the future.

Medical imaging is today becoming one of the most important visualization and interpretation methods in biology and medicine. The past decade has witnessed a tremendous development of new, powerful instruments for detecting, storing, transmitting, analyzing, and displaying images. These instruments are greatly amplifying the ability of biochemists, biologists, medical scientists, and physicians to see their objects of study and to obtain quantitative measurements to support scientific hypotheses and medical diagnoses. An awareness of the power of computer-aided analytical techniques, coupled with a continuing need to derive more information from medical images, has led to a growing application of digital processing techniques for the problems of medicine. The most challenging aspect in medical imaging lies in the development of integrated systems for use in the clinical sector. Design, implementation, and validation of complex medical systems require not solely medical expertise but also a tight collaboration between physicians and biologists, on the one hand, and engineers and physicists, on the other. It is well known that it was the interdisciplinary collaboration between a physicist, G. N. Hounsfield, and a neuroradiologist, J. Ambrose, that led in the late 1960s to the development of the first computer tomographic scanner. Noise, artifacts, and weak contrast are the cause of a decrease in image quality and make the interpretation of medical images very difficult. These sources of interference, which are of a different nature for mammograms than for ultrasound images, are responsible for the fact that conventional or traditional analysis and detection algorithms are not always successful. Poor image quality always leads to problematic and unreliable feature extraction, analysis, and recognition in many medical applications. Therefore much of the research done today is geared toward improvement of the reduced quality of the available image material.

The goal of the present book is to present a complete range of proven and new methods, which play a leading role in the improvement of image quality, as well as analysis and interpretation, in the modern medical imag-

ing of this decade. These methods offer solutions to a vast number of problems, for which the classical methods provide only insufficient solutions. Chapter I provides an overview of the diverse methods of microscopic and macroscopic image analysis. Imaging with ionization radiation, magnetic resonance imaging, ultrasound and ultrasonic imaging, and biomagnetic imaging plays a central role in the present book and is described in detail. Chapter II contains a description of methods for feature selection and extraction. Feature selection methods presented are nontransforming and transforming signal characteristics, graphical and structural descriptors, and texture. Methods for feature extraction are exhaustive search, branch and bound algorithm, max-min feature selection, and Fisher's linear discriminant function. Wavelets, for example, are leaders in edge extraction, compression, noise cancellation, feature extraction, image enhancement, and image fusion and occupy, therefore, a central role in this book. Two chapters are dedicated for discussion of wavelets: A mathematical basic part, Chapter III, and an application part, Chapter IV, regarding the application of the wavelets to the above-mentioned problems. Another basic feature extraction method having its roots in evolution theory is genetic algorithms, discussed in Chapter V. Both genetic algorithms and neural networks are among the few approaches for large-scale feature extraction providing an optimal solution for extraction of relevant parameters. Chapters VI to X describe cognitive and noncognitive classification methods relevant for medical imaging. Chapter VI develops the traditional statistical classification methods, presenting both parametric and nonparametric estimation methods, and the less known syntactic or structural approach. The potential of the methods presented in Chapters II through VI is illustrated by means of relevant applications in radiology, digital mammography, and fMRI. Neural networks have been an emerging technique since the early 1980s and have established themselves as an effective parallel processing technique in pattern recognition. The foundations of these networks are described in Chapter VII. Chapter VIII reviews neural implementations of principal and independent component analysis and presents their application in medical image coding and exploratory data analysis in functional MRI. Besides neural networks, fuzzy logic methods represent one of the most recent techniques applied to data analysis in medical imaging. They are always of interest when we have to deal with imperfect knowledge, when precise modeling of a system is difficult, and when we have to cope with both uncertain and imprecise knowledge. Chapter IX develops the foundations of fuzzy logic and that of several fuzzy clustering algorithms and their application in radiology, fMRI, and MRI. Chapter X details the emerging complex neural architectures for medical imaging. Specialized architectures such as invariant neural networks, context-based neural networks, optimization networks, and elastic contour models are very detailed. The chapter also includes the application of convolutional neural networks, hierarchi-

cal pyramidal neural networks, neural networks with receptive fields, and modified Hopfield networks to almost all types of medical images. Principal component analysis and independent component analysis for fMRI data analysis based on self-organizing neural networks is also shown as a comparative procedure. Compression of radiological images based on neural networks is compared to JPEG and SPHIT wavelet compression. Chapter XI describes integrated complex computer-aided diagnosis systems for medical imaging and shows the application of most feature selection and classification methods from the previous chapters.

The emphasis of the book lies in the compilation and organization of a breadth of new approaches, modeling, and applications from pattern recognition relevant to medical imaging. Many references are included and are the basis of an in-depth study. Only basic knowledge of digital signal processing, linear algebra, and probability is necessary to fully appreciate the topics considered in this book. Therefore, I hope that the book will receive widespread attention in an interdisciplinary scientific community.

ACKNOWLEDGMENTS

A book does not just "happen," but requires a significant commitment from its author as well as a stimulating and supporting environment. The author has been very fortunate in this respect. The environment in the Department of Electrical and Computer Engineering and the College of Engineering at Florida State University was also conducive to this task. My thanks to Dean Ching-Jen Chen and to the Chair, Reginald Perry.

I would like to thank my graduate students, who used earlier versions of the notes and provided both valuable feedback and continuous motivation.

I am deeply indebted to Prof. Heinrich Werner, Thomas Martinetz, Heinz-Otto Peitgen, Dagmar Schipanski, Maria Kallergi, Claudia Berman, Leonard Tung, Jim Zheng, Simon Foo, Bruce Harvey, Krishna Arora, Rodney Roberts, Uwe Meyer-Bäse, Helge Ritter, Henning Scheich, Werner Endres, Rolf Kammerer, DeWitt Sumners, Monica Hurdal, and Mrs. Duo Liu.

I am grateful to Prof. Andrew Laine from Columbia University who provided me with data, and support and inspired this book project. My thanks to Dr. Axel Wismüller from the University of Munich in Germany, who is the only "real" bioengineer I met who provided me with material and expertise and is one of the most helpful colleagues. I also wish to thank Dr. habil. Marek Ogiela from the AGH University of Science and Technology in Poland for proofreading the sections regarding syntactic pattern

recognition and for his valuable expert tips in applications of structural pattern recognition techniques in bioimaging.

The efforts of the professional staff at Elsevier Science/Academic Press, especially Michael Forster, Tiffany Gasbarrini. and Justin Palmeiro, deserve special thanks. Finally, watching my daughter Lisa-Marie laugh and play rewarded me for the many hours spent with the manuscript.

My thanks to all, many unmentioned, for their help.

Anke Meyer-Bäse

1

INTRODUCTION

Medical imaging deals with the interaction of all forms of radiation with tissue and the design of technical systems to extract clinically relevant information, which is then represented in image format. Medical images range from the simplest such as a chest X-ray to sophisticated images displaying temporal phenomena such as the functional magnetic resonance imaging (fMRI).

The past decades have witnessed a tremendous development of a new, powerful technology for detecting, storing, transmitting, analyzing, and displaying digital medical images. This technology is helping biochemists, biologists, medical scientists, and physicians to obtain quantitative measurements, which facilitate the validation of scientific hypothesis and accurate medical diagnosis.

This chapter gives an overview of microscopic and macroscopic image analysis and describes the basic model for diagnostic systems as a common platform for studying various problems of medical-imaging-based diagnostics.

1.1 MICROSCOPIC IMAGE ANALYSIS

The visualization and analysis of small particles are important tasks in many biomedical sciences, biology, and medicine. Cells are part of all living organisms, and the critical examination of living material at cell level represents an important tool for the analysis of biological and metabolic processes associated with pathological phenomena. The understanding and visualization of such formerly unrevealed processes were facilitated by the development of new imaging techniques.

The applications of digital image processing techniques in microscopic image analysis are vast: they range from enhancement and filtering to reconstruction and segmentation. The new techniques do not represent a substitute for the skills of a radiologist, but they can be considered as a useful tool for making a better diagnosis.

There are many types of microscopic imaging techniques, and each of them provides a different information representation or produces an unique information. Some of the best-known microscopic image analysis methods such as optical microscopy, electron microscopy. and morphometry are described in the following.

1.1.1 Optical Microscopy

The advantage of light microscopy over electron microscopy lies in the ability to visualize the interior of biological specimens without detrimental preparation. Light can be an ideal carrier of microscopic information if resolution is not limited by diffraction which is the reason why standard light microscopes are not able to distinguish objects closer than 200 nm in the focal plane. Perpendicular to the focal plane, the resolution is even poorer, namely about 600 nm. The immediate advantage of light microscopy motivated to search for methods that overcome the present resolution limits. Such approaches are confocal or multiphoton excitation scanning systems.

Biological material that has been stained with chemical compounds of a fluorescent nature can be visualized using a confocal fluorescence microscope. Confocal fluorescence microscopy is a modern version of light microscopy capable of imaging translucent biological specimens in three dimensions. The advantages of this method are: (1) detection at low light levels due to a higher signal-to-noise ratio, (2) visualization of fluorescently labeled ligands implanted into living cells and tissue, and (3) three-dimensional visualization of biological specimens.

1.1.2 Electron Microscopy

Electron microscopes produce images of biological material at a substantially higher resolution than the light microscope. For imaging, biological material must be specifically stained with salts of heavy metals such as uranium or osmanium. The produced images correspond to the electron

density of the heavy metal stains used during specimen preparation. The most frequent image processing techniques applied to these images pertain to morphometric measurement, image reconstruction, and visualization of crystalline structures.

The scanning electron microscope (SEM) is a microscope that uses electrons rather than light to form an image. It represents one of today's most frequently used devices for research. There are many advantages to using the SEM instead of a light microscope: (1) SEM has a large depth of field, which allows a large amount of the sample to be in focus at one time; (2) high-resolution images are produced such that closely spaced features can be examined at a high magnification; and (3) the sample has to be conductive, which means a short preparation time is required.

1.1.3 Autoradiography

Autoradiography is the self-recording of radiation by exposure of a radioactive source to a sensitive medium. In medical and biological research, tissue autoradiography is used to reveal the location of radiolabeled molecules and to monitor bioactive compounds in tissues, cells, and subcellular elements at set time periods. Therefore, autoradiography represents a histochemical technique that adds functional and time-related information to the structure. It is an essential technique for determining the imaging characteristics of a radioligand concentration.

Autoradiography is based on the principle of amplification, which means that during exposure the radiation from radiolabeled molecules in the tissues is recorded and held in an image. Weak signals, thus amplified, may become detectable after an exposure time of several days, weeks, or months, increasing the sensitivity of the method.

1.1.4 Morphometry

Morphometry quantifies structural features from a biological three-dimensional object, while stereology derives three-dimensional information from measurements on two-dimensional images. Many contour descriptors are part of automated image processing and are employed for the classification of biological objects.

Size and contour descriptors such as surface, radius, and perimeter are extremely useful for characterizing biological objects and were previously manually extracted from the image material.

Integrated morphometry analysis represents a powerful tool for image measurement and analysis of monochrome or color images. Scientists can perform morphometric measurements of objects in an image whenever classification of complex biological objects such as single cells is necessary and high measurement precision is important.

1.2 MACROSCOPIC IMAGE ANALYSIS

Macroscopic images, for example, images of human organs such as the heart, breast, lung, and brain, play an important role in diagnostic radiology, psychiatry, cardiology, and internal medicine.

During the past decade medical imaging technologies have diversified and progressed with amazing speed. Nowadays, clinical diagnostics rely mainly on the various medical imaging systems providing both two-dimensional and three-dimensional images.

There are several imaging modalities that can be applied to obtain more detailed information of an abnormality. For example, in diagnostic radiology four different media can be used. This is illustrated in Table 1.1 where US means ultrasound and MR means magnetic resonance.

TABLE 1.1 Range of application of the most important radiologic imaging modalities.

X-rays	Breast, lung, bone
γ-rays	Brain, organ parenchyma, heart function
MR	Soft tissue, disks, brain
US	Fetus, pathological changes, internal organs

The most frequently used medical imaging modalities are illustrated in Figure 1.1.

Figure 1.1a illustrates the concept of X-ray imaging with X-ray source, X-ray detector, and the subject under examination. The result, the X-ray image, provides a two-dimensional representation of the attenuation of X-rays in tissue.

The initial primary application of radionuclides in medicine was not in diagnosis but in radiotherapeutic applications. Combined with radiopharmaceuticals, radioactive isotopes can be used as tracers for radionuclide imaging and then detected by a gamma camera, which produces two-dimensional images of the radioactive tracer, as shown in Figure 1.1b. It is important to note that whereas the goal of X-ray imaging is to visualize structure, the objective of nuclear medicine imaging is to visualize physiological function or functional metabolism.

Clinical MRI systems visualize the distribution of water molecules in the tissue. Normally, the water molecules are randomly oriented in the tissue, but under exposure to a strong magnetic field they orient themselves in the direction of the magnetic field and spin at a specific angular frequency, known as the Larmour angular frequency. The fundamental concept of MRI is illustrated in Figure 1.1c. Supplementary energy can be introduced by a radiofrequency (RF) transmitter tuned to the resonance frequencies of the water molecules producing a short energy burst. This causes the molecules to align to the magnetic field. As they return to the

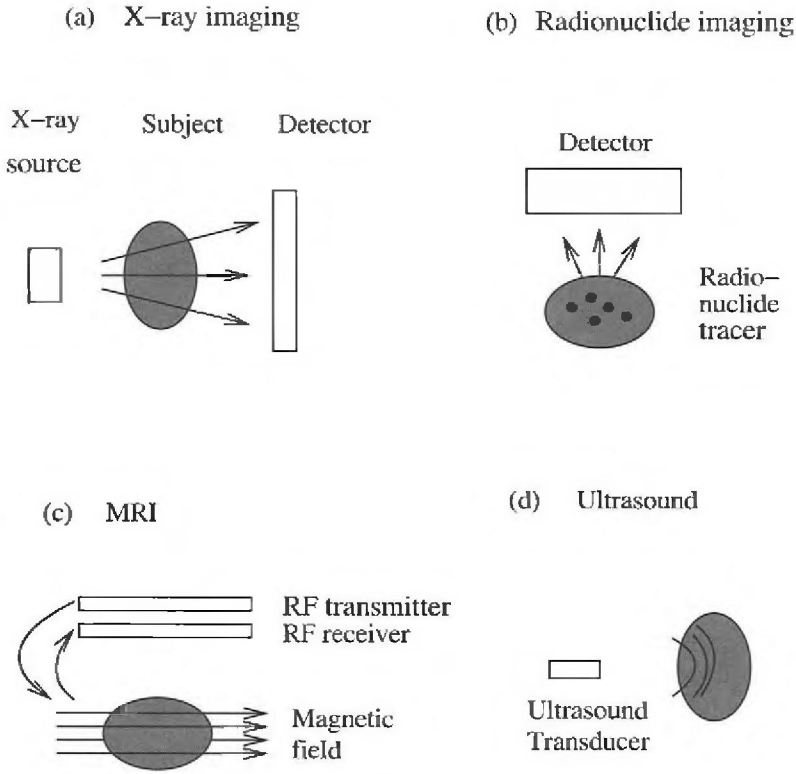


FIGURE 1.1 Schematic schemes of the most frequent used medical imaging modalities [137].

original orientation, they release energy in the form of an RF signal. The received RF signal's amplitude is proportional to the quantity of water molecules. Two time constants are relevant for MRI: T_1, the spin–lattice relaxation time, describing the rate at which the molecules return to their equilibrium, and T_2, the spin–spin relaxation time, showing the exponential decay time constant of the received RF signal. T_2 is usually broken down further into a pure T_2 due to the molecular interactions, and one due to the inhomogenities of the water molecules in their surrounding environment:

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_{2_{inhomogenities}}} \tag{1.1}$$

The overall T^* is referred to as T_2 star. The MR images can be reconstructed by determining the Fourier transform of the received RF signals.

The ultrasound principle is illustrated in Figure 1.1d where an ultrasound probe serves for both transmitting and receiving ultrasound waves.

Across an array of piezoelectric transducers, phase-shifted signals are both transmitted and received. The echoes of the ultrasound burst for each given angle are recorded over a time period and then transformed into image intensity along the scan line. A detailed explanation of the physics and the technical realization is given in the later chapters.

In this chapter, we introduce some basic knowledge of the most commonly used medical imaging systems in order to obtain a better understanding of the problems nature under investigation. Some of the best-known macroscopic image analysis methods such as imaging with ionizing radiation, magnetic resonance imaging, and ultrasound imaging are described in the following. A three-level model is developed to provide a common basis for studying the various problems of medical-imaging-based diagnostics. For further details on medical imaging, readers are referred to [39, 239, 148].

1.2.1 Model for Medical Image Processing

The analysis and interpretation of medical images represent two of the most responsible and complex tasks and usually consist of multiple processing steps. However, it is not difficult to generalize this procedure for all medical imaging modalities, and the resulting three-level processing model is shown in Figure 1.2. Image formation represents the bottom level of a diagnostic system. Some imaging modalities, such as conventional X-ray, do not rely on any computation, while others such as single-photon emission computed tomography employ image reconstruction as an image processing technique. Image processing is performed in two steps: a lower-level and a higher-level step. The former performs filtering, image enhancement and segmentation, feature extraction, and selection, directly on the raw pixels, while the latter uses the preprocessed pixel data and provides a medical diagnosis based on it. The most important tasks associated with this processing level are feature classification, tumor detection, and, in general, diagnosis for several diseases.

The basic image processing operations can be classified into five categories:

- Preprocessing: Preprocessing serves to better visualize object contours exhibiting a low resolution. The most common techniques include histogram transformation, filters, or Laplace-operators.
- Filtering: Filtering includes enhancement, deblurring, and edge detection. Enhancement techniques consist of linear or nonlinear, local or global filters or are wavelet-based. Deblurring techniques may consist of inverse or Wiener filters. Edge-detection techniques include the Haar transform, local operators, prediction, and/or classification methods.

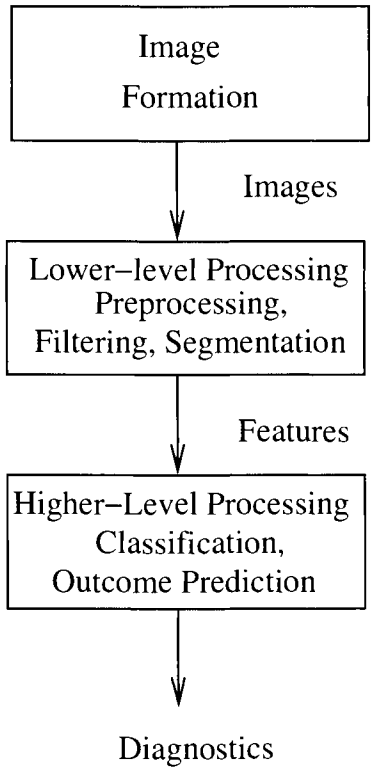


FIGURE 1.2 Model for diagnostic system using medical images [137].

- Segmentation: Segmentation can be both region-based and curve-based. There are several different kinds of segmentation algorithms including the classical region growers, clustering algorithms, and line and circular arc detectors. A critical issue in medical imaging is whether or not segmentation can be performed for many different domains using general bottom-up methods that do not use any special domain knowledge.
- Shape modeling: Shape modeling is performed based on features that can be used independently of, or in combination with, size measurements. For medical images, it is sometimes useful to describe the shape of an object in more detail than that offered by a single feature but more compactly than is reflected in the object image itself. A shape descriptor represents in such cases a more compact representation of an object's shape.
- Classification: Classification is based on feature selection, texture characterization, and a decision regarding the feature class. Each

abnormality or disease is recognized as belonging to a particular class, and the recognition is implemented as a classification process.

1.2.2 Computer-Aided Diagnosis Systems

Biomedical diagnosis systems can be implemented in both hardware and software. The most important steps for the realization of such a system are:

- Collection of data stemming from one or more biological sensors
- Preprocessing of the data: analog-to-digital conversion, localization of these data in time and space, normalization of data
- Extraction of classification-relevant features
- Selection of only relevant features from the whole feature set
- Concatenation of features in a logical form with the goal of describing properties
- Clustering of features and properties in groups based on similarities
- Interpretation of the obtained results based on importance of ideas and concepts

For the past several decades, researchers in the medical imaging field have focused on bringing new imaging modalities to clinicians while improving the performance of existing systems. Now, software designers are beginning to take the next step by introducing software improvements, enabling computers to assist doctors in interpreting the acquired noninvasive medical information. An important example of software to be evaluated using these image databases is referred to as computer-aided diagnosis (CAD). Computer-aided diagnosis describes a variety of artificial intelligence techniques applied to medical images. CAD systems work with physicians either by identifying features that might have been overlooked or by providing a second opinion for physicians, making their efforts more efficient.

For large-scale screening applications, CAD methods are important for:

- Prototyping robust systems working independently of patients
- Improving sensitivity in cancer detection
- Reducing observer variation in image interpretation
- Facilitating remote reading by experts (e.g., telemammography).

1.2.3 Imaging with Ionizing Radiation

X-ray is one of the most popular and, at the same time, oldest imaging techniques. It was discovered by W. C. Röntgen in 1895. X-rays are electromagnetic waves produced by the X-ray tube. The energy range for diagnostic X-rays lies between 10 and 150 KeV such that the human tissue appears semitransparent to this radiation. Thus, only within this range can adequate contrast in the X-ray image be achieved. A mapping of the three-dimensional absorption coefficient onto the two-dimensional image plane is thus obtained. Figure 1.3 displays a thorax X-ray.

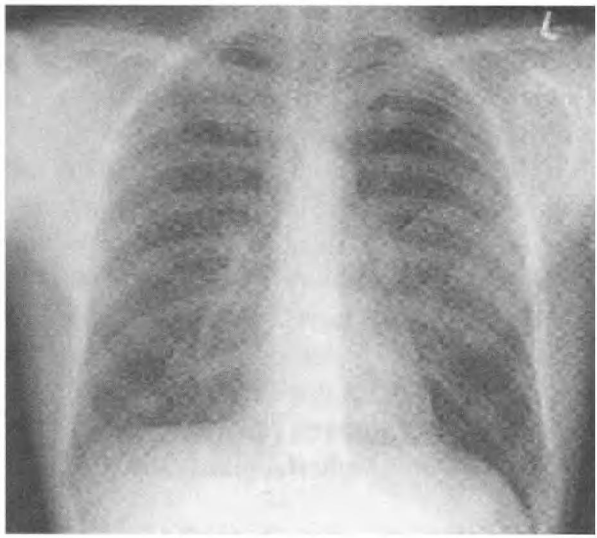


FIGURE 1.3 Thorax X-ray. (Courtesy of Publicis-MCD-Verlag.)

Another imaging modality is computer-assisted tomography (CAT), introduced by Hounsfield in 1972 with the idea of eliminating the major disadvantage posed by X-rays: the overlapping tissues create superimposed shadows that obscure critical diagnostic details. In CAT, X-ray projections are collected around the patient. The two-dimensional cross-sectional slices of the subject are reconstructed from the projection data based on the Radon transform [39]. By eliminating the interfering patterns stemming from over- and underlying bones and organs, CAT can provide a detailed contrast among the various soft tissues, far better than that achieved by traditional X-rays. The major advantages of CAT are: (1) quantitative imaging, (2) extracting more diagnostically relevant information by altering and enhancing an image after its realization, and (3) as a tomographic and potentially three-dimensional method allowing the analysis of isolated cross-sectional visual slices of the body. In the clinical environment, CAT is

routinely employed for detailed diagnosis of abdominal and pelvic organs, the lungs, and the brain. An example of a CAT image is shown in Figure 1.4.

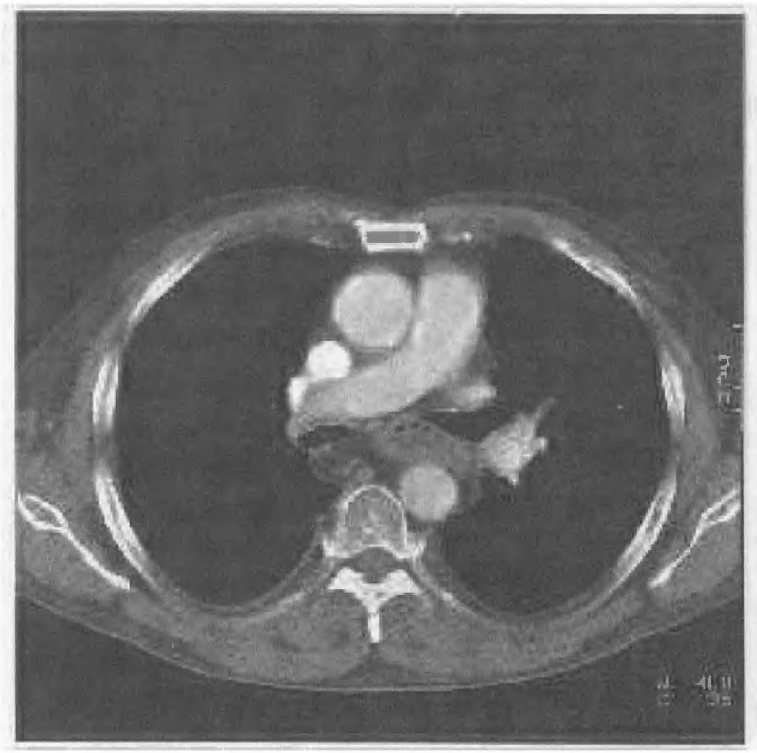


FIGURE 1.4 CAT of mediastinum and lungs. (Courtesy of Publicis-MCD-Verlag.)

Nuclear medicine began in the late 1930s when radioactive iodine was used to treat thyroid disease. Nuclear medicine is based on ionizing radiation, and image generation is similar to an X-ray's. However, in nuclear medicine the source of emission is inside the body. Nuclear medicine techniques, deriving from the physiologic localization of isotopically labeled, biologically active molecules to particular tissues, are intrinsically tissue-characterizing imaging techniques. This technique can be employed in tumor detection, imaging of abnormal body functions, and quantitative measurement of heart function. Based on the method of radiopharmaceutical disintegration, nuclear medicine is usually divided into two main areas: (1) single-photon emission and (2) positron emission. Single photon emission uses nuclei that disintegrate by emitting a single γ photon, which is measured with a gamma camera system. Single-photon emission

computed tomography (SPECT) is a slice-oriented technique, in the sense that the obtained data are tomographically reconstructed to produce two-dimensional slices. SPECT represents an important imaging technique to generate functional images of organs and to measure local blood flow and metabolism. Figure 1.5 shows an image of a brain SPECT study.

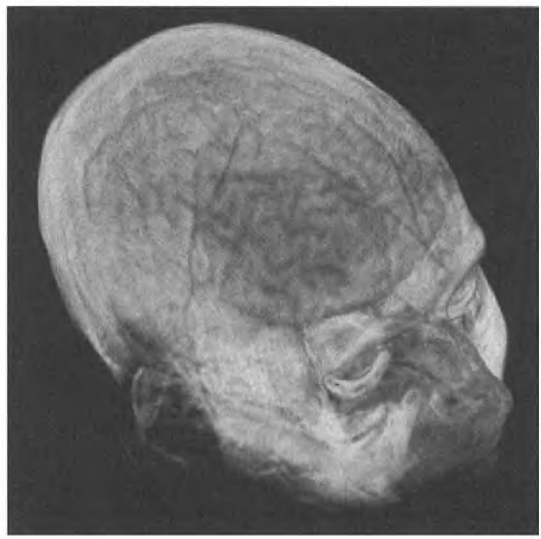


FIGURE 1.5 SPECT brain study. (Image courtesy Dr. A. Wismüller, Dept. of Radiology, University of Munich.)

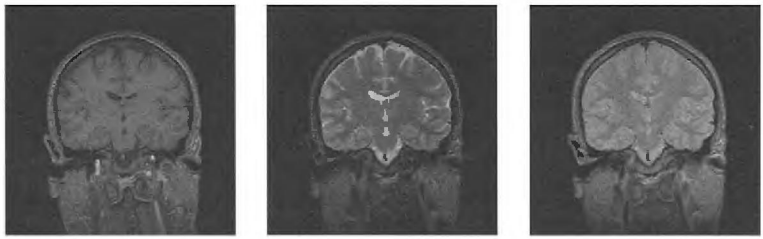
On the other hand, the radionuclides employed for positron emission tomography (PET) emit positrons instead of γ-rays. These positrons, antiparticles of electrons, are measured and their positions are computed. The reconstruction is produced by using algorithms of filtered back projection. PET has its main clinical application in brain tumor detection. Also, epilepsy, strokes and Alzheimer disease can be studied using PET.

1.2.4 Magnetic Resonance Imaging

The principle of magnetic resonance imaging (MRI) was discovered in the late 1940s, and during the past 20 years MRI has become an important diagnostic tool in many medical areas such as neurology, oncology, and gynecology. MRI provides structural details of the various organs like CAT, and additionally information on their physiological status and pathologies like nuclear medicine. The energy emission stemming from the spinning molecules is determined by two time constants, T_1 and T_2. MRI also produces a map of the hydrogen density throughout a volume of the patient. Simple T_1 and T_2 weighted images are very common in many medical ap-

plications. All three types of images are used clinically. Figure 1.6 shows a brain scan as T_1, T_2, and hydrogen density weighted images.

Summarizing, the advantages of MRI as an imaging tool are: (1) excellent contrasts between the various organs and tumors essential for image quality, (2) three-dimensional nature of the image, and (3) the contrast provided by the T_1 and T_2 relaxation mechanism as one of the most important imaging modalities.



(a) (b) (c)

FIGURE 1.6 Brain MRI showing (a) T_1, (b) T_2, and (c) hydrogen density weighted images. (Image courtesy Dr. A. Wismüller, Dept. of Radiology, University of Munich.)

A significant advance in MRI is the development of an MRI method for the noninvasive localization of human dynamic brain processes without using exogenous contrast agents [173], the so-called functional magnetic resonance imaging (fMRI). The most commonly used fMRI signal is the blood oxygenation level-dependent (BOLD) contrast which is derived from the fact that deoxyhemoglobin acts as an endogenous paramagnetic contrast agent. This means that changes in the local concentration of deoxyhemoglobin correspond to alterations in the MRI signal reflected in T_2^*- and T_2-weighted MR images. The evident correlation between neuronal activities and MR signal changes enables a variety of functional brain studies related to motor cortex, language, auditory cortex, memory, and attention.

In a typical fMRI study, T_2^*-weighted images of the brain are acquired while the subject is instructed to perform experimental and control tasks in an alternating sequence of 20–40 sec blocks. Changes in fMRI signal related to alternating performance of experimental and control tasks are identified and used to reveal the localized neural activation associated with the task. It should be mentioned that the detection of the fMRI response is not a trivial process, because the response is usually very small, and its exact form is generally unknown.

1.2.5 Ultrasound and Acoustic Imaging

Currently used clinical ultrasound presents an imaging technique based on construction of an image from backscattered echo strength. The interaction

of sound with tissue structures is very sophisticated and not well understood. It is speculated that the so-called "ultrasonic signatures" represent particular features emerging from the interaction of ultrasound energy with different tissues types. The goal is to use these characteristic features to make certain tissue diagnoses noninvasively.

Unlike imaging modalities based on ionizing radiation, ultrasound does not require electromagnetic radiation. Ultrasound produces oscillations of 1 to 10MHz traveling through soft tissues and fluids. When ultrasound passes through different tissues, echoes are produced at the boundaries that describe or separate them. Images are then produced which represent the acoustic reflectivity of the tissue. For example, two-dimensional echocardiography provides dynamic imaging of the cardiovascular system.

Ultrasound is used on an everyday basis in many areas of medicine such as pediatrics, cardiology, neurology, and gynecology. Often ultrasound provides critical clinical information quickly and inexpensively, such as distinguishing a fluid-filled cyst from a solid tumor in the abdomen or the breast. It can assist in diagnosing a wide range of diseases, such as pathological changes in the liver, gallbladder, pancreas, kidneys, heart, and blood vessels.

II

FEATURE SELECTION AND EXTRACTION

2.1 INTRODUCTION

Pattern recognition tasks require the conversion of pattern in features describing the collected sensor data in a compact form. Ideally, this should pertain only to relevant information. Feature selection methods can be either classical methods (statistic or of syntactic nature) or biologically oriented (neural or genetic algorithm based) methods.

Feature extraction and selection in pattern recognition are based on finding mathematical methods for reducing dimensionality of pattern representation. A lower-dimensional representation based on pattern descriptors is a so-called feature. It plays a crucial role in determining the separating properties of pattern classes. The choice of features, attributes, or measurements has an important influence on: (1) accuracy of classification, (2) time needed for classification, (3) number of examples needed for learning, and (4) cost of performing classification.

The cost of performing classification is very important in medical diagnosis where patterns are described by using both observable symptoms and results of diagnostic tests. Different diagnostic tests have different costs as

well as risks associated with them. For example, needle biopsy is more expensive than X-ray mammography.

Generally speaking, for each pattern recognition problem there are many solution strategies. Table 2.1 illustrates this aspect. Deciding on one specific solution is not always simple.

TABLE 2.1 Standard approaches in pattern recognition.

Measured features	Transformation of features	Structural features
Amplitude	Polynomials	Peaks
Bias	Harmonic analysis	Derivatives
Duration	Fourier transform	Lines
Phase	Wavelet transform	Edges
Energy	Haar transform	LPC coefficients
Moments	Karhunen-Loeve transform	Parametric models
Singular values		
Karhunen-Loeve eigenvalues		

Feature selection	Classifiers	Clustering methods
Discriminant analysis	Euclidian distance	Isodata algorithm
Chernoff bound	Mahalanobis distance	Fisher's linear discriminant
Bhattacharya	Linear discriminant functions	Parsing
Divergence	Bayesian linear classifier	
Exhaustive search	Maximum likelihood	
Dynamic programming	Production rules	
	Density functions	
	Parzen estimator	
	k-NN algorithm	
	Histogram	

This chapter gives an overview of the most relevant feature selection and extraction methods for biomedical image processing. Feature extraction methods encompass, besides the traditional transformed and nontransformed signal characteristics and texture, structural and graph descriptors. The feature selection methods described in this chapter are the exhaustive search, branch and bound algorithm, max-min feature selection, sequential forward and backward selection, and also Fisher's linear discriminant.

2.2 ROLE OF FEATURE SELECTION AND EXTRACTION

Dimensionality reduction plays an important role in classification performance. A recognition system is designed using a finite set of inputs. While the performance of this system increases if we add additional features, at some point a further inclusion leads to a performance degradation. Thus a dimensionality reduction may not always improve a classification system.

A model of the pattern recognition system including the feature selection and extraction stages is shown in Figure 2.1.

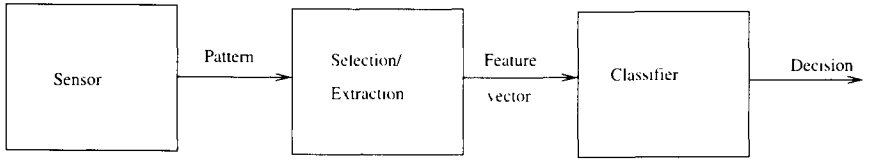


FIGURE 2.1 Pattern recognition system including feature selection and extraction.

The sensor data are subject to a feature extraction and selection process for determining the input vector for the subsequent classifier. This makes a decision regarding the class associated with this pattern vector.

Dimensionality reduction is accomplished based on either feature selection or feature extraction. Feature selection is based on omitting those features from the available measurements which do not contribute to class separability. In other words, redundant and irrelevant features are ignored. This is illustrated in Figure 2.2.

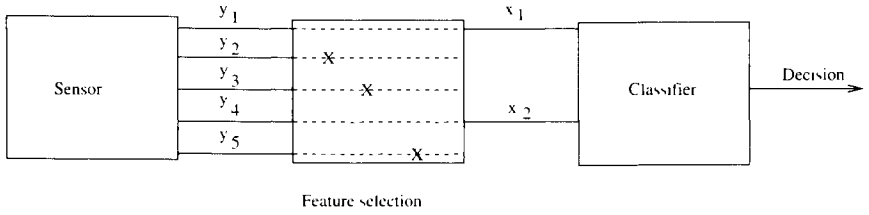


FIGURE 2.2 Dimensionality reduction based on feature selection.

Feature extraction, on the other hand, considers the whole information content and maps the useful information content into a lower dimensional feature space. This is shown in Figure 2.3. In feature extraction, the mapping type A has to be specified beforehand.

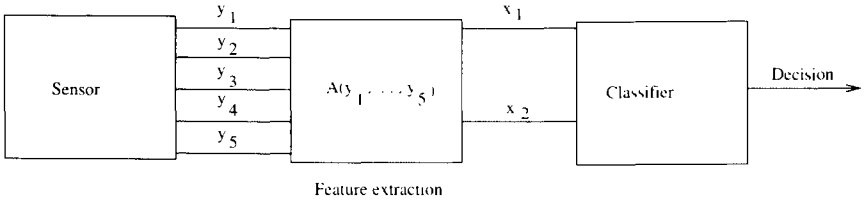


FIGURE 2.3 Dimensionality reduction based on feature extraction.

We see immediately that for feature selection or extraction the following is required: (1) feature evaluation criterion, (2) dimensionality of the feature space, and (3) optimization procedure.

2.3 PRELIMINARY NOTATIONS FOR FEATURE SELECTION AND EXTRACTION

We will introduce the necessary notations to explain feature selection and extraction methods.

The available sensor outputs are given either in the form of a vector component y_k, $k = 1, 2, \ldots, D$ or in the form of a vector $\mathbf{y}$ with $\mathbf{y} = [y_1, \ldots, y_D]^T$. Each pattern vector $\mathbf{y}$ is associated with one of the possible M classes ω_i, $i = 1, \ldots, M$. A set of candidate features $\zeta_j, j = 1, \ldots, d$, is described by χ while $X = \{\mathbf{x}_j | j = 1, \ldots, d\}$ describes the optimized set based on some criterion function J.

For feature selection optimization of all possible candidate feature sets has to be considered:

$$J(X) = \overset{max}{\chi} J(\chi) \tag{2.1}$$

whereas for feature extraction the optimization concerns only all admissible mappings

$$J(A) = \overset{max}{A} J(A(\mathbf{y})) \tag{2.2}$$

where A represents an optimal feature extractor. Based on this specification, we can determine the feature vector $\mathbf{x} = [x_1, \ldots, x_d]^T$ based on the transformation $\mathbf{x} = A(\mathbf{y})$.

2.4 FEATURE EXTRACTION METHODS

A good feature should remain unchanged if variations take place within a class, and it should reveal important differences when discriminating between patterns of different classes. In other words, patterns are described with as little loss as possible of pertinent information.

There are four known categories in the literature for extracting features [43]:

1. Nontransformed structural characteristics: moments, power, amplitude information, energy, etc.
2. Transformed structural characteristics: frequency and amplitude spectra, subspace transformation methods, etc.
3. Structural descriptions: formal languages and their grammars, parsing techniques, and string matching techniques
4. Graph descriptors: attributed graphs, relational graphs, and semantic networks

One of the most frequent problems when applying any classification technique to a pattern recognition problem is the so-called curse of dimensionality [59]. Methods that are adequate for a low-dimensional feature space might be completely impracticable for a high-dimensional space

(number of features > 50). This shows that techniques for dimensionality reduction in the feature space had to be developed in order to obtain a more manageable problem.

We will tackle the problem of feature extraction applied to medical image analysis. Our goal will be for a given image, or a region within an image, to generate the features that will be the input to a classifier, which has to assign this image or region of an image to one of the possible classes.

A digital monochrome image is obtained by sampling the corresponding continuous image function $I(x, y)$ and storing the discretized values for x and y in form of a two-dimensional $N_x \times N_y$ image matrix $I(m, n)$ with $m = 0, 1, \ldots, N_x - 1$ and $n = 0, 1, \ldots, N_y - 1$. Every (m, n) element is called a pixel whose gray value is determined by $I(m, n)$. There are N_g quantization levels leading to 2^{N_g} distinct gray levels. To achieve a smooth perception of the image, we need a resolution of $N_g = 8$ bits per pixel.

A small 128×128 image has 16,384 pixels, and it is evident that we cannot use this raw information for classification purposes. Therefore, we have to generate new features from the available image matrix $I(m, n)$, and those features should extract precisely the relevant information contained in the original image.

This section will review the most important feature extraction methods in biomedical image analysis.

2.4.1 Nontransformed Signal Characteristics

To describe the properties of random variables and processes, we need statistical parameters that are obtained from these random variables and processes. Although to describe a Gaussian process second-order statistics (mean value and variance) is perfectly sufficient. signal separation problems usually need higher order statistics [186].

2.4.1.1 Moments Moments represent extracted features that are derived from raw measurements. For two-dimensional signals, such as images, they can be used to achieve rotation, scale, and translation invariant.

By $I(x, y)$ we define a continuous image function. The geometric moment of order $p + q$ is given by

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q I(x, y) dx dy \tag{2.3}$$

The geometric moment provides important information for the purpose of image reconstruction [186]. In other words, every single coefficient contributes to providing image information.

Invariance in geometric transformation is required for many classification tasks [229]. Moments defined by eq. (2.3) do not provide the desired feature invariance. However, a remedy can be found by defining moments that are invariant to

- *Translation:*

$$x' = x + a, \quad y' = y + b \tag{2.4}$$

- *Scaling:*

$$x' = \alpha x, \quad y' = \alpha y \tag{2.5}$$

- *Rotations:*

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.6}$$

We also can show that central moments are invariant to translations

$$\mu_{pq} = \int\int I(x,y)(x-\bar{x})^p(y-\bar{y})^q dxdy, \tag{2.7}$$

where

$$\bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}} \tag{2.8}$$

On the other hand, normalized central moments η_{pq} are both scaling and translation invariant [229].

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \gamma = \frac{p+q+2}{2} \tag{2.9}$$

The preceding equations represent the definitions regarding continuous image functions. Similarly, we can define for a digital image $I(x,y)$ with $i = 0, 1, \ldots, N_x - 1, j = 0, 1, \ldots, N_y - 1$ the corresponding moments by just replacing integrals by summations

$$m_{pq} = \sum_i \sum_j I(x,y) i^p j^q \tag{2.10}$$

2.4.1.2 Parametric Modeling Parametric modeling is based on representing a signal as a weighted combination of previous samples. For a detailed introduction, see [221]. By applying this method to digital images, we can obtain a useful feature set to be used in conjunction with a subsequent classifier. For this, we will assume that $I(m,n)$ is a real nondiscrete random variable.

There are two ways to proceed. One thinks of an image as a successive sequence of rows or columns and assumes it is a one-dimensional random process $I(n)$, while the other considers a two-dimensional random process $I(m,n)$ or a so-called random field.

The basic idea is that a random sequence can be generated at the output of a linear, causal, stable, time-invariant system with impulse response

$h(n)$, whose input is a white noise sequence [186]. Let $I(n)$ be a random stationary sequence with the autocorrelation $R(k)$:

$$R(k) = E[I(n)I(n-k)] \tag{2.11}$$

With $\mu(n)$ representing a white noise sequence, we obtain equivalently for $I(n)$

$$I(n) = \sum_{k=0}^{\infty} h(k)\mu(n-k) \tag{2.12}$$

Such a process is referred to as an autoregressive (AR) process, and is generated recursively by

$$I(n) = \sum_{k=1}^{p} a(k)I(n-k) + \mu(n) \tag{2.13}$$

We immediately see that the random sequence $I(n)$ represents a linear combination of past values of $I(n-k)$ and an additive constant $\mu(n)$. p is the order of the AR model. The coefficients $a(k)$ with $k = 1, 2, \ldots, p$ are the parameters of the AR model and at the same time the predictor parameters of the sequence $I(n)$. In other words, they represent the weighting terms of the past sampled values $I(n-1), \ldots, I(n-p)$ and serve as a prediction of the actual value $I(n)$

$$\hat{I}(n) = \sum_{k=1}^{p} a(k)I(n-k) \equiv \mathbf{a}^T\mathbf{I}(n-1) \tag{2.14}$$

with $\mathbf{I}^T(n-1) \equiv [I(n-1), \ldots, I(n-p)]$ and $\mu(n)$ the prediction error. The unknown parameter vector $\mathbf{a}^T = [a(1), a(2), \ldots, a(p)]$ can optimally be estimated by minimizing the mean square prediction error:

$$E[\mu^2(n)] = E[(I(n) - \hat{I}(n))^2] = E[(I(n) - \mathbf{a}^T\mathbf{I}(n-1))^2] \tag{2.15}$$

The unknown parameters can be determined from

$$E[\mathbf{I}(n-1)\mathbf{I}^T(n-1)]\mathbf{a} = E[I(n)\mathbf{I}(n-1)] \tag{2.16}$$

This leads to the following equation system

$$\begin{bmatrix} R(0) & R(-1) & \cdots & R(-p+1) \\ R(1) & R(0) & \cdots & R(-p+2) \\ \vdots & \vdots & \vdots & \vdots \\ R(p-2) & R(p-3) & \cdots & R(-1) \\ R(p-1) & R(p-2) & \cdots & R(0) \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \\ \vdots \\ a(p-1) \\ a(p) \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(p-1) \\ r(p) \end{bmatrix} \tag{2.17}$$

or equivalently in matrix notation to

$$\tilde{\mathbf{R}}\mathbf{a} = \mathbf{R} \tag{2.18}$$

with $\mathbf{r} \equiv [r(1), \ldots, r(p)]^T$. The variance of the prediction error σ_μ^2 is determined based on eqs. (2.15) and (2.18):

$$\sigma_\mu^2 = E[\mu^2(n)] = R(0) - \sum_{k=1}^{p} a(k)R(k) \tag{2.19}$$

The resulting autocorrelation matrix is Töplitz. This property is very desirable because it provides, based on the Levinson-Durbin-algorithm [186], a solution of the linear system described in eq. (2.18).

The AR-parameters can also serve as features to distinguish between different classes.

EXAMPLE 2.4.1: Let the AR random sequence of order $p = 2$ be

$$I(n) = \sum_{k=1}^{2} a(k)I(n-k) + \mu(k) \tag{2.20}$$

with $R(0) = 1, R(1) = 0.3, and R(2) = 0.55$. The AR parameters can be computed from eq. (2.18):

$$\begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \end{bmatrix} = \begin{bmatrix} 0.3 \\ 0.55 \end{bmatrix} \tag{2.21}$$

We obtain $a(1) = 0.15$ and $a(2) = 0.5$.

As a generalization we also give the two-dimensional AR random sequence $I(m,n)$

$$\hat{I}(m,n) = \sum_k \sum_l a(k,l)I(m-k, n-l), \quad (k,l) \in W \tag{2.22}$$

and

$$I(m,n) = \hat{I}(m,n) + \mu(m,n) \tag{2.23}$$

Figure 2.4 illustrates the pixel domain W, which leads to the prediction of $I(m,n)$. The figure illustrates four different two-dimensional predictor models [229]. Figure 2.4a illustrates a strict causal predictor model. The coordinates of every single pixel in the contributing frame are smaller than the coordinates m, n of the predicted pixel represented by an unshaded circle. The corresponding frame is given by $W_1 = \{0 \leq k \leq p, 0 \leq l \leq q, (k,l) \neq (0,0)\}$. Besides causal predictors, there are also noncausal predictors defined by

$$I(m,n) = \sum_{k=-p}^{p} \sum_{l=-q}^{q} a(k,l) I(m-k, n-l) + \mu(m,n) \tag{2.24}$$

Figure 2.4d illustrates the corresponding frame for $p = q = 2$. Figure 2.4c shows a semicausal predictor, while Figure 2.4b shows a causal predictor. In most applications, strict causality is not of primary importance. Thus, the other three relevant cases are summarized in Table 2.2.

TABLE 2.2 Domain W for three application relevant predictors.

Causal:	W_2	=	$\{(-p \le k \le p, 1 \le l \le q) \cup (1 \le k \le p, l = 0)\}$
Semicausal:	W_3	=	$\{-p \le k \le p, 0 \le l \le q, (k,l) \ne (0,0)\}$
Noncausal:	W_4	=	$\{-p \le k \le p, -q \le l \le q, (k,l) \ne (0,0)\}$

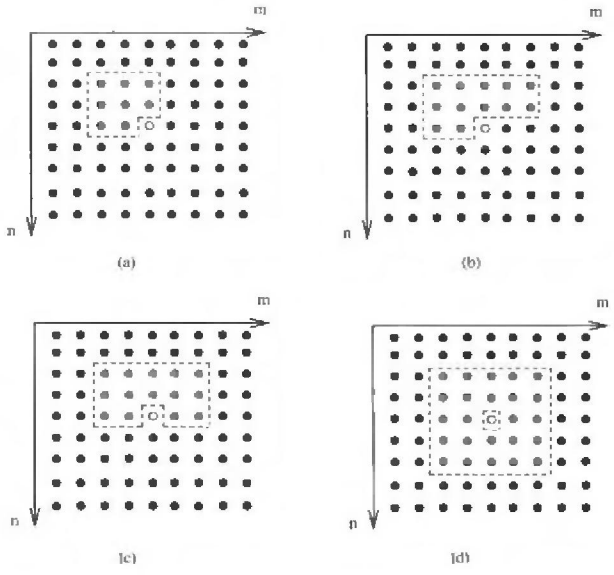


FIGURE 2.4 Different types of 2-D predictor models. The dotted polygon corresponds to the region W. (a) Strict causal, (b) causal, (c) semicausal, (d) noncausal.

In the 2-D case, the predictor parameters can be determined similarly to the 1-D case:

$$\hat{I}(m,n) = \sum_{k} \sum_{l} a(k.l) I(m-k, n-l) \tag{2.25}$$

From the orthogonality condition between error and signal, we obtain

$$R(i,j) = \sum_k \sum_l a(k,l)R(i-k, j-l), \quad (i,j) \in W \tag{2.26}$$

$R(i,j)$ is a 2-D autocorrelation of the image array $I(m,n)$. The system of equations shown in (2.26) is linear, and its solution provides an estimation of $a(k,l)$. By choosing a causal window with $p = q$, we obtain from the symmetry property of the window that for every single index pair (i,j) there also exists an associated index pair $(-i,-j)$. The variance of the prediction error is determined by

$$\sigma_\mu^2 = R(0,0) - \sum_k \sum_l a(k,l)R(k,l) \tag{2.27}$$

It is important to note that the correlation $R(k,l)$ of a homogeneous and direction invariant (isotropic) image depends only on the relative distance between pixels:

$$R(k,l) = R(\sqrt{k^2 + l^2}) \tag{2.28}$$

This means that the resulting autocorrelation matrix is symmetric and also Töplitz,

$$\tilde{\mathbf{R}} = \begin{bmatrix} \tilde{\mathbf{R}}_0 & \tilde{\mathbf{R}}_1 & \cdots & \tilde{\mathbf{R}}_{2p} \\ \tilde{\mathbf{R}}_1 & \tilde{\mathbf{R}}_0 & \cdots & \tilde{\mathbf{R}}_{2p-1} \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{\mathbf{R}}_{2p} & \tilde{\mathbf{R}}_{2p-1} & \cdots & \tilde{\mathbf{R}}_0 \end{bmatrix} \tag{2.29}$$

while each matrix element again represents a matrix which is also Töplitz:

$$\tilde{\mathbf{R}}_i = \begin{bmatrix} R(i,0) & \cdots & R(i,2p) \\ \vdots & \vdots & \vdots \\ R(i,2p) & \cdots & R(i,0) \end{bmatrix} \tag{2.30}$$

By choosing a symmetric frame W, we get symmetric AR parameters, $a(k,l) = a(-k,-l)$, and the system can be solved based on the Levinson-Durbin algorithm.

The AR parameters represent useful features used for many image processing classification tasks [106, 153].

EXAMPLE 2.4.2: Let $R(k,l)$ be the autocorrelation sequence for a medical image

$$R(k,l) = 0.4^{\sqrt{k^2+l^2}}$$

Estimate the AR parameters for a noncausal $p = q = 1$ window.

From eq. (2.22) we obtain

$$\begin{aligned}\hat{I}(m,n) &= a(1,1)I(m-1,n-1) + a(1,0)I(m-1,n) \\ &+ a(1,-1)I(m-1,n+1) + a(0,1)I(m,n-1) \\ &+ a(0,-1)I(m,n+1) + a(-1,1)I(m+1,n-1) \\ &+ a(-1,0)I(m+1,n) + a(-1,1)I(m+1,n+1)\end{aligned}$$

The resulting matrix $\tilde{\mathbf{R}}$ is a $(2p+1) \times (2p+1)$ matrix, whose elements are 3×3 matrices

$$\tilde{\mathbf{R}} = \begin{bmatrix} \tilde{\mathbf{R}}_0 & \tilde{\mathbf{R}}_1 & \tilde{\mathbf{R}}_2 \\ \tilde{\mathbf{R}}_1 & \tilde{\mathbf{R}}_0 & \tilde{\mathbf{R}}_1 \\ \tilde{\mathbf{R}}_2 & \tilde{\mathbf{R}}_1 & \tilde{\mathbf{R}}_0 \end{bmatrix}$$

with

$$\tilde{\mathbf{R}}_0 = \begin{bmatrix} R(0,0) & R(0,1) & R(0,2) \\ R(0,1) & R(0,0) & R(0,1) \\ R(0,2) & R(0,1) & R(0,0) \end{bmatrix}$$

$$\tilde{\mathbf{R}}_1 = \begin{bmatrix} R(1,0) & R(1,1) & R(1,2) \\ R(1,1) & R(1,0) & R(1,1) \\ R(1,2) & R(1,1) & R(1,0) \end{bmatrix}$$

and

$$\tilde{\mathbf{R}}_2 = \begin{bmatrix} R(2,0) & R(2,1) & R(2,2) \\ R(2,1) & R(2,0) & R(2,1) \\ R(2,2) & R(2,1) & R(2,0) \end{bmatrix}$$

We obtain the following AR parameters: $a(1,1) = a(-1,-1) = -0.0055$, $a(1,0) = a(-1,0) = -0.125$, $a(1,-1) = a(-1,1) = -0.0055$, $a(0,1) = a(0,-1) = -0,125$, and $\sigma_\mu^2 = 0.585$.

2.4.2 Transformed Signal Characteristics

The basic idea employed in transformed signal characteristics is to find such transform-based features with a high information density of the original input and a low redundancy. To understand this aspect better, let us consider a mammographic image. The pixels (input samples) at the various positions have a large degree of correlation. By using only the pixels, irrelevant information for the subsequent classification is also considered. By

using, for example, the Fourier transform we obtain a feature set based on the Fourier coefficients which retains only the important image information residing in low-frequency coefficients. These coefficients preserve the high correlation between the pixels.

There are several methods for obtaining transformed signal characteristics. For example, Karhunen-Loeve transform and singular value decomposition are problem-dependent and the result of an optimization process [59, 229]. They are optimal in terms of decorrelation and information concentration properties, but at the same time too computationally expensive. On the other hand, transforms which use fixed basis vectors (images) such as the Fourier and wavelet transform, exhibit low computational complexity while being suboptimal in terms of decorrelation and redundancy.

In this section, we will review the most important methods for obtaining transformed signal characteristics, such as principal component analysis, the discrete Fourier transform, and the discrete cosine and sine transform.

2.4.2.1 Principal Component Analysis (PCA) Principal component analysis is a basic technique used for data reduction in pattern recognition, signal processing, and bioengineering. It has been widely used in data analysis [54, 225, 177, 178, 156, 101] and compression [234, 97, 35]. PCA is also referred to as Karhunen-Loeve transformation or the Hotelling transform.

The idea is that similar input patterns belong to the same class. Thus, the input data can be normalized within the unit interval and then chosen based on their variances. In this sense, the larger the variances, the better discriminatory properties the input features have.

Sometimes, combining two features provides a better recognition result than either one alone.

PCA involves a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components. This is a highly desirable property, since besides being optimally uncorrelated, the redundancy in data information is removed.

If the data are concentrated in a linear subspace, this provides a way to compress them without losing much information and simplifying the representation. By selecting the eigenvectors having the largest eigenvalues we lose as little information as possible in the mean-square sense. A fixed number of eigenvectors and their respective eigenvalues can be chosen to obtain a consistent representation of the data.

Let $\mathbf{x} = [x_1, \ldots, x_m]^T$ be a random vector generated by a stationary stochastic process. The correlation matrix $\mathbf{R}$ of this vector is defined as

$$\mathbf{R} = E[\mathbf{x}\mathbf{x}^T] \tag{2.31}$$

Also, let $\mathbf{q}_i$ be the ith eigenvector and λ_i the corresponding ith eigenvalue of the matrix $\mathbf{R}$. The eigenvalues λ_i are also known as singular values. Let $\mathbf{Q}$ be an $m \times m$ eigenvector matrix

$$\mathbf{Q} = [\mathbf{q}_1, \cdots, \mathbf{q}_m] \tag{2.32}$$

and $\mathbf{\Lambda}$ be the $m \times m$ diagonal matrix given by the eigenvalues of matrix $\mathbf{R}$

$$\mathbf{\Lambda} = \text{diag}[\lambda_1, \cdots, \lambda_m] \tag{2.33}$$

Without loss of generality, we may assume that the eigenvalues are arranged in decreasing order:

$$\lambda_1 > \lambda_2 \cdots \lambda_m \tag{2.34}$$

such that the first eigenvalue represents the maximal eigenvalue, $\lambda_1 = \lambda_{max}$.

We immediately see that matrix $\mathbf{Q}$ is an orthogonal matrix because its column vectors satisfy the conditions of orthonormality:

$$\mathbf{q}_i^T \mathbf{q}_i = \begin{cases} 1, & j = i \\ 0, & \text{otherwise} \end{cases} \tag{2.35}$$

In matrix representation, this becomes

$$\mathbf{Q}^T \mathbf{Q} = \mathbf{I} \tag{2.36}$$

Thus, the correlation matrix $\mathbf{R}$ can be rewritten in terms of the eigenvalues as

$$\mathbf{R} = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^T \tag{2.37}$$

This is known as the spectral theorem. By selecting the eigenvectors $\mathbf{q}_i$ as new basis vectors of the data space, we can rewrite the original data vector $\mathbf{x}$ in the following form:

$$\mathbf{x} = \sum_{i=1}^{m} \mathbf{q}_i a_i = \mathbf{Q}\mathbf{a} \tag{2.38}$$

where $\mathbf{a} = [a_1, \ldots, a_m]^T$ is a coefficient vector. The coefficients a_i represent the projections of $\mathbf{x}$ onto the principal directions represented by the vectors $\mathbf{q}_i$. In other words, they are produced by the linear transformation from the vector space of the original data vector $\mathbf{x}$. They are determined based on the inner product of the transposed eigenvector matrix $\mathbf{Q}$ with the data vector $\mathbf{x}$

$$\mathbf{a} = \mathbf{Q}^T \mathbf{x} \tag{2.39}$$

or

$$a_j = \mathbf{q}_j^T \mathbf{x} = \mathbf{x}^T \mathbf{q}_j, \quad j = 1, 2, \ldots, m \tag{2.40}$$

The a_i represent the principal components and have the same physical dimensions as the original data vector $\mathbf{x}$.

The importance of PCA lies in the dimensionality reduction. The number of features of the input vector can be reduced by eliminating those linear combinations in eq. (2.38) that have small variances and keeping those with large variances.

Let us assume that the eigenvalues $\lambda_1, \ldots, \lambda_l$ have larger variances than the others $\lambda_{l+1}, \ldots, \lambda_m$. By omitting the latter we obtain an approximation $\hat{\mathbf{x}}$ of vector $\mathbf{x}$ as

$$\hat{\mathbf{x}} = \sum_{i=1}^{l} \mathbf{q}_i a_i = \sum_{i=1}^{l} \mathbf{q}_i \mathbf{q}_i^T \mathbf{x} = \tilde{\mathbf{Q}} \tilde{\mathbf{Q}}^T \mathbf{x} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_l] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix}, \quad l \neq m \tag{2.41}$$

where $\tilde{\mathbf{Q}} = [\mathbf{q}_1, \cdots, \mathbf{q}_l]$. Equation (2.41) is the orthogonal projection of vector $\mathbf{x}$ onto the subspace spanned by the principal components with the l largest eigenvalues. Based on eq. (2.40), we are able to determine the set of the principal components with the l largest eigenvalues:

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_l^T \end{bmatrix} \mathbf{x}, \quad l \leq m \tag{2.42}$$

These l largest eigenvalues correspond to the most relevant features of the input vector $\mathbf{x}$. It is evident that the elimination of features yields an approximation error $\mathbf{e}$ that is given by the difference between the original data vector $\mathbf{x}$ and the new reduced-order data vector $\tilde{\mathbf{x}}$. The approximation error is given by

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}} = \sum_{i=l+1}^{m} a_i \mathbf{q}_i \tag{2.43}$$

The total variance of the considered l components of the vector $\mathbf{x}$ is

$$\sum_{i=1}^{l} \sigma_i^2 = \sum_{i=1}^{l} \lambda_i \tag{2.44}$$

while the total variance of the remaining $(l - m)$ components is

$$\sum_{i=l+1}^{m} \sigma_i^2 = \sum_{i=l+1}^{m} \lambda_i \tag{2.45}$$

The smallest eigenvalues $\lambda_{l+1}, \ldots, \lambda_m$ of the correlation matrix $\mathbf{R}$ are thus discarded from eq. (2.38). The most effective dimensionality reduction is achieved if the discarded eigenvalues are close to zero, and represent only redundant information. Thus, the strategy for dimensionality reduction based on PCA is to determine the eigenvalues and eigenvectors of the correlation matrix of the input data vector, and then project the data orthogonally onto the subspace that belongs to the largest eigenvalues. This technique is known as subspace decomposition [178].

By analyzing eqs. (2.42) and (2.41), we observe an interesting aspect: eq. (2.42) describes a mapping from the data space $\mathcal{R}^m$ to a reduced-order feature space $\mathcal{R}^l$ and represents an encoder for the approximate representation of the data vector $\mathbf{x}$. Equation (2.41) describes the inverse transformation as a projection from the reduced-order feature space $\mathcal{R}^l$ to the original data space $\mathcal{R}^m$ and represents a decoder for the reconstruction of the data vector $\mathbf{x}$. Figure 2.5 illustrates this.

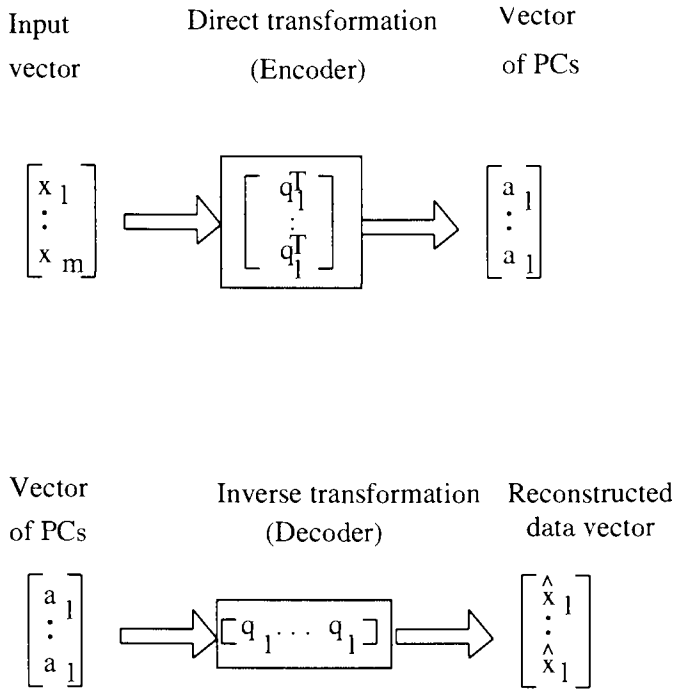


FIGURE 2.5 Coding and decoding mechanism of PCA.

It is important to point out the major difference between PCA and a related transformation, singular value decomposition (SVD). SVD is related only to a single set of samples, whereas PCA is related to an ensemble of them. PCA is playing a major role as a preprocessing technique in pattern recognition and also in context with other transformations such as independent component analysis.

There are also some failures of PCA when used for feature extraction. Figure 2.6 illustrates the case when the first principal component does not contribute toward class separability but only the second principal component does.

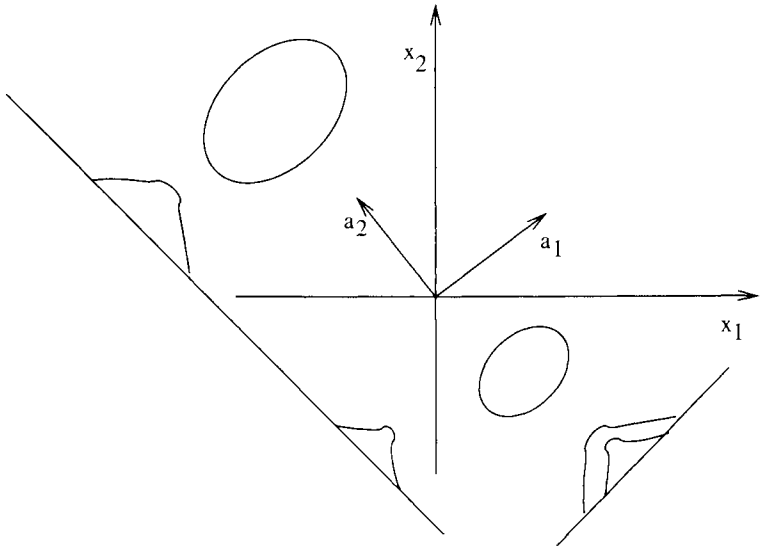


FIGURE 2.6 PCA is not always associated with class separability. Different contributions of the first two principal components to class separability are shown.

2.4.2.2 Discrete Fourier Transform In the following we will review the discrete Fourier transform (DFT) for both one- and two-dimensional cases.

Let us consider N sampled values $x(0), \ldots, x(N-1)$. Their DFT is given by

$$y(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) \exp\left(-j \frac{2\pi}{N} kn\right), \quad k = 0, 1, \ldots, N-1 \tag{2.46}$$

while the corresponding inverse transform is

$$x(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} y(k) \exp\left(j \frac{2\pi}{N} kn\right), \quad n = 0, 1, \ldots, N-1 \tag{2.47}$$

with $j \equiv \sqrt{-1}$. All $x(n)$ and $y(k)$ can be concatenated in form of two $N \times 1$ vectors. Let us also define

$$W_N \equiv \exp\left(-j \frac{2\pi}{N}\right) \tag{2.48}$$

such that eqs. (2.46) and (2.47) can be written in matrix form

$$\mathbf{y} = \tilde{\mathbf{W}}\mathbf{x}, \quad \mathbf{x} = \mathbf{W}\mathbf{y} \tag{2.49}$$

with

$$\tilde{\mathbf{W}} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \tag{2.50}$$

where $\mathbf{W}$ is an unitary and symmetric matrix

$$\mathbf{W}^{-1} = \tilde{\mathbf{W}} = \mathbf{W}^*$$

Let us choose as an example the case $N = 2$.

EXAMPLE 2.4.3: We then obtain for $N = 2$

$$\mathbf{W} = \frac{\sqrt{2}}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

We see that the columns of $\mathbf{W}$ correspond to the basis vectors

$$\mathbf{w_0} = \frac{1}{\sqrt{2}}[1, 1]^T$$

$$\mathbf{w_1} = \frac{1}{\sqrt{2}}[1, -1]^T$$

and based on them we can reconstruct the original signal:

$$\mathbf{x} = \sum_{i=0}^{1} y(i)\mathbf{w_i}$$

The two-dimensional DFT for an $N \times N$-image is defined as

$$Y(k,l) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} X(m,n) W_N^{km} W_N^{ln} \tag{2.51}$$

while its inverse DFT is given by

$$X(m,n) = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} Y(k,l) W_N^{-km} W_N^{-ln} \tag{2.52}$$

The corresponding matrix representation yields

$$\mathbf{Y} = \tilde{\mathbf{W}}\mathbf{X}\tilde{\mathbf{W}}, \quad \mathbf{X} = \mathbf{W}\mathbf{Y}\mathbf{W} \tag{2.53}$$

We immediately see that the two-dimensional DFT represents a separable transformation with the basis images $\mathbf{w_i}\mathbf{w_j^T}, i, j = 0, 1, \ldots, N-1$.

2.4.2.3 Discrete Cosine and Sine Transform Another very useful transformation is the discrete cosine transform (DCT) which plays an important role in image compression. In the following, we will review the DCT for both the one- and two-dimensional cases.

For N given input samples the DCT is defined as

$$y(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad k = 0, 1, \ldots, N-1 \tag{2.54}$$

Its inverse transform is given by

$$x(n) = \sum_{k=0}^{N-1} \alpha(k) y(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad n = 0, 1, \ldots, N-1 \tag{2.55}$$

with

$$\alpha(0) = \sqrt{\frac{1}{N}}, \quad k = 0 \quad \text{and} \quad \alpha(k) = \sqrt{\frac{2}{N}}, \quad 1 \leq k \leq N-1 \tag{2.56}$$

The vector form of the DCT is given by

$$\mathbf{y} = \mathbf{C}^{\mathrm{T}}\mathbf{x} \tag{2.57}$$

while we have for the elements of the matrix $\mathbf{C}$

$$C(n,k) = \sqrt{\frac{1}{N}}, \quad k = 0, \quad 0 \leq n \leq N-1$$

and

$$\begin{aligned} C(n,k) &= \sqrt{\frac{2}{N}} \cos\left(\frac{\pi(2n+1)k}{2N}\right), \\ & 1 \leq k \leq N-1, \quad 0 \leq n \leq N-1 \end{aligned}$$

$\mathbf{C}$ represents an orthogonal matrix with real numbers as elements:

$$\mathbf{C}^{-1} = \mathbf{C}^{\mathrm{T}}$$

In the two-dimensional case the DCT becomes

$$\mathbf{Y} = \mathbf{C}^{\mathrm{T}}\mathbf{X}\mathbf{C}, \quad \mathbf{X} = \mathbf{C}\mathbf{Y}\mathbf{C}^{\mathrm{T}} \tag{2.58}$$

Unlike the DFT, the DCT is real valued. It has found application in image compression.

Another orthogonal transform is the discrete sine transform (DST) defined as

$$S(k,n) = \sqrt{\frac{2}{N+1}} \sin\left(\frac{\pi(n+1)(k+1)}{N+1}\right), \quad k.n = 0,1,\ldots,N-1 \tag{2.59}$$

Both DCT and DST have excellent information concentration properties since they concentrate most of the energy in a few coefficients.

Other important transforms are the Haar, wavelet, Hadamard, and Walsh transforms [229, 34]. Because of the powerful properties of the wavelet transform and its extensive application opportunities in biomedical imaging, Chapters III and IV are dedicated to the wavelet transform and its applications.

2.4.3 Structural Descriptors

The basic idea of this modeling technique is that the signal characteristics of interest are separable and deterministic, and together they describe the concept of interest. The feature information is not provided in numerical form but it is given in structural form. In other words, the signals can be accurately described in symbolic form. To make the description

of structural information feasible, we need to define a suitable grammar. A grammar is defined as a tuplet $\mathbf{G} = (Z, V, S, P)$, with Z describing an alphabet of letters while V is an auxiliary alphabet of letter-valued variables. The start variable from V is denoted by S, while P describes a set of production rules [216].

2.4.4 Graph Descriptors

Graphs are a useful tool in structural pattern recognition in case the training set is too small to correctly infer pattern class grammars, or in case each pattern can be considered as class prototype. A directed graph is defined as $G_r = [V, E]$, where V is a set of nodes, and E is a subset of $N \times N$ edges in the graph [216].

The most important graph types are relational and attributed graphs.

- In general, a relational graph describes the structure of a pattern. Such graphs represent semantic networks where nodes describe pattern while edges describe the relations between nodes. A given input pattern first has to be converted into a relational graph, and then the correct class is determined by finding the matching between the given pattern and a relational graph belonging to that class. In other words, a library of relational graphs has to be designed providing for reference prototypes for every class.
- An attributed graph is a relational graph with attributes that represent measures of either symbolic or numeric patterns. These attributes are located at graph nodes. The attributed graph is defined as a 3-tuple $G_{ri} = [V_i, P_i, E_i]$, where V_i is the set of nodes, P_i the properties of nodes, and E_i the relations between nodes.

Graph descriptors form the basis for the model-based techniques in medical image processing. For instance, knowledge of the anatomic structure of certain body parts is essentially for designing models for these structures. On the other hand, a graph is very useful to describe these models. Its nodes determine the objects with associated attributes, while the edges define the relations between these objects. The model has to be patient insensitive and generalize very well. This modeling is frequently used in PET [204].

The extracted image features make it possible to establish a graph prototype of these properties. The model adaptation is accomplished by finding a corresponding subgraph.

2.4.5 Texture

Texture represents a very useful feature for medical imaging analysis by helping to segment images into regions of interest and to classify those regions.

Texture provides information about the spatial arrangement of the colors or intensities in an image. A more relevant definition is given in [34], saying that texture describes the spatial distribution of the gray levels of the pixels in a region. To measure texture is equivalent to quantifying the nature of the variation in gray level within an object. Texture is represented in an image if the gray levels vary significantly. Figure 2.7 shows examples of textured images taken from [25].

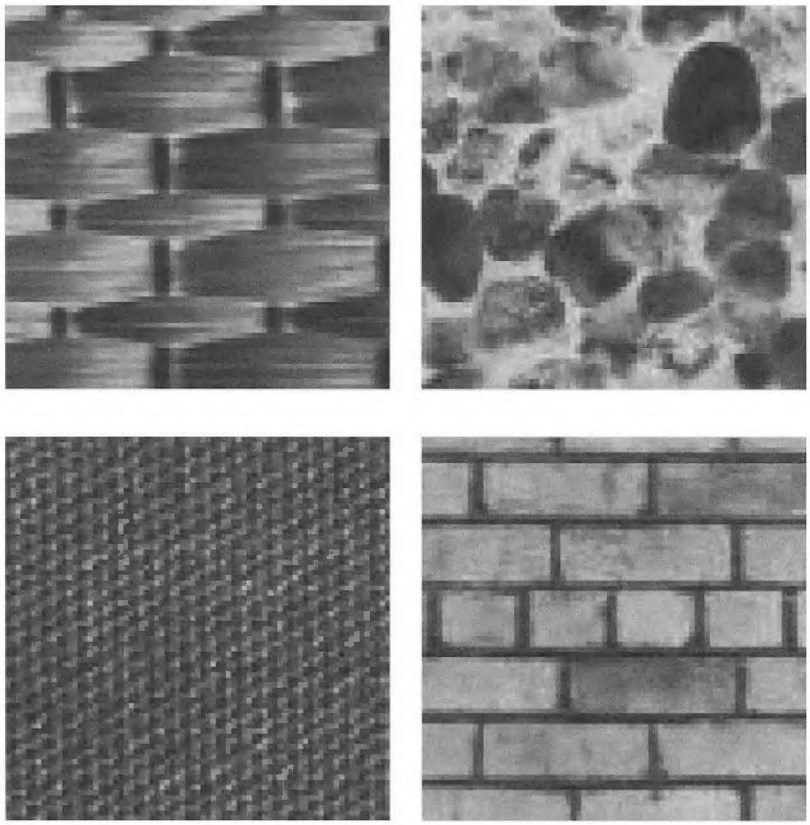


FIGURE 2.7 Some examples of textures. (Images courtesy of Dr. T. Randen.)

The literature distinguishes between random texture and pattern texture. Random texture induced by film grain noise exhibits no visible regularity while pattern texture such as cross-hatching does.

The problem of measuring texture leads us to search for methods to describe texture. A texture feature is derived from an image, and exploits

the spatial relations underlying the gray level distribution. It does not depend on the object's size, shape, orientation, and brightness.

A mathematical model for a texture pattern ρ is given in [138] as a function R of a small region S_k of pixels

$$\rho = R(S_k) \tag{2.60}$$

S_k is at the same time a function of the input image $I(m,n)$, which makes the above equation recursive.

Texture analysis based on eq. (2.60) is illustrated in Figure 2.8.

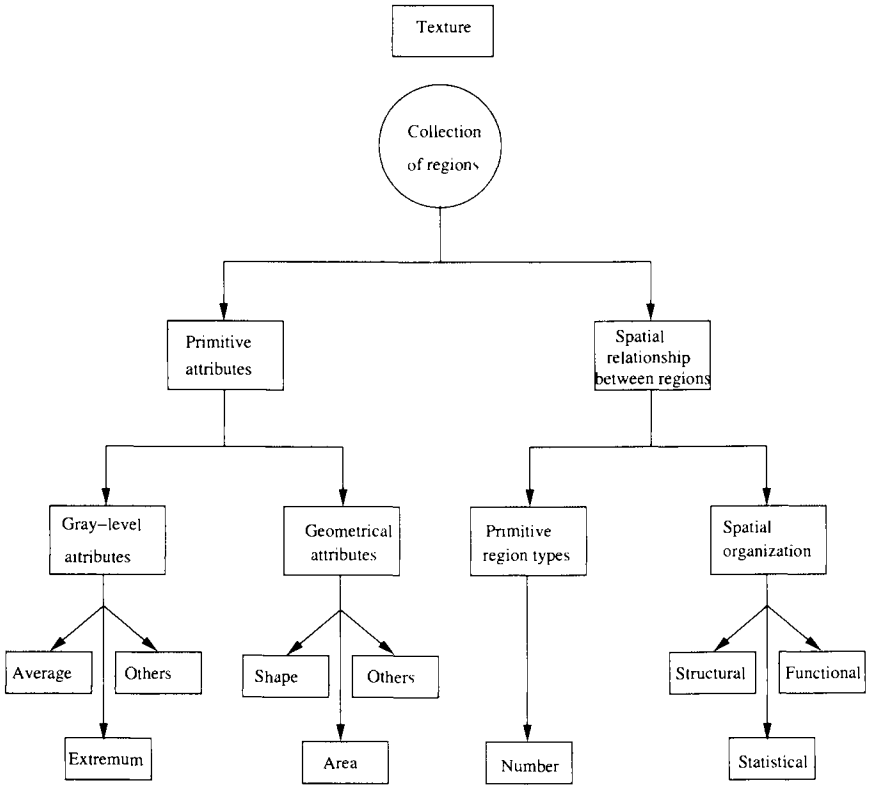


FIGURE 2.8 Texture analysis [138].

A different way to look at texture analysis is based on region segmentation. This is always the case when an object differs from the rest of the image in texture but not in average brightness. This leads to image segmentation based on texture. Instead of analyzing neighboring pixels, the focus is shifted towards neighboring regions. In other words, we have to define regions to be compared. Then the comparison is still performed

based on similarity and proximity of local patterns. A model for texture segmentation is illustrated in Figure 2.9.

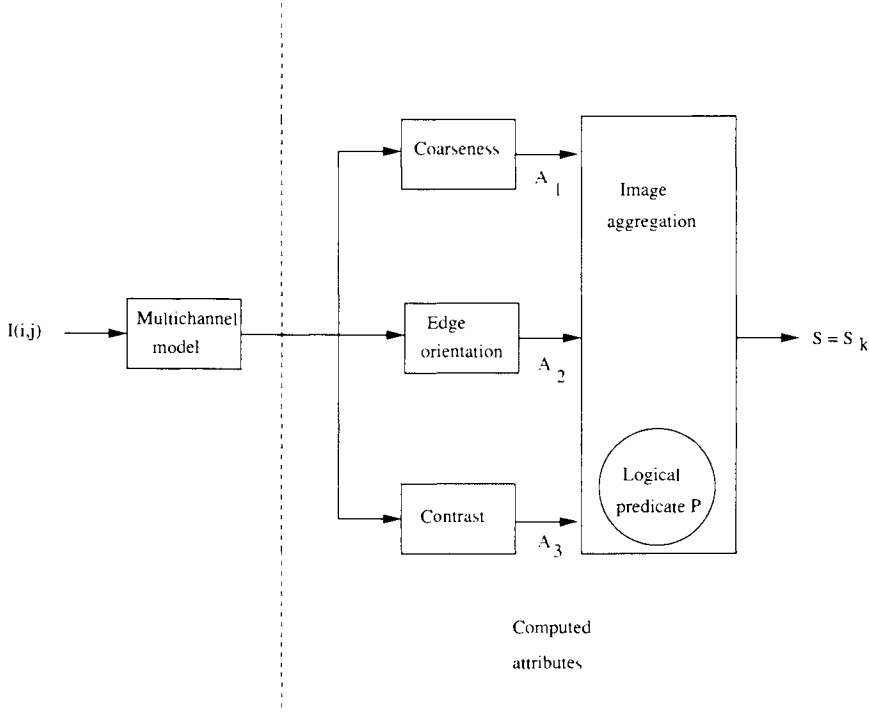


FIGURE 2.9 Model for texture segmentation [138].

To define exactly what texture is, we have to examine two main approaches: statistical and structural. In the statistical approach, texture represents a quantitative measure of the arrangement of intensities in a region. The structural approach considers texture as a set of primitive texels ordered in some regular or repeated relationship. These can be small objects that represent one unit of a repeated pattern. Then feature extraction is equivalent to locating the primitives and quantifying their spatial distribution. For example, cell nuclei in healthy liver tissue are uniformly distributed whereas unhealthy tissue has lost this ordered relationship. We can choose the nuclei as primitives, and the mean and standard deviations reflecting the spatial nuclei distribution are structural texture features.

While the structural approach works well for man-made, regular patterns, the statistical approach is more general and easy to compute, and therefore more appealing for practical problems. In the following, we will concentrate our discussion on statistical texture analysis. Figure 2.10 shows the block diagram for texture computation.

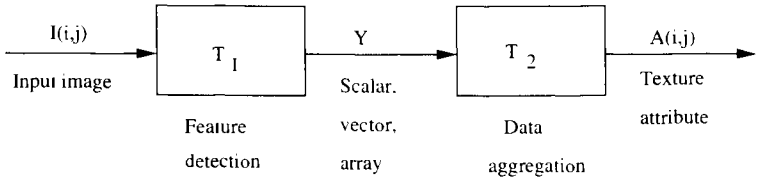


FIGURE 2.10 Statistical texture analysis.

A transformation T_1 measures a statistical feature from the image region $I(i,j)$ and provides as an output a scalar, vector, or matrix. In case of a matrix or vector an additional transformation T_2 is needed in order to compress the output of transformation T_1. The goal is to achieve a texture attribute $A(i,j)$ that is a scalar and describes the texture in region $I(i,j)$.

The statistical approach is based on the Julesz conjecture [138]. It stated that the human visual system cannot distinguish between pattern with identical first- and second-order statistics. First-order statistics measure point properties while second-order statistics capture the spatial relationship in a texture pattern. Unfortunately, many counterexamples to the conjecture have been found. But this approach has remained successful in machine vision.

Before defining these features based on first- and second-order statistics, we need to define first- and second-order statistics. In [138] it is done in terms of "dropping" a point and a line, respectively, at random on an image. First-order statistics is determined by the pixel gray level which was randomly hit and represents the probability distribution in terms of a histogram. Second-order statistics is determined by considering the gray levels at the two extremities of a randomly dropped line of varying length and orientation.

2.4.5.1 First-Order Statistics Features Starting with Figure 2.10, we see that in case of first-order statistics, Y is the image array and the transformation T_2 computes a histogram, from which the scalar A representing the mean value is easily found. Thus, the histogram (or first-order probability distribution) provides precise image information. Let us assume that $Y_k(i,j)$ represents the kth feature of a local region $W(i,j)$ of (i,j). Additionally it is assumed that Y_k takes only discrete values $\{y_1, \cdots, y_s, \ldots, y_t\}$.

Instead of considering the gray levels themselves, it is more useful to look at the gray-level difference between two pixels. Let's define a distance vector

$$\mathbf{d} = (\Delta_x, \Delta_y) \tag{2.61}$$

where Δ_x and Δ_y are both integers. The gray-level difference at the distance $\mathbf{d}$ is given by

$$Y(\mathbf{d}) = |I(i,j) - I(i+\Delta_x, j+\Delta_y)| \tag{2.62}$$

Let $H(y_s, \mathbf{d})$ be the probability that a given difference y_s derived from equation (2.62) occurs at a distance $\mathbf{d}$.

In [138], four distinct transformations T_2 were proposed for determining the texture attribute A:

1. Contrast:

$$A_1 = \sum_{y_s=y_1}^{y_t} y_s^2 H(y_s, \mathbf{d}) \tag{2.63}$$

   This is the second moment of the histogram.

2. Angular second moment:

$$A_2 = \sum_{y_s=y_1}^{y_t} [H(y_s, \mathbf{d})]^2 \tag{2.64}$$

   Flat histograms have a small A_2, whereas those concentrated around the origin yield a large A_2.

3. Entropy:

$$A_3 = -\sum_{y_s=y_1}^{y_t} H(y_s, \mathbf{d}) \log H(y_s, \mathbf{d}) \tag{2.65}$$

   Entropy is maximal for uniform $H(y_s, \mathbf{d})$.

4. Mean:

$$A_4 = \left(\frac{1}{t}\right) \sum_{y_s=y_1}^{y_t} y_s H(y_s, \mathbf{d}) \tag{2.66}$$

   A_4 is small for histograms concentrated around $y_s = y_l$ and larger otherwise.

2.4.5.2 Second-Order Statistics Features While first-order statistics provide only point properties (or gray-level distribution of the image), second-order statistics describe the spatial relationships in a texture pattern. In other words, they provide information about the relative positions of various gray levels within the image.

For notational purposes we will introduce the co-occurrence matrix or spatial dependence matrix. It describes the probability of occurrence of particular pairs y_q and y_r of gray levels at the distance $\mathbf{d}$. It is interesting to point out the relationship between the first-order $H(y_s, \mathbf{d})$ and second-order difference statistics $H(y_q, y_r, \mathbf{d})$:

$$H(y_s, \mathbf{d}) = \sum_{|y_q - y_r| = y_s} H(y_q, y_r, \mathbf{d}) \tag{2.67}$$

Five transformations T_2 are mentioned in the literature [138] for computing texture attributes A:

1. Contrast:

$$A_1 = \sum_{y_r=y_1}^{y_t} \sum_{y_s=y_1}^{y_t} \delta(y_r, y_s) H(y_r, y_s, \mathbf{d}) \tag{2.68}$$

where $\delta(y_r, y_s)$ represents a feature-dependent dissimilarity measure.

2. Angular second moment:

$$A_2 = \sum_{y_r=y_1}^{y_t} \sum_{y_s=y_1}^{y_t} [H(y_r, y_s, \mathbf{d})]^2 \tag{2.69}$$

Describes uniformity or homogeneity.

3. Inverse difference moment:

$$A_3 = \sum_{y_r=y_1}^{y_t} \sum_{y_s=y_1}^{y_t} \left[\frac{H(y_r, y_s, \mathbf{d})}{1 + \delta(y_r, y_s)} \right] \tag{2.70}$$

with $\delta(y_r, y_s) = (y_r - y_s)^2$. Describes the local homogeneity of an image.

4. Entropy:

$$A_4 = -\sum_{y_r=y_1}^{y_t} \sum_{y_s=y_1}^{y_t} H(y_r, y_s, \mathbf{d}) \log H(y_r, y_s, \mathbf{d}) \tag{2.71}$$

A_4 is a measure of randomness. It t akes low values for smooth images.

5. Correlation:

$$A_5 = \left[\frac{1}{\sigma_{y_r} \sigma_{y_s}} \right] \sum_{y_r=y_1}^{y_t} \sum_{y_s=y_1}^{y_t} (y_r - \mu_{y_r})(y_s - \mu_{y_s}) H(y_r, y_s, \mathbf{d}) \tag{2.72}$$

where σ_{y_r} and σ_{y_s} are the standard deviations, and μ_{y_r} and μ_{y_s} are the means of y_s and y_r. A_5 becomes larger if the elements of $H(y_q, y_r, \mathbf{d})$ become more similar.

2.4.5.3 Laws Texture Energy Measures A different approach to obtain texture features is to employ local masks to detect various types of texture. Each mask is designed to respond to a different local property. A set of texture energy measures (one for each convolved image) is computed for each pixel by taking the average absolute value of pixel values in a square 15×15 window. Since the texture energy measures are computed for each pixel, the feature value for a segmented object is computed by averaging the Laws texture energy of the object pixels.

Laws [133, 132] proposed a texture-energy approach that determines the variation amount within a fixed-size window. A set of nine 5×5 convolution masks is applied to an image in order to determine the texture energy, which is then represented by a vector of nine numbers for each pixel of the image being analyzed [218].

Laws provides a set of five element, one-dimensional convolution kernels named $L5, E5, S5, W5$ and $R5$:

$$\begin{array}{lll} \text{L5} & \text{(Level)} & = [\ \ 1 \quad\ 4 \quad\ 6 \quad\ 4 \quad\ 1\ \ ] \\ \text{E5} & \text{(Edge)} & = [-1 \quad -2 \quad\ 0 \quad\ 2 \quad\ 1\ \ ] \\ \text{S5} & \text{(Spot)} & = [\ \ 1 \quad\ 0 \quad\ 2 \quad\ 0 \quad -1\ \ ] \\ \text{W5} & \text{(Wave)} & = [-1 \quad\ 2 \quad\ 0 \quad -2 \quad\ 1\ \ ] \\ \text{R5} & \text{(Ripple)} & = [\ \ 1 \quad -4 \quad\ 6 \quad -4 \quad\ 1\ \ ] \end{array}$$

The names of the kernels are mnemonics for level, edge, spot, wave, and ripple. Two-dimensional kernels are created by taking the outer product of the one-dimensional kernels.

After preprocessing which removes the effects of illumination, each of the sixteen 5×5 masks is applied to the preprocessed image, producing 16 filtered images.

With $F_k[i, j]$ we denote the result of the filtering with the kth mask at pixel $[i, j]$. The texture energy map E_k for filter k is given by

$$E_k[r.c] = \sum_{j=c-7}^{c+7} \sum_{i=r-7}^{r+7} |F_k[i, j]| \tag{2.73}$$

We immediately see that each texture energy map represents an image, describing the application of the kth mask to the input image.

From the 16 energy maps based on the combination of certain symmetric pairs, the following nine final maps are obtained:

$E5L5$ measures the horizontal edge content while $L5E5$ measures the vertical edge content, and the total edge content is given by the average of these two maps.

In summary, nine energy maps are obtained, or equivalently a single image is produced with a vector of nine texture attributes at each pixel.

L5E5/E5L5	L5S5/S5L5
L5R5/R5L5	E5E5
E5S5/S5E5	E5R5/R5E5
S5S5	S5R5/R5S5
R5R5	

Feature vectors derived from Laws masks provide a good spatial discrimination since the determined measure is well localized. However, the masks only operate at a single scale which reduces their ability to characterize texture. In theory a multiscale approach for masks could be developed, but only at the cost of impracticability.

EXAMPLE 2.4.4: A problem with Laws texture energy measures are the errors they introduce along the separating line between two texture regions [80] since the computed textural energy for a pixel over a given window contains more than one texture.

This poses a serious problem for microcalcification detection in mammography [244]. Microcalcifications are known for being very small, and the 15×15 window of each pixel most likely crosses these textural boundaries. A solution to this problem was given in [243] by choosing a set of modified Laws texture energy measures. The 5×5 convolution mask was preserved but instead 12 measures were determined: six measures were obtained by averaging the absolute values of the object pixels, while another six are obtained by averaging the absolute values of the border pixels.

This substantial improvement is based on the fact that pixels of different textural regions are not combined and thus it eliminates the smoothing effect at pixel level introduced by the original Laws energy measures.

2.5 FEATURE SELECTION METHODS

There are several criteria to measure the classification performance of individual features and/or feature vectors. In this section we will review techniques to select a subset of l features out of m originally available.

The most important techniques described here are exhaustive search [216], branch and bound algorithm, max-min feature selection [229], and Fisher's linear discriminant [248].

2.5.1 Exhaustive Search

Let $\mathbf{y}$ with $\mathbf{y} = [y_1, \ldots, y_D]^T$ be a pattern vector. Exhaustive search selects the d best features out of the maximal available features D as to minimize the classification error.

The resulting number of total combinations is

$$\binom{D}{d} = \frac{D!}{(D-d)!d!} \tag{2.74}$$

The main disadvantage of exhaustive search is that the total number of combinations increases exponentially with the dimension of the feature vector.

2.5.2 Branch and Bound Algorithm

The computational cost associated with the exhaustive search is very huge. Therefore, new techniques are necessary to determine the optimal feature set without explicit evaluation of all the possible combinations of d features. Such a technique is the branch and bound algorithm.

This technique is applicable when the separability criterion is monotonic, that is,

$$\text{If} \quad \chi_1 \subset \chi_2 \subset \cdots \chi_D, \quad \text{then} \quad J(\chi_1) \leq J(\chi_2) \leq \cdots J(\chi_D). \tag{2.75}$$

χ_i is one of the possible feature subsets with i features while $J(\chi_i)$ is the corresponding criterion function. This technique is visualized for an example in Figure 2.11. The problem is to select out of the four features y_1, y_2, y_3, y_4 the two most relevant. At each stage, we determine the maximal value of the criterion function and determine the correct path based on this. At each node then, we eliminate one feature from the initial feature set. The criterion function J is updated as well: if J is larger than its initial value J_0, then continue along the same path, else choose the next path at left.

In summary, the branch and bound method represents a computationally efficient way by formulating the problem as a combinatorial optimization task. The optimal solution is obtained without involving exhaustive enumeration of all possible combinations.

2.5.3 Max-Min Feature Selection

The main idea here is to perform feature selection based on the individual and pairwise merit of features [113]. Thus the criterion function has to be evaluated only in one- and two-dimensional spaces.

This technique determines, based on a given criterion function J, the individual and pairwise relevance of features. Let us assume that we have already determined k features and thus the remaining features are $Y - X_k$. By writing $Y = \{y_j | j = 1, \ldots, D\}$, we denote the initial available feature set, while by X_k we denote the already chosen features from Y. Let $y_j \in Y - X_k$ and $x_l \in X_k$.

The max-min feature selection first determines

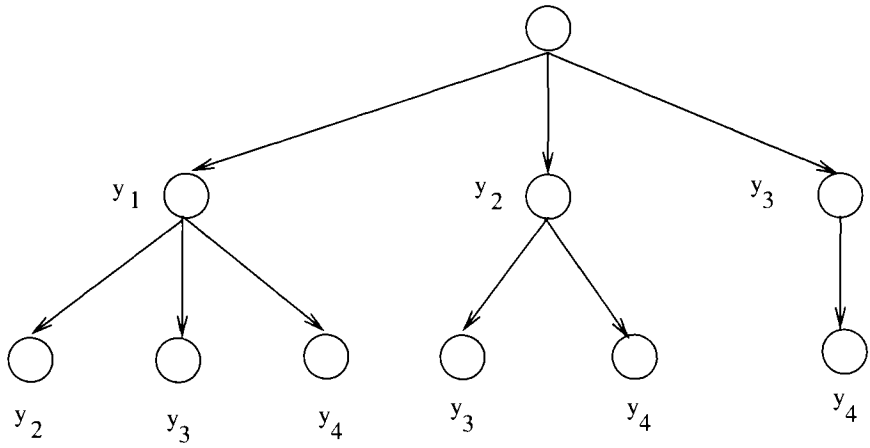


FIGURE 2.11 Feature selection based on branch and bound algorithm.

$$\Delta J(y_j, x_l) = J(y_j, x_l) - J(x_l) \tag{2.76}$$

and then it chooses

$$\Delta J(x_{k+1}, x_l) = \overset{max}{j}\overset{min}{l} \Delta J(y_j, x_l) \quad \text{with} \quad x_l \in X_k \tag{2.77}$$

2.5.4 Sequential Forward and Sequential Backward Selection

There are many situations in medical imaging when the determination of the optimal feature set based on the branch and bound algorithm is computationally too intensive. Therefore the question of trade-off between the optimality and efficiency of algorithms leads us to seek suboptimal solutions.

The suboptimal search methods reviewed in this section represent an alternative to the above-mentioned problems and are shown to give very good results and to be computationally more effective than other techniques [248]. The simplest suboptimal techniques are the sequential forward and sequential backward selection algorithms [200]. We follow here the approach described in [113].

Sequential forward selection (SFS) is a bottom-up technique. We start with an empty feature set, and as the first feature we choose the individually best measurement. In the subsequent steps, we choose from the remaining feature set only the feature that, together with the previously selected ones, yields the maximum value of the criterion function. For a better understanding, let us assume that the feature set X_k consists of k features out of the original complete feature set $Y = \{y_j | j = 1, \ldots, D\}$. We choose

the $(k+1)$th feature from the set of the remaining features, $Y - X_k$, such that

$$J(X_{k+1}) = \max \quad J(X_k \cup y_j), \quad y_j \in Y - X_k \tag{2.78}$$

$$\text{Initialization:} \quad X_0 = \Phi \tag{2.79}$$

Sequential backward selection (SBS), on the other hand, is a top-down technique. Here, we start with the whole feature set Y, and at each step we eliminate only that feature which yields the smallest decrease of the criterion function.

We assume that k features are already eliminated from the initial feature set, $Y = \{y_j | j = 1, \ldots, D\}$, such that the remaining feature set is given by X_{D-k}. The $(k+1)$th feature to be removed is selected from the set X_{D-k} such that

$$J(X_{D-k-1}) = \max J(X_{D-k} - y_j), \quad y_j \in X_{D-k} \tag{2.80}$$

$$\text{Initialization:} \quad X_D = Y \tag{2.81}$$

SFS is computationally less expensive than SBS since the criterion function is evaluated in d-dimensional spaces, while for the SBS it has to be evaluated in spaces of the dimensionality from d to D. The advantage of the SBS as a top-down technique is that at each stage the information loss can be supervised.

However, the disadvantage with both SBS and SFS lies in the so-called nesting effect. This means that once eliminated by SBS a feature cannot be reconsidered later, while a feature added by SFS cannot be removed later. To overcome these disadvantages associated with SBS and SFS, a new technique, the so-called floating search method was proposed in [200]. Based on the traditional SBS and SFS, it has the flexibility to allow revision at later stages of added or discarded features in previous stages. The computational efficiency of the floating search technique makes it attractive for large-scale feature selection, when the number of features approaches 100. Besides avoiding the nesting of features, this technique compared to the branch and bound method is also tolerant to deviations from monotonic behavior of the feature selection criterion function.

2.5.5 Fisher's Linear Discriminant

The motivation of this technique is based on achieving a maximum class separability. This simple method is built upon information related to the way the sample vectors are scattered in the feature space.

Fisher's approach determines out of D given features the d best features based on a $d \times D$ linear transformation matrix $\mathbf{A}$. Let us assume

that we are given a D-dimensional feature vector. and obtain a new d-dimensional feature vector $\mathbf{x}$ by

$$\mathbf{x} = \mathbf{A}\mathbf{y} \tag{2.82}$$

Let us now define the between-class scatter matrix $\mathbf{S_b}$ and the within-class scatter matrix $\mathbf{S_w}$ as

$$\mathbf{S_b} = \sum_{i}^{M} P(\omega_i) E[(\mu_i - \mu_0)(\mu_i - \mu_0)^T] \tag{2.83}$$

and

$$\mathbf{S_w} = \sum_{i}^{M} P(\omega_i) E[(\mathbf{y} - \mu_i)(\mathbf{y} - \mu_i)^T] \tag{2.84}$$

$P(\omega_i)$ is the a priori probability of class ω_i, μ_i is the mean vector of class ω_i, and μ_0 is the global mean vector over all classes ω_i with $i = 1, \ldots, M$.

Figure 2.12 illustrates three different cases of classes at different locations and within-class variances [229].

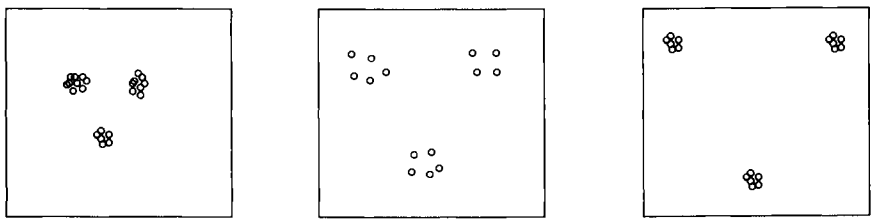


FIGURE 2.12 Classes with (a) small within-class variance and between-class distances, (b) large within-class variance and small between-class distances, and (c) small within-class variance and large between-class distances [229].

A reasonable measure of projected data separability is given by the following criterion function $J(\mathbf{A})$:

$$J(\mathbf{A}) = \frac{\mathbf{A}^T \mathbf{S_b} \mathbf{A}}{\mathbf{A}^T \mathbf{S_w} \mathbf{A}} \tag{2.85}$$

The goal is to determine an adequate $\mathbf{A}$ which maximizes the above equation. The solution yields a generalized eigenvector problem

$$\mathbf{S_b A} = \Lambda \mathbf{S_w A} \tag{2.86}$$

where Λ is the matrix of the significant eigenvalues, and $\mathbf{A}$ is at the same time also the matrix of the corresponding eigenvalues. Thus, $\mathbf{A}$ represents

a projection matrix which transforms the feature vector $\mathbf{y}$ into a lower dimensional vector $\mathbf{x}$.

The algorithmic description of determining the relevant features is as follows:

1. Determine Fisher's covariance matrix [216]

$$J_{wb} = \mathbf{S_w^{-1}S_b} \tag{2.87}$$

2. Determine D eigenvalues of the matrix J_{wb} and the corresponding eigenvectors based on

$$J_{wb}\mathbf{Q} = \mathbf{Q}\Lambda \tag{2.88}$$

where $\Lambda = [\lambda_1, \ldots, \lambda_D]$ is the matrix of the eigenvalues and $\mathbf{Q}$ is the matrix of the corresponding eigenvectors. Let us assume that $\lambda_1 > \lambda_2 > \ldots > \lambda_D$ holds.

3. Determine the rank d of matrix J_{wb}.
4. Determine for each $i = 1, \ldots, D$ the term $U(i)$ which represents the sum of the absolute value of the first d eigenvectors:

$$U(i) = \sum_{j=1}^{d} |\mathbf{Q}(i,j)|, \quad j = 1, 2, \ldots, D \tag{2.89}$$

where D is the dimension of the feature vector $\mathbf{y}$.

5. The feature ranking is based on the value of U: the larger the value of U the better the selection.

EXAMPLE 2.5.1: Acoustic neuromas are benign tumors which generally grow along the track of the acoustic nerve and are detected in MR images.

An automatic system for detection and classification of acoustic neuromas or candidate tumor regions (CTRs) works only as well as the relevance of the selected features describing such neuromas.

The following features were found to be relevant in characterizing acoustic neuromas [55]:

- Shape: These neuromas being roughly globular objects, a contour-based moment scheme can be employed requiring only four moment measurements.
- Compactness: This feature is concerned with shape and measures how closely the CTR combination approximates a circular disc.
- Position: Acoustic neuromas grow on and along the acoustic nerve and are therefore site specific. The position of a CTR combination was represented by the position of its centroid in relationship to

the centroid of the head in the image. Two resulting values, the position and the angle, are treated as separate features.

- Symmetry: Acoustic neuromas induce a nonsymmetry in an MRI image as can be seen from Figure 2.13. A descriptor consisting of four parameters describes the position of the two clusters F_1 and F_2.

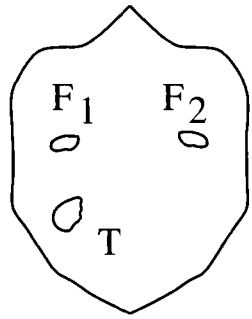


FIGURE 2.13 Typical position of fp-tumors F_1 and F_2. The rp-tumor is marked with T [55].

- Average pixel gray level: This encodes brightness and texture.
- Gray level standard deviation: Also serves for representing brightness and texture.

The collection of these seven distinct features (position counts as two separable features) results in a vector of dimension of 13.

TABLE 2.3 Results of assessing the 127 subsets. The eight best feature subsets achieving class separability are presented. The higher the J_{wb} the better the class separability between tumor and non-tumor classes. * shows that a feature was used in the feature subset, while – shows it was not considered [55].

Shape	Inter-clus. posit.	Global angle	Global dist.	Compactness	Mean intens.	Stand. deviat.	Dim. vector	J_{wb}
*	*	*	*	*	*	*	13	8.079
*	*	*	*	*	–	*	12	8.065
*	*	*	–	*	*	*	12	8.061
*	*	*	–	*	–	*	11	8.006
*	*	*	*	*	*	–	12	7.819
*	*	*	*	*	–	–	11	7.811
*	*	*	–	*	*	–	11	7.743
*	*	*	–	*	–	–	10	7.739

The data set consists of 105 pattern vectors with 70 describing tumors and 35 nontumors. The criterion function chosen as a measure for class separability was proposed by Fukunaga [69]

$$J_{wb} = \text{Trace}(\mathbf{S}_\mathbf{w}^{-1}\mathbf{S}_\mathbf{b}) \tag{2.90}$$

The best feature subset yields the highest value for J_{wb}. Table 2.3 gives the details of the feature subsets that give the eight best class separabilities.

EXERCISES

1. Show that the central moments shown in eq. (2.7) are translation invariant while the normalized moments shown in eq. (2.9) are both translation and scaling invariant.
2. Write a program to compute both the central and normalized moments described in eq. (2.7) and (2.9) and apply it to any digitized image of your choice.
3. Write a program that determines the AR parameters for a noncausal prediction model. Then apply it to homogeneous and isotropic image with the following autocorrelation sequence

$$r(k,l) = \exp\left(-2\sqrt{k^2 + l^2}\right) \tag{2.91}$$

   for a window W of order $p = q = 1$.
4. Write a program to determine the first- and second-order moments and apply it to any digitized image of your choice.
5. Given the following image array

$$\begin{bmatrix} 1 & 3 & 1 \\ 0 & 1 & 3 \\ 2 & 4 & 2 \end{bmatrix} \tag{2.92}$$

   compute its two-dimensional DFT transform.
6. For the image given in Exercise 2.5, compute the two-dimensional discrete cosine transform (DCT) and discrete sine transform (DST).
7. Write a program to compute the two-dimensional DFT transform and apply it to any digitized image of your choice.
8. Write a program to compute the two-dimensional DCT and DST transform and apply it to any digitized image of your choice.
9. Write a program for sequential forward and sequential backward selection.
10. Apply Fishers's linear discriminant to separate the following data belonging to the two classes ω_1 and ω_2.

Class	ω_1	Class	ω_2
−0.25	1.83	1.18	−2.55
1.13	3.18	1.65	−2.60
−2.18	3.77	4.20	−0.28
0.55	−0.10	2.10	−0.94
−2.31	−0.41	2.21	−1.10

(2.93)

a) Calculate the within- and between-class scatter matrices, $\mathbf{S_W}$ and $\mathbf{S_B}$.

b) Write a program that computes the relevant features and apply it to the given data set.

III

SUBBAND CODING AND WAVELET TRANSFORM

3.1 INTRODUCTION

New transform techniques that specifically address the problems of image enhancement and compression, edge and feature extraction, and texture analysis received much attention in recent years especially in biomedical imaging. These techniques are often found under the names multiresolution analysis, time-frequency analysis, pyramid algorithms, and wavelet transforms. They became competitors to the traditional Fourier transform, whose basis functions are sinusoids. The wavelet transform is based on wavelets, which are small waves of varying frequency and limited duration. In addition to the traditional Fourier transform, they provide not only frequency but also temporal information on the signal.

In this chapter, we review the basics of subband coding and present in great detail the different types of wavelet transforms. By subband coding, we mean a transformation created by filtering and subsampling. The signal is separated approximately into frequency bands for efficient coding. The subband signals are quantized such that the objective measure is maximized. A wavelet is a basis function in continuous time. We know that a function $f(t)$ can be represented by a linear combination of basis functions,

such as wavelets. The most important aspect of the wavelet basis is that all wavelet functions are constructed from a single mother wavelet. This wavelet is a small wave or a pulse.

This chapter will also reveal one of the most important aspects of signal processing: the connections among filters, filter banks, and wavelets. Historically, they were separately developed but they must be seen together. The lowpass filter coefficients of the filter bank determine the scaling functions; the highpass filter coefficients produce the wavelets.

3.2 THE THEORY OF SUBBAND CODING

In the following we will review the basics of subband decomposition which is strongly related to filter bank theory. A filter bank is a set of filters. The analysis bank splits the input signal into M subbands using a parallel set of bandpass filters. In other words, the input signal is separated into frequency bands. Those subsignals can be compressed more efficiently than the original signal. Then they can be transmitted or stored. This processing principle is called subband coding or subband decomposition. The synthesis bank reconstructs the original signal from the subband signals. The essential information is extracted from the subband signals in the processing block between the banks. In audio/image compression, the spectral contents are coded depending on their energies.

The analysis bank often has two filters, a lowpass and highpass filter. The lowpass filter takes "averages": it smooths out the high-frequency components. The response of this filter is small or near zero for the highest discrete-time frequency $\omega = \pi$. The highpass filter takes "differences": it smooths the low-frequency components. The response of this filter is small or zero for the frequencies near $\omega = 0$. As mentioned before, these two filters separate the signal into frequency bands. One difficulty arises: the signal length has doubled. This means twice as much storage is needed to keep two full-length outputs. Since this is unacceptable, the solution is to downsample or decimate the signal. Downsampling is an essential part of the filter bank. Only the even-numbered components of the two outputs from the lowpass and highpass filters are saved, while the others are removed. We see that the analysis part has two steps, filtering and downsampling. The synthesis part is the inverse of the analysis bank. The synthesis part also has two steps, upsampling and filtering. The order must be reversed as it is always for inverses. The first step is to recreate full-length vectors. Since downsampling is not invertible, upsampling is the solution. The odd-numbered components are replaced by zeros. The second step of the filter bank is the filtering. The responses of a lowpass and highpass filter are added together in order to reproduce the original input signal up to a delay.

When the synthesis bank recovers the input signal exactly (apart from a time delay), the filter bank is called a perfect reconstruction filter bank.

In the following we will describe mathematically the concepts of downsampling, upsampling, and perfect reconstruction filter banks.

3.2.1 Decimation and Interpolation

We previously have seen that in order to reduce the amount of data to be transmitted, the operation of downsampling or decimation is needed. The filters' outputs must be downsampled. Then in order to regain the original signal, upsampling or interpolation is needed. The removed components are replaced by zeros. This section will review the operation of decimation and interpolation in the time-domain as well as in the frequency domain.

The interpolation and decimation techniques being a basic part of the theory of subband decomposition are described in Figures 3.1 and 3.3. The decimation accomplishes a reduction of the sampling rate by a factor M of a given signal $\{x(n)\}$ after this signal passes through an antialiasing filter $h(n)$. The subsampler is represented in Figure 3.1 by a circle containing the integer number M and a downward arrow. The subsampling considers only each Mth sample value and achieves such a new scaling of the samples on the x-axis.

The retained intermediate signal x' is shown in Figure 3.1, as well as the new downsampled signal $y(n)$.

$$x' = \begin{cases} x(n), & n = 0. \pm M. \pm 2M. \ldots \\ 0. & \text{else} \end{cases} \tag{3.1}$$

$$y(n) = x'(Mn) = x(Mn) \tag{3.2}$$

The intermediate signal $x'(n)$ has the same sampling rate as $x(n)$ and can be expressed mathematically as the product of $x(n)$ with a Dirac train

$$x' = \left[\sum_{r=-\infty}^{\infty} \delta(n - rM) \right] x(n) \tag{3.3}$$

The Dirac train can be written as a Fourier sum

$$\sum_{r} \delta(n - rM) = \frac{1}{M} \sum_{k=0}^{M-1} e^{j\frac{2\pi}{M}nk} \tag{3.4}$$

We see immediately that we can obtain

$$x'(n) = \frac{1}{M} \sum_{k=0}^{M-1} x(n) e^{j\frac{2\pi}{M}nk} \tag{3.5}$$

We now obtain as a Fourier transform $X'(z)$ of $x'(n)$:

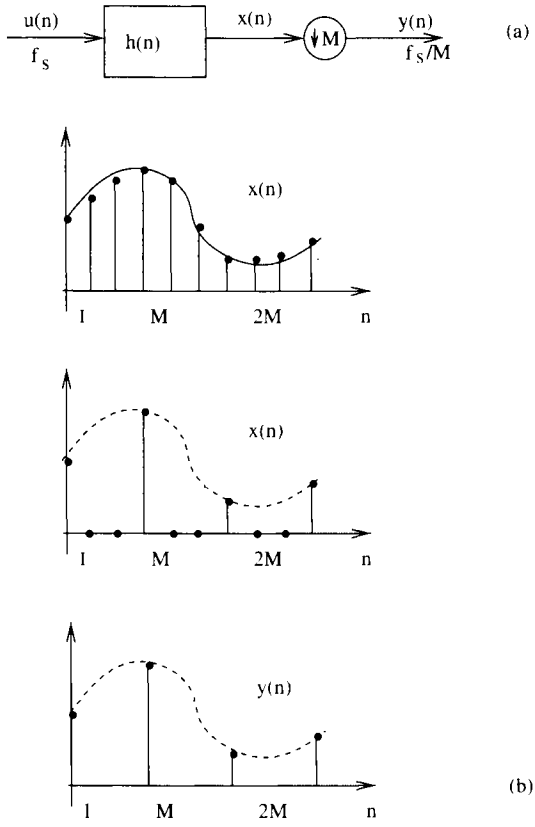


FIGURE 3.1 Representation of decimation technique: (a) filter and downsampler, (b) typical time sequences of the intermediate signals.

$$X'(z) = \frac{1}{M}\sum_{k} Z\{x(n)\left(e^{j2\pi k/M}\right)^n\} = \frac{1}{M}\sum_{k=0}^{M-1} X(ze^{-j2\pi k/M}) \tag{3.6}$$

By choosing $W = e^{\frac{-j2\pi}{M}}$, we obtain

$$X'(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(zW^k) \tag{3.7}$$

Then we get for the frequency response on the unit circle $z = e^{j\omega}$

$$X'(e^{j\omega}) = \frac{1}{M}\sum_{k=0}^{M-1} X(e^{j(\omega - \frac{2\pi k}{M})}) \tag{3.8}$$

The preceding expression shows the Fourier transform yields a sum of M delayed by $2\pi/M$ copies of the original signal. Next, we rename the time axis, such that we get a compression of the time scale with the factor M. We see immediately

$$Y(z) = \sum_{n=-\infty}^{\infty} x'(Mn)z^{-n} = \sum_{k=-\infty}^{\infty} x'(k)(z^{\frac{1}{M}})^{-k} \tag{3.9}$$

or

$$Y(z) = X'(z^{\frac{1}{M}}) \tag{3.10}$$

and

$$Y(e^{j\omega}) = X'(e^{\frac{j\omega}{M}}) \tag{3.11}$$

For the M subsampler, we now obtain

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(z^{\frac{1}{M}}W^k) \tag{3.12}$$

or

$$Y(e^{j\omega}) = \frac{1}{M}\sum_{k=0}^{M-1} X(e^{j(\frac{\omega-2\pi k}{M})}) \tag{3.13}$$

It is important to realize that a compression in the time domain leads to a dilation in the frequency domain in such a way that the interval 0 to $\frac{\pi}{M}$ now covers the band from 0 to π. The spectra of the intermediate signals occurring based on the subsampling are shown in Figure 3.2.

Interpolation, on the other hand, is a technique to increase the sampling rate of a given signal by a factor M. As shown in Figure 3.3, this can be accomplished by a combination of an upsampler and a lowpass filter $g(n)$.

The upsampler is symbolically shown by a circle containing an upward arrow. The upsampler is described mathematically as

$$y(n) = \begin{cases} x(n/M), & n = 0, \pm M, \pm 2M, \ldots \\ 0, & \text{else} \end{cases} \tag{3.14}$$

Upsampling can be achieved by inserting $M-1$ zeros between the sampled values and by reindexing the time scale. The sampling rate is thus increased by a factor M. The upsampling has two major effects: it leads to a dilation in the time domain while achieving a compression in the frequency domain, and based on the insertion of zeros it generates high-frequency signals and images. It is easy to prove these effects in the transformation domain:

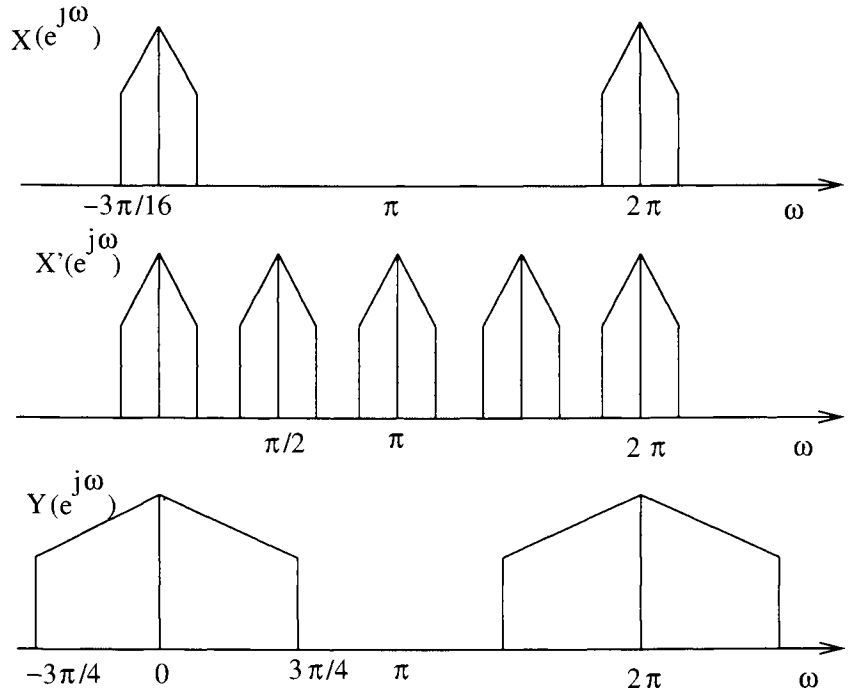


FIGURE 3.2 Frequency spectra of downsampled signals for $M = 4$.

$$Y(z) = \sum_{n=-\infty}^{\infty} y(n)z^{-n} = \sum_{n=-\infty}^{\infty} x(\frac{n}{M})z^{-n} = \sum_{k=-\infty}^{\infty} x(k)(z^M)^{-k} \quad (3.15)$$

or

$$Y(z) = X(z^M), \quad Y(e^{j\omega}) = X(e^{j\omega M}) \quad (3.16)$$

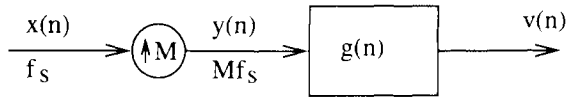
To remove the undesired high-frequency components, a lowpass filter must follow the upsampler.

The interpolators and decimators are represented in the time domain as

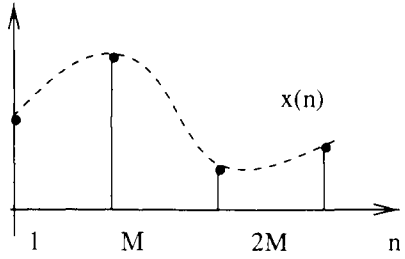
$$y(n) = \sum_k h(Mn - k)u(k) \quad (3.17)$$

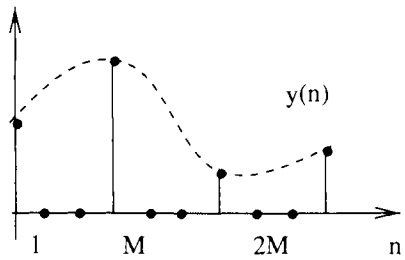
and

$$v(n) = \sum_k g(n - Mk)x(k) \quad (3.18)$$



(a)





(b)

FIGURE 3.3 Upsampling or interpolation: (a) upsampling operation, (b) input and output waveforms for M=3.

EXAMPLE 3.2.1: Consider the following alternating vector $\mathbf{x}(n) = (\ldots, 1, -1, 1, -1, 1, -1, \ldots)$. Downsampling produces the vector $(\downarrow)\mathbf{x} = (\ldots, 1, 1, 1, \ldots)$, while upsampling produces the vector $(\uparrow)\mathbf{x} = (\ldots, 1, 0, -1, 0, 1, 0, -1, 0, 1, 0, -1, \ldots)$.

We will finish this subsection with the following important remarks:

1. The result of a series circuit of a downsampler and an upsampler is shown in Figure 3.4. The signal at the output of the interpolator is identical with the signal $x'(n)$ from Figure 3.1. The spectra of the signals outputting both an upsampler and a downsampler are corrupted by aliasing.

2. There are equivalencies regarding transfer functions across down- and upsamplers as illustrated in Figure 3.5. Filter and interpolator, or respectively decimator, can be interchanged, if the filter transfer functions are correspondingly changed.

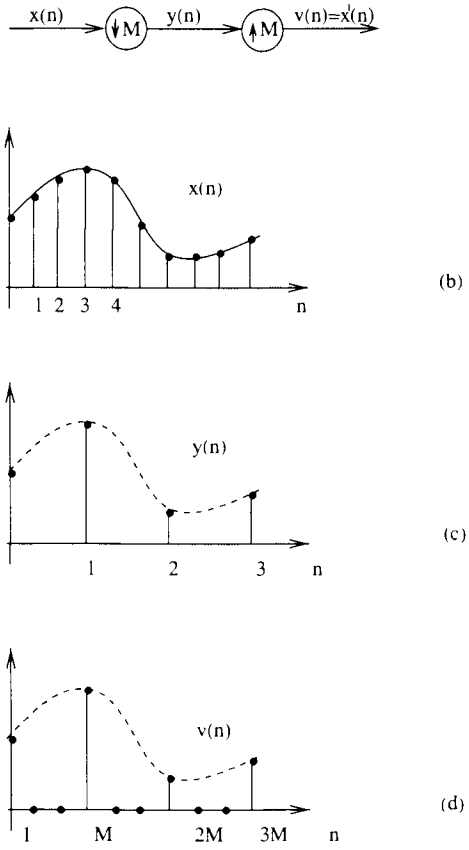


FIGURE 3.4 Decimation and interpolation signals: (a) downsampling and upsampling operation, (b) sampled input signal, (c) downsampled sequence, (d) upsampled sequence.

3. A quadrature-mirror-filter [2] was first used in the elimination of aliasing in two-channel subband coders. Let $h_0(n)$ be a FIR lowpass filter having real coefficients. A mirror filter is described as

$$h_1(n) = (-1)^n h_0(n) \tag{3.19}$$

or, equivalently, in the transform domain as

$$\begin{aligned} H_1(z) &= H_0(-z) \\ H_1(e^{j\omega}) &= H_0(e^{j(\omega-\pi)}) \end{aligned} \tag{3.20}$$

In the following, we make the substitution $\omega \to \frac{\pi}{2} - \omega$, and note that the absolute value is an even function of ω. We thus obtain

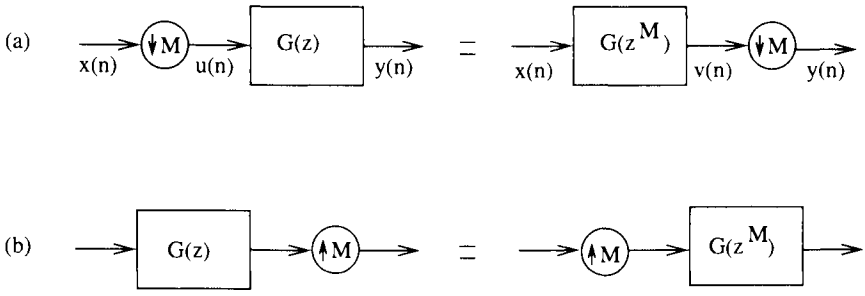


FIGURE 3.5 Equivalent structures: (a) equivalence for downsampling, (b) equivalence for upsampling.

$$|H_1(e^{j(\frac{\pi}{2}-\omega)})| = |H_0(e^{j(\frac{\pi}{2}+\omega)})| \tag{3.21}$$

The above equation shows the mirror property of H_0 and H_1 with respect to $\omega = \pi/2$, as illustrated by Figure 3.6. The highpass response is a mirror image of the lowpass magnitude with respect to the middle frequency $\frac{\pi}{2}$ — the quadrature frequency. This explains the term quadrature mirror filter.

3.2.2 Two-Channel Filter Banks

A filter bank is a set of filters, linked by sampling operators and sometimes by delays. In a two-channel filter bank, the analysis filters are normally lowpass and highpass. Those are the filters $H_0(z)$ and $H_1(z)$. The filters in the synthesis bank $G_0(z)$ and $G_1(z)$ must be specially adapted to the analysis filters in order to cancel the errors in the analysis bank or to achieve perfect reconstruction. This means that the filter bank is biorthogonal. The synthesis bank, including filters and upsamplers, is the inverse of the analysis bank. In the next chapter we will talk about biorthogonal filter banks and wavelets. If the synthesis is the transpose of the analysis, we deal with an orthogonal filter bank. A paraunitary (orthonormal) filter bank is a special case of a biorthogonal (perfect reconstruction) filter bank.

This section derives the requirements and properties for the perfect reconstruction by means of a two-channel filter bank. Such a filter bank is shown in Figure 3.7. The input spectrum $X(e^{j\omega}), 0 \leq \omega\pi$ is going to be split in two subbands. The filter bank consists of two main parts: analysis and synthesis. The analysis part is performing the signal decomposition while the synthesis part is reconstructing the signal.

The analysis filters $H_0(z)$ and $H_1(z)$ separate the signal in two equal subbands. The filter output signals θ_0 and θ_1 are each downsampled by 2, and then we obtain, as output signals of the analysis part, the subband signals $v_0(n)$ and $v_1(n)$. These signals are quantized by employing a sub-

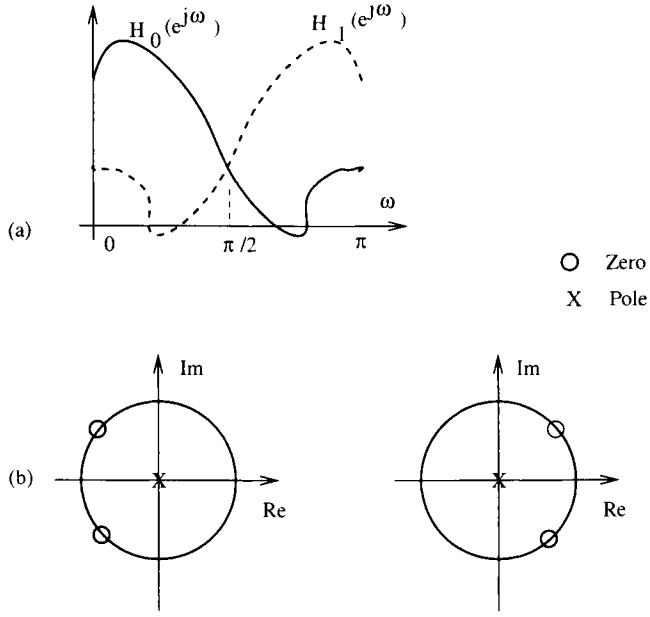


FIGURE 3.6 (a) Frequency response of the quadrature mirror filter and (b) pole-zero distribution.

band coder, coded, and sent to a receiver. Assuming ideal conditions, we expect to receive exactly $v_0(n)$ and $v_1(n)$ at the synthesis part. These signals are interpolated by 2, and the newly obtained signals are denoted by $f_0(n)$ and $f_1(n)$. These signals are processed by the interpolation filters $G_0(z)$ and $G_1(z)$. The reconstructed signal $\hat{x}(n)$ is obtained by adding the signals $y_0(n)$ and $y_1(n)$. The requirements for a perfect reconstruction are described in great detail in [235].

Here, we derive these requirements for the time and frequency domain. The output signals for the interpolation and decimation filter are given by

$$V_0(z) = \frac{1}{2}[\theta_0(z^{1/2}) + \theta_0(-z^{1/2})] \tag{3.22}$$

$$F_0(z) = V_0(z)$$

and also by

$$Y_0(z) = \frac{1}{2}G_0(z)[H_0(z)X(z) + H_0(-z)X(-z)] \tag{3.23}$$

and

$$Y_1(z) = \frac{1}{2}G_1(z)[H_1(z)X(z) + H_1(-z)X(-z)] \tag{3.24}$$

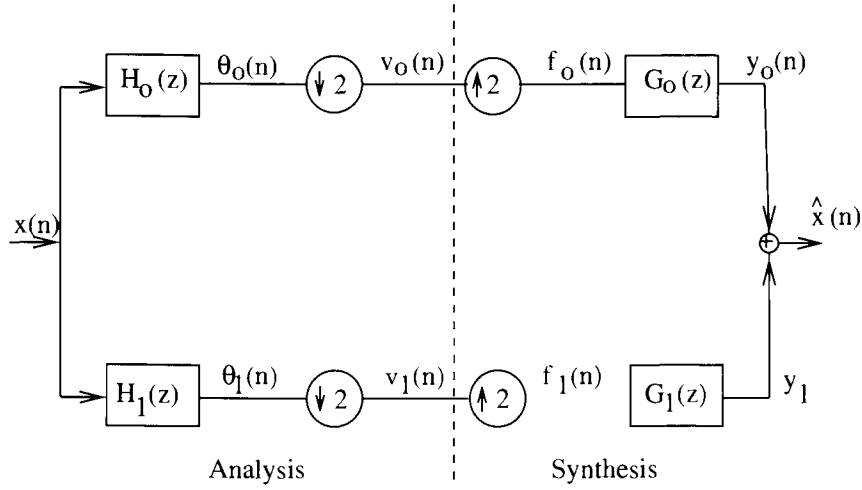


FIGURE 3.7 Two-channel filter bank.

Next we z-transform the reconstructed signal and obtain

$$\begin{aligned}\hat{X}(z) &= \frac{1}{2}X(z)[H_0(z)G_0(z) + H_1(z)G_1(z)] \\ &\quad + \frac{1}{2}X(-z)[H_0(-z)G_0(-z) + H_1(-z)G_1(-z)]\end{aligned} \tag{3.25}$$

To achieve a perfect reconstruction, we must impose two conditions:

$$H_0(-z)G_0(z) + H_1(z)G_1(z) = 0 \tag{3.26}$$

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 2 \tag{3.27}$$

Equation (3.26) eliminates aliasing, while eq. (3.27) eliminates amplitude distortion. The two distinct equations can be rewritten in matrix form as

$$\begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix} = \frac{2}{\det(\mathbf{H}_m(z))} \begin{bmatrix} H_1(-z) \\ -H_0(-z) \end{bmatrix} \tag{3.28}$$

where the nonsingular analysis modulation matrix $\mathbf{H}_m(z)$ is given by

$$\mathbf{H}_m = \begin{bmatrix} H_0(z) & H_0(-z) \\ H_1(z) & H_1(-z) \end{bmatrix} \tag{3.29}$$

Equation (3.28) describes the cross-modulation of the analysis and synthesis filters. This means the diagonally opposed filters in Figure 3.7 are functionally related by $-z$ in the Z-domain: $G_1(z)$ is a function of $H_0(-z)$,

while $G_0(z)$ is a function of $H_1(-z)$. For finite impulse response (FIR) filters, the determinate is a pure delay.

Three different solutions to eqs. (4.7) and (4.8) are given in Table 3.1 [74].

TABLE 3.1 Filter solutions for perfect reconstruction.

Filter	QMF	CQF	Orthonormal
$H_0(z)$	$H_0^2(z) - H_0^2(-z) = 2$	$H_0(z)H_0(z^{-1}) + H_0(-z)H_0(-z^{-1}) = 2$	$G_0(z^{-1})$
$H_1(z)$	$H_0(-z)$	$z^{-1}H_0(-z^{-1})$	$G_1(z^{-1})$
$G_0(z)$	$H_0(z)$	$H_0(z^{-1})$	$G_0(-z)G_0(z^{-1}) + G_0(-z)G_0(-z^{-1}) = 2$
$G_1(z)$	$-H_0(-z)$	$zH_0(-z)$	$-z^{-2K+1}G_0(-z^{-1})$

The second and third columns in Table 3.1 represent the quadrature mirror filter (QMF) solution and the conjugate quadrature mirror filters (CQFs) solution, while the fourth column represents the orthonormal filters solution. $2K$ in Table 3.1 denotes the length or number of coefficients in each filter.

The orthonormality condition for perfect reconstruction filter banks is defined in the time domain as

$$< g_i(n), g_j(n + 2m) >= \delta(i - j)\delta(m), \quad i, j = \{0, 1\} \tag{3.30}$$

This condition is different from the *biorthogonality* condition, which is given in the time domain as

$$< h_i(2n - k), g_j(k) >= \delta(i - j)\delta(n), \quad i, j = \{0, 1\} \tag{3.31}$$

It is useful to note that the biorthogonality condition holds for all two-band, real-coefficient perfect reconstruction filter banks.

Multirate techniques are considered among the most important tools for multiresolution spectral analysis. In this context, the perfect reconstruction quadrature mirror filter (PR QMF) bank is the most efficient signal decomposition block. The PR QMF bank splits the input spectrum into two equal subbands, a low (L) and high (H) band. This decomposition can be applied again to these (L) and (H) half bands to produce the quarter bands: (LL), (LH), (HL), and (HH). Two of these decomposition levels are illustrated in Figure 3.8. The original signal at a data rate f_s is decomposed into four subband signals $v_0(n), \ldots, v_3(n)$, each operating at a rate $f_s/4$.

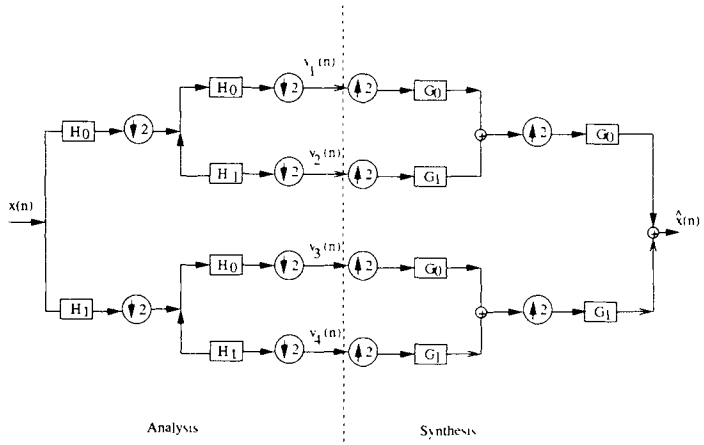


FIGURE 3.8 Four-band, analysis-synthesis tree structure.

For perfect reconstruction, eqs. (3.26) and (3.27) must be fulfilled. This two-level structure is equivalent to the four-band bank shown in Figure 3.9, if the following holds [2]:

$$H_0'(z) = H_0(z)H_0(z^2) \tag{3.32}$$

$$H_1'(z) = H_0(z)H_1(z^2) \tag{3.33}$$

$$H_2'(z) = H_1(z)H_0(z^2) \tag{3.34}$$

$$H_3'(z) = H_1(z)H_1(z^2) \tag{3.35}$$

The corresponding representation in the frequency domain given ideal bandpass filters is given in Figure 3.9b.

Almost all real-world signals such as speech concentrate most of their energy content in only subregions of their spectrum, indicating that only some spectral bands are significant, while the others are not. A logical consequence is that not all subbands of the regular binary tree are needed, and therefore some of the fine frequency resolution subbands can be combined to achieve larger bandwidth frequency bands. This leads to a new irregular tree structure having unequal bandwidths. Figure 3.10 displays an arbitrary irregular subband tree with a maximum tree level $L = 3$ and the corresponding frequency bands. It represents a dyadic band tree, and it

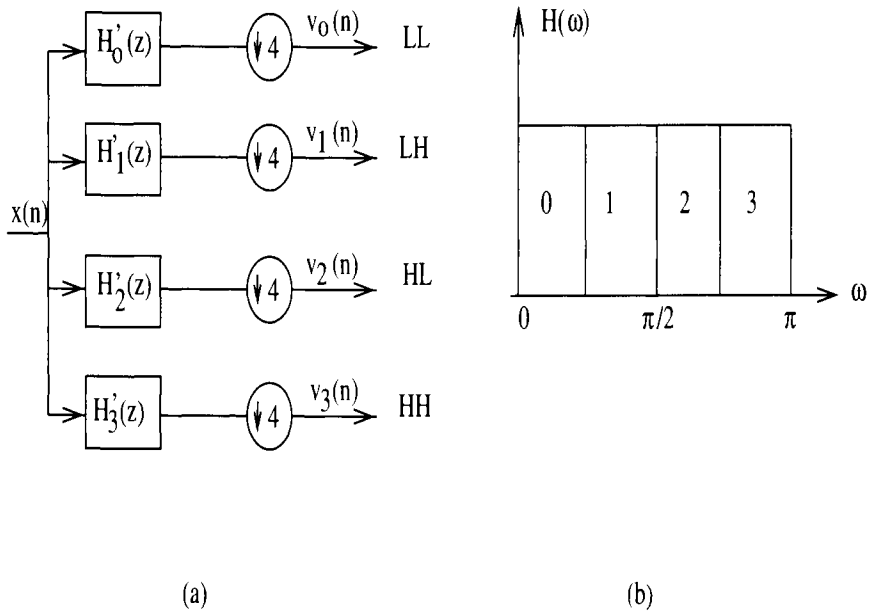


FIGURE 3.9 (a) Four-band equivalent to two-level regular binary tree and (b) frequency bands corresponding to the four-bands with ideal bandpass filters.

splits only the lower half of the spectrum at any level into two equal bands. The higher (detail) half-band component at any level is decomposed no further.

A closer look at the analysis-synthesis structure reveals that a half-resolution frequency step is used at any level. Based on this fact, this structure is also called the octave-band or constant-Q subband tree structure. As a first step, we need to obtain the low (L) and high (H) signal bands. It is relevant to note that band (L) gives a coarse representation of a signal while band (H) contains the detail information. Figure 3.11 shows the analysis-synthesis structure for a three-level dyadic tree. This structure is perfectly able to recover the original signal.

3.2.3 The Laplacian Pyramid for Signal Decomposition

The idea of multi-scale signal decomposition (coarse to fine) goes back to Burt and Adelson (1988) [30] and was first applied to tasks related to image coding [30]. The basic motivation of multiresolution processing or image pyramids is seen in context with the examination of small and large objects or of low- and high-contrast objects in an image. Then it becomes evident that studying them at different resolution, either high or low, is of great relevance.

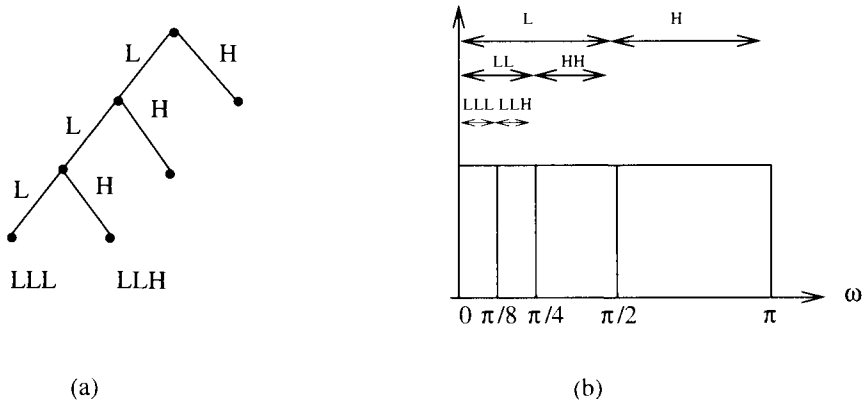


FIGURE 3.10 (a) A dyadic tree structure and the (b) corresponding frequency band split.

A pyramid represents a hierarchical data structure that contains successively compressed information, usually of an image [151]. Each level of such a pyramid produces a successively lower-resolution (more blurred) version of the given image. The detail information is given by the difference between the blurred representations of two neighboring levels. The image decoding process based on a pyramid reconstruction is achieved in a progressive manner: It starts with the coarsest approximation of the image and adds successively finer details at each image level until the original image is obtained. The Laplacian pyramid is nothing other than but a spectral or subband analysis accomplished by a dyadic tree [30].

The signal $x(n)$ in Figure 3.12 is first low-pass filtered and then undersampled by 2. This newly obtained signal is denoted by $x_D^1(n)$. Next we upsample this signal by 2 and interpolate it, such that we get $x_I^1(n)$. The approximation error represents the high-resolution detail and is given by

$$x_L^1(n) = x(n) - x_I^1(n) \tag{3.36}$$

where L means Laplacian since the approximation error in most cases has a Laplacian-shaped pdf. For perfect reconstruction, we need to add the detail signal to the interpolated low-pass signal:

$$x(n) = x_I^1(n) + x_L^1(n) \tag{3.37}$$

Since $x_I^1(n)$ is obtained from $x_D^1(n)$, we achieve a perfect reconstruction of $x(n)$ based only on $x_L^1(n)$ and $x_D^1(n)$. The data rate of $x_D^1(n)$ is half of the data rate of $x(n)$. Therefore, we obtain a coarse-to-fine signal decomposition in term of time and space. The downsampling and upsampling steps

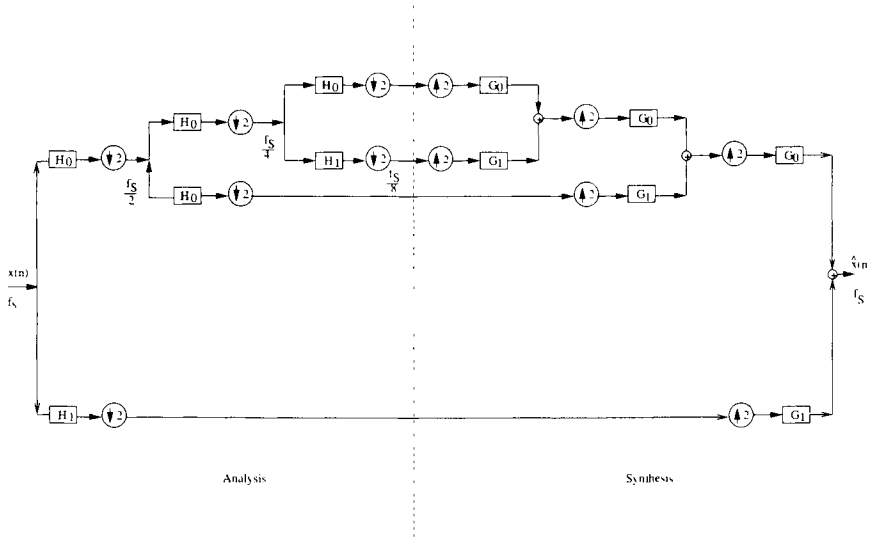


FIGURE 3.11 Dyadic wavelet filter bank.

are repeated on the low-pass content of the signal until the desired level L is reached. Figure 3.12 illustrates a Laplacian pyramid and its frequency resolution for $L = 3$ [2]. We can see that $x(n)$ is reconstructed from the low-pass signal and the detail signals $x_L^3(n)$, $x_L^2(n)$, and $x_L^1(n)$.

In general, the base of the pyramid contains the highest resolution, while the apex contains the lowest resolution. Moving upward, both size and resolution decrease. It is important to state that the Laplacian pyramid contains the prediction residuals needed to compute the Gaussian pyramid. The first-order statistics of the prediction residuals in the Laplacian pyramid are highly peaked around zero. This is different from their Gaussian counterparts, and therefore these images can be highly compressed by assigning fewer bits to the more probable values.

3.3 THE WAVELET TRANSFORM

3.3.1 Time-Frequency Representation

The discrete Fourier transform (DFT) analyzes a signal in terms of its frequency components by finding the signal's magnitude and phase spectra. Unfortunately, the DFT cannot find the times at which various frequency components occur within the window. In other words, the DFT fails to distinguish signals whose characteristics change with time, known as nonstationary signals, from those whose characteristics do not change with time, known as stationary signals. Since the DFT is windowed, only the

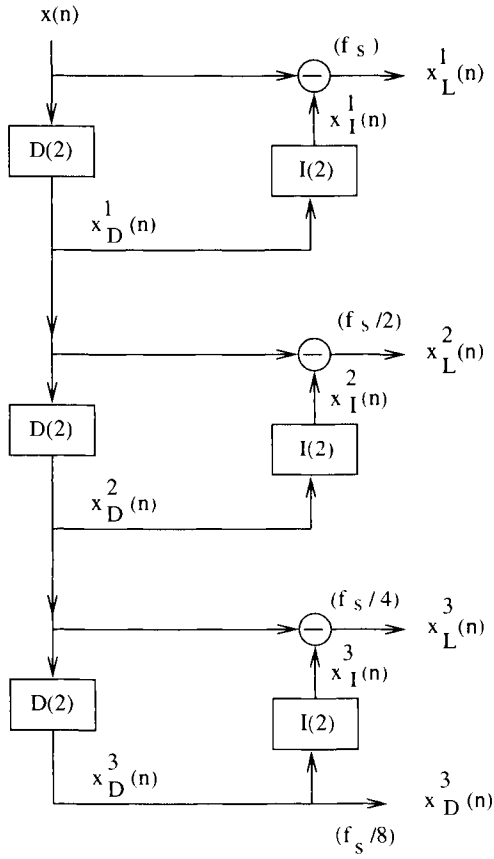


FIGURE 3.12 Laplacian pyramid representation [2].

signal's behavior within the window is important. For signals that are stationary within the window, the DFT provides an accurate picture of the frequency content since the same signal behavior persists for the length of the window. For signals with changing behavior the DFT can report the frequencies but not when they occur. Choosing a variable window length is a solution to this problem. But while shorter windows can improve the DFT's ability to analyze nonstationary signals, there are resolution implications. If the sampling rate remains unchanged, a shorter window has fewer points, which means that the DFT cannot provide enough detail about the signal spectrum. So a shorter window means good time resolution by providing very local detail, but at the same time poor frequency resolution, since the time to detect signal characteristics is too short. A solution is to choose a larger window, but this reduces the time resolution since the DFT cannot pinpoint changes in signal behavior within its window. Thus good

time and frequency resolution cannot be achieved at the same time. Table 3.2 illustrates this fact.

TABLE 3.2 Time and frequency resolution by window width.

Narrow window	Good time resolution	Poor frequency resolution
Wide window	Poor time resolution	Good frequency resolution

We can conclude that one of the most important tasks in signal analysis is to find a transform which represents the signal features simultaneously in time and frequency. Based on the standard Fourier analysis it is only possible to decompose a signal in its frequency components and to determine the corresponding amplitudes. Thus it is not possible to determine when a signal exhibits a particular frequency characteristics.

Therefore, the Fourier transform

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}dt \longleftrightarrow f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{j\omega t}d\omega \tag{3.38}$$

is not suitable for nonstationary signals. It sweeps over the whole time axis and does not detect local changes, such as high-frequency bursts. A solution is the short-time Fourier transform which was proposed by Gabor in 1946 [70] and does not have the above-mentioned disadvantages. This transform works by sweeping a short-time window over the time signal, and thus determines the frequency content in each considered time interval. This technique has been successfully applied to speech processing [202, 149, 212].

The short-time Fourier transform works by positioning a window $g(t)$ at some point τ on the time axis and determining the Fourier transform of the signal within this window:

$$F(\omega, \tau) = \int_{-\infty}^{\infty} f(t)g^*(t-\tau)e^{-j\omega t}dt \tag{3.39}$$

The basis functions of this transformation are produced based on the modulation and translation of the window function $g(t)$. ω and τ represent in this context the modulation and translation parameters. It can be shown that this is equivalent to filtering the signal $f(t)$ using a bank of filters, each centered at a different frequency but all of them having the same bandwidth. This is its drawback, because low- and high-frequency signal components are analyzed through the same window in time, resulting in poor overall localization of events. The solution to this is to find a long window to analyze slowly time-varying low-frequency components and a narrow window to detect high-frequency short-time activities. As we saw, this is offered by a tree-structured octave-band filter bank associated with the discrete time wavelet transform.

In summary, the major problem with the short-time Fourier transform is that the window $g(t)$ has a fixed time duration, and at the same time it also has a fixed frequency resolution. The product of frequency interval and time interval is a stable quantity. The Heisenberg Uncertainty Principle [186] gives in this respect a precise definition and also a lower bound for the product. The uncertainty principle states that for each transformation pair $g(t) \longleftrightarrow G(\omega)$ the relationship

$$\sigma_t \sigma_\omega \geq \frac{1}{2} \tag{3.40}$$

holds. The lower bound is given by the Gaussian function $f(t) = e^{-t^2}$. σ_T and σ_ω represent the squared variances of $g(t)$ and $G(\omega)$:

$$\sigma_T^2 = \frac{\int t^2 |g(t)|^2 dt}{\int |g(t)|^2 dt} \tag{3.41}$$

$$\sigma_\omega^2 = \frac{\int \omega^2 |G(\omega)|^2 d\omega}{\int |G(\omega)|^2 d\omega}$$

The window function $g(t)$ is defined as a prototype function. As τ increases, the prototype function is shifted on the time axis such that the window length remains unchanged. Figure 3.13 illustrates that each element σ_T and σ_ω of the resolution rectangle of the area $\sigma_T \sigma_\omega$ remains unchanged for each frequency ω and time-shift τ. The rectangles have the same form and area in the entire time-frequency plane.

Wavelet transforms are an alternative to the short-time Fourier transform. Their most important feature is that they analyze different frequency components of a signal with different resolutions. In other words, they address exactly the concern raised in connection with the short-time Fourier transform. To implement different resolutions at different frequencies requires the notion of functions at different scales. Like scales on a map, small scales show fine details while large scales show coarse features only. A scaled version of a function $\psi(t)$ is the function $\psi(t/a)$, for any scale a. When $a > 1$, a function of lower frequency is obtained that is able to describe slowly changing signals. When $a < 1$ a function of higher frequency is obtained that can detect fast signal changes. It is important to note that the scale is inversely proportional to the frequency.

Wavelet functions are localized in frequency in the same way sinusoids are, but they differ from sinusoids by being localized in time, as well. There are several wavelet families, each having a characteristic shape, and the basic scale for each family covers a known, fixed interval of time. The time spans of the other wavelets in the family widen for larger scales and narrow for smaller scales. Thus, wavelet functions can offer either good time resolution or good frequency resolution: Good time resolution is asso-

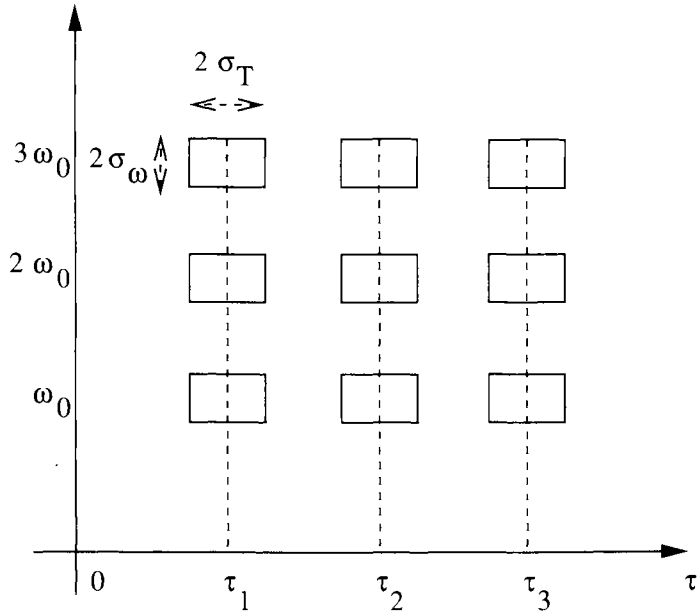


FIGURE 3.13 Short-time Fourier transform: time-frequency space and resolution cells.

ciated with narrow, small-scale windows, while good frequency resolution is associated with wide, large-scale windows.

To determine what frequencies are present in a signal and when they occur, the wavelet functions at each scale must be translated through the signal, to enable comparison with the signal in different time intervals. A scaled and translated version of the wavelet function $\psi(t)$ is the function $\psi(\frac{t-b}{a})$, for any scale a and translation b. A wavelet function similar to the signal in frequency produces a large wavelet transform. If the wavelet function is dissimilar to the signal, a small transform will arise. A signal can be coded using these wavelets if it can be decomposed into scaled and translated copies of the basic wavelet function. The widest wavelet responds to the slowest signal variations and thus describes the coarsest features in the signal. Smaller scale wavelets respond best to high frequencies in the signal and detect rapid signal changes, thus providing detailed information about this signal. In summary, smaller scales correspond to higher frequencies, and larger scales to lower frequencies. A signal is coded through the wavelet transform by comparing the signal to many scalings and translations of a wavelet function.

The wavelet transform (WT) is produced by a translation and dilation of a so-called prototype function ψ. Figure 3.14 illustrates a typical wavelet

and its scalings. The bandpass characteristics of ψ and the time-frequency resolution of the WT can be easily detected.

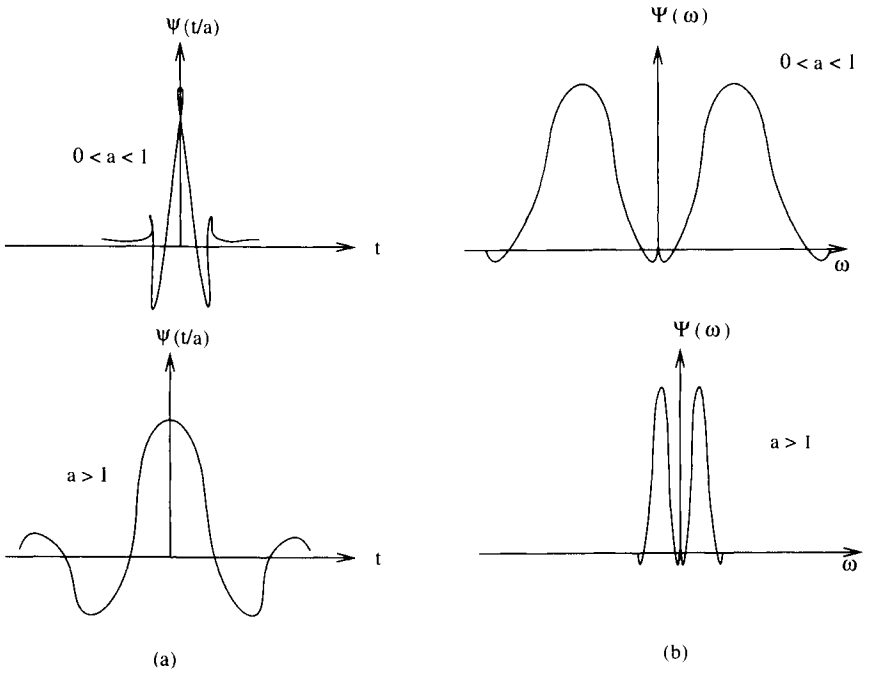


FIGURE 3.14 Wavelet in time and frequency domain: (a) scale parameter 0¡a¡1, (b) a¿1.

The foundation of the WT is based on the scaling property of the Fourier transform. If

$$\psi(t) \longleftrightarrow \Psi(\omega) \tag{3.42}$$

represents a Fourier transform pair, then we have

$$\frac{1}{\sqrt{a}}\Psi\left(\frac{t}{a}\right) \longleftrightarrow \sqrt{a}\Psi(a\omega) \tag{3.43}$$

with $a > 0$ being a continuous variable. A contraction in the time-domain produces an expansion in the frequency domain, and vice-versa. Figure 3.15 illustrates the corresponding resolution cells in the time-frequency domain. The figure makes visual the underlying property of wavelets: they are localized in both time and frequency. While the functions $e^{j\omega t}$ are perfectly localized at ω they extend over all time: wavelets on the other hand, that are not at a single frequency are limited to finite time. As we rescale, the frequency goes up by a certain quantity and at the same time the time

interval goes down by the same quantity. Thus the uncertainty principle holds.

A wavelet can be defined by the scale and shift parameters a and b,

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \tag{3.44}$$

while the WT is given by the inner product

$$W(a,b) = \int_{-\infty}^{\infty} \psi_{ab}(t) f^*(t) dt = < \psi_{ab}, f > \tag{3.45}$$

with $a \in R^+, b \in R$.

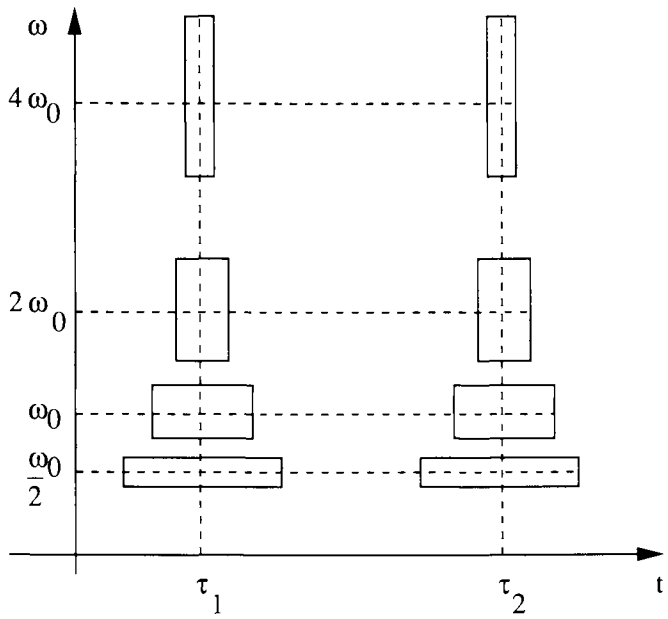


FIGURE 3.15 Wavelet transform: time-frequency domain and resolution cells.

The WT defines a $L^2(R) \rightarrow L^2(R^2)$ mapping which has a better time-frequency localization than the short-time Fourier transform.

In the following, we will describe the continuous wavelet transform (CWT) and show an admissibility condition which is necessary to ensure the inversion of the WT. Also, we will define the discrete wavelet transform (DWT) which is generated by sampling the wavelet parameters (a, b) on a grid or lattice. The quality of the reconstructed signals based on the transform values depends on the coarseness of the sampling grid. A finer sampling grid leads to more accurate signal reconstruction at the cost of

redundancy, a coarse sampling grid is associated with loss of information. To address these important issues, the concept of frames is now presented.

3.3.2 The Continuous Wavelet Transform

The CWT transforms a continuous function into a highly redundant function of two continuous variables, translation and scale. The resulting transformation is important for time-frequency analysis and is easy to interpret.

The CWT is defined as the mapping of the function $f(t)$ on the time–scale space by

$$W_f(a,b) = \int_{-\infty}^{\infty} \psi_{ab}(t) f(t) dt = < \psi_{ab}(t), f(t) > \tag{3.46}$$

The CWT is invertible if and only if the resolution of identity holds:

$$f(t) = \underbrace{\frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{0}^{\infty} \frac{dadb}{a^2}}_{\text{summation}} \underbrace{W_f(a,b)}_{\text{Wavelet coefficients}} \underbrace{\psi_{ab}(t)}_{\text{Wavelet}} \tag{3.47}$$

where

$$C_\upsilon = \int_{o}^{\infty} \frac{|\Psi(\omega)|^2}{\omega} d\omega \tag{3.48}$$

assuming that a real-valued $\upsilon(t)$ fulfills the admissibility condition. If $C_\psi < \infty$, then the wavelet is called admissible. Then we get for the DC gain

$$\Psi(0) = \int_{-\infty}^{\infty} \upsilon(t) dt = 0 \tag{3.49}$$

We immediately see that $\psi(t)$ corresponds to the impulse response of a bandpass filter and has a decay rate of $|t|^{1-\epsilon}$. It is important to note that based on the admissibility condition, it can be shown that the CWT is complete if $W_f(a,b)$ is known for all a, b.

The Mexican-hat wavelet

$$\psi(t) = (\frac{2}{\sqrt{3}} \pi^{-\frac{1}{4}})(1-t^2) e^{-\frac{t^2}{2}} \tag{3.50}$$

is visualized in Figure 3.16. It has a distinctive symmetric shape, and it has an average value of zero and dies out rapidly as $|t| \to \infty$. There is no scaling function associated with the Mexican hat wavelet.

Figure 3.17 illustrates the multi-scale coefficients describing a spiculated mass. Figure 3.17a shows the scanline through a mammographic image with a mass (8 mm) while Figure 3.17b visualizes the multi-scale coefficients at various levels.

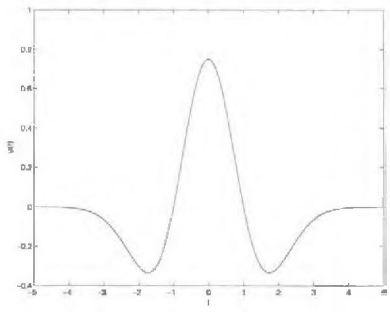
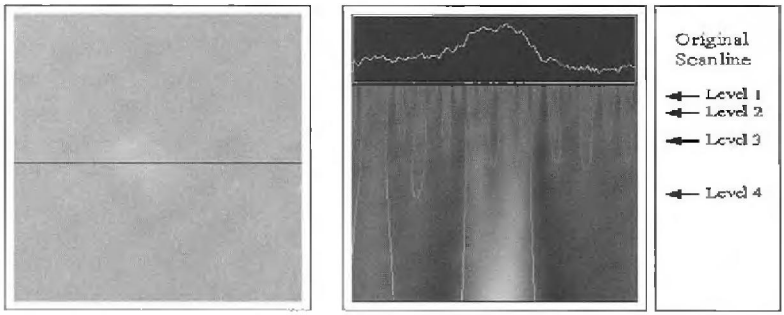


FIGURE 3.16 Mexican hat wavelet.



(a) (b)

FIGURE 3.17 Continuous wavelet transform: (a) scan line and (b) multi-scale coefficients. (Images courtesy of Dr. A. Laine, Columbia University.)

The short-time Fourier transform finds a decomposition of a signal into a set of equal-bandwidth functions across the frequency spectrum. The WT provides a decomposition of a signal based on a set of bandpass functions that are placed over the entire spectrum. The WT can be seen as a signal decomposition based on a set of constant-Q bandpasses. In other words, we have an octave decomposition, logarithmic decomposition, or constant-Q decomposition on the frequency scale. The bandwidth of each of the filters in the bank is the same in a logarithmic scale, or equivalently, the ratio of the filters' bandwidth to the respective central frequency is constant.

3.4 THE DISCRETE WAVELET TRANSFORMATION

The CWT has two major drawbacks: redundancy and lack of practical relevance. The first is based on the nature of the WT; the latter is because the transformation parameters are continuous. A solution to these prob-

lems can be achieved by sampling both parameters (a, b) such that a set of wavelet functions in the form of discrete parameters is obtained. We also have to look into the following problems:

1. Is the set of discrete wavelets complete in $L^2(R)$?
2. If complete, is the set at the same time also redundant?
3. If complete, then how coarse must the sampling grid be, such that the set is minimal or nonredundant?

A response to these questions will be given in this section, and we also will show that the most compact set is the orthonormal wavelet set.

The sampling grid is defined as follows [2]:

$$a = a_0^m b = nb_0 a_0^m \tag{3.51}$$

where

$$\psi_{mn}(t) = a^{-m/2}\psi(a_0^{-m}t - nb_0) \tag{3.52}$$

with $m, n \in Z$. If we consider this set complete in $L^2(R)$ for a given choice of $\psi(t), a, b$, then $\{\psi_{mn}\}$ is an affine wavelet. $f(t) \in L^2(R)$ represents a wavelet synthesis. It combines the components of a signal together again to reproduce the original signal $f(t)$. If we have a wavelet basis, we can determine a wavelet series expansion. Thus, any square-integrable (finite energy) function $f(t)$ can be expanded in wavelets:

$$f(t) = \sum_m \sum_n d_{m,n}\psi_{mn}(t) \tag{3.53}$$

The wavelet coefficient $d_{m,n}$ can be expressed as the inner product

$$d_{m,n} = < f(t), \psi_{mn}(t) > = \frac{1}{a_0^{m/2}} \int f(t)\psi(a_0^{-m}t - nb_0)dt \tag{3.54}$$

These complete sets are called frames. An analysis frame is a set of vectors ψ_{mn} such that

$$A||f||^2 \leq \sum_m \sum_n | < f, \psi_{mn} > |^2 \leq B||f||^2 \tag{3.55}$$

with

$$||f||^2 \triangleq \int |f(t)|^2 dt \tag{3.56}$$

$A, B > 0$ are the frame bounds. A tight, exact frame that has $A = B = 1$ represents an orthonormal basis for $L^2(R)$. A notable characteristic of orthonormal wavelets $\{\psi_{mn}(t)\}$ is

$$\int \psi_{mn}(t)\psi_{m'n'}(t)dt = \begin{cases} 1, & m = m', n = n' \\ 0, & \text{else} \end{cases} \tag{3.57}$$

They are additionally orthonormal in both indices. This means that for the same scale m they are orthonormal both in time and across the scales.

For the scaling functions the orthonormal condition holds only for a given scale

$$\int \phi_{mn}(t)\phi_{ml}(t)dt = \delta_{n-l} \tag{3.58}$$

The scaling function can be visualized as a low pass filter. While scaling functions alone can code a signal to any desired degree of accuracy, efficiency can be gained by using the wavelet functions. Any signal $f \in L^2(R)$ at the scale m can be approximated by its projections on the scale space.

The similarity between ordinary convolution and the analysis equations suggests that the scaling function coefficients and the wavelet function coefficients may be viewed as impulse responses of filters, as shown in Figure 3.18. The convolution of $f(t)$ with $\psi_m(t)$ is given by

$$y_m(t) = \int f(\tau)\psi_m(\tau - t)d\tau \tag{3.59}$$

where

$$\psi_m(t) = 2^{-m/2}\psi(2^{-m}t) \tag{3.60}$$

Sampling $y_m(t)$ at $n2^m$ yields

$$y_m(n2^m) = 2^{-m/2}\int f(\tau)\psi(2^{-m}\tau - n)d\tau = d_{m,n} \tag{3.61}$$

Whereas in the filter bank representation of the short-time Fourier transform all subsamplers are identical, the subsamplers of the filter bank corresponding to the wavelet transform are dependent on position or scale.

The DWT dyadic sampling grid in Figure 3.19 visualizes this aspect. Every single point represents a wavelet basis function $\psi_{mn}(t)$ at the scale 2^{-m} and shifted by $n2^{-m}$.

3.5 MULTISCALE SIGNAL DECOMPOSITION

The goal of this section is to highlight an important aspect of the wavelet transform that accounts for its success as a method in pattern recognition: the decomposition of the whole function space into subspaces. That implies that there is a piece of the function $f(t)$ in each subspace. Those pieces (or projections) give finer and finer details of $f(t)$. For audio signals, these scales are essentially octaves. They represent higher and higher frequencies.

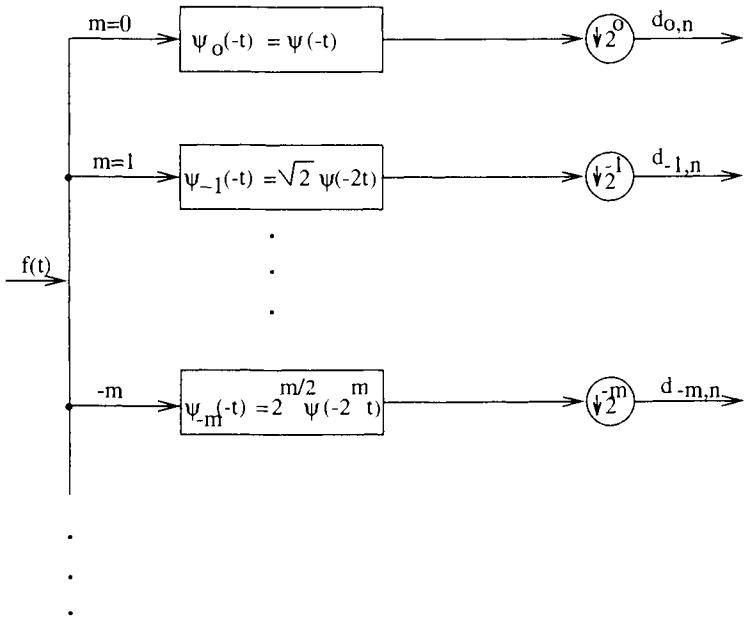


FIGURE 3.18 Filter bank representation of DWT.

For images and all other signals, the simultaneous appearance of multiple scales is known as multiresolution.

Mallat's and Meyer's method [151] for signal decomposition based on orthonormal wavelets with compact carrier is going to be reviewed here. We will establish a link between these wavelet families and the hierarchic filter banks. In the last part of this section, we will show that the FIR PR-QMF hold the regularization property, and produce orthonormal wavelet bases.

3.5.1 Multiscale-Analysis Spaces

Multiscale signal analysis provides the key to the link between wavelets and pyramidal dyadic trees. A wavelet family is used to decompose a signal into scaled and translated copies of a basic function. As stated before, the wavelet family consists of scaling and wavelet functions. Scaling functions $\phi(t)$ alone are adequate to code a signal completely, but a decomposition based on both scaling and wavelet functions is most efficient.

In mathematical terminology, a function $f(t)$ in the whole space has a piece in each subspace. Those pieces contain more and more of the full information in $f(t)$. These successive approximations converge to a limit which represents the function $f \in L^2$. At the same time they describe different resolution levels, as is known from the pyramidal representation.

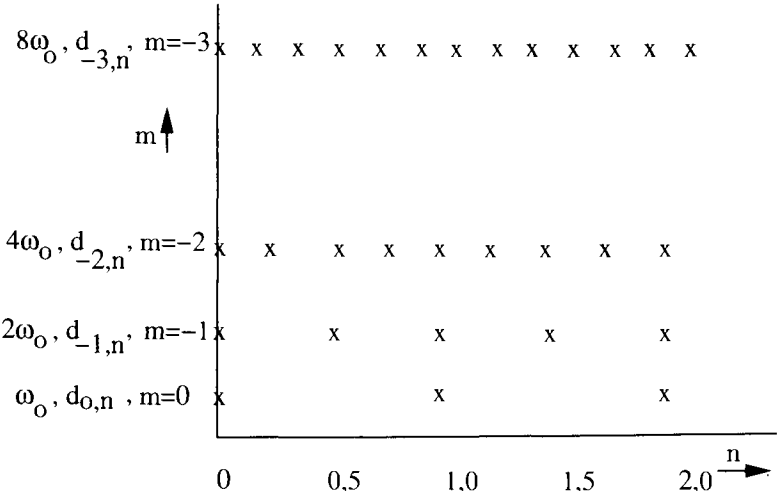


FIGURE 3.19 Dyadic sampling grid for the DWT.

A multiscale analysis is based on a sequence of subspaces $\{V_m | m \in Z\}$ in $L^2(R)$ satisfying the following requirements:

- Inclusion: Each subspace V_j is contained in the next subspace. A function $f \in L^2(R)$ in one subspace is in all the higher (finer) subspaces:

$$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots \qquad (3.62)$$
$$\leftarrow coarser \qquad finer \rightarrow$$

- Completeness: A function in the whole space has a part in each subspace

$$\bigcap_{m \in Z} V_m = 0 \qquad \bigcup_{m \in Z} V_m = L^2(R) \qquad (3.63)$$

- Scale invariance:

$$f(x) \in V_m \Longleftrightarrow f(2x) \in V_{m-1} \quad \text{for any function} \quad f \in L^2(R) \qquad (3.64)$$

- Basis-frame property: This requirement for multiresolution concerns a basis for each space V_j. There is a scaling function $\phi(t) \in V_0$, such that $\forall m \in Z$, the set

$$\{\phi_{mn}(t) = 2^{-m/2}\phi(2^{-m}t - n)\} \qquad (3.65)$$

forms an orthonormal basis for V_m:

$$\int \phi_{mn}(t)\phi_{mn'}(t)dt = \delta_{n-n'} \tag{3.66}$$

In the following, we will review mathematically the multiresolution concept based on scaling and wavelet functions, and thus define the approximation and detail operators.

Let $\phi_{mn}(t)$ with $m \in Z$ be defined as

$$\{\phi_{mn}(t) = 2^{-m/2}\phi(2^{-m}t - n)\} \tag{3.67}$$

Then the approximation operator P_m on functions $f(t) \in L^2(R)$ is defined by

$$P_m f(t) = \sum_n < f, \phi_{mn} > \phi_{mn}(t) \tag{3.68}$$

and the detail operator Q_m on functions $f(t) \in L^2(R)$ is defined by

$$Q_m f(t) = P_{m-1} f(t) - P_m f(t) \tag{3.69}$$

It can be easily shown that $\forall m \in Z, \{\phi_{mn}(t)\}$ is an orthonormal basis for V_m [237], and that for all functions $f(t) \in L^2(R)$

$$\lim_{m \to -\infty} ||P_m f(t) - f(t)||_2 = 0 \tag{3.70}$$

and

$$\lim_{m \to \infty} ||P_m f(t)||_2 = 0 \tag{3.71}$$

An important feature of every scaling function $\phi(t)$ is that it can be built from translations of double-frequency copies of itself, $\phi(2t)$, according to

$$\phi(t) = 2\sum_n h_0(n)\phi(2t - n) \tag{3.72}$$

This equation is called a multiresolution analysis equation. Since $\phi(t) = \phi_{00}(t)$, both m, n can be set to 0 to obtain the above simpler expression. The equation expresses the fact that each scaling function in a wavelet family can be expressed as a weighted sum of scaling functions at the next finer scale. The set of coefficients $\{h_0(n)\}$ are called the scaling function coefficients and behave as a lowpass filter.

Wavelet functions can also be built from translations of $\phi(2t)$:

$$\psi(t) = 2\sum_n h_1(n)\phi(2t - n) \tag{3.73}$$

This equation is called the fundamental wavelet equation. The set of coefficients $\{h_1(n)\}$ are called the wavelet function coefficients and behave as a highpass filter. This equation expresses the fact that each wavelet function in a wavelet family can be written as a weighted sum of scaling functions at the next finer scale.

The following theorem provides an algorithm for constructing a wavelet orthonormal basis given a multiscale analysis.

THEOREM 3.5.1:

Let $\{V_m\}$ be a multiscale analysis with scaling function $\phi(t)$ and scaling filter $h_0(n)$.

Define the wavelet filter $h_1(n)$ by

$$h_1(n) = (-1)^{n+1} h_0(N-1-n) \tag{3.74}$$

and the wavelet $\psi(t)$ by eq. (3.73).

Then

$$\{\psi_{mn}(t)\} \tag{3.75}$$

is a wavelet orthonormal basis on R.

Alternatively, given any $L \in Z$,

$$\{\phi_{Ln}(t)\}_{n\in Z} \bigcup \{\psi_{mn}(t)\}_{m,n\in Z} \tag{3.76}$$

is an orthonormal basis on R.

The proof can be found in [237]. Some very important facts representing the key statements of multiresolution follow:

(a) $\{\psi_{mn}(t)\}$ is an orthonormal basis for W_m.

(b) If $m \neq m'$ then $W_m \perp W_{m'}$.

(c) $\forall m \in Z$, $V_m \perp W_m$ where W_m is the orthogonal complement of V_m in V_{m-1}.

(d) $\forall m \in Z$, $V_{m-1} = V_m \oplus W_m$. $\oplus$ stands for orthogonal sum. This means that the two subspaces are orthogonal and that every function in V_{m-1} is a sum of functions in V_m and W_m. Thus every function $f(t) \in V_{m-1}$ is composed of two subfunctions $f_1(t) \in V_m$ and $f_2(t) \in W_m$ such that $f(t) = f_1(t) + f_2(t)$ and $< f_1(t), f_2(t) >= 0$. The most important part of multiresolution is that the spaces W_m represent the differences between the spaces V_m, while the spaces V_m are the sums of the W_m.

(e) Every function $f(t) \in L^2(R)$ can be expressed as

$$f(t) = \sum_m f_m(t) \tag{3.77}$$

where $f_m(t) \in W_m$ and $< f_m(t), f_{m'} >= 0$. This can be usually written as

$$\cdots \oplus W_j \oplus W_{j-1} \cdots \oplus W_0 \cdots \oplus W_{-j+1} \oplus W_{-j+2} \cdots = L^2(R) \quad (3.78)$$

Although scaling functions alone can code a signal to any desired degree of accuracy, efficiency can be gained by using the wavelet functions. This leads to the following new understanding of the concept of multiresolution. Multiresolution can be described based on wavelet W_j and scaling subspaces V_j. This means that the subspace formed by the wavelet functions covers the difference between the subspaces covered by the scaling functions at two adjacent scales.

3.5.2 A Very Simple Wavelet: The Haar Wavelet

The Haar wavelet is one of the simplest and oldest known orthonormal wavelets. However, it has didactic value because it helps to visualize the multiresolution concept.

Let V_m be the space of piecewise constant functions

$$V_m = \{f(t) \in L^2(R); \quad \text{f is constant in} \quad [2^m n, 2^m(n+1)] \quad \forall n \in Z\} \quad (3.79)$$

Figure 3.20 illustrates such a function.

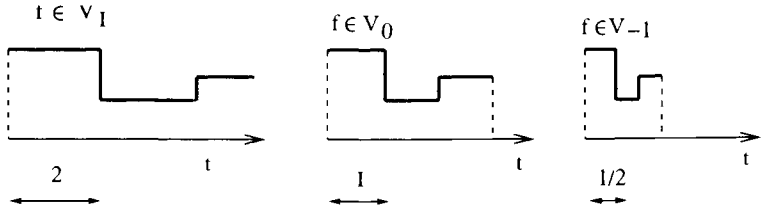


FIGURE 3.20 Piecewise constant functions in V_1, V_2 and V_{-1}.

We can easily see that $\cdots V_1 \subset V_0 \subset V_{-1} \cdots$ and $f(t) \in V_0 \longleftrightarrow f(2t) \in V_{-1}$ and that the inclusion property is fulfilled. The function $f(2t)$ has the same shape as $f(t)$ but is compressed to half the width.

The scaling function of the Haar wavelet $\phi(t)$ is given by

$$\phi(t) = \begin{cases} 1, & 0 \le t \le 1 \\ 0, & \text{else} \end{cases} \quad (3.80)$$

and defines an orthonormal basis for V_0. Since for $n \neq m$, $\phi(t-n)$ and $\phi(t-m)$ do not overlap, we obtain

$$\int \phi(t-n)\phi(t-m)dt = \delta_{n-m} \tag{3.81}$$

The Fourier transform of the scaling function yields

$$\Phi(\omega) = e^{-j\frac{\omega}{2}} \frac{\sin \omega/2}{\omega/2} \tag{3.82}$$

Figure 3.21 shows that $\phi(t)$ can be written as the linear combination of even and odd translations of $\phi(2t)$

$$\phi(t) = \phi(2t) + \phi(2t-1) \tag{3.83}$$

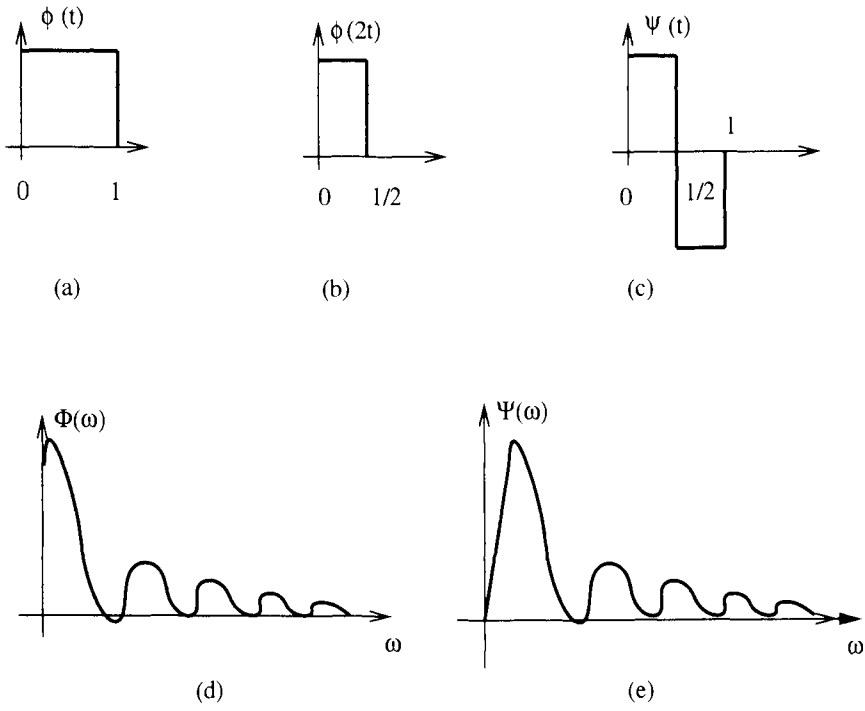


FIGURE 3.21 (a) and (b) Haar basis functions; (c) Haar wavelet; (d) Fourier transform of the scaling function; and (e) Haar wavelet function.

Since $V_{-1} = V_0 \oplus W_0$ and $Q_0 f = (P_{-1} f - P_0 f) \in W_0$ represent the details from scale 0 to -1, it is easy to see that $\psi(t-n)$ spans W_0. The Haar mother wavelet function is given by

$$\psi(t) = \phi(2t) - \phi(2t-1) = \begin{cases} 1, & 0 \le t < 1/2 \\ -1, & 1/2 \le t < 1 \\ 0, & \text{else} \end{cases} \tag{3.84}$$

The Haar wavelet function is an up-down square wave and can be described by a half-box minus a shifted half-box. We also can see that the wavelet function can be computed directly from the scaling functions. In the Fourier domain it describes a bandpass, as can be easily seen from Figure 3.21e. This is given by

$$\Psi(\omega) = je^{-j\frac{\omega}{2}} \frac{\sin^2 \omega/4}{\omega/4} \tag{3.85}$$

We can easily show that

$$\phi_{m+1,n} = \frac{1}{\sqrt{2}}[\phi_{m,2n} + \phi_{m,2n+1}] \tag{3.86}$$

and

$$\psi_{m+1,n} = \frac{1}{\sqrt{2}}[\phi_{m,2n} - \phi_{m,2n+1}] \tag{3.87}$$

Figure 3.22 illustrates a typical Haar wavelet for the scales 0 and 1. Figure 3.23 shows the approximations $P_0 f$, $P_{-1} f$ and the detail $Q_0 f$ for a function f. As stated in context with multiresolution. the detail $Q_0 f$ is added to the coarser approximation $P_0 f$ in order to obtain the finer approximation $P_{-1} f$.

The scaling function coefficients for the Haar wavelet at scale m are given by

$$c_{m,n} = < f, \phi_{mn} > = 2^{-m/2} \int_{2^m n}^{2^m (n+1)} f(t) dt \tag{3.88}$$

This yields an approximation of f at scale m

$$P_m f = \sum_n c_{m,n} \phi_{mn}(t) = \sum_n c_{m,n} 2^{-m/2} \phi(2^{-m} t - n) \tag{3.89}$$

In spite of their simplicity, the Haar wavelets exhibit some undesirable properties, which pose a difficulty in many practical applications. Other wavelet families such as Daubechies wavelets and Coiflet basis [2, 237] are more attractive in practice. Daubechies wavelets are quite often used in image compression. The scaling function coefficients $h_0(n)$ and the wavelet function coefficients $h_1(n)$ for the Daubechies-4 family are nearly impossible to determine. They were obtained based on iterative methods [29].

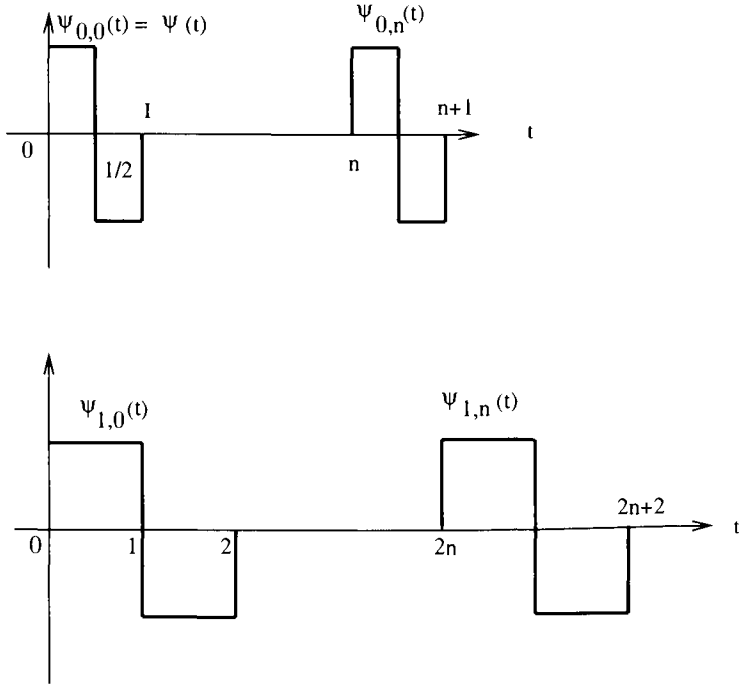


FIGURE 3.22 Typical Haar wavelet for the scales 0,1.

3.5.3 Analogy between Filter Banks and Wavelet Bases

In this section we will show the main approach to wavelets through two-channel filter banks. We will point out the connections between wavelets and filters. Historically, their development was separate; however, they are very closely related. We remember that the lowpass filter coefficients determine the scaling function while the highpass coefficients produce the wavelets.

The underlying problem here is that we again have an analysis and synthesis filter bank. The decimated lowpass and highpass filter can be expressed in one matrix. This matrix represents the whole analysis bank. It executes the lowpass and highpass channel (both decimated). There is also a synthesis bank. When the analysis and synthesis banks are transposes as well as inverses, we have an orthogonal filter bank. When they are inverses but not necessarily transposes, then we deal with a biorthogonal filter bank.

When the filter bank is orthonormal in discrete time, we hope for orthogonal basis functions in continuous time. All wavelets should be orthogonal to the scaling functions. Furthermore, the wavelets and the scaling functions should be mutually orthogonal.

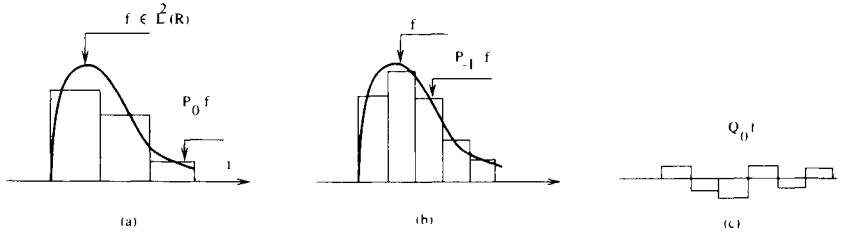


FIGURE 3.23 Approximation of (a) P_0f, (b) $P_{-1}f$ and (c) the detail signal Q_0f, with $P_0f+Q_0f=P_{-1}f$.

Our starting point is the dilation equation given by (3.72). The Fourier transform of the dilation equation in (3.72) considering $h_0(n) \leftrightarrow H_1(e^{j\omega})$ and $\phi(t) \leftrightarrow \Phi(\omega)$ is given by

$$\Phi(\omega) = H(\frac{\omega}{2})\Phi(\frac{\omega}{2}) = \Phi(0)\prod_{k=1}^{\infty} H_0(e^{j\omega/2^k}) \tag{3.90}$$

where, based on completeness and orthonormality property, we have

$$|\Phi(0)| = |\int_{-\infty}^{\infty} \phi(t)dt| = |H_0(e^{j\omega})|_{\omega=0} = 1 \tag{3.91}$$

Next we will establish the orthogonality conditions in the frequency domain. In wavelet terminology, this translates to finding for a given scaling function $\phi(t)$ and scale m the orthonormal set $\{\phi_{mn}\}$. If $\{\phi(t-n)\}$ spans V_0, then $\Phi(\omega)$ and H fulfill in the frequency domain

$$\sum_k |\Phi(\omega + 2\pi k)|^2 = 1 \tag{3.92}$$

and

$$|H_0(e^{j\omega})|^2 + |H_0(e^{j(\omega+\pi)}|^2 = 1 \tag{3.93}$$

The last equation gives the condition on the frequency response $H_0(e^{j\omega})$ to produce an orthonormal filter bank.

Similarly, we assume $h_1(n) \leftrightarrow H_1(e^{j\omega})$, and with eq. (3.90), we obtain for the Fourier transform of the wavelet equation (3.73)

$$\Psi(\omega) = H_1(e^{j\omega/2})\Phi\left(\frac{\omega}{2}\right) = H_1(e^{j\omega/2})\prod_{k=2}^{\infty} H_0(e^{j\omega/2^k}) \tag{3.94}$$

The orthonormal wavelet bases are complementary to the scaling bases. They fulfill the intra- and interscale orthonormalities

$$< \psi_{mn}(t), \psi_{kl}(t) >= \delta_{m-k}\delta_{n-l} \tag{3.95}$$

with m and k being scale parameters, and n and l being translation parameters. Because the wavelets are orthonormal at all scales, this implies for the frequency domain

$$\sum_k |\Psi(\omega + 2\pi k)|^2 = 1 \tag{3.96}$$

Considering $\Psi(2\omega) = H_1(e^{j\omega})\Phi(\omega)$, we immediately obtain from the above equation

$$|H_1(e^{j\omega})|^2 + |H_1(e^{j(\omega+\pi)})|^2 = 1 \tag{3.97}$$

or

$$H_1(z)H_1(-z) + H_1(-z)H_1(-z^{-1}) = 1 \tag{3.98}$$

The scaling functions are orthogonal to the wavelets

$$< \phi_{mn}(t), \psi_{kl}(t) >= 0 \tag{3.99}$$

This implies for the frequency domain

$$\sum_k \Phi(\omega - 2\pi k)\Psi^*(\omega - 2\pi k) = 0 \tag{3.100}$$

Considering eqs. (3.90), (3.94), and (3.98), we get for the frequency domain

$$H_0(e^{j\omega})H_1(e^{-j\omega}) + H_0(e^{j(\omega+\pi)})H_1(e^{-j(\omega+\pi)}) = 0 \tag{3.101}$$

or equivalently for the z-domain

$$H_0(z)H_1(-z) + H_0(-z)H_1(-z^{-1}) = 0 \tag{3.102}$$

For proving the analogy between filter banks and wavelet bases, we will introduce the paraunitary filter bank. The two-band paraunitary analysis filters must obey

$$\begin{bmatrix} H_0(z^{-1}) & H_0(-z^{-1}) \\ H_1(z^{-1}) & H_1(-z^{-1}) \end{bmatrix} \begin{bmatrix} H_0(z) & H_1(z) \\ H_0(-z) & H_1(-z) \end{bmatrix} = 2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.103}$$

The conditions stated in eqs. (3.93), (3.98), and (3.101) are sufficient for a paraunitary two-band FIR perfect reconstruction QMF as given in Table 3.1.

Equation (3.101) holds if

$$H_1(z) = z^{-(N-1)}H_0(-z^{-1}), \quad \text{N even} \tag{3.104}$$

or equivalently for the time domain we get

$$h_1(n) = (-1)^{n+1}h_0(N-1-n) \tag{3.105}$$

Thus, we have shown that compactly supported orthonormal wavelet bases imply FIR paraunitary QMF filter banks. In [2], it was shown that the converse is also true if $H_0(z)$ and $H_1(z)$ have at least one zero at $z = -1$ and $z = 1$.

We thus have shown the connection between the wavelet theory and filter banks.

The mathematical properties of orthonormal wavelets with compact carriers are summarized in Table 3.3 [2].

TABLE 3.3 Properties of orthonormal wavelets.

$$\begin{aligned}
\Phi(\omega) &= \prod_{k=1}^{\infty} H_0(e^{j\omega/2^k}) \\
\Psi(\omega) &= H_1(e^{j\omega/2}) \prod_{k=2}^{\infty} H_0(e^{j\omega/2^k}) \\
H_1(z) &= z^{-(N-1)}H_0(-z^{-1}) \\
\phi(t) &= \sum h_0(n)\phi(2t-n) \\
\psi(t) &= \sum h_1(n)\psi(2t-n) \\
h_1(n) &= (-1)^{n+1}h_0(N-1-n) \\
< \psi_{mn}(t), \psi_{kl}(t) > &= \delta_{m-k}\delta_{n-l} \\
< \phi_{mn}(t), \phi_{mn'}(t) > &= \delta_{n-n'} \\
< \phi_{mn}(t), \psi_{kl}(t) > &= 0
\end{aligned} \tag{3.106}$$

3.5.4 Multiscale Signal Decomposition and Reconstruction

In this section we will illustrate multiscale pyramid decomposition. Based on a wavelet family, a signal can be decomposed into scaled and translated copies of a basic function. As discussed in the last sections, a wavelet family consists of scaling functions, which are scalings and translations of a father wavelet, and wavelet functions, which are scalings and translations of a mother wavelet. We will show an efficient signal coding that uses scaling and wavelet functions at two successive scales. In other words, we give a recursive algorithm which supports the computation of wavelet coefficients of a function $f(t) \in L^2(R)$.

Assume we have a signal or a sequence of data $\{c_0(n)|n \in Z\}$, and $c_0(n)$ is the nth scaling coefficient for a given function $f(t)$:

$$c_{0,n} =< f, \phi_{0n} > \tag{3.107}$$

for each $n \in Z$. This assumption makes the recursive algorithm work.

The decomposition and reconstruction algorithm is given by the following theorem [237].

THEOREM 3.5.2:

Let $\{V_k\}$ be a multiscale analysis with associated scaling function $\phi(t)$ and scaling filter $h_0(n)$. The wavelet filter $h_1(n)$ is defined by eq. (3.73), and the wavelet function is defined by eq. (3.74).

Given a function $f(t) \in L^2(R)$, define for $n \in Z$

$$c_{0,n} =< f, \phi_{0n} > \tag{3.108}$$

and for every $m \in N$ and $n \in Z$,

$$c_{m,n} =< f, \phi_{mn} > \quad and \quad d_{m,n} =< f, \psi_{mn} > \tag{3.109}$$

Then, the decomposition algorithm is given by

$$c_{m+1,n} = \sqrt{2}\sum_k c_{m,k}h_0(k-2n) \quad d_{m+1,n} = \sqrt{2}\sum_k d_{m,k}h_1(k-2n) \tag{3.110}$$

and the reconstruction algorithm is given by

$$c_{m,n} = \sqrt{2}\sum_k c_{m+1,n}h_0(n-2k) + \sqrt{2}\sum_k d_{m+1,n}h_1(n-2k) \tag{3.111}$$

From eq. (3.110) we obtain for $m = 1$ at resolution $1/2$ the following wavelet $d_{1,n}$ and scaling coefficients $c_{1,n}$:

$$c_{1,n} = \sqrt{2}\sum h_0(k-2n)c_{0,k} \tag{3.112}$$

and

$$d_{1,n} = \sqrt{2}\sum h_1(k-2n)c_{0,k} \tag{3.113}$$

These last two so-called analysis equations relate the DWT coefficients at a finer scale to the DWT coefficients at a coarser scale. The analysis operations are similar to ordinary convolution. The similarity between ordinary convolution and the analysis equations suggests that the scaling function coefficients and wavelet function coefficients may be viewed as

impulse responses of filters. In fact, the set $\{h_0(-n), h_1(-n)\}$ can be viewed as a paraunitary FIR filter pair. Figure 3.24 illustrates this.

The discrete signal $d_{1,n}$ is the WT coefficient at the resolution 1/2 and describes the detail signal or difference between the original signal $c_{0,n}$ and its smooth undersampled approximation $c_{1,n}$.

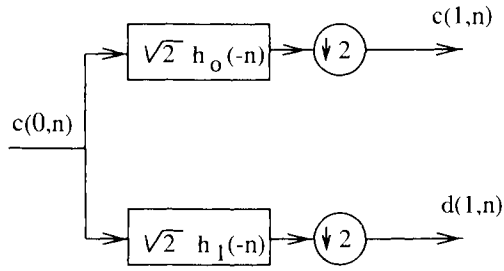


FIGURE 3.24 First level of the multiscale signal decomposition.

For $m = 2$, we obtain at the resolution 1/4 the coefficients of the smoothed signal (approximation) and the detail signal (approximation error) as

$$c_{2,n} = \sqrt{2} \sum c_{1,k} h_0(k - 2n) \tag{3.114}$$

$$d_{2,n} = \sqrt{2} \sum c_{1,k} h_1(k - 2n) \tag{3.115}$$

These relationships are illustrated in the two-level multiscale pyramid in Figure 3.25.

Wavelet synthesis is the process of recombining the components of a signal to reconstruct the original signal. The inverse discrete wavelet transformation, or IDWT, performs this operation. To obtain $c_{0,n}$, the terms $c_{1,n}$ and $d_{1,n}$ are upsampled and convoluted with the filters $h_0(n)$ and $h_1(n)$ as shown in Figure 3.26.

The results of the multiscale decomposition and reconstruction of a dyadic subband tree are shown in Figure 3.27 and describe the analysis and synthesis part of a two-band PR-QMF bank.

It is important to mention that the recursive algorithms for decomposition and reconstruction can be easily extended for a two-dimensional signal (image) [237] and play an important role in image compression.

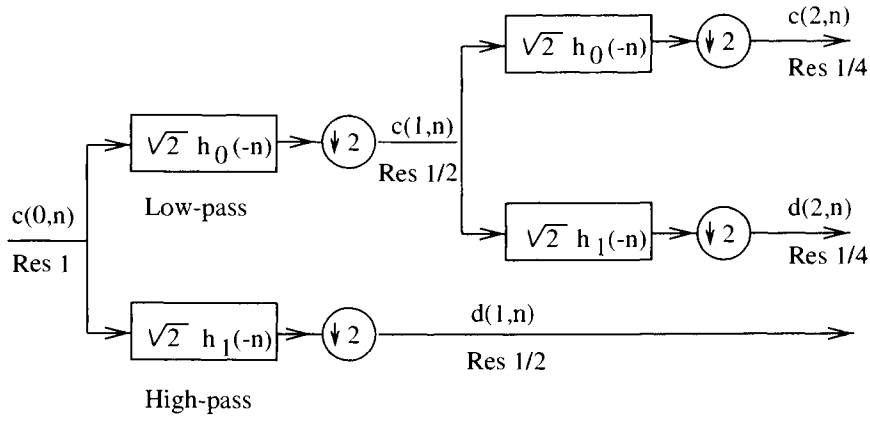


FIGURE 3.25 Multiscale pyramid decomposition.

3.5.5 Wavelet Transformation at a Finite Resolution

In this section we will show that a function can be approximated to a desired degree by summing together the scaling function and as many wavelet detail functions as necessary. Let $f \in V_0$ be defined as

$$f(t) = \sum c_{0,n}\phi(t-n) \tag{3.116}$$

As stated in previous sections, it also can be represented as a sum of signal at a coarser resolution (approximation) plus a detail signal (approximation error):

$$f(t) = f_v^1(t) + f_w^1(t) = \sum c_{1,n}2^{\frac{1}{2}}\phi\left(\frac{t}{2}-n\right) + \sum d_{1,n}2^{\frac{1}{2}}\psi\left(\frac{t}{2}-n\right) \tag{3.117}$$

The coarse approximation $f_v^1(t)$ can be rewritten as

$$f_v^1(t) = f_v^2(t) + f_w^2(t) \tag{3.118}$$

such that

$$f(t) = f_v^2(t) + f_w^2(t) + f_w^1(t) \tag{3.119}$$

Continuing with this procedure we have at scale J for $f_v^J(t)$

$$f(t) = f_v^J(t) + f_w^J(t) + f_w^{J-1}(t) + \cdots + f_w^1 \tag{3.120}$$

or

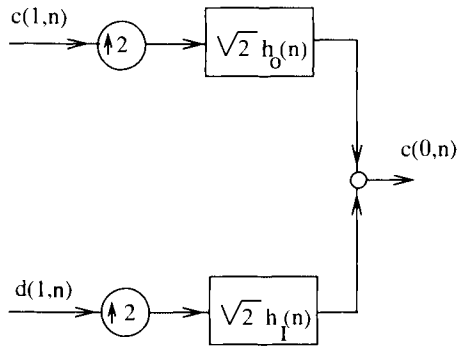


FIGURE 3.26 Reconstruction of a one-level multiscale signal decomposition.

$$f(t) = \sum_{n=-\infty}^{\infty} c_{J,n}\phi_{J,n}(t) + \sum_{m=1}^{J}\sum_{n=-\infty}^{\infty} d_{m,n}\psi_{m,n}(t) \tag{3.121}$$

This equation describes a wavelet series expansion of function $f(t)$ in terms of the wavelet $\psi(t)$ and scaling function $\phi(t)$ for an arbitrary scale J. In comparison, the pure WT

$$f(t) = \sum_{m}\sum_{n} d_{m,n}\psi_{mn}(t) \tag{3.122}$$

requires an infinite number of resolutions for a complete signal representation.

From eq. (3.122) we can see that $f(t)$ is given by a coarse approximation at the scale L and a sum of L detail components (wavelet components) at different resolutions.

EXAMPLE 3.5.1: Consider the simple function

$$y = \begin{cases} t^2, & 0 \le t \le 1 \\ 0, & \text{else} \end{cases} \tag{3.123}$$

Using Haar wavelets and the starting scale $J = 0$ we can easily determine the following expansion coefficients:

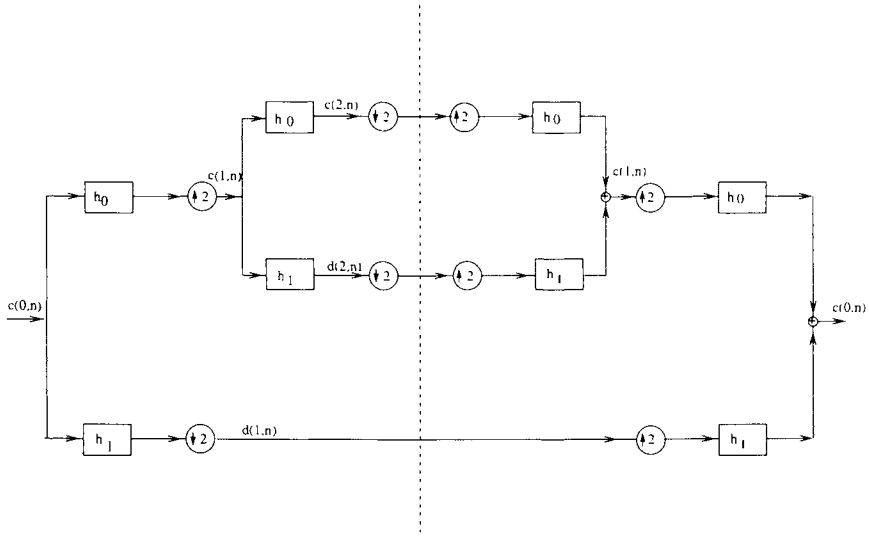


FIGURE 3.27 Multiscale analysis and synthesis.

$$
\begin{aligned}
c_{0,0} &= \int_0^1 t^2 \phi_{0,0}(t)dt = \frac{1}{3} \\
d_{0,0} &= \int_0^1 t^2 \psi_{0,0}(t)dt = -\frac{1}{4} \\
d_{1,0} &= \int_0^1 t^2 \psi_{1,0}(t)dt = -\frac{\sqrt{2}}{32} \\
d_{1,1} &= \int_0^1 t^2 \psi_{1,1}(t)dt = -\frac{3\sqrt{2}}{32}
\end{aligned} \tag{3.124}
$$

Thus, we obtain the wavelet series expansion

$$
y = \frac{1}{3}\phi_{0,0}(t) - \frac{1}{4}\psi_{0,0}(t) - \frac{\sqrt{2}}{32}\psi_{1,0}(t) - \frac{3\sqrt{2}}{32}\psi_{1,1}(t) + \cdots \tag{3.125}
$$

3.6 OVERVIEW: TYPES OF WAVELET TRANSFORMS

The goal of this section is to provide an overview of the most frequently used wavelet types. Figure 3.28 illustrates the block diagram of the generalized time-discrete filter bank transform.

The mathematical representation of the direct and inverse generalized time–discrete filterbank transform is

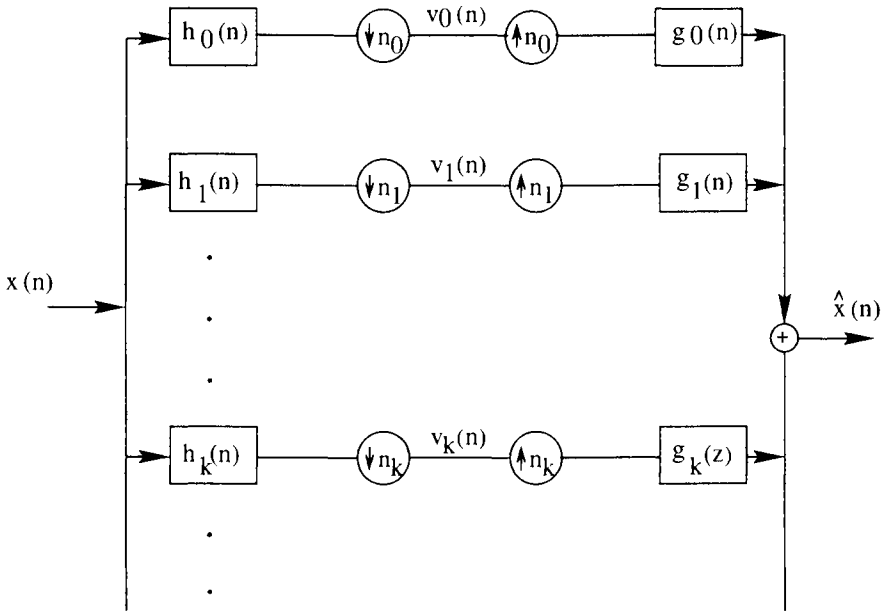


FIGURE 3.28 Generalized time-discrete filter bank transform.

$$v_k(n) = \sum_{m=-\infty}^{\infty} x(m)h_k(n_k n - m), \quad 0 \leq k \leq M-1 \tag{3.126}$$

and

$$\hat{x}(n) = \sum_{k=0}^{M-1} \sum_{m=-\infty}^{\infty} v_k(m)g_k(n - n_k m) \tag{3.127}$$

Based on this representation, we can derive as functions of n_k, $h_k(n)$, and $g_k(n)$ the following special cases [62]:

1. **Orthonormal wavelets:** $n_k = 2^k$ with $0 \leq k \leq M-2$ and $n_{M-1} = n_{M-2}$. The basis function fulfills the orthonormality condition (3.57).
2. **Orthonormal wavelet packets:** They represent a generalization of the orthonormal wavelets because they make use of the recursive decomposition-reconstruction structure which is applied to all bands as shown in Figure 3.8. The following holds: $n_k = 2^L$ with $0 \leq k \leq 2^L - 1$.

3. **Biorthogonal wavelets:** They have properties similar to those of the orthogonal wavelets but are less restrictive. They will be explained in detail in section 4.3.
4. **Generalized filter bank representations:** They represent a generalization of the (bi)orthogonal wavelet packets. Each band is split in two subbands. The basis functions fulfill the biorthonormality condition:

$$\sum_{m=-\infty}^{\infty} g_c(m - n_c l)h_k(n_k n - m) = \delta(c - k)\delta(l - n) \qquad (3.128)$$

5. **Oversampled wavelets:** There is no downsampling or oversampling required, and $n_k = 1$ holds for all bands. An example of an oversampled WT is shown in Figure 3.29.

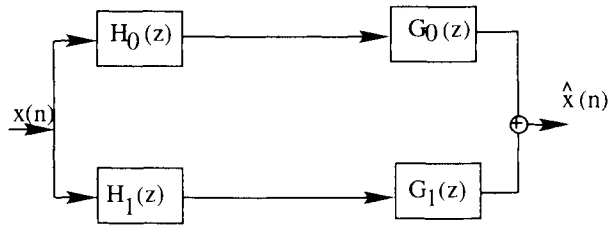


FIGURE 3.29 Oversampled wavelet transform.

The first four wavelet types are known as *nonredundant wavelet representations.* For the representation of oversampled wavelets more analysis functions ($\{u_k(n)\}$) than basis functions are required. The analysis and synthesis functions must fulfill

$$\sum_{k=0}^{M-1} \sum_{m=-\infty}^{\infty} g_k(m - l)h_k(n - m) = \delta(l - n), \quad \text{or} \quad \sum_{k=0}^{M-1} G_k(z)H_k(z) = 1 \qquad (3.129)$$

This condition holds only in case of linear dependency. This means that some functions are represented as linear combinations of others.

EXERCISES

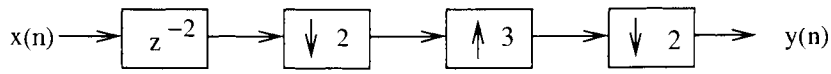
1. Given

$$x(n) = 3\lambda^n u(n) \qquad (3.130)$$

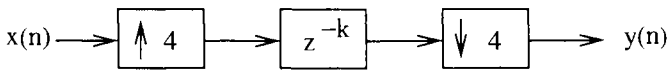
Let $y(n)$ be the downsampled signal of $x(n)$ for $M = 2$. Determine $X(z), Y(z)$ and show the pole–zero plots. Determine and plot

$|X(e^{j\omega})|$, and $|Y(e^{j\omega})|$. Next consider the upsampling of $x(n)$ and perform the same steps as for downsampling.

2. Consider the following alternating vector $\mathbf{x}(n) = \{\ldots, 1, -1, 1, -1, 1, -1, 1, -1, 1 \ldots\}$, which is delayed to $\mathbf{x}(n-2)$. Determine $(\downarrow 2)\mathbf{x}(n-2)$.
3. What are the components of $(\uparrow 3)(\downarrow 2)\mathbf{x}$ and $(\downarrow 2)(\uparrow 3)\mathbf{x}$ with $\mathbf{x}$ given as in Exercise 2?
4. Show that the equivalence in Figure 3.5a holds both in the z-domain and in the time domain.
5. Show that the equivalence in Figure 3.5b holds both in the z-domain and in the time domain.



(a)



(b)

FIGURE 3.30 Systems for Exercises 6 and 7.

6. Simplify the following system given in Figure 3.30a: Express $Y(z)$ in terms of $X(z)$. What is $\mathbf{y}(n)$ given
   a) $\mathbf{x}(n) = \delta(n)$ and
   b) $\mathbf{x}(n) = (\ldots, 1, 1, 1, 1, 1, 1, \ldots)$?
7. Simplify the following system given in Figure 3.30b:
   Express $Y(z)$ in terms of $X(z)$. What is $\mathbf{y}(n)$ in terms of $\mathbf{x}(n)$?
8. Given a $2^L \times 2^L$ image, does a $(L-1)$-level pyramid reduce or increase the amount of data required to represent the image? What compression or expansion ratio is thus achieved?
9. Show that the conjugate quadrature filter from Table 3.1 forms a perfect reconstruction filter bank.
10. Determine if the quadrature mirror filters are orthonormal filters.
11. Express the scaling space V_4 as a function of scaling function $\phi(t)$. Employ the Haar scaling function to plot the Haar V_4 scaling functions at translations $k = \{0, 1, 2\}$.

12. Plot the wavelet $\psi_{4,4}(t)$ for the Haar wavelet function. Express $\psi_{4,4}$ in terms of the Haar scaling function.

13. Discuss the differences between the CWT and DWT. When is the CWT and when is the DWT more useful for practical applications.

14. Verify the following for the Haar wavelet family:
a) $\phi(4t) = \sum h_0(n)\phi(8t-k)$ and
b) $\psi(4t) = \sum h_1(n)\psi(8t-k)$.

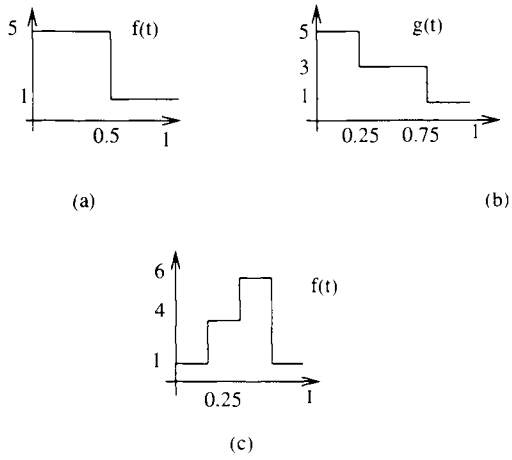


FIGURE 3.31 Examples of functions in Exercises 15, 16, and 18.

15. Split the function $f(t)$ from Figure 3.31a into a scaling function and a wavelet.

16. Split the function $g(t)$ from Figure 3.31b into its parts in the spaces V_0, W_0, and W_1.

17. The function $f(t)$ is given as:

$$f(t) = \begin{cases} 4, & 0 \le t < 2 \\ 0, & \text{else} \end{cases} \tag{3.131}$$

Plot the following scaled and/or translated versions of $f(t)$:
a) $f(t-1)$
b) $f(2t)$
c) $f(2t-1)$
d) $f(4t)$
e) $f(4t-5)$
f) $f(t/2)$
g) $f(t/2-1)$.

18. The Haar wavelet family must be used to code function $f(t)$ from Figure 3.31c. Determine a coding using:

a) $\phi(4t), \phi(4t-1), \phi(4t-2), \phi(4t-3)$.

b) $\phi(2t), \phi(2t-1), \psi(2t), \psi(2t-1)$.

c) $\phi(t), \psi(t), \psi(2t), \psi(2t-1)$.

IV

THE WAVELET TRANSFORM IN MEDICAL IMAGING

4.1 INTRODUCTION

Tremendous research efforts during the past 10 years have established the wavelet transform as the most important tool in biomedical imaging. This valuable technique specifically addresses for medical imaging the problems of image compression, edge and feature selection, denoising, contrast enhancement, and image fusion.

In this chapter, we present the theory of 2D discrete wavelet transforms, and of biorthogonal wavelets, and we show several applications of the wavelet transform in medical imaging. The most remarkable applications are: (a) ability of the WT to make visible simple objects in a noisy background, which were previously considered to be invisible to a human viewer, (b) demonstrated superiority of the WT over existing techniques for unsharp mask enhancement and median filtering, and (c) enhancing the visibility of clinically important features.

4.2 THE TWO-DIMENSIONAL DISCRETE WAVELET TRANSFORM

For any wavelet orthonormal basis $\{\psi_{j,n}\}_{(j,n)\in Z^2}$ in $L^2(R)$, there exists a separable wavelet orthonormal basis also in $L^2(R)$:

$$\{\psi_{j,n}(x)\psi_{l,m}(y)\}_{(j,l,n,m)\in Z^4} \tag{4.1}$$

The functions $\psi_{j,n}(x)\psi_{l,m}(y)$ mix the information at two different scales 2^j and 2^l across x and y. This technique leads to a building procedure based on separable wavelets whose elements represent products of function dilation at the same scale. These multiscale approximations are mostly applied in image processing because they facilitate the processing of images at several detail levels. Low resolution images can be represented using fewer pixels while preserving the features necessary for recognition tasks.

The theory presented for the one-dimensional WT can be easily extended to two-dimensional signals such as images. In two dimensions, a 2D scaling function, $\phi(x,y)$ and three 2D wavelets $\psi^1(x,y), \psi^2(x,y)$ and $\psi^3(x,y)$ are required. Figure 4.1 shows a 2D filter bank. Each filter $\psi_a(x,y)$ represents a 2D impulse response and its output a bandpass filtered version of the original image. The set of the filtered images describes the WT.

In the following, we will assume that the 2D scaling functions are separable, that is,

$$\phi(x,y) = \phi(x)\phi(y) \tag{4.2}$$

where $\phi(x)$ is a one–dimensional scaling function. If we define $\psi(x)$, the companion wavelet function, as shown in eq (3.73), then based on the following three basis functions,

$$\psi^1(x,y) = \phi(x)\psi(y) \quad \psi^2(x,y) = \psi(x)\phi(y) \quad \psi^3(x,y) = \psi(x)\psi(y) \tag{4.3}$$

we set up the foundation for the 2D wavelet transform. Each of them is the product of a one-dimensional scaling function ϕ and a wavelet function ψ. They are "directionally sensitive" wavelets because they measure functional variations, either intensity or gray-level variations, along different directions: ψ^1 measures variations along the columns (horizontal edges), ψ^2 is sensitive to variations along rows (vertical edges), and ψ^3 corresponds to variations along diagonals. This directional sensitivity occurs as an implication of the separability condition.

To better understand the 2D WT, let us consider $f_1(x,y)$, an $N \times N$ image, where the subscript describes the scale and N is a power of 2. For $j = 0$, the scale is given by $2^j = 2^0 = 1$ and corresponds to the original image. Allowing j to become larger doubles the scale and halves the resolution.

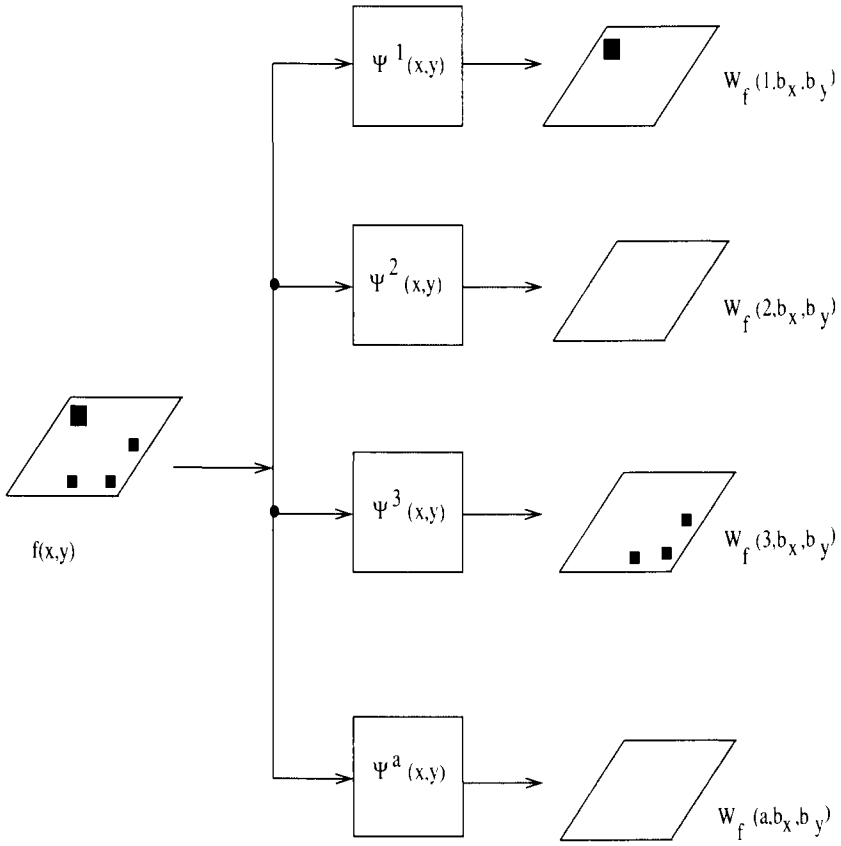


FIGURE 4.1 Filter-bank analogy of the WT of an image.

An image can be expanded in terms of the 2D WT. At each decomposition level, the image can be decomposed into four subimages a quarter of the size of the original, as shown in Figure 4.2. Each of these images stems from an inner product of the original image with the subsampled version in x and y by a factor 2. For the first level ($j = 1$), we obtain

$$
\begin{aligned}
f_2^0(m,n) &= < f_1(x,y), \phi(x-2m, y-2n) > \\
f_2^1(m,n) &= < f_1(x,y), \psi^1(x-2m, y-2n) > \\
f_2^2(m,n) &= < f_1(x,y), \psi^2(x-2m, y-2n) > \\
f_2^3(m,n) &= < f_1(x,y), \psi^3(x-2m, y-2n) >
\end{aligned}
\tag{4.4}
$$

For the subsequent levels ($j > 1$), $f^0_{2^j}(x,y)$ is decomposed in a similar way, and four quarter-size images at level 2^{j+1} are formed. This procedure is visualized in Figure 4.2.

The inner products can also be written as a convolution:

$$\begin{aligned} f^0_{2^{j+1}}(m,n) &= \{[f^0_{2^j}(x,y) * \phi(x,y)](2m,2n)\} \\ f^1_{2^{j+1}}(m,n) &= \{[f^0_{2^j}(x,y) * \psi^1(x,y)](2m,2n)\} \\ f^2_{2^{j+1}}(m,n) &= \{[f^0_{2^j}(x,y) * \psi^2(x,y)](2m,2n)\} \\ f^3_{2^{j+1}}(m,n) &= \{[f^0_{2^j}(x,y) * \psi^3(x,y)](2m,2n)\} \end{aligned} \tag{4.5}$$

The scaling and the wavelet functions are separable, and therefore we can replace every convolution by a 1D convolution on the rows and columns of $f^0_{2^j}$. Figure 4.4 illustrates this fact. At level 1, we convolve the rows of the image $f_1(x,y)$ with $h_0(x)$ and with $h_1(x)$ and then eliminate the odd-numbered columns (the leftmost is set to zero) of the two resulting arrays. The columns of each $N/2 \times N$ are then convolved with $h_0(x)$ and $h_1(x)$, and the odd-numbered rows are eliminated (the top row is set to zero). As an end result we obtain the four $N/2 \times N/2$-arrays required for that level of the WT. Figure 4.3 illustrates the localization of the four newly obtained images in the frequency domain. $f^0_{2^j}(x,y)$ describes the low-frequency information of the previous level, while $f^1_{2^j}(x,y)$, $f^2_{2^j}(x,y)$ and $f^3_{2^j}(x,y)$ represent the horizontal, vertical, and diagonal edge information.

The inverse WT is shown in Figure 4.4. At each level, each of the arrays obtained on the previous level is upsampled by inserting a column of zeros to the left of each column. The rows are then convolved with either $h_0(x)$ or $h_1(x)$, and the resulting $N/2 \times N$-arrays are added together in pairs. As a result, we get two arrays which are oversampled to achieve an $N \times N$-array by inserting a row of zeros above each row. Next, the columns of the two new arrays are convolved with $h_0(x)$ and $h_1(x)$, and the two resulting arrays are added together. The result shows the reconstructed image for a given level.

4.3 BIORTHOGONAL WAVELETS AND FILTER BANKS

The functions that qualify as orthonormal wavelets with compact support lack a very important property, that of symmetry. By choosing two different wavelet bases, $\psi(x)$ and $\tilde{\psi}(x)$, one for the analysis (decomposition) and the other for the synthesis (reconstruction), we get symmetric wavelets with compact support. The two wavelets are duals of each other, and the wavelet families $\{\psi_{jk}(x)\}$ and $\{\tilde{\psi}_{jk}(x)\}$ are biorthogonal.

Let us look again at the standard two-band filter bank. The conditions for achieving perfect reconstructions are

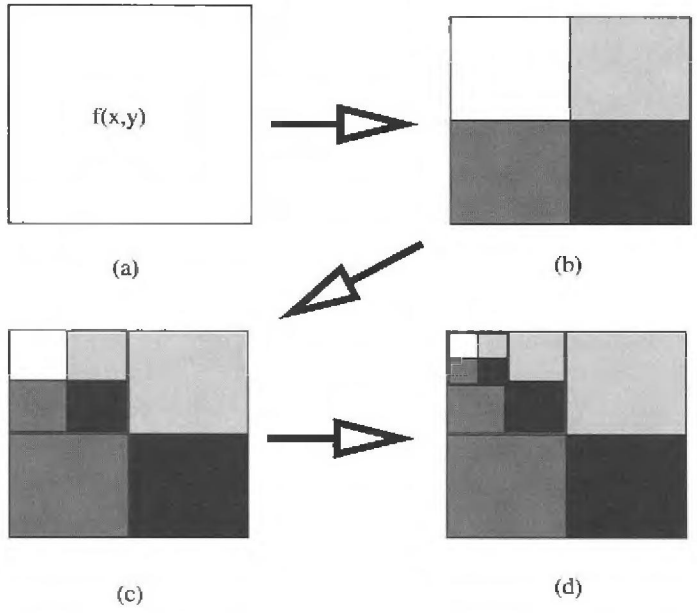


FIGURE 4.2 2D discrete wavelet transform: (a) original image; (b) first, (c) second, and (d) third level.

$$\begin{aligned}\hat{X}(z) &= \frac{1}{2}X(z)[H_0(z)G_0(z) + H_1(z)G_1(z)] \\ &\quad + \frac{1}{2}X(-z)[H_0(-z)G_0(-z) + H_1(-z)G_1(-z)]\end{aligned} \tag{4.6}$$

To achieve a perfect reconstruction, we must impose two conditions:

$$H_0(-z)G_0(z) + H_1(z)G_1(z) = 0 \tag{4.7}$$

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 2 \tag{4.8}$$

Equation (4.7) eliminates aliasing, while eq. (4.8) eliminates amplitude distortion. Based on the perfect reconstruction conditions, we can demonstrate the biorthogonality of the analysis and synthesis filters. Let $P(z)$ be the product of the transfer functions of the lowpass analysis and synthesis filters. From eq. (3.28), we obtain

$$P(z) = G_0(z)H_0(z) = \frac{2}{\det(\mathbf{H}_m(z))}H_0(z)H_1(-z) \tag{4.9}$$

Similarly, we can write for $G_1(z)H_1(z)$, considering $\det(\mathbf{H}_m(z))$ $= -\det(\mathbf{H}_m(-z))$

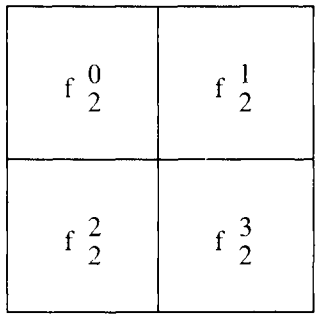


FIGURE 4.3 DWT decomposition in the frequency domain.

$$P(-z) = G_1(z)H_1(z) = \frac{-2}{\det(\mathbf{H}_m(z))} H_0(-z)H_1(z) = G_0(-z)H_0(-z) \tag{4.10}$$

Thus, we obtain

$$G_0(z)H_0(z) + G_0(-z)H_0(-z) = 2 \tag{4.11}$$

Taking the inverse Z-transform, we obtain

$$\sum_k g_0(k)h_0(2n-k) =< g_0(k), h_0(2n-k) >= \delta(n) \tag{4.12}$$

Similarly, by expressing G_0 and H_0 as a function of G_1 and H_1, we obtain the more general expression for biorthogonality:

$$< h_i(2n-k), g_j(k) >= \delta(i-j)\delta(n), \quad i,j = \{0,1\} \tag{4.13}$$

We immediately see the main advantage with biorthogonal filters: linear phase and unequal filter length.

Let us now analyze this from the wavelet perspective. For this we must define two new hierarchies of subspaces :

$$\begin{aligned} &\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots \\ &\cdots \tilde{V}_2 \subset \tilde{V}_1 \subset \tilde{V}_0 \subset \tilde{V}_{-1} \subset \tilde{V}_{-2} \cdots \end{aligned} \tag{4.14}$$

The orthogonal complementary properties hold for $W_j \perp \tilde{V}_j$ and $\tilde{W}_j \perp V_j$

This leads to

$$V_{j-1} = V_j \oplus \tilde{W}_j \quad \text{and} \quad \tilde{V}_{j-1} = \tilde{V}_j \oplus W_j \tag{4.15}$$

and to the following scaling and wavelet functions:

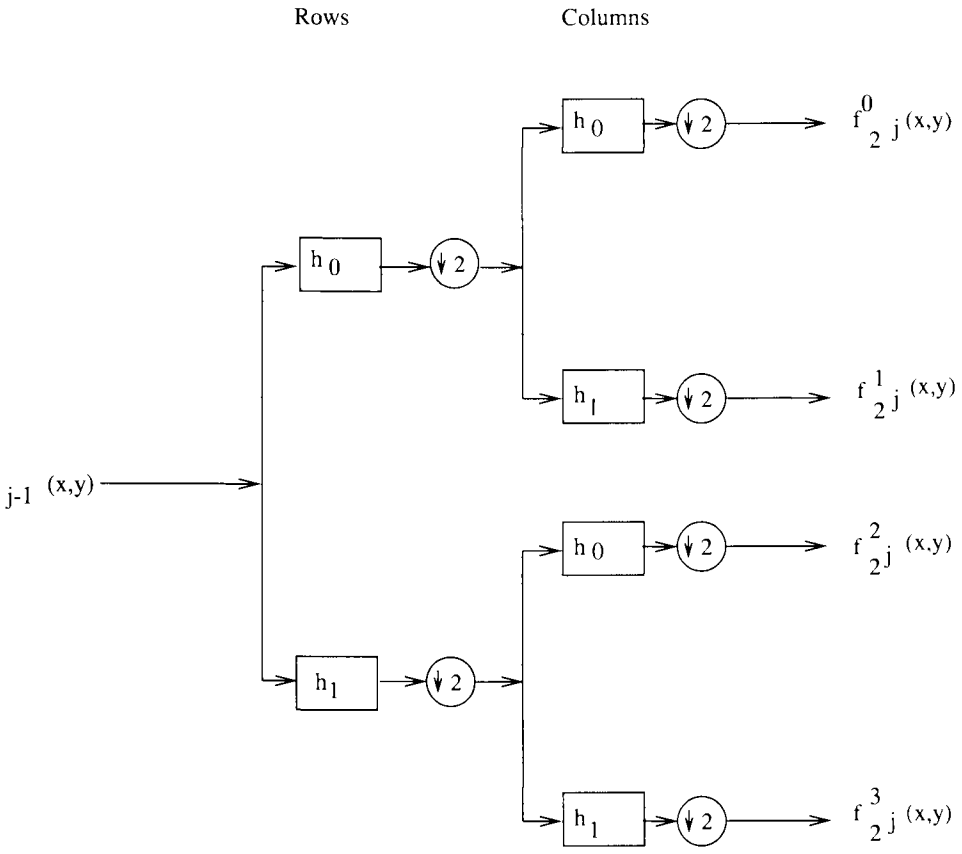


FIGURE 4.4 Image decomposition based on discrete WT.

$$\phi(t) = 2\sum h_0(n)\phi(2t-n) \tag{4.16}$$
$$\bar{\phi}(t) = 2\sum \tilde{g}_0(n)\bar{\phi}(2t-n)$$

and

$$\psi(t) = 2\sum h_1(n)\phi(2t-n) \tag{4.17}$$
$$\bar{\psi}(t) = 2\sum \tilde{g}_1(n)\bar{\phi}(2t-n)$$

It is easy to show that the scaling and wavelet function are interrelated for the biorthogonal case

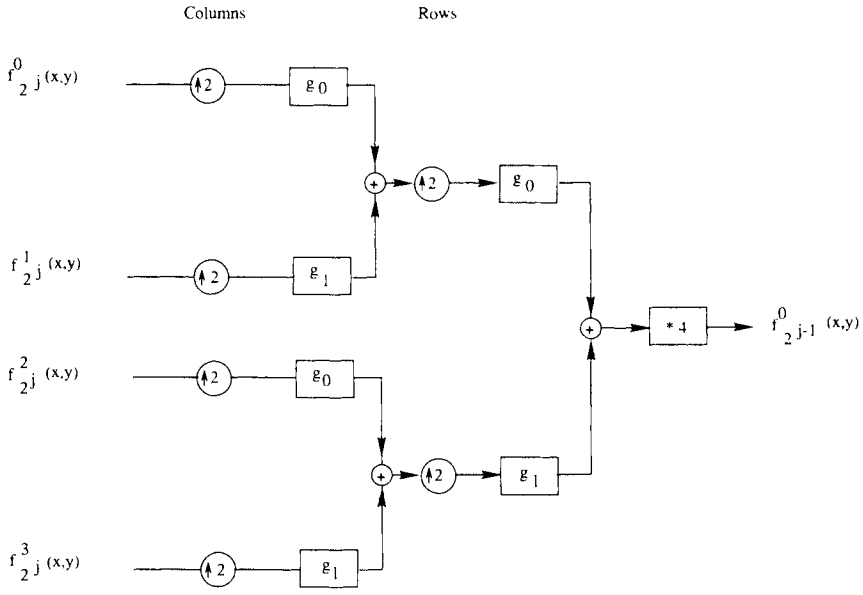


FIGURE 4.5 Image reconstruction based on the DWT.

$$\begin{aligned} &< \bar{\phi}(t-k), \phi(t-l) >= \delta_{k-l} \\ &< \bar{\psi}(t-k), \psi(t-l) >= \delta_{k-l} \end{aligned} \tag{4.18}$$

and

$$\begin{aligned} &< \bar{\phi}(t-k), \psi(t-l) >= 0 \\ &< \bar{\psi}(t-k), \phi(t-l) >= 0 \end{aligned} \tag{4.19}$$

Based on the foregoing, we see that any function $f \in L^2(R)$ can be written as

$$\begin{aligned} f(t) &= \sum_j \sum_k < f, \bar{\psi}_{jk} > \psi_{jk}(t) \\ &= \sum_j \sum_k < f, \psi_{jk} > \bar{\psi}_{jk}(t) \end{aligned} \tag{4.20}$$

The most important aspects achieved simultaneously based on biorthogonality are perfect reconstruction and linear phase filters. However, a drawback is the different filter lengths in the decomposition part, which means

that the signal spectrum is unevenly distributed over the low-band and high-band segments. Figure 4.5 illustrates the image reconstruction based on the DWT.

4.4 APPLICATIONS

4.4.1 Multiscale Edge Detection

The transient properties of signals and images are best described by points of sharp variations, which are mostly located at the boundaries of relevant image structures. To detect contours of small structures as well as boundaries of larger objects, a new wavelet-based concept, the so-called multiscale edge detection, was introduced [150, 152]. The scale describes the size of the neighborhood where the signal variations are determined. The WT provides a better insight in these algorithms, because it is closely related to multiscale edge detection. For example, the Canny edge detector [31] corresponds to determining the local maxima of a wavelet transform modulus. Since the multiscale edges carry important information, it is possible to reconstruct images from multiscale edges. In [152] a compact image coding algorithm is proposed that keeps only the important edges. The image can be recovered from these main features, and although some small details are lost, it preserves its quality. In the following, we will describe how multiscale edge detection algorithms are related to the WT.

Multiscale edge detectors typically smooth the signal at various scales and determine the sharp variation of points from their first- or second-order derivatives. The extrema of the first derivative describe the zero crossings of the second derivative and also the inflection points of the smoothed signal.

A function $\theta(x)$ is defined as a smoothing function, if the following properties hold:

$$\int_{-\infty}^{\infty} \theta(x)dx = 1 \tag{4.21}$$

and

$$\lim_{x\to\infty} \theta(x) = 0 \tag{4.22}$$

The Gaussian function can be chosen as a smoothing function. For a smoothing function $\theta(x)$, we assume that it is twice differentiable and denote by $\psi^a(x)$ and $\psi^b(x)$ the first and second derivative of $\theta(x)$

$$\psi^a(x) = \frac{d\theta(x)}{dx} \quad \text{and} \quad \psi^b(x) = \frac{d^2\theta(x)}{dx^2} \tag{4.23}$$

We immediately can see that $\psi^a(x)$ and $\psi^b(x)$ represent wavelets because the following holds:

$$\int_{-\infty}^{\infty} \psi^a(x)dx = 0 \quad \text{and} \int_{-\infty}^{\infty} \psi^b(x)dx = 0 \tag{4.24}$$

Based on the new definition

$$\zeta_s(x) = \frac{1}{s}\zeta\left(\frac{x}{s}\right) \tag{4.25}$$

we can represent a dilation by a scaling factor s of the given function $\zeta(x)$.

The WT of a function $f(x)$ with respect to $\psi^a(x)$ and $\psi^b(x)$ is defined as

$$W_f(a,s) =< \psi_{as}(x), f(x) > \tag{4.26}$$

and

$$W_f(b,s) =< \psi_{bs}(x), f(x) > \tag{4.27}$$

From eqs. (4.23), (4.26), and (4.27) we obtain

$$W_f(a,s) =< f(x), \left(s\frac{d\theta_s(x)}{dx}\right) >= s\frac{d < f(x), \theta_s(x) >}{dx} \tag{4.28}$$

and

$$W_f(b,s) =< f(x), \left(s^2\frac{d^2\theta_s(x)}{dx^2}\right) >= s^2\frac{d^2 < f(x), \theta_s(x) >}{dx^2} \tag{4.29}$$

We can see that the WTs $W_f(a,s)$ and $W_f(b,s)$ describe the first and second derivative of the signal smoothed at scale s. The local extrema of $W_f(a,s)$ thus correspond to the zero crossings of $W_f(b,s)$ and to the inflection points of $< f(x), \theta_s(x) >$. If we assume that the smoothing function $\theta(x)$ is Gaussian, then the zero-crossing detection corresponds to a Marr-Hildreth edge detection, and the extremum detection is equivalent to Canny edge detection [152]. The scale s plays an important role in edge detection: by choosing a large s and convolving it with the smoothing function corresponding to that scale, small signal fluctuations are removed, and only edges of large objects can be detected.

An inflection point of $< f(x), \theta_s(x) >$ corresponds to either a maximum or minimum value of the first derivative. It is useful to point to the role of those values: the maximum value describes the fast fluctuations of $< f(x), \theta_s(x) >$, while the minimum value describes the slow ones.

To apply the derived theoretical aspects to image processing, we need to extend the Canny edge detector in two dimensions. We will use the following notations

$$\zeta_s(x,y) = \frac{1}{s^2}\zeta(\frac{x}{s},\frac{y}{s}) \tag{4.30}$$

describing the dilation by s of any 2D function $\zeta(x,y)$.

We assume that the smoothing function has the properties

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\theta(x,y)dxdy = 1 \tag{4.31}$$

and

$$\lim_{x,y\to\infty}\theta(x,y) = 0 \tag{4.32}$$

The image $f(x,y)$ is convolved with $\theta_s(x,y)$ and thus smoothed at different scales s. In the next step, we need to determine the gradient vector $\vec{\nabla}(< f(x,y),\theta_s(x,y) >)$. The gradient vector points into the direction of the largest absolute value of the directional derivative of the given image. Edges are defined as points (x_0,y_0) where the gradient vector exhibits its greatest magnitude and are at the same time the inflection points of the surface $< f(x,y),\theta_s(x,y) >$ [152].

As for the 1D case, we will point out the equivalence between edge detection and the 2D WT. Let $\psi_1(x,y)$ and $\psi_2(x,y)$ be two wavelet functions

$$\psi_1(x,y) = \frac{\partial\theta(x,y)}{\partial x} \quad \text{and} \quad \psi_2(x,y) = \frac{\partial\theta(x,y)}{\partial y} \tag{4.33}$$

and

$$\psi_{1s}(x,y) = \frac{1}{s^2}\psi_1(\frac{x}{s},\frac{y}{s}) \tag{4.34}$$

and, respectively,

$$\psi_{2s}(x,y) = \frac{1}{s^2}\psi_2\left(\frac{x}{s},\frac{y}{s}\right) \tag{4.35}$$

Let us assume that $f(x,y) \in L^2(R^2)$. The WT of $f(x,y)$ at the scale s is given by two distinct parts,

$$W_f(1,s) = < f(x,y),\psi_{1s}(x,y) > \quad \text{and} \quad W_f(2,s) = < f(x,y),\psi_{2s}(x,y) > \tag{4.36}$$

As seen in eqs. (4.26) and (4.27), we can also show here that the following relationship holds:

$$\begin{pmatrix} W_{1s}f(x,y) \\ W_{2s}f(x,y) \end{pmatrix} = s\begin{pmatrix} \frac{\partial < f(x,y),\theta_s(x,y) >}{\partial x} \\ \frac{\partial < f(x,y),\theta_s(x,y) >}{\partial y} \end{pmatrix} = s\vec{\nabla} < f(x,y),\theta_s(x,y) > \tag{4.37}$$

We see here as well that edges can be determined based on the two components $W_{1s}f(x,y)$ and $W_{2s}f(x,y)$ of the WT.

Multiscale edge detection can be illustrated by means of a mammographic image. An overcomplete and continuous wavelet representation is used in the following for the extraction of features for mass detection. Overcomplete wavelet representations are more desirable for detection and enhancement because (1) they avoid the aliasing effects introduced by critically sampled representations, and (2) they yield a shift-invariant representation. Continuous wavelet representations are also useful for analysis because they yield wavelet representations at nondyadic scales.

Figure 4.6 shows a mammogram containing a small mass. The typical mammographic features are visualized at four different scales. It is important to note that across each scale, the spatial location of the mass is preserved because of the translation invariance property. Also, the transform coefficient matrix size at each scale remains the same as the spatial resolution of the original image since we do not down sample across scales.

4.4.2 Wavelet-Based Denoising and Contrast Enhancement

The DWT decomposes an image into components of different magnitudes, positions, and orientations. Similar to the linear filtering in the Fourier frequency domain, it is possible to modify the WT coefficients before we perform the inverse WT. This means to keep only the coefficients reflecting the relevant information and to omit those containing redundant information. Figure 4.7 elucidates this technique.

The middle part represents either a linear or nonlinear processing unit for modifying the WT coefficients.

Contrast enhancement can make more obvious unseen or barely seen features. In mammography, for example, early detection of breast cancer relies on the ability to distinguish between malignant and benign mammographic features. However, the detection of small malignancies and subtle lesions is in most cases very difficult. Contrast enhancement provides the physician with a better mammographic image without requiring additional radiation. Figure 4.8 illustrates a mammographic image containing a suspicious mass and the corresponding global enhanced image. In the new image we can observe that subtle features are made more clear, and the geometric shape of the spiculated mass is made more visible.

A problem associated with image enhancement in digital mammography is the need to emphasize mammographic features while reducing the enhancement of noise. Local enhancement techniques apply only to some selected parts of an image, while global enhancement pertains to the whole image. A locally enhanced suspicious mass is shown in Figure 4.9.

In the beginning of this section, we mentioned that we are interested in techniques for modifying the DWT before performing the inverse DWT. The crudest way is to remove layers of high-frequency detail from the signal

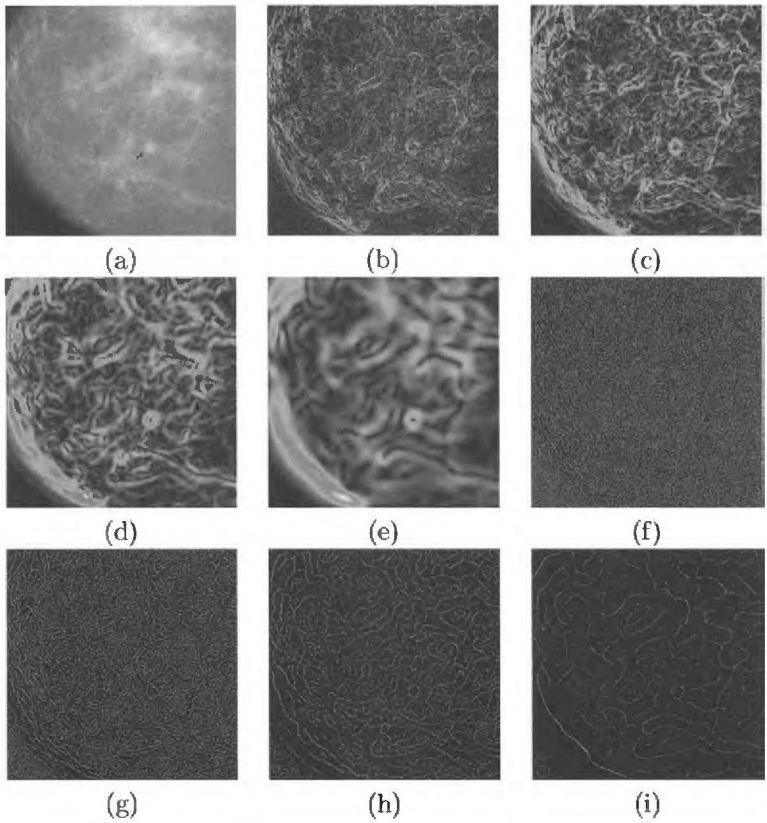


FIGURE 4.6 (a) Original mammogram containing a small spiculated mass. (b) – (e) Multiscale coefficients. (f) – (i) Multiscale edges. (Images courtesy of Dr. A. Laine, Columbia University.)

by eliminating high-frequency coefficients from the DWT. In other words, the details can be dropped by simply zeroing all the high-frequency coefficients above a certain scale. At the same time, useful approximations to the original signal can be retained. However, scale reduction is not always the most effective way to compress a signal. A more selective zeroing of DWT coefficients can be achieved based on thresholding. Donoho and Johnstone [57, 58] proposed a new technique for achieving high compression rates and denoising at the same time. Any DWT coefficient with a magnitude below a specified threshold is set to zero. Thresholding generally gives a lowpass version of the original signal and suppresses the noise amplitude. Based on the inverse WT, we then obtain an almost denoised version of the original signal. It is important to point out the difference between the

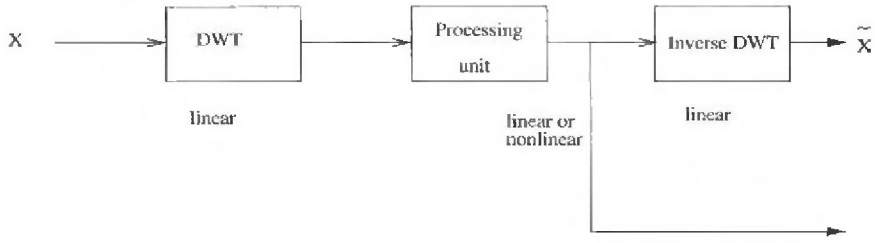
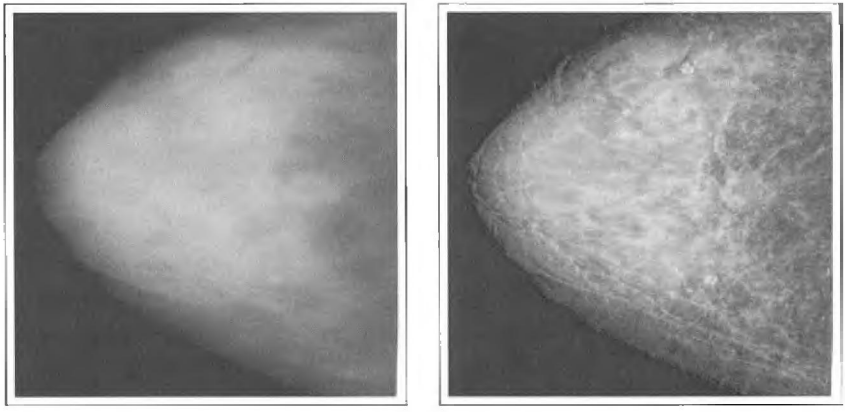


FIGURE 4.7 Wavelet-based signal processing.



(a) Original mammogram (b) Enhanced mammogram

FIGURE 4.8 Global image enhancement via a sub-octave WT and adaptive nonlinear processing. (Images courtesy of Dr. A. Laine, Columbia University.)

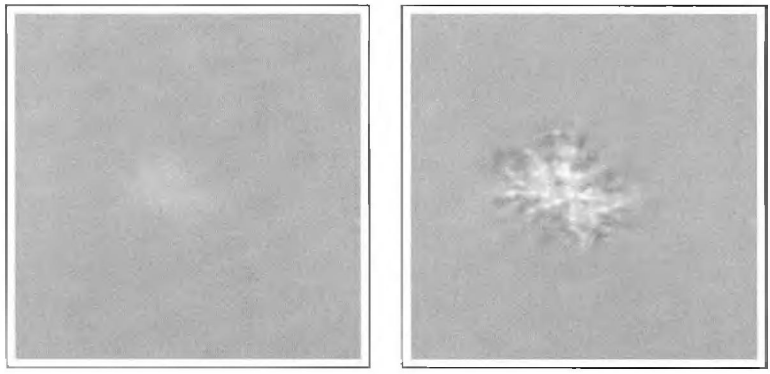
linear filtering and wavelet shrinkage proposed by Donoho and Johnstone. The first technique requires that signal and noise spectra not overlap. The latter does not impose a separability condition; it requires only different amplitudes.

4.4.3 Denoising by Thresholding

This section reviews the concept of denoising based on Donoho's work [57]. A signal x_i is corrupted by noise n_i and produces a new signal y_i. The noise power ϵ is assumed to be much smaller than the signal power. The new signal y_i is given by

$$y_i = x_i + \epsilon n_i, \quad i = 1, \ldots, N \tag{4.38}$$

The goal is to obtain the original signal x_i from the observed noisy signal y_i. The equation becomes, in the transformation domain,



(a) (b)

FIGURE 4.9 Local enhancement of mass via manual segmentation: original mammogram (a) and local multiscale enhanced image (b). (Images courtesy of Dr. A. Laine, Columbia University.)

$$Y = X + N \tag{4.39}$$

with $y = Wy$ and W the inverse WT-matrix of the DWT. For the inverse transformation matrix we obtain

$$W^{-1}W = I \tag{4.40}$$

and

$$W^{-1} = W^T \tag{4.41}$$

$\widehat{X}$ describes the estimation of X based on the observed values Y. Next, we introduce the diagonal linear projection

$$\Delta = diag(\delta_1, \ldots, \delta_N), \quad \delta_i \in \{0, 1\}, \quad i = 1 \ldots, N \tag{4.42}$$

which gives the estimation

$$\widehat{x} = W^{-1}\widehat{X} = W^{-1}\Delta Y = W^{-1}\Delta W y \tag{4.43}$$

The estimated value $\widehat{X}$ is obtained by zeroing the wavelet coefficients. The resulting L_2-error becomes

$$R(\widehat{X}, X) = E\left[||\widehat{x} - x||_2^2\right] = E\left[||W^{-1}(\widehat{X} - X)||_2^2\right] = E\left[||\widehat{X} - X||_2^2\right] \tag{4.44}$$

The last equal sign stems from the orthogonality of W. The optimal coefficients of the diagonal matrix Δ must be given by $\delta_i = 1$ for $x_i \geq \epsilon$ and $\delta_i = 0$ for $x_i < \epsilon$. Thus, only those values of Y are retained for which the corresponding X-value is larger than ϵ. The others are set to zero.

Donoho's algorithm for denoising is given below:

1. Compute the DWT $Y = Wy$
2. Apply either hard or soft thresholding for the wavelet coefficients

$$\widehat{X} = T_h(Y,t) = \begin{cases} Y, & |Y| \geq t \\ 0, & |Y| < t \end{cases} \tag{4.45}$$

or

$$\widehat{X} = T_s(Y,t) = \begin{cases} \text{sgn}(Y)(|Y| - t), & |Y| \geq t \\ 0, & |Y| < t \end{cases} \tag{4.46}$$

3. Compute the inverse DWT $\widehat{x} = W^{-1}\widehat{X}$.

Donoho's algorithm exhibits some interesting properties. The soft thresholding guarantees that $|\widehat{X}_i| < |X_i|$ holds and that $\widehat{X}$ has the same smoothness degree as X. Soft thresholding can be considered as an optimal estimation. On the other hand, the hard thresholding achieves the smallest L_2-error but lacks the required smoothness.

4.4.4 Nonlinear Contrast Enhancement

Image enhancement techniques have been widely applied in fields such as radiology, where the subjective quality of images is important for diagnosis. Low-contrast structures need to be resolved in all kinds of medical images. Obtaining high contrast in the raw image directly from the imaging device is almost always expensive in examination time or X-ray dose to the patient. Therefore, digital postprocessing can play a very important role.

From an image-processing point of view, the low contrast can be considered a result of "bad" distribution of pixel intensities over the dynamic range of the display device. This suggests the application of contrast enhancement methods in an attempt to modify the intensity distribution of the image. Many known enhancement algorithms have been proposed and applied to medical imaging. A comprehensive survey of the most relevant enhancement algorithms is given in [238]. The most commonly used algorithms are the histogram modification and the edge enhancement techniques.

Histogram modification is a simple and fast procedure and has achieved satisfactory results for some applications. This technique works as follows: a nonlinear transformation function is applied on a histogram of an input image in order to achieve a desired histogram. However, when applied to digital images information loss occurs as a result of quantization errors.

For medical imaging this can lead to diagnostic losses because a subtle edge can be fused with the neighboring pixels and disappear.

A different approach is used for contrast enhancement. It is observed that any area in an image having highly visible edges has to be of high contrast. To achieve contrast enhancement, either the magnitude of edge gradients or the pixel intensities between both sides of edges must be amplified. Also, contrast enhancement appears to be spatially dependent, and an increase in contrast has to be perpendicular to edge contours. Standard techniques for edge enhancement are pointwise intensity transformation for edge pixels [18] or filter design for magnitude amplification of edge gradients [112]. The latter technique is identical to "unsharp masking" [34].

Recent advancement of wavelet theory has been the driving force in developing new methods for contrast enhancement [4, 62, 128, 147]. The advantage of this new technique over traditional contrast enhancement techniques is illustrated in Figure 4.10.

(a) (b) (c)

FIGURE 4.10 Multiscale contrast enhancement: original mammogram (a), histogram modified mammogram (b), and (c) multiscale contrast enhancement. (Reprinted from [130] with permission from IEEE.)

In the following, we will review simple nonlinear enhancement by point-wise functional mapping. Linear enhancement can be described as a mapping of wavelet coefficients by a linear function $E_m(x) = G_m x$ with m being the level and G_m the amplification. On the other hand, nonlinear enhancement represents a direct extension of the linear enhancement.

The idea of nonlinear contrast enhancement applied to digital mammography was first pioneered by Laine [3, 130]. The main difficulties are to choose a nonlinear function and to incorporate the dyadic wavelet information to accomplish contrast enhancement.

Laine [128] gives the following guidelines for designing a nonlinear enhancement function:

1. An area of low contrast should be enhanced more than an area of high contrast.
2. A sharp edge should not be blurred.

3. The enhancement function should be monotone in order to preserve the position of local extrema and not create new extrema.
4. The enhancement function should be antisymmetric in order to preserve phase polarity for "edge crispening."

A simple piecewise function that satisfies these requirements is shown in Figure 4.11:

$$E(x) = \begin{cases} x - (K-1)T, & x < -T \\ Kx, & |x| \leq T \\ x + (K-1)T, & x > T \end{cases} \tag{4.47}$$

where $K > 1$.

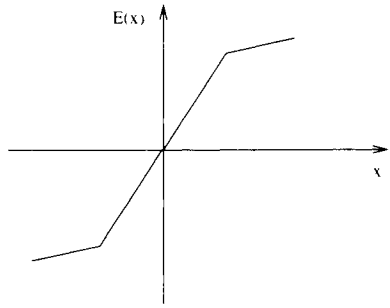


FIGURE 4.11 Nonlinear function for contrast enhancement.

For early detection of breast cancer, the ability to distinguish between malignant and mammographic features is crucial. However, the detection of small malignancies and subtle lesions encounters major difficulties. Global and local enhancement has the ability to make more visible unseen or barely seen features of a mammogram without requiring additional radiation. Figures 4.12, 4.13, 4.14, and 4.15 illustrate the necessary processing steps for global and local contrast enhancement in diagnostic radiology [129, 255]. The starting point is as usual a digitized radiograph. A unique component in the described technique is the "fusion" step shown in Figures 4.12 and 4.13. The importance of this step lies in producing images that look more familiar to radiologists than do those produced using only multiscale global enhancement.

Therefore, by integrating the original mammogram with one or more enhanced versions, a more familiar image for radiologists is achieved. Image fusion combines two or more registered images of the same object into a single image that is more easily interpreted than any of the originals.

In [117] an image fusion method based on a steerable dyadic WT is described that enables multiscale image processing along arbitrary orientations. While steerable dyadic WTs when applied to image fusion share

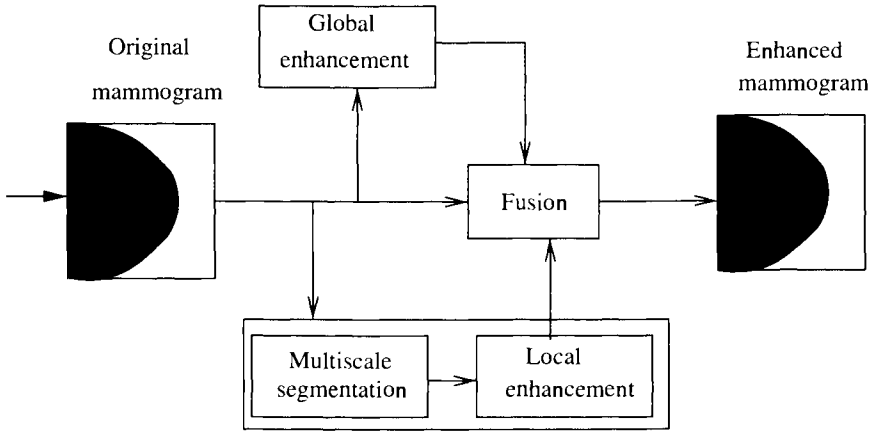


FIGURE 4.12 Processing overview for multiscale enhancement.

many properties with pyramids and WTs, they do not exhibit aliasing and translation non-invariance as sources for unwanted artifacts in a fused image. A steerable dyadic wavelet transform combines the properties of a discrete dyadic wavelet transform with the computational framework for analysis along arbitrary orientations. A steerable dyadic WT is implemented as a filter bank consisting of polar separable filters [126]. Input images are convolved with rotations, dilations, and translations of a mother wavelet. The coefficients corresponding to the greatest local oriented energy are included for reconstruction and thus produce the fused image.

Figure 4.16 shows the results of enhancement via fusion.

In [128] it was shown that local enhancement techniques, focused on suspicious areas, provide a clear advantage over traditional global processing methods where enhancement is carried out within a specified ROI or within the entire mammogram.

An example of selection of localized transform coefficients is shown in Figures 4.17 and 4.19.

4.4.5 Image Fusion

Image fusion combines two or more registered images of the same object into a single image that is more easily interpreted than any of the originals. The goal of image fusion, especially in medical imaging, is to create new images that are more suitable for the purposes of human visual perception. The simplest image fusion technique is to take the average of two input images. Applied directly, this leads to a feature contrast reduction. A solution to this technique offers a Laplacian pyramid-based image fusion [231], but at the cost of introducing blocking artifacts. Better fusion results were achieved based on the WT [139]. Figure 4.20 describes the

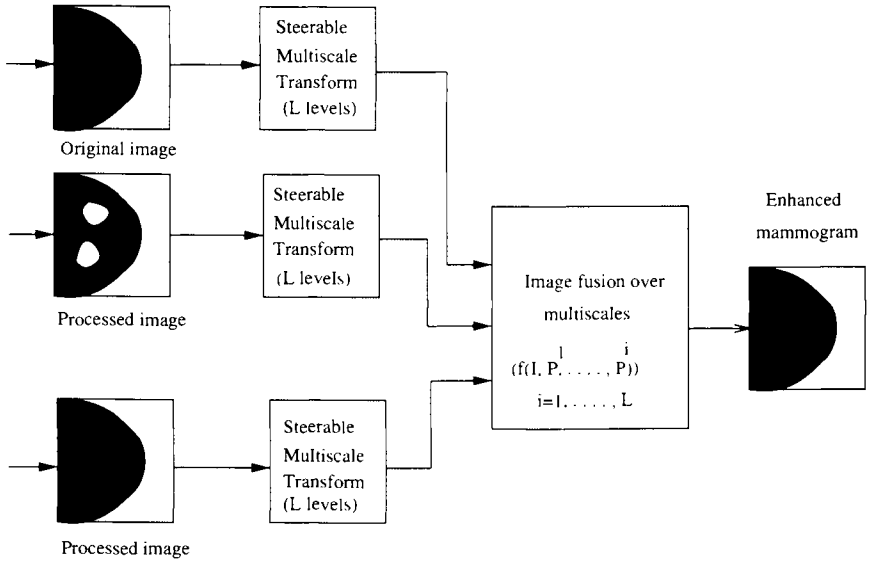


FIGURE 4.13 Fusion of locally and globally enhanced mammograms.

fusion framework proposed in [139]. The steps are: (1) computation of WT for each image, (2) selection at every point of the coefficients having the highest absolute value, and (3) computing the inverse WT for the new image. It is well known that the larger absolute value transform coefficients describe sharper brightness changes, and thus represent the salient features at different scales.

Figure 4.18 shows the image fusion results for an example from mammography. An original image is fused with a WT image and yields a better result than the original mammogram.

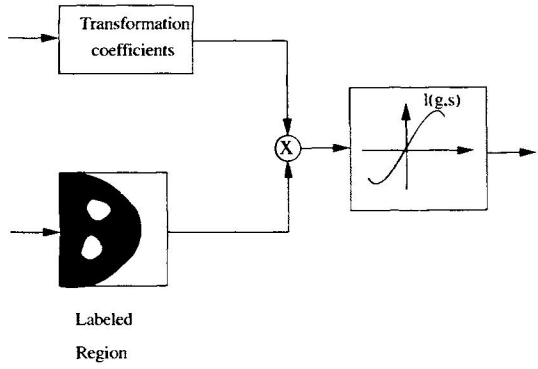


FIGURE 4.14 Local enhancement of segmented regions.

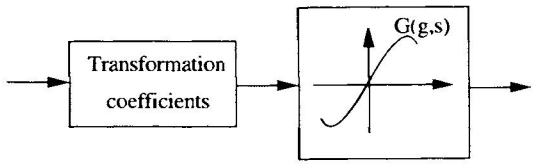


FIGURE 4.15 Global enhancement of multiscale coefficients.

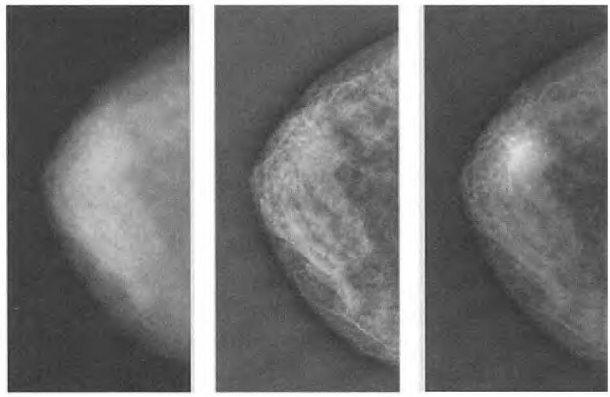


FIGURE 4.16 Enhancement via fusion: (a) original mammogram, (b) contrast enhancement by multiscale analysis, and (c) enhancement obtained by fusion of enhanced features. (Reprinted from [118] with permission from IEEE.)

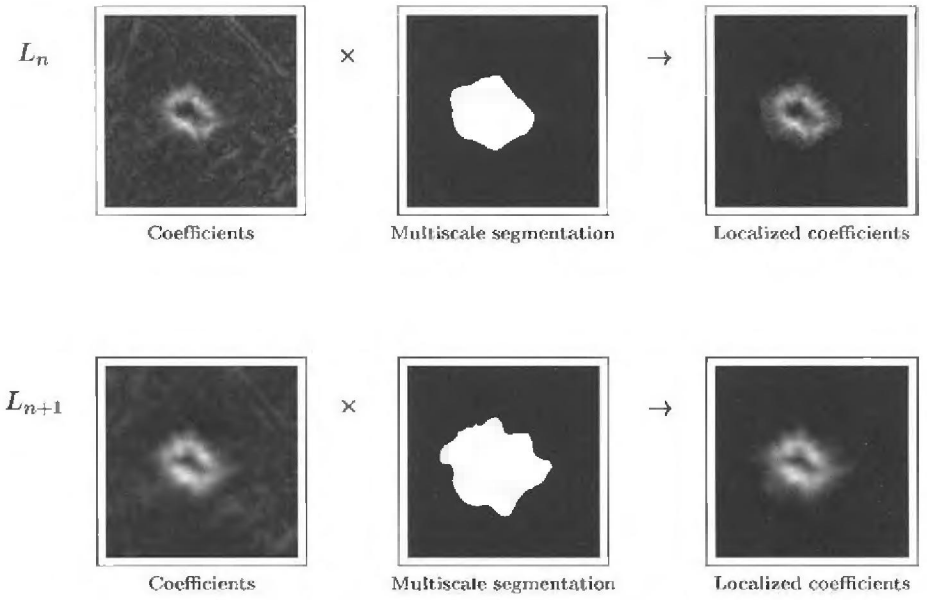


FIGURE 4.17 Localized wavelet coefficients of a suboctave band for a first-order derivative approximation of a smoothing function at two levels of analysis. Top row: finer scale. Bottom row: coarser scale. (Images courtesy of Dr. A. Laine, Columbia University.)

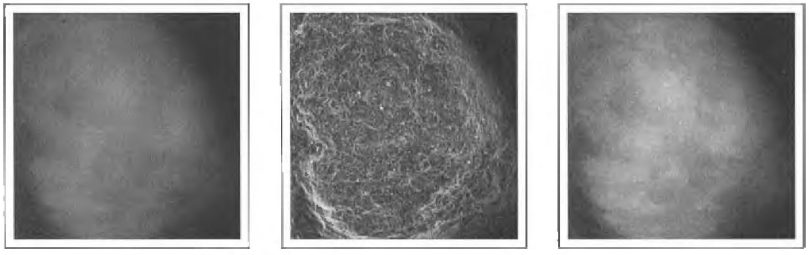


FIGURE 4.18 Fused image (c) of an original mammogram (a) with the selected modulus of dyadic wavelet coefficients (b). (Images courtesy of Dr. A. Laine, Columbia University.)

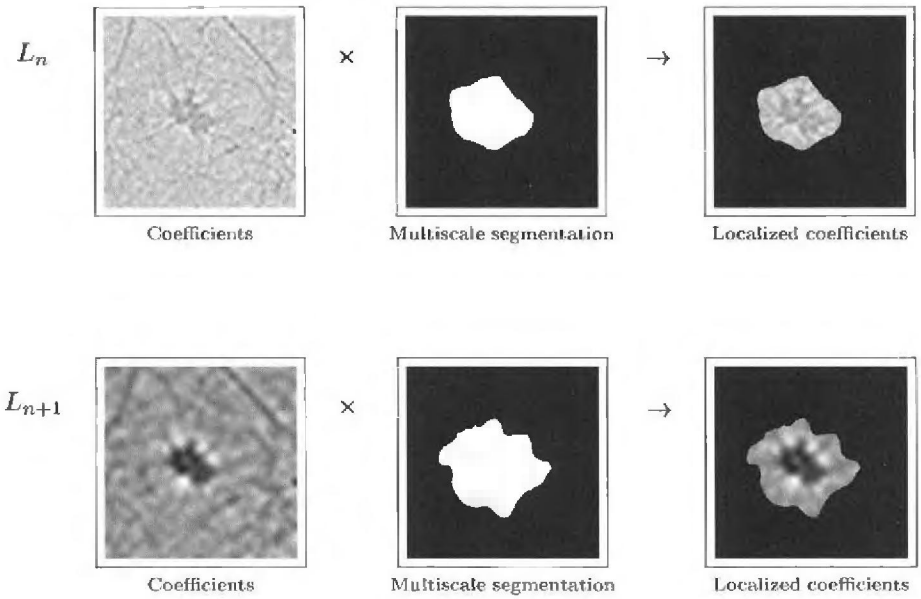


FIGURE 4.19 Localized wavelet coefficients of a suboctave band for a first-order derivative approximation of a smoothing function at two levels of analysis. Top row: finer scale. Bottom row: coarser scale. (Images courtesy of Dr. A. Laine, Columbia University.)

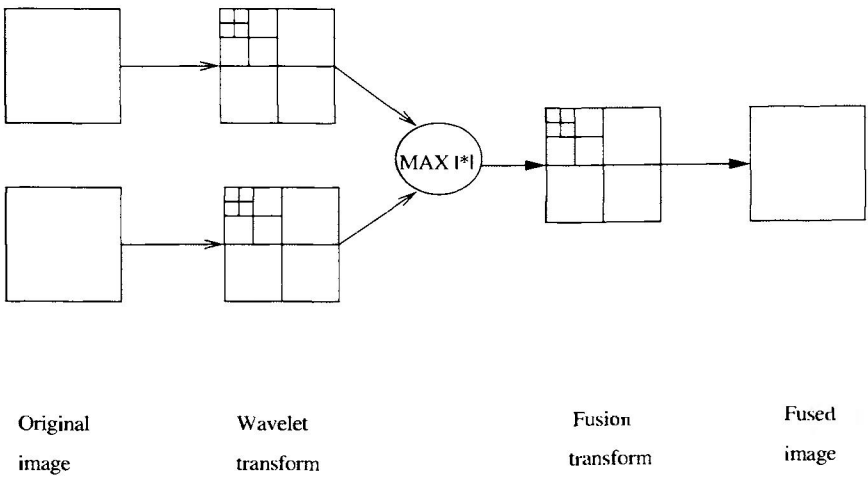


FIGURE 4.20 The block diagram of the image fusion scheme.

EXERCISES

1. Compute the two-dimensional WT with respect to Haar wavelets of the following 2×2 image:

$$\mathbf{F} = \begin{bmatrix} 4 & -1 \\ 8 & 2 \end{bmatrix} \tag{4.48}$$

   Draw the corresponding filter bank.

2. Determine the most suitable wavelet transform in an image for:
   a) Detecting lines.
   b) Compressing mammograms.
   c) Performing image fusion.

3. Write a program to compute the CWT of an image and use it to determine small objects in the image.

4. Write a program to compute a wavelet series expansion of a medical image and use this program to compress the image.

5. Write a program to compute the DWT of a medical image and use this program to detect edges in the image.

6. Write a program to compute the DWT of a medical image and use this program to denoise the image by thresholding.

7. Write a program to compute the DWT of a medical image and use this program to detect edges in the image.

8. Write a program to compute the DWT of a medical image and use this program for nonlinear contrast enhancement of the image.

V

GENETIC ALGORITHMS

5.1 INTRODUCTION

Genetic algorithms (GA) like neural networks are biologically inspired and represent a new computational model having its roots in evolutionary sciences. Usually GAs represent an optimization procedure in a binary search space, and unlike traditional hill climbers they do not evaluate and improve a single solution but a set of solutions or hypotheses, a so-called population. The GAs produce successor hypotheses by mutation and recombination of the best currently known hypotheses. Thus, at each iteration a part of the current population is replaced by offspring of the most fit hypotheses. In other words, a space of candidate hypotheses is searched in order to identify the best hypothesis, which is defined as the optimization of a given numerical measure, the so-called hypothesis fitness. Consider the case of function approximation based on given input-output samples: The fitness is the accuracy of the hypothesis (solution) over the training set.

The strength of this parallel process is enhanced by the mechanics of population modification, making GAs adequate candidates even for NP-hard problems. Mathematically, they are function optimizers and they encode a potential solution based on chromosome-like data structures. The

critical information is preserved by applying recombination operators to these structures. Their most interesting properties are [164]:

- Efficiency
- Simple programmability
- Extraordinary robustness regarding the input data

The most important property is robustness, and this represents an emulation of nature's adaptation algorithm of choice. Mathematically, it means that it is possible to find a solution even if the input data do not facilitate finding such a solution. GAs are mostly applied to nondifferentiable functions and functions with many local optima. Besides the abovementioned advantages, there is one major disadvantage: GAs have to be carefully designed. An unfavorable choice of operators might affect the outcome of the application. Therefore, a precise knowledge of the basics and the context is crucial for any problem solution based on GAs.

Usually, the implementation of a GA starts with the random definition of a chromosome population. The structures are evaluated and then better reproductive opportunities are allocated for those chromosomes, leading to better solutions. The adequacy of a solution is typically related to the current population average, or the median population.

Let us look again at the definition of a GA. In a strict sense, the GA defines a model introduced and investigated by John Holland in 1975 [86]. This computational paradigm — known also as the canonical genetic algorithm — is still a relevant research topic. In a more detailed sense, the GA represents a population-based model, which employs selection and recombination operators to generate new data points in a search space [241]. There are several GA models known in the literature, most of them designed as optimization tools for several applications.

GAs differ from classical optimization and search procedures: (1) direct manipulation of a coding, (2) search from a population of points and not a single solution, (3) search via sampling, a so-called blind search, and (4) search using stochastic operators, not deterministic rules.

In this section, we will review the basics of GAs, briefly describe the schema theorem and the building block hypothesis, and describe feature selection based on GAs, as one of the most important applications of GAs.

5.2 ENCODING AND OPTIMIZATION PROBLEMS

GAs have generally two basic parts that are problem-oriented: the problem encoding and the evaluation problem [65].

Let us consider the problem of minimizing (or maximizing) a function $F(x_1, x_2, \ldots, x_M)$. The goal is to optimize an output by finding the corresponding parameter combination. Since most of the problems are nonlinear, it is not possible to optimize every single parameter, but the

whole set of parameters revealing the interactions between them must be considered.

To solve this optimization problem, two assumptions have to be made. The first assumption concerns the representation of parameter variables as bit strings (concatenation of bits from the set $\{0, 1\}$). Most test functions have a length of at least 30 bits. Considering the nonlinear nature of the optimization problem, we see immediately that the size of the search space is related to the number of bits used for the problem encoding. Considering a bit string of length L, we obtain a hypercube for the search space with a size of 2^L. In other words, the GA samples the corners of this L-dimensional hypercube. The success of the algorithm depends on the correct encoding. Binary encoding has a major disadvantage: the weighted representation. A bit has a larger weight if it is positioned more to the left in a bit string. This means that the same operator applied to a bit string achieves different results depending on the bit position in a string. This is not always acceptable, but applying the Gray code to the string corrects this.

The second assumption concerns the evaluation function, which is usually given as part of the problem description. It is desired that the evaluation function be easily computed. Although this is valid for all optimization problems, it is of special interest for GAs. GAs do not employ just one solution, but are based on a population of solutions being replaced for every iteration with their offspring. The offspring represent a new population that has to be evaluated anew. It is evident that the time factor plays an important role in that sense.

Most of the definitions used in context with GAs have their roots in genetics but have an equivalent in information theory. For a better visualization, we can find those correspondents in Table 5.1.

TABLE 5.1 Definition analogies.

Information theory	Biology/genetics
Vector, string	Chromosome
Feature, character	Gene
Feature value	Allele
Set of all vectors	Population

5.3 THE CANONICAL GENETIC ALGORITHM

Like all other optimization methods, GAs are searching iteratively for the optimal solution to a posed problem. Unlike the other methods, GAs are employing some very simple techniques for information processing. These usually are operations for string manipulations such as copying, inserting, deleting, and replacing of string parts and random number generation. Usually, the initial population is generated randomly. The simplicity of

operation with the achieved computational power makes the genetic algorithm approach very appealing for a multitude of practical problems.

The most common operators are [241]:

- **Selection:**
  Based on selection, we can choose the best strings out of a set. The strings in the current population are copied in proportion to their fitness and placed in an intermediate generation. Through selection, we can ensure that only the fittest strings perpetuate. In a broader sense, we have a natural selection which enables that the strings of the new generation are closer to the desired final results. Mathematically, it means that the convergence of the population is given.
- **Crossover:**
  Crossover describes the swapping of fragments between two binary strings at a randomly choosen crossover point. In other words, it creates two new offspring from two parents. After recombination, two new strings are formed, and these are inserted into the next population. In summary, new sample points are generated by recombining two parent strings. Consider the following two strings 000111000 and 111000111. Using a single randomly chosen crossover point, recombination occurs as follows:

  $$000|111000$$
  $$111|000111$$

  The following offspring are produced by swapping the fragments between the two parents

  $$000000111 \quad \text{and} \quad 111111000 \tag{5.1}$$

  This operator also guarantees the convergence of the population.
- **Mutation:**
  This operator does not represent a critical operation for GAs. It produces new alleles on a random basis and is relatively simple in its concept. Mutation randomly generates a new bit or a new bit sequence in a given string, usually by flipping the bit or bits in the sequence. Mutation prevents an early convergence since it produces divergence and inhomogeneity within the population. On the other hand, it ensures that the population converges to a global maximum instead of a local maximum. What is much more, based on the random changes applied to the genetic material, new combinations are produced, which lead to better solutions. Mutation can be viewed as a way out of getting stuck in local minima and is often

performed after crossover has been applied. Based on the arbitrary changes produced by this operator, one should employ mutation with care. At most one out of 1,000 copied bits should undergo a mutation.

Apart from these very simple operations, many others have been proposed in the literature [164].

The implementation of a GA requires the generation of an initial population. Each member of this population represents a binary string of length L, called the chromosome, and thus determines the problem encoding. Usually, the initial population takes random values. After generating an initial population, every single string is evaluated and then its fitness value is determined. One should carefully distinguish between the fitness function and the evaluation function in the context of GAs. The evaluation function represents a performance measure for a particular set of parameters. The fitness function gives, based on the measured performance, the chance of reproductive opportunities. In other words, it defines the criterion for ranking potential hypotheses and for probabilistically selecting them for inclusion in the population of the next generation. In the case of learning classification rules, the fitness function determines the ranking of classification accuracy of the rule over a set of available training samples. The evaluation of a string describing a particular set of parameters is not related to any other string evaluation. However, the fitness of that string is related to the other chromosomes of the current population. Thus, the probability that a hypothesis is chosen is directly proportional to its own fitness, and inversely proportional to the rest of competing hypotheses for the given population.

For canonical GAs the definition of the fitness is given by $f_i/\bar{f}$, where f_i is the evaluation associated with chromosome i and $\bar{f}$ is the average evaluation of all strings in the population:

$$\bar{f} = \frac{1}{n}\sum_{i=1}^{n} f_i \tag{5.2}$$

The execution of a GA can be viewed as a two-level process. In the beginning, we have the current population. Based on selection we obtain an intermediate population, and afterward based on recombination and mutation we obtain the next population. The procedure of generating the next population from the current population represents one generation in the execution of a GA. Figure 5.1 shows this procedure [241].

The intermediate population is generated from the current population. In the beginning, the current population is given by the initial population. Next $f_i/\bar{f}$ is determined for all chromosomes of the current population, and then the selection operator is employed. The strings of the current population are copied or duplicated proportional to their fitness and then

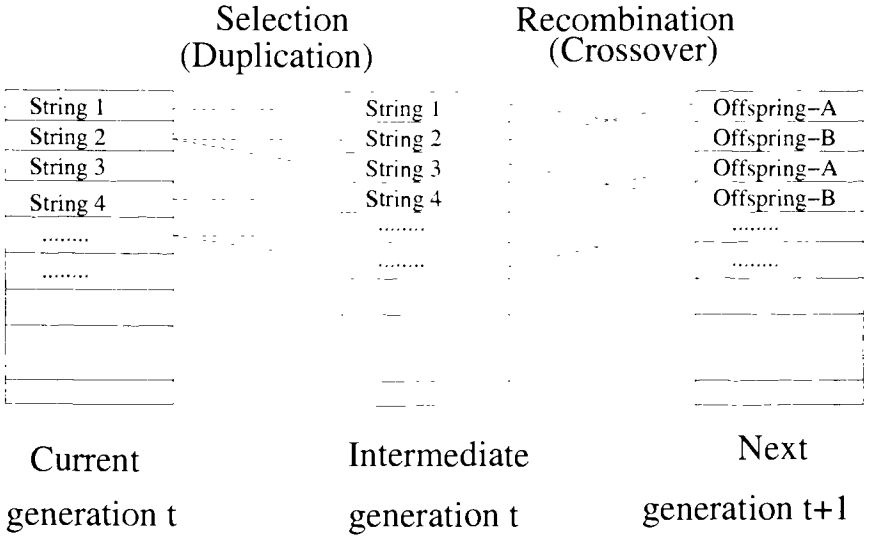


FIGURE 5.1 Splitting of a generation into a selection and recombination phase.

entered in the intermediate generation. If, for a string i, we obtain $f_i/\bar{f} > 1.0$, then the integer portion of fitness determines the number of copies of this string that directly enter the intermediate population. A string with a fitness of $f_i/\bar{f} = 0.74$ has a 0.74 chance of placing one string in the intermediate population while a string with a fitness of $f_i/\bar{f} = 1.53$ places one copy in the intermediate population. The selection process continues until the intermediate population is generated.

Next, the recombination operator is carried out. The recombination is a process of generating the next population from the intermediate population. Then crossover is applied to a pair of strings chosen in a random fashion to produce a pair of offspring chromosomes which are syntheses of their parents. These strings are recombined with a probability of p_c and the newly generated strings are included in the next population. The last operation needed for producing the next population is the mutation operator. The mutation probability of a bit p_m is very small, usually $p_m \ll 1\%$. For practical applications, we normally choose p_m close to 0.01. Mutation changes the bit values and produces a near-identical copy with some components of the chromosome altered. Selection, recombination, and mutation are applied for each population. Combined, these operators form one generation in the execution of a GA.

The algorithmic description of a GA is given below:

ALGORITHM 1.

1. Generate the initial population randomly for the strings a_i:
2. $\Pi = \{a_i\}, i = 1, \ldots, n.$
3. For $i \leftarrow 1$ to *Numberofgenerations* do
4. **begin**
5. Initialize mating set $M \leftarrow \varnothing$ and Offspring O
6. For $j \leftarrow 1$ to n do
7. Add $f(a_i)/\bar{f}$ copies from a_i to M.
8. For $j \leftarrow 1$ to $n/2$ do
9. Choose two parents a_j and a_k from M and perform with the probability p_c $O = O \cup Crossover(a_j, a_k)$.
10. For $i \leftarrow 1$ to n do
11. For $j \leftarrow 1$ to d do
12. Mutate with the probability p_m the jth bit from $a_i \in O$
13. Update the population $\Pi \leftarrow$ combine(Π, O).
14. **end**

5.4 OPTIMIZATION OF A SIMPLE FUNCTION

A GA represents a general optimization method that searches a large space of candidates looking for a candidate that performs best according to the fitness function. It is not always guaranteed that an optimal candidate will be found; however, in most cases GAs succeed in finding a candidate with high fitness. Their most important application area is machine learning, where they have been successfully applied both to choosing the network topology for neural networks and to function-approximation.

In this section, we will apply the most important basic operations of a GA to an example of function optimization [164].

The following function is considered:

$$g(x) = x \cdot \sin(10\pi \cdot x) + 1 \tag{5.3}$$

The goal is to find, based on a GA, the maximum of this function in the interval $[-1 \ldots 2]$:

$$g(x_0) \geq g(x), \quad \text{for all} \quad x \in [-1 \ldots 2] \tag{5.4}$$

To solve this optimization problem, some typical GA operators are employed.

5.4.1 Number Representation

The real-valued x have to be transformed into a binary vector (chromosome). The vector length is determined by the required precision, which for this example is six decimal places. The real-valued domain of the number x has the length of 3. The required precision means that the interval

$[-1 \ldots 2]$ is split in 3,000,000 equal parts. This means that 22 bits are needed to represent a binary vector (chromosome):

$$2{,}097{,}152 = 2^{21} < 3{,}000{,}000 \leq 2^{22} = 4{,}194{,}304 \tag{5.5}$$

The transformation of a binary number $< b_{21}b_{20} \ldots b_0 >$ into a real number x is performed in two distinct steps:

- Transform the binary number $< b_{21}b_{20} \ldots b_0 >$ from basis 2 into basis 10:

$$(< b_{21}b_{20} \ldots b_0 >)_2 = (\sum_{i=0}^{21} b_i \cdot 2^i)_{10} = x' \tag{5.6}$$

- Determine the corresponding real number x:

$$x = -1 + x' \cdot \frac{3}{2^{22} - 1} \tag{5.7}$$

where -1 is the left limit of the interval and 3 is the interval length.

5.4.2 Initial Population

The initial population is randomly generated. Each chromosome represents a 22-bit binary vector.

5.4.3 Evaluation Function

The evaluation function f of the binary vector $\mathbf{v}$ is equivalent to the function $g(x)$:

$$f(\mathbf{v}) = g(x) \tag{5.8}$$

The three given x-values $x_1 = 0.637197$, $x_2 = -0.958973$, and $x_3 = 1.627888$ correspond to the following three chromosomes:

$$\mathbf{v}_1 = (1000101110110101000111),$$
$$\mathbf{v}_2 = (0000001110000000010000),$$
$$\mathbf{v}_3 = (1110000000111111000101)$$

The evaluation function provides the following values:

$$f(\mathbf{v}_1) = g(x_1) = 1.586345$$
$$f(\mathbf{v}_2) = g(x_2) = 0.0078878$$
$$f(\mathbf{v}_3) = g(x_3) = 2.250650$$

We immediately see that $\mathbf{v}_3$ is the fittest chromosome since its evaluation function provides the largest value.

5.4.4 Genetic Operators

While the GA is executed, two distinct operators are employed to change the chromosomes: mutation and crossover.

We mutate first the fifth and then the 10th gene of the chromosome $\mathbf{v}_3$. We thus obtain

$$\mathbf{v}_3' = (11101000001111111000101) \tag{5.9}$$

and, respectively,

$$\mathbf{v}_3'' = (11100000011111111000101) \tag{5.10}$$

This leads for $x_3'' = 1.630818$ and for $g(x_3'') = 2.343555$ to an increase in value to $g(x_3) = 2.250650$. Crossover is now applied to chromosomes $\mathbf{v}_2$ and $\mathbf{v}_3$. The fifth gene is chosen as a crossover point:

$$\mathbf{v}_2 = (00000|01110000000010000)$$
$$\mathbf{v}_3 = (11100|00000111111000101)$$

Swapping the fragments between the two parents produces the following offspring:

$$\mathbf{v}_2' = (00000|00000111111000101)$$
$$\mathbf{v}_3' = (11100|01110000000010000)$$

Their evaluation functions is determined as

$$f(\mathbf{v}_2') = g(-0.998113) = 0.940865$$
$$f(\mathbf{v}_3') = g(1.666028) = 2.459245$$

The second offspring has a better evaluation function value than its parents.

5.4.5 Simulation Parameters

To determine the solution of the given optimization problem, we will choose the following parameters: the population consists of 50 distinct chromosomes, the crossover probability is $p_c = 0.25$, and the mutation probability is $p_m = 0.01$.

5.4.6 Simulation Results

Table 5.2 shows the generations where both the evaluation function and the value of the function have improved at the same time. The best chromosome after 150 generations is

$$\mathbf{v}_{max} = (11110011010000100000101) \tag{5.11}$$

Its value is $x_{max} = 1.850773$. We could demonstrate that the GA converges toward the maximum of the given function.

TABLE 5.2 Simulation results after 150 generations.

Generation	Evaluation function
1	1.441942
6	2.250003
8	2.250283
9	2.250284
10	2.250363
12	2.328077
39	2.344251
40	2.345087
51	2.738930
99	2.849246
137	2.850217
145	2.850227

5.5 THEORETICAL ASPECTS OF GENETIC ALGORITHMS

To obtain an understanding of GA performance, we realize that by exploiting important similarities in the coding we use for the raw available data, the search algorithm becomes more efficient. Thus, appropriate coding leads to good performance. This led to the development of the notion of similarity template, or schema, and further to a keystone of the genetic algorithm approach, the building block hypothesis.

The theory of GA relies on two concepts: representation of solutions in the form of binary strings, and the schemata concept for determination of similarity between chromosomes [164]. Strings that contain $*$ are referred to as schemata. They are more general than completely specified strings, since $*$ means don't care and stands for both 0 and 1. By specifying all the $*$ in a schema, we obtain a string named an instance.

The mathematical interpretation of this situation is very interesting. We recall that a GA can result in complex and robust search by implicitely sampling hyperplane partitions of a search space. A vector of length n defines a hyperspace M^n. Each schema corresponds to a hyperplane in the search space. For example, by using the data set $\{0, 1, *\}$ we have in M^n exactly 3^n schemata. Each element is an instance of 2^n schemata. A population of p chromosomes (strings of length n) has between 2^n and $p2^n$ schemata.

5.5.1 The Schema Theorem

The order of a schema $o(h)$ refers to the number of actual bit values contained in the schema. In other words, it is the number of fixed positions (in a binary alphabet, the number of 0's and 1's) present in the template. For a schema $1**0**1*$ we obtain $o(h) = 3$. The *defining length* of a schema is determined by the distance between the first and the last bit in the schema with value either 0 or 1. Thus, the defining length of the schema $1**0**1*$ is $l(h) = 6$, because the index of the leftmost occurrence of either 0 or 1 is 7 and the index of the rightmost occurrence is 1.

Schemata and their properties are important tools for discussing and classifying string similarities. Furthermore, they provide the basic means of analyzing reproduction and genetic operators on building blocks contained in the population.

There are some important aspects with respect to a schema:

- Schemata with a large defining length are very sensitive when carrying out crossover. It might happen that after crossover a string is no longer an instance of a schema.
- Schemata of a high order are very sensitive when mutation is carried out. However, schemata with many $*$ are less sensitive.
- The greater the fitness of a schema, the higher the probability that its instances are considered by the selection process. Therefore, schemata with a small fitness have a lower chance because their instances do not survive.

These observations can be summarized in the following theorem:

THEOREM 5.5.1:
The best chances to reproduce in the next generations belong only to those schematas of GAs that have a large fitness value, a short defining length, and a low order.

In other words, short, low-order above-average schemata will tend to grow in influence. Thus, the schema theorem roughly states that more fit schemas will gain influence, especially those containing a small number of defined bits preferably close to each other in the binary string.

Some very important conclusions can be drawn from the above theorem:

- Parts of a chromosome that encode similar information should be placed in neighboring positions on the string. This minimizes the probability of disruption during crossover.
- The segments of a string contributing decisively to the fitness value of that string should be efficiently coded so that the best chromosomes can perpetuate.

The schema theorem is a basic concept for the theory of GAs. However, in a way it is incomplete since it fails to consider the assumed positive contributions of crossover and mutation.

5.5.2 The Building Block Hypothesis

Schemata provide a better understanding of GAs. Short, low-order, and more fit schemata are sampled, recombined, and resampled to form strings of improved fitness. Using these schemata, the complexity of the optimization problem can be dramatically reduced: improved strings are formed continuously from the best partial solutions of past samplings.

Schemata with a high fitness value and at the same time of a low order are defined as building blocks [164]. They have several advantages compared with the regular schemata. They are hardly affected by crossover and mutation and have the best chances to be selected. In other words, they represent good partial solutions.

In this regard, we can state the following hypothesis:

Hypothesis: The fitness and thus the performance of the GA can be dramatically improved by combining building blocks.

This reduces the complexity of the posed problem. It is longer necessary to test all possible allele combinations, because highly fit strings can be formed based on the combination of building blocks. In other words, the experience of the last populations is included.

It is important to note that simple GAs depend upon the recombination of building blocks to seek the best points. In cases where the building blocks are misleading because of the coding used or the function itself, the problem may require long computational time to achieve near optimal solutions, and thus the performance of the GA is reduced.

As a closing remark to GAs. it is useful to emphasize the search method employed by evolutionary computing. GAs use a randomized beam search method to find a maximally fit hypothesis. This search method is quite different from the gradient descent method employed by most unsupervised statistical or cognitive methods. The gradient descent method moves smoothly from one hypothesis to a very similar new one, while GAs move quickly to a radically different hypothesis. Thus, the GA search is not as prone to getting stuck in local minima like gradient descent methods. However, there is one drawback with GAs: crowding. Crowding describes the fact that the fittest chromosomes reproduce quickly. and thus copies of those and similar chromosomes dominate the population. This means that the diversity of the population is reduced, and thus learning slows down.

5.6 FEATURE SELECTION BASED ON GENETIC ALGORITHMS

GAs are excellent candidates to reduce the dimension of a feature vector. They are among the few large-scale feature selection algorithms that can be efficiently applied to many classification problems [219].

The search space can be represented by an undirected graph, where the nodes correspond to points in the search space while an existing edge between two nodes shows that the feature set associated to one of these two connected nodes is either a subset or superset of the associated set of the second node. A four-dimensional feature selection lattice is shown in Figure 5.2.

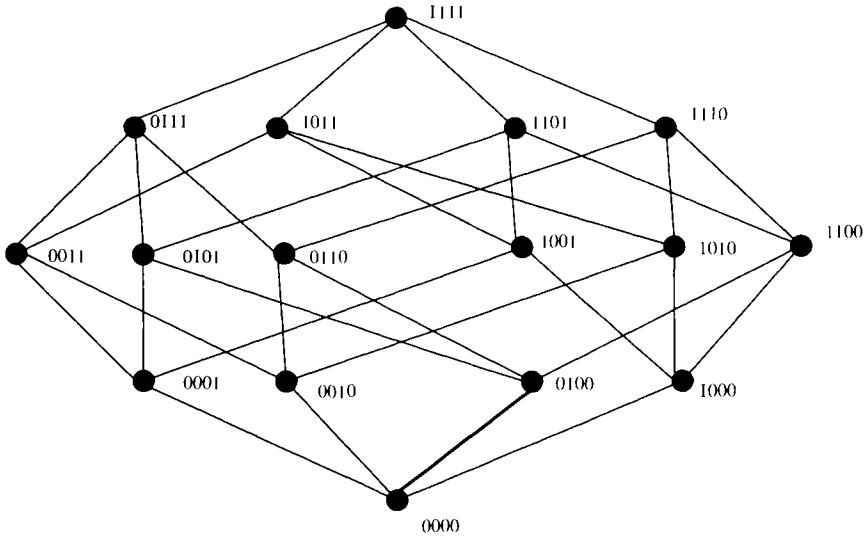


FIGURE 5.2 Four-dimensional feature selection graph [219].

The top node in the graph contains the whole feature subset while the bottom node contains the empty set. The topology of the graph in Figure 5.2 reveals its level-structure with the bottom node being at level 0 and the top node being at level 4.

To find the subsets, we have to code each subset as a d-dimensional binary string, $a = (\alpha_1, \ldots, \alpha_d)$, the so-called feature selection vector. In this notation, d represents the initial number of features and α_i the feature selection variable with

$$\alpha_i = \begin{cases} 1, & \text{feature present in subset} \\ 0, & \text{else} \end{cases} \tag{5.12}$$

We immediately see that the search takes place in a binary d-dimensional space.

To better understand the concept of feature selection based on GAs, the monotonicity property for nodes was introduced in [219]. It is said that a function $e(\cdot)$ is monotonic, if for every pair of linked nodes in the lattice, a and b, the following holds:

$$\text{if} \quad l(a) < l(b) \quad \text{then} \quad e(a) > e(b) \tag{5.13}$$

where $l(\cdot)$ represents the level function

$$l(a) = \sum_{i=1}^{n} \alpha_i \tag{5.14}$$

In the following we will review the algorithm described in [219]. The approach determines the feasible subset of features for which the classifier's error rate is below the so-called feasibility threshold. In other words, it is searched for the smallest subset of features among all feasible subsets.

This yields to a constrained optimization problem, which is adapted for GAs by introducing a penalty function $p(e)$:

$$p(e) = \frac{\exp((e-t)/m) - 1}{\exp(1) - 1} \tag{5.15}$$

e is the error rate, t the feasibility threshold, and m a scale factor describing the tolerance margin. $p(e)$ is monotonic with respect to e. It can be shown that $p(e)$ approaches its minimum for $e < t$ and $e \to 0$:

$$p(0) = \frac{\exp(-t)/m) - 1}{\exp(1) - 1} > -\frac{1}{\exp(1) - 1} \tag{5.16}$$

Also, we obtain $p(t) = 0$ and $p(t + m) = 1$. There is a trade-off between penalty function and error rate: A large value for an error rate leads to a penalty function going toward infinity. The score $J(a)$ is obtained by adding the obtained penalty function to the level function

$$J(a) = l(a) + p(e(a)) \tag{5.17}$$

$l(a)$ is the level function representing the number of features in the evaluated binary subset a and represents the cost of extracting features.

Based on the properties of the penalty function described in eq. (5.15), it was shown in [219] that:

1. Feature subsets or chromosomes in GAs parlance producing an error rate below a feasibility threshold receive a negative penalty.
2. The adaptation of feature subsets is based on the associated error rates measured by the score from eq. (5.17).
3. It is possible to achieve a better subset at a certain level than that associated with the next higher level. This can be accomplished if

feature subsets whose error rate e fullfills $t < e < t + m$ receive a small penalty.

4. On the other hand, if the error rate $e > t + m$, then the feature subsets receive a high penalty and cannot compete with subsets at the next higher level in the feature selection graph.

It remains now only to define the fitness function and the reproduction rule. Let $\Pi = \{a_1, \ldots, a_n\}$ be the population of feature selection vectors or chromosomes. The goal is to determine a minimum of the score $J(a)$ from eq. (5.17) and thus to find an adequate fitness function,

$$f(a_i) = (1 + \epsilon) \max_{a_j \in \Pi} J(a_j) - J(a_i) \tag{5.18}$$

with ϵ a small positive constant such that $\min f(a_i) > 0$. In other words, even the least fit chromosomes get a chance to reproduce. The reproduction operator selects the best n chromosomes from $\Pi \cup O$.

Examples of how to use the GA in a large-scale feature selection are shown in Chapter XI.

EXERCISES

1. Consider the function

$$g(x) = 21.5 + x sin(4\pi x) \tag{5.19}$$

Find the maximum of this function in the interval $[-3 \ldots 12.1]$ by employing a GA.

2. Apply the GA to determine an appropriate set of weights for a $3 \times 2 \times 1$ multilayer perceptron. Encode the weights as bit strings and apply the required genetic operators. Discuss the advantages and disadvantages of using a GA for the weights' learning instead of the traditional backpropagation.

3. Consider the function

$$J = \sum_{i=1}^{N} d(\mathbf{x}, C_{\mathbf{v}_i}) \tag{5.20}$$

where $d(\mathbf{x}, C_{\mathbf{v}_i})$ describes the distance between an input vector $\mathbf{x}$ and a set using no representatives for the set. Propose a coding of the solutions for a GA that uses this function. Discuss the advantages and disadvantages of this coding.

VI

STATISTICAL AND SYNTACTIC PATTERN RECOGNITION

6.1 INTRODUCTION

Modern classification paradigms such as neural networks, genetic algorithms, and neuro-fuzzy methods have gained an increasing importance over the past decade. However, the traditional and noncognitive classification methods, though challenged by these new approaches, are still in place and sometimes provide a better solution than the modern ones.

Noncognitive classification paradigms encompass statistical and structural (syntactic) methods. If there is an underlying and quantifiable statistical basis for the generation of patterns, then we refer to statistical pattern recognition. Otherwise, the fundamental information is provided by the underlying pattern structure, and we refer to structural or syntactic pattern recognition. The latter classification paradigm is less popular than the statistical classification methods.

This chapter gives an overview of the most important approaches in statistical and syntactic pattern recognition and their application to biomedical imaging. Parametric and nonparametric estimation methods and binary decision trees form the basis for most classification problems related to bioimaging, whereas grammatical inference and graphical meth-

ods are the basic classification paradigms in syntactic pattern recognition. The chapter also reviews the diagnostic accuracy of classification measured by ROC-curves and presents application examples based on statistical classification methods.

6.2 LEARNING PARADIGMS IN STATISTICAL PATTERN RECOGNITION

Pattern recognition techniques are widely used for many miscellaneous practical problems. It can be either the statistical nature of pattern generation of special interest or the relevant information that is captured by the pattern structure. This chapter deals with the design of a classifier in a pattern recognition system. There are several classification paradigms which lead to a reasonable solution: statistical, syntactic, or neural classification. Statistical and neural pattern recognition can be further differentiated into supervised and unsupervised.

6.2.1 Supervised Pattern Recognition

Here we assume that a set of training data is available, and the classifier is designed by exploiting this a priori known information. The most relevant techniques which belong to this group are [59]:

- Maximum likelihood method: estimates unknown parameters using a set of known feature vectors in each class.
- Bayes method: assumes a known a priori probability and minimizes the classification error probability. Also the class-conditional probability density functions describing the distribution of the feature vectors in each class must be known. If not known, they can be estimated from the available training data.
- Minimum distance classifier: classifies an input vector based on its distance to the learned prototypes.
- Entropy criteria: classify an input vector based on the minimization of the randomness of this vector.

6.2.2 Unsupervised Pattern Recognition

Unsupervised pattern recognition concerns another type of pattern recognition task for which training data, of known class labels, are not available. Here, we deal with a given set of feature vectors, and the goal to find out the underlying similarities and cluster (group) "similar" vectors together. Classifiers can be designed as either linear or nonlinear depending on the data separability.

The most relevant techniques which belong to this group are [59]:

- Isodata or k-means or c-means clustering: the goal is to achieve a close partitioning of the data space. The number of classes and the

initial values of the class centers need to be known a priori. Learning is an iterative process which adapts the class centers according to the training data.

- Vector quantization: the input data are mapped onto a given number of code vectors (prototypes) which together form the code book. The goal is to achieve a code book with least possible distortion.
- Cluster swapping: is applied mostly to a large data space, aims to avoid a suboptimal class partitioning. Based on a predefined criterion function, the classification as a whole is supervised during each iteration. This can prevent two feature vectors of different classes but erroneously assigned to the same class from belonging forever to this class.
- Hierarchical clustering: produces instead of a single clustering, a hierarchy of nested clustering. More specifically, these algorithms have as many steps as the number of data vectors. At each step a new clustering is obtained based on the clustering produced at the previous step. There are two main categories of hierarchical algorithms: agglomerative and divisive hierarchical algorithms. Agglomerative clustering is a bottom-up procedure, which starts with as many clusters as the number of feature vectors. The cluster number is reduced by merging clusters describing the same classes. In this context, the dendrogram is a useful tool of representing the sequence of clustering produced by an agglomerative algorithm. This procedure continues until the final clustering, which contains as a single set the set of data. The divisive clustering is a top-down procedure and follows the inverse path of the agglomerative clustering. In the first step, the clustering consists of a single set. The following steps increase the number of clusters.

Statistical classification approaches are built on probabilistic arguments stemming from the statistical nature of generated features. The goal is to design a classifier that classifies a pattern into the most probable of the classes.

Decision rules necessary for performing the classification task can be formulated in several ways:

- An a priori known class-related probability $P(\omega_i)$ is transformed into an a posteriori probability $P(\omega_i|\mathbf{x})$. $\mathbf{x}$ is an n-dimensional feature vector.
- Choice of a classification error rate and of a decision rule.

In the following, we will consider the standard classification problem. An unknown feature vector $\mathbf{x}$ has to be assigned to one of the M classes, $\omega_1, \omega_2, \ldots, \omega_M$. M conditional or a posteriori probabilities $P(\omega_i|\mathbf{x})$ with $i = 1, 2, \ldots, M$ can be defined. Each of them describes the probability

that an unknown pattern belongs to a given class ω_i. The goal of the classification is to determine the maximum of these M probabilities, or equivalently, the maximum of an appropriately defined function of them.

6.3 PARAMETRIC ESTIMATION METHODS

Statistical pattern recognition assumes the knowledge of $p(\mathbf{x}_i|\omega_i)$ and $P(\omega_i)$ for each class. In addition, the number of classes needs to be known a priori.

Let us also assume that we have a set of training samples representative of the type of features and underlying classes, with each labeled as to its correct class. This yields a learning problem. When the form of the densities is known, we are faced with a parameter estimation problem.

This section reviews techniques for estimating probability density functions (pdfs) based on the available experimental data that are described by the feature vectors corresponding to the patterns of the training set.

Every pdf can be characterized by a specific parameter set. One of the most frequently used pdfs is the Gaussian or normal density function. This pdf is completely characterized by the mean value and covariance matrix, which have to be estimated for each class ω_i from the training data set. This technique is known under the name of parametric estimation.

There are two approaches to parameter estimation known in the literature:

- Maximum likelihood estimation: It is assumed the parameters are fixed, but unknown. This approach seeks the "best" parameter-set estimate that maximizes the probability of obtaining the (given) training set.
- Bayesian estimation: It uses the training set to update the training-set conditioned density function of the unknown parameters. The training set acts as "observations" and allows the conversion of the a priori information into an a posteriori density.

Both techniques yield an uncertainty in the parameter estimation. Whereas the Bayesian approach determines a density that approximates an impulse, the maximum likelihood approach determines parameter estimates that maximize a likelihood function.

6.3.1 Bayes Decision Theory

Bayes decision theory represents a fundamental statistical approach to the problem of pattern classification. This technique is based on the assumption that the decision problem is formulated in probabilistic terms, and that all relevant probability values are given. In this section, we develop the fundamentals of this theory.

A simple introduction to this approach can be given by an example focusing on the two–class case ω_1, ω_2. The a priori probabilities $P(\omega_1)$ and

$P(\omega_2)$ are assumed to be known since they can be easily determined from the available data set. Also known are the pdfs $p(\mathbf{x}_i|\omega_i)$, $i = 1, 2$. $p(\mathbf{x}_i|\omega_i)$ is also known under the name of the likelihood function of ω_i with respect to $\mathbf{x}$.

Recalling the Bayes rule, we have

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{p(\mathbf{x})} \tag{6.1}$$

where $p(\mathbf{x})$ is the pdf of $\mathbf{x}$, and for which it holds that

$$p(\mathbf{x}) = \sum_{i=1}^{2} p(\mathbf{x}|\omega_i)P(\omega_i) \tag{6.2}$$

The Bayes classification rule can now be stated for the two-class case ω_1, ω_2:

$$\begin{aligned} &\text{If} \quad P(\omega_1|\mathbf{x}) > P(\omega_2|\mathbf{x}), \quad \mathbf{x} \text{ is assigned to } \omega_1 \\ &\text{If} \quad P(\omega_1|\mathbf{x}) < P(\omega_2|\mathbf{x}), \quad \mathbf{x} \text{ is assigned to } \omega_2 \end{aligned} \tag{6.3}$$

We can immediately conclude from above this that a feature vector can be assigned to either one class or the other. Equivalently, we now can write

$$p(\mathbf{x}|\omega_1)P(\omega_1) \gtrless p(\mathbf{x}|\omega_2)P(\omega_2) \tag{6.4}$$

This corresponds determining the maximum of the conditional pdfs evaluated at $\mathbf{x}$. Figure 6.1 visualizes two equiprobable classes and the conditional pdfs $p(x|\omega_i)$, $i = 1, 2$ as function of x. The dotted line at x_0 corresponds to a threshold splitting the one-dimensional feature space into two regions R_1 and R_2. Based on the Bayes classification rule, all values of $x \in R_1$ are assigned to class ω_1, while all values $x \in R_2$ are assigned to class ω_2.

The probability of the decision error is given by

$$P_e = \int_{-\infty}^{x_0} p(x|\omega_2)dx + \int_{x_0}^{+\infty} p(x|\omega_1)dx \tag{6.5}$$

The Bayes classification rule achieves a minimal error probability. In [59] it was shown that the classification error is minimal if the partition of the feature set into the two regions R_1 and R_2 is chosen such that

$$\begin{aligned} R_1: &\quad P(\omega_1|\mathbf{x}) > P(\omega_2|\mathbf{x}) \\ R_2: &\quad P(\omega_2|\mathbf{x}) > P(\omega_1|\mathbf{x}) \end{aligned} \tag{6.6}$$

The generalization for M classes $\omega_1, \omega_2, \ldots, \omega_M$ is very simple. A feature vector $\mathbf{x}$ is assigned to class ω_i if

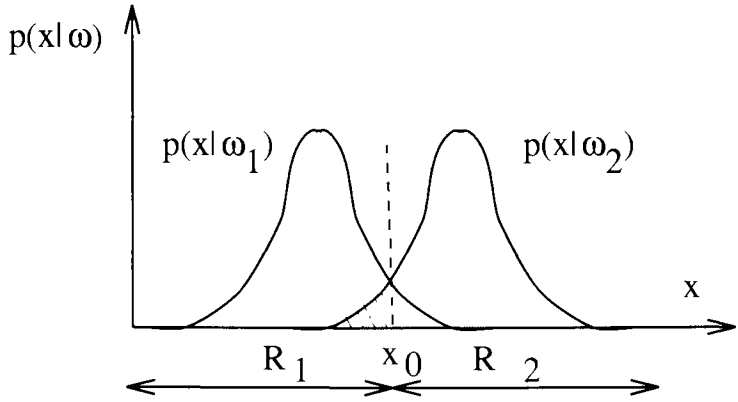


FIGURE 6.1 Two equiprobable classes and the corresponding regions R_1 and R_2.

$$P(\omega_i|\mathbf{x}) > P(\omega_j|\mathbf{x}) \quad \forall j \neq i \tag{6.7}$$

Every time we assign an object to a class, we risk making an error. In multiclass problems, some misclassifications can have more serious repercussions than others. A quantitative way to measure this, is given by a so-called cost function. Let $L(i,j)$ be the cost (or "loss") of assigning an object to class i when it really belongs to class j.

From the above, we see that a different classification possibility is achieved by defining a so-called cost term $L(i,j)$ with $i,j = 1,2,\ldots,M$. The penalty term is equal to zero, $L(i,j) = 0$, if the feature vector $\mathbf{x}$ is correctly assigned to its class, and larger than zero, $L(i,j) > 0$, if $\mathbf{x}$ is assigned to class ω_j instead of the correct class ω_i. In other words, there is only loss if misclassification occurs.

The conditional loss term $R_i(\mathbf{x})$ with respect to the class assignment of $\mathbf{x}$ is

$$R_i(\mathbf{x}) = \sum_{j=1}^{M} L(i,j)P(\omega_j|\mathbf{x}) \tag{6.8}$$

or, equivalently,

$$R_i(\mathbf{x}) = \sum_{j=1}^{M} L(i,j)p(\mathbf{x}|\omega_j)P(\omega_j) \tag{6.9}$$

For practical applications we choose $L(i,j) = 0$ for $i = j$, and $L(i,j) = 1$ for $i \neq j$.

Thus, given the feature vector, there is a certain risk involved in assigning the object to any group.

Based on the above definitions, we obtain a slightly changed Bayes classification rule: a feature vector $\mathbf{x}$ is assigned to a class ω_i for which $R_i(\mathbf{x})$ is minimal.

6.3.2 Discriminant Functions

There are many different possibilities for defining pattern classifiers. One method, which can be considered as a canonical form for classifiers, is the so-called discriminant functions. In the M class case, they are used to partition the feature space.

Also, in many situations it is simpler instead to work directly with probabilities, to deal with an equivalent probability function, for example, $g_i(\mathbf{x}) = f(P(\omega_i|\mathbf{x}))$, where $f(\cdot)$ is a monotonically increasing function. $g_i(\mathbf{x})$ is called the discriminant function.

Based on the above relationships, we get for eq. (6.7) the following equivalent representation or, simply, decision rule:

$$\mathbf{x} \quad \text{belongs to class} \quad \omega_i \quad \text{if} \quad g_i(\mathbf{x}) > g_j(\mathbf{x}) \quad \forall j \neq i \tag{6.10}$$

Assuming two regions R_1 and R_2 are neighboring, then they can be separated by a hyperplane in multidimensional space. The equation defining the separation plane

$$g_{ij}(\mathbf{x}) = g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0, \quad i, j = 1, 2, \ldots, M \quad i \neq j \tag{6.11}$$

The discriminant functions are extremely useful when dealing with Gaussian pdf. Let us consider the normal density function

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma_i^{-1}(\mathbf{x} - \mu_i)\right), \quad i = 1, \ldots, M \tag{6.12}$$

where μ_i is the mean value and Σ_i the covariance matrix of the class ω_i. The covariance matrix Σ_i is given by

$$\Sigma_i = E[(\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T] \tag{6.13}$$

In this case, we choose a monotonic logarithmic discriminant function $\ln(\cdot)$

$$g_i(\mathbf{x}) = \ln\left(p(\mathbf{x}|\omega_i)P(\omega_i)\right) = \ln p(\mathbf{x}|\omega_i) + \ln P(\omega_i) \tag{6.14}$$

and we thus obtain for the normal density function

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma_i^{-1}(\mathbf{x} - \mu_i) + \ln P(\omega_i) + c_i \tag{6.15}$$

where $c_i = -(n/2)\ln 2\pi - (1/2)\ln|\Sigma_i|$ is a constant.

6.3.3 Bayesian Classification for Tumor Detection

When dealing with tumor detection, we are considering only two pattern classes: abnormal and normal. In other words, two discriminant functions have to be considered: $g_{abnormal}(\mathbf{x})$ and $g_{normal}(\mathbf{x})$. A pattern vector is assigned to the abnormal class, if $g_{abnormal}(\mathbf{x}) > g_{normal}(\mathbf{x})$.

Assuming the a priori probabilities are equal for each class, we obtain from eq. (6.15) the following discriminant function for tumor detection:

$$g(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu) + \log|\Sigma| \tag{6.16}$$

If each of the two classes now has the same covariance matrix, they can be separated by a hyperplane, and hence the classifier is referred to as a linear classifier. If the covariance matrices are unequal, then the decision surfaces are hyperquadrics and the classifier is referred to as a quadratic classifier.

6.3.4 Minimum Distance Classifiers

Here we assume that two classes are equiprobable with the same covariance matrix $\Sigma_1 = \Sigma_2 = \Sigma$. This simplifies eq. (6.15) to

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma^{-1}(\mathbf{x} - \mu_i) \tag{6.17}$$

considering the constant terms as being neglected.

Based on the covariance matrix shape, we can distinguish between two types of minimum distance classifiers:

- $\Sigma = \sigma^2 \mathbf{I}$: In this case, the maximum of $g_i(\mathbf{x})$ corresponds to the minimum of

$$\text{Euclidean distance: } \mathrm{d}_e = ||\mathbf{x} - \mu_i|| \tag{6.18}$$

  Thus, the feature vectors are assigned to classes according to the Euclidean distance between the class mean points and these vectors. Figure 6.2a visualizes the curves of equal distance $d_e = c$ to the stored class mean points. In this case, they are hyperspheres of radius c.
- Nondiagonal covariance matrix: In this case, the maximum of $g_i(\mathbf{x})$ is equivalent to the minimum of

$$\text{Mahalanobis distance: } d_m = \left((\mathbf{x} - \mu_i)^T \Sigma^{-1}(\mathbf{x} - \mu_i)\right)^{1/2} \tag{6.19}$$

  The curves of constant distance $d_m = c$ represent hyperellipses. The covariance matrix Σ is a symmetric function. All points having the same distance from a class mean point are located on an ellipse. The center of mass of the ellipse is at μ_i, and the principal axes are

aligned with the corresponding eigenvectors and have a length of $2\sqrt{\lambda_k}c$, as we can see from Figure 6.2b.

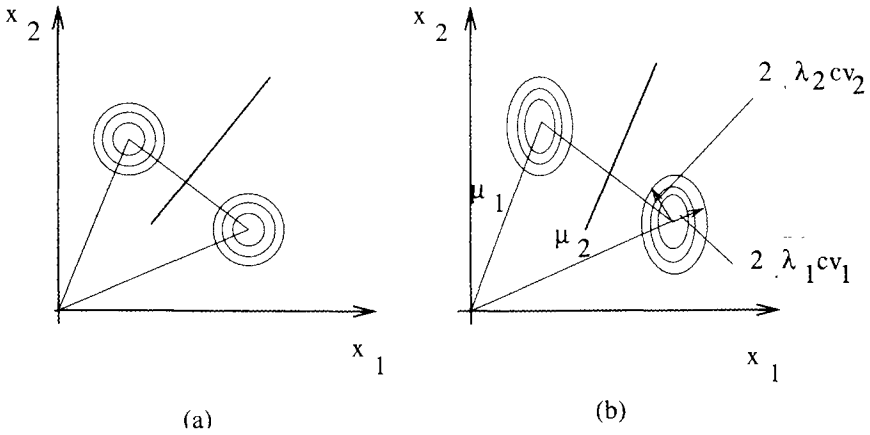


FIGURE 6.2 Curves of equal (a) Euclidean distance and (b) equal Mahalanobis distance from the mean values of each class.

EXAMPLE 6.3.1: Consider a two-class, two-dimensional classification problem. The feature vectors are generated by Gaussian pdfs with the same covariance matrix $\Sigma_1 = \Sigma_2 = \Sigma$;

$$\Sigma = \begin{bmatrix} 1,1 & 0,3 \\ 0,3 & 1,9 \end{bmatrix} \tag{6.20}$$

and the mean vectors given by $\mu_1 = [1,1]^T$ and respectively by $\mu_2 = [4,4]^T$.

The vector $[2,2]^T$ should be assigned to one of the two given classes according to the Bayesian classifier.

By determining the Mahalanobis distance of $[2,2]^T$ from the two mean vectors, we obtain

$$\begin{aligned} d_m^2(\mu_1, \mathbf{x}) &= (\mathbf{x} - \mu_1)^T \Sigma^{-1} (\mathbf{x} - \mu_1) \\ &= [1,1] \begin{bmatrix} 0.95 & -0.15 \\ -0.15 & 0.55 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 1.5 \end{aligned}$$

and

$$\begin{aligned} d_m^2(\mu_2, \mathbf{x}) &= (\mathbf{x} - \mu_2)^T \Sigma^{-1} (\mathbf{x} - \mu_2) \\ &= [-2,-2] \begin{bmatrix} 0.95 & -0.15 \\ -0.15 & 0.55 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} = 4.8 \end{aligned}$$

Thus the feature vector is assigned to the class ω_1 based on the minimal distance.

6.3.5 Estimation of Unknown Probability Density Functions

So far, we have assumed that the pdfs are known. However, in many practical problems this is not always the case, and they have to be estimated from the available data.

Many elegant solutions to this problem are known. Sometimes the type of the pdf is known (Gaussian, Laplacian), and we need only to estimate some parameters such as mean value and variance. Or the opposite is the case: We have knowledge of certain statistical parameters, but we have no information about the pdf type. Depending on the available information, different approaches to determine the missing information are known in the literature. This section reviews the maximum likelihood parameter estimation and the maximum entropy estimation as a solution to the above-stated problem.

6.3.5.1 Maximum Likelihood Parameter Estimation Let us consider the following classification task: Feature vectors of a given distribution $p(\mathbf{x}|\omega_i)$, $i = 1, 2, \ldots, M$ have to be assigned to the corresponding class out of M classes. Further, we are assuming that these likelihood functions are given in a parametric form and that the corresponding parameters form the unknown vectors θ. This dependence is reflected by writing $p(\mathbf{x}|\omega_i; \theta_i)$. The goal is to estimate the unknown parameters by using a set of known feature vectors for each class.

The joint pdf is given by $p(X; \theta)$, with $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x}_N\}$ being the set of the sample vectors:

$$p(X; \theta) = p(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x}_N; \theta) = \prod_{k=1}^{N} p(\mathbf{x_k}; \theta) \tag{6.21}$$

We are further assuming that the sample vectors are statistically independent.

The resulting function of θ is known as the likelihood function with respect to X. The estimation of θ is based on the maximum likelihood (ML) method and determines the maximum value of the likelihood function, that is,

$$\widehat{\theta}_{ML} = \arg \max_{\theta} \prod_{k=1}^{N} p(\mathbf{x_k}; \theta) \tag{6.22}$$

The necessary condition that $\widehat{\theta}_{ML}$ is a maximum is that the gradient of the likelihood function equals zero:

$$\frac{\partial \prod_{k=1}^{N} p(\mathbf{x_k};\theta)}{\partial \theta} = 0 \tag{6.23}$$

Computational simplicity can be achieved by considering the monotonicity of the logarithmic function. A new function, the so-called loglikelihood function, can be defined as

$$L(\theta) \equiv \ln \prod_{k=1}^{N} p(\mathbf{x_k};\theta) \tag{6.24}$$

and based on this we obtain from eq. (6.23) the equivalent expression

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{k=1}^{N} \frac{\partial \ln p(\mathbf{x_k}:\theta)}{\partial \theta} = \sum_{k=1}^{N} \frac{1}{p(\mathbf{x_k};\theta)} \frac{\partial p(\mathbf{x_k};\theta)}{\partial \theta} = 0 \tag{6.25}$$

In other words, we are now determining the maximum of the loglikelihood function.

EXAMPLE 6.3.2: Let us assume that the sample vectors $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}$ have a Gaussian pdf. Their covariance matrix is known, whereas the mean vector is unknown and has to be estimated. The pdf takes the form

$$p(\mathbf{x_k};\mu) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x_k}-\mu)^T\Sigma^{-1}(\mathbf{x_k}-\mu)\right), \quad i = 1,\ldots,M \tag{6.26}$$

For the N available samples, we have to determine the loglikelihood function

$$L(\mu) \equiv \ln \prod_{k=1}^{N} p(\mathbf{x_k};\mu) = -\frac{N}{2}\ln\left((2\pi)^n|\Sigma|\right) - \frac{1}{2}\sum_{k=1}^{N}(\mathbf{x_k}-\mu)^T\Sigma^{-1}(\mathbf{x_k}-\mu) \tag{6.27}$$

Taking the gradient of $L(\mu)$ with respect to μ, we get

$$\frac{\partial L(\mu)}{\partial \mu} \equiv \begin{bmatrix} \frac{\partial L}{\partial \mu_1} \\ \frac{\partial L}{\partial \mu_2} \\ \vdots \\ \frac{\partial L}{\partial \mu_l} \end{bmatrix} = \sum_{k=1}^{N} \Sigma^{-1}(\mathbf{x_k}-\mu) = 0 \tag{6.28}$$

For the mean value we obtain

$$\hat{\mu}_{ML} = \frac{1}{N}\sum_{k=1}^{N} \mathbf{x_k} \tag{6.29}$$

We immediately see that the ML estimate for normal densities is the mean of all sample vectors. In other words, the maximum likelihood estimate for the unknown population mean is just the arithmetic average of the samples — the sample mean.

6.3.6 Maximum Entropy Estimation

The definition of entropy stems from Shannon's information theory and gives in pattern recognition applications a measure of the randomness of the feature vectors. The entropy H for the density function $p(\mathbf{x})$ is given by

$$H = -\int_{\mathbf{X}} p(\mathbf{x}) \ln p(\mathbf{x}) d\mathbf{x} \tag{6.30}$$

Further, we assume that $p(\mathbf{x})$ is not known, and that some of the related constraints (mean value, variance, etc.) are known.

EXAMPLE 6.3.3: Let us assume that the random variable x is nonzero for $a \leq x \leq b$ and is equal to zero outside of the interval. We want to determine the maximum entropy of its pdf.

In this sense, we have to determine the maximum of eq. (6.30) under the condition

$$\int_a^b p(x) dx = 1 \tag{6.31}$$

Employing Lagrange multipliers, we obtain an equivalent representation

$$H_L = -\int_a^b p(x)(\ln p(x) - \lambda) dx \tag{6.32}$$

The derivative of H_L with respect to $p(x)$ is given by

$$\frac{\partial H_L}{\partial p(x)} = -\int_a^b \{\ln p(x) - \lambda) + 1\} dx \tag{6.33}$$

Setting the above equation equal to zero, we have

$$\widehat{p}(x) = \exp(\lambda - 1) \tag{6.34}$$

From eq. (6.31) we obtain $\exp(\lambda - 1) = \frac{1}{b-a}$. The estimated value of $p(x)$ is thus

$$\widehat{p}(x) = \begin{cases} \frac{1}{b-a}, & a \leq x \leq b \\ 0, & \text{else} \end{cases} \tag{6.35}$$

In other words, the maximum entropy estimate of the unknown pdf $p(x)$ is the uniform distribution.

6.4 NONPARAMETRIC ESTIMATION METHODS

For classification tasks we need to estimate class-related pdfs since they determine the classifier's structure. So far, each pdf was characterized by a certain parameter set. For Gaussian distributions, the covariance and the mean value are needed, and they are estimated from the sample data. Techniques described in the last section fall under parametric estimation methods.

This section will deal with nonparametric estimation techniques. There is no information available about class-related pdfs, and so they have to be estimated directly from the data set. There are many types of nonparametric techniques for pattern recognition. One procedure is based on estimating the density functions $p(\mathbf{x}|\omega_i)$ from sample patterns. If the achieved results are good, they can be included in the optimal classifier. Another approach estimates directly the a posteriori probabilities $P(\omega_i|\mathbf{x})$ and is closely related to nonparametric decision procedures, which bypass probability estimation and go directly to decision functions.

The following nonparametric estimation techniques will be reviewed:

- Histogram method
- Parzen windows
- k-nearest-neighbor
- Potential function

6.4.1 Histogram Method

This is one of the simplest and most popular pdf estimation methods. However, to achieve a robust estimation we need a relatively large data set. For the simple, one-dimensional case, the x-axis is divided into successive bins of length h, and then the probability of a sample x being located in a bin is estimated for each of these bins. Let N be the total number of samples and k_N of these samples are in a certain bin. Then the probability of the corresponding class is given by $P \approx k_N/N$. Figure 6.3 illustrates this procedure.

6.4.2 Parzen Windows

One of the most important nonparametric methods for pdf estimation is "Parzen windows" [160, 195]. For a better understanding, we will take the simple one-dimensional case. The goal is to estimate the pdf $p(x)$ at the point x. This requires determining the number of the samples N_h within the interval $[x-h, x+h]$ and then dividing by the total number of all feature vectors M and by the interval length $2h$. Based on the described procedure, we will obtain an estimate for the pdf at x:

$$\widehat{p}(\mathbf{x}) = \frac{N_h(x)}{2hM} \tag{6.36}$$

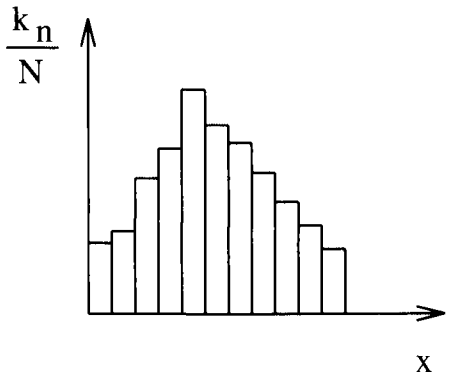


FIGURE 6.3 Histogram method applied to pdf estimation

As a support function K_h, we will choose

$$K_h = \begin{cases} 0,5 & : \quad |m| \leq |1| \\ 0 & : \quad |m| > |1| \end{cases} \tag{6.37}$$

From eq. (6.36) we get

$$\widehat{p}(x) = \frac{1}{hM} \sum_{i=1}^{M} K(\frac{x - m_i}{h}) \tag{6.38}$$

with the ith component of the sum being equal to zero if m_i falls outside the interval $[x - h, x + h]$. This leads to

$$\gamma(x, m) = \frac{1}{h} K(\frac{x - m}{h}) \tag{6.39}$$

as can be seen from Figure 6.4.

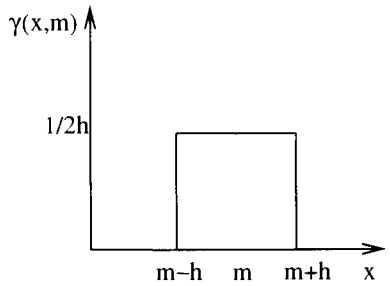


FIGURE 6.4 Clustering process of a two-dimensional vector table. The maximum number of vectors in one cluster was set to 7.

If $\widehat{p}(x)$ is considered to be a function corresponding to the number of samples, we thus obtain

$$\widehat{p}(x) = \widehat{p}(x, M) \tag{6.40}$$

Parzen showed that the estimate $\widehat{p}$ with $M \to \infty$ is bias free if $h = h(M)$ and

$$\lim_{M \to \infty} h(M) = 0 \tag{6.41}$$

In practice, where only a finite number of samples is possible, a right compromise between M and h has to be made. The choice of h is crucial, and it is recommended to start with an initial estimate of h and then modify it iteratively to minimize the misclassification error. Theoretically, a large M is necessary for acceptable performance. But in practice, a large number of data points increases the computational complexity unnecessarily.

Typical choices for the function $K(m)$ are

$$K(m) = (2\pi)^{\frac{-1}{2}} e^{-\frac{m^2}{2}} \tag{6.42}$$

$$K(m) = \frac{1}{\pi(1 + m^2)} \tag{6.43}$$

or

$$K(m) = \begin{cases} 1 - |m| & : \quad |m| \leq 1 \\ 0 & : \quad |m| > 1 \end{cases} \tag{6.44}$$

6.4.3 k-Nearest-Neighbor Density Estimation

In the Parzen windows estimation the length of the interval is fixed while the number of samples falling inside an interval varies from point to point. For the k-nearest-neighbor density estimation exactly the reverse holds: the number of samples k falling inside an interval is fixed while the interval length around x will be varied each time, to include the same number of samples k. We can generalize for the n-dimensional case: In low-density areas the hypervolume $V(\mathbf{x})$ is large while in high-density areas it is small.

The estimation rule can now be given as

$$\widehat{p}(\mathbf{x}) = \frac{k}{NV(\mathbf{x})} \tag{6.45}$$

and reflects the dependence of the volume $V(\mathbf{x})$. N represents the total number of samples while k describes the number of points falling inside the volume $V(\mathbf{x})$.

This procedure can be very easily elucidated based on a two-class classification task: an unknown feature vector $\mathbf{x}$ should be assigned to

one of the two classes ω_1 or ω_2. The decision is made by computing its Euclidean distance d from all the training vectors belonging to various classes. By r_1 we denote the radius of the hypersphere centered at $\mathbf{x}$ that contains k points from class ω_1, while r_2 is the corresponding radius of the hypersphere belonging to class ω_2. V_1 and V_2 are the two hypersphere volumes.

The k-nearest-neighbor classification rule in the case of the two classes ω_1 and ω_2 can now be stated:

$$\text{Assign} \quad \mathbf{x} \quad \text{to class} \quad \omega_1 \quad (\omega_2) \quad \text{if} \quad \frac{V_2}{V_1} > (<) \frac{N_1}{N_2} \frac{P(\omega_2)}{P(\omega_1)} \tag{6.46}$$

If we adopt the Mahalanobis distance instead of the Euclidean distance, then we will have hyperellipsoids instead of hyperspheres.

6.4.3.1 k-Nearest-Neighbor for Tumor Detection The classification rule has to be slightly modified for abnormal and normal classes. A new pattern vector is assigned to a particular class, if at least l of the k nearest neighbors are in that particular class. This is especially attractive for applications where the penalty for misclassifying one class is much greater than that associated with the other class. Also, if there are more samples from one class than from the other class, in other words if we deal with an unbalanced training set, a majority vote might not be useful. For example, consider a training set with 50 sample vectors for class 1, and 500 sample vectors for class 2. In this case, we get more accurate classifications if $l \approx k/10$ for class 1, because there are 10 times more sample vectors in class 2 than in class 1. Thus, a new pattern vector is assigned to class 1 if approximately 10% of the k-nearest-neighbors belong to class 1.

6.4.4 Potential Functions

Potential functions represent a useful method for estimating an unknown pdf $\widehat{p}(\mathbf{x})$ from the available feature vectors [10, 160].

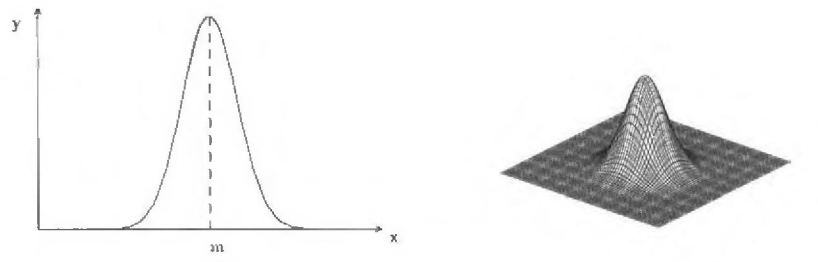
The estimated pdf is given by a superposition of potential functions $\gamma(\mathbf{x}, \mathbf{m})$:

$$\widehat{p}(\mathbf{x}) = \frac{1}{M} \sum_{j=1}^{M} \gamma(\mathbf{x}, \mathbf{m}_j) \tag{6.47}$$

Figure 6.5a and b illustrate an example of a potential function for the one-dimensional and two-dimensional cases.

A possible potential function is

$$\gamma(\mathbf{x}, \mathbf{m}) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} e^{\left(\frac{-\|\mathbf{x}-\mathbf{m}\|^2}{2\sigma^2}\right)} \tag{6.48}$$



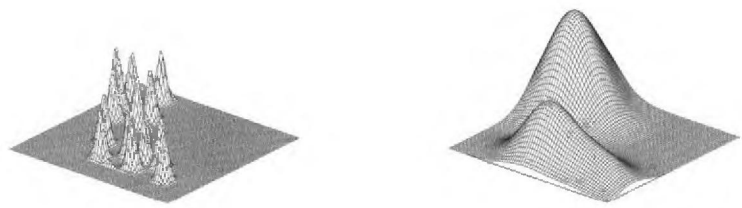
(a) (b)

FIGURE 6.5 Potential functions for the (a) one-dimensional and (b) two-dimensional cases.

where $||\mathbf{x}||$ is a norm in the n–dimensional space. The potential function defines a distance measure (Mahalanobis distance) between two feature vectors $\mathbf{x}$ and $\mathbf{m}$.

Equation (6.47) describes the complete algorithm for a prespecified potential function. $\widehat{p}(\mathbf{x})$ can be estimated for every $\mathbf{x}$ directly from eq. (6.47).

The choice of the potential function is not so trivial since the width of the potential function plays an important role here. The smaller its width, the more it peaks and the higher it is. This means it considers only feature vectors in its immediate neighborhood. Larger widths produce potential functions of smoother shape, as can be seen from Figure 6.6.



(a) (b)

FIGURE 6.6 Importance of the width of the potential function (a) sharper surfaces and (b) smoother surfaces.

The choice of a certain variance has a considerable importance on the overlapping degree of neighboring potential functions. This is especially critical when potential functions describing feature vectors belonging to different classes overlap.

There are several possible ways of determining σ [16]:

1. Let us assume that within the distance σ from a specific feature vector there are L other feature vectors. The average value describes the distance $D_L(\mathbf{m})$ from the feature vector to the Lth feature vector:

$$\sigma = \frac{1}{M} \sum_{i=1}^{M} D_L(\mathbf{m}_i) \tag{6.49}$$

   To determine L, we have to take into account the distribution of the feature vectors. In many practical problems, $L = 10$ is a good choice.

2. σ can also be chosen as a multiple of the minimal distance between two feature vectors. This is necessary to achieve a certain overlapping degree. In [16] it is recommended to set the multiple equal to 4.

The potential function has the following general properties [160]:

1. $\gamma(\mathbf{x}, \mathbf{m})$ has its maximum at $\mathbf{x} = \mathbf{m}$.
2. $\gamma(\mathbf{x}, \mathbf{m})$ goes asymptotically toward zero if the distance between the two feature vectors is very large. This is of special importance for multimodal distributions. There are more than one cluster for each class.
3. $\gamma(\mathbf{x}, \mathbf{m})$ is a continuous function decreasing monotonically on both sides from the maximum.
4. If $\gamma(\mathbf{x}_1, \mathbf{m}_1) = \gamma(\mathbf{x}_2, \mathbf{m}_1)$, then the feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ have the same similarity degree with respect to $\mathbf{m}_1$.

There are several types of known potential functions: Besides the above-mentioned unimodal or multimodal normal distributions, potential functions built from orthonormal functions are also of interest [160]. They have the form

$$\gamma(\mathbf{x}, \mathbf{m}) = \sum_{i=1}^{R} \lambda_i^2 \Phi_i(\mathbf{x}) \Phi_i(\mathbf{m}) \tag{6.50}$$

where $\Phi_i(\mathbf{x})$ is an orthonormal function and λ is a constant. Orthonormal functions fulfill

$$\int \Phi_i(\mathbf{x}) \Phi_j(\mathbf{x}) dx = \begin{cases} 1 & i = j \\ 0 & \text{else} \end{cases} \tag{6.51}$$

with $\lambda_i = 1$.

It is easy to see that they are potential functions because

$$\sum_{i=1}^{\infty} \Phi_i(\mathbf{x})\Phi_i(\mathbf{m}) = \delta(\mathbf{x} - \mathbf{m}) \tag{6.52}$$

where $\delta(\mathbf{x})$ is a Dirac function and $\{\Phi_i\}$ describes a set of orthonormal functions.

The resulting discriminant function can be determined from eq. (6.51) and is given by

$$\hat{p}(\mathbf{x}) = \frac{1}{M}\sum_{i=1}^{M} \gamma(\mathbf{x}, \mathbf{m}_i) = \frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{R} \Phi_k(\mathbf{x})\Phi_k(\mathbf{m}_i) \tag{6.53}$$

where $\hat{p}(\mathbf{x})$ is the estimated pdf, that is,

$$\hat{p}(\mathbf{x}) = \sum_{k=1}^{R} c_k \Phi_k(\mathbf{x}) \tag{6.54}$$

The estimated coefficients c_k are

$$c_k = \frac{1}{M}\sum_{i=1}^{M} \Phi_k(\mathbf{m}_i) \tag{6.55}$$

It is important to note that the potential function estimator is both unbiased, and asymptotically consistent. We also can easily see that the potential function method is related to Parzen windows. In fact, the smooth function used for estimation is known as either *kernels* or potential functions or Parzen windows.

6.5 BINARY DECISION TREES

Binary decision trees are nonlinear multistage classifiers. This classification system operates by rejecting classes sequentially until the correct class is found. In other words, the correct class corresponding to a feature vector is determined by searching a tree-based decision system. The feature space is divided into regions corresponding to the different classes.

This classification scheme is extremely useful when a large number of classes are given. Binary decision trees divide the search space into hyperrectangles with sides parallel to the axis. The tree is searched in a sequential manner and a decision of the form $x_i \leq \alpha$, with x_i being a feature and α a threshold value, is made at each node for individual features. This processing scheme is an essential part of many tree-based vector quantization algorithms.

Figure 6.7 illustrates an example of a binary decision tree. It shows the regions created by the successive sequential splitting of 23 reference vectors.

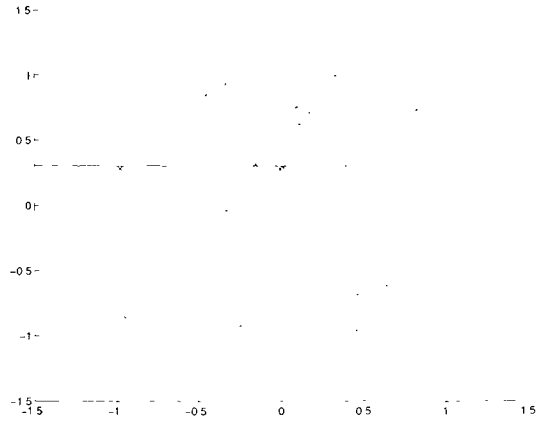


FIGURE 6.7 Clustering process of a two-dimensional vector table. The maximum number of vectors in one cluster was set to 7.

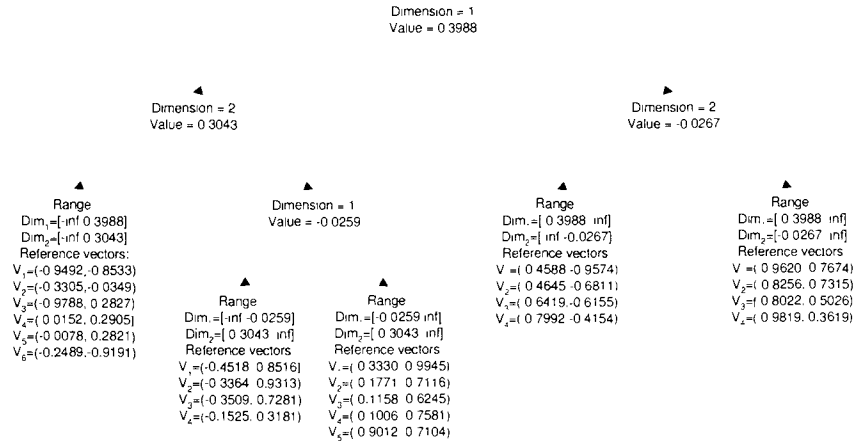


FIGURE 6.8 The internal tree structure after clustering.

Figure 6.8 shows the corresponding binary decision tree classification. A decision based on this classification scheme can be made without testing all available features.

6.5.1 Binary Decision Trees for Tumor Detection

Since binary decision trees have the tendency to generalize new sample vectors poorly, overtraining is one of the unwanted side effects. In a pruning process which replaces subtrees by leaf nodes the new sample is assigned to the most frequent class. In other words, instead of a single class being associated with each leaf, there is a probability for each class that a new

pattern vector belongs to that class. For example, if a leaf node contains 7 training samples from the abnormal class and 3 samples from the normal class, then a new pattern vector that reaches the leaf node is 70% abnormal.

6.6 SYNTACTIC PATTERN RECOGNITION

In some practical applications the pertinent information is not always available in the form of a numerical-valued feature vector, as is the case with classification based on decision-theoretic methods. Therefore, it is not surprising that patterns contain structural information that is difficult or impossible to quantify in feature vector form. Structure describes the way in which elements or components of a pattern are related. This structural relation can be either implicit in the model or explicitly defined in detail.

This important aspect of structural information leads to a new classification paradigm, the so-called structural or syntactic classification concept. Syntactic pattern recognition is accomplished based on the the following steps: (1) definition, (2) extraction of structural information, and (3) comparison based on measures of pattern structural similarity. Standard references for syntactic pattern recognition are [67, 68, 165].

Structural pattern recognition can be employed for both classification and description. For classification, metrics of pattern structural similarity must be determined first. Classes can be defined according to a common structural description. Summarizing the above, we can state that the basis of syntactic pattern classification lies in the definition of the structure of an entity, which can be used afterward for classification and description. The concept of syntactic pattern recognition is described in Figure 6.9.

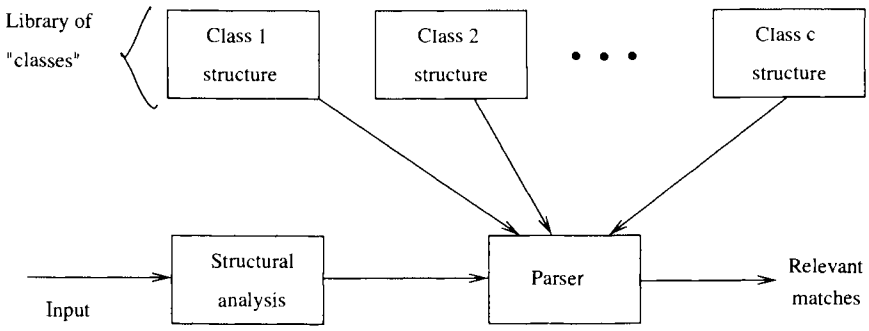


FIGURE 6.9 Syntactic pattern classification.

To make syntactic pattern recognition feasible, we have to require that "structure" be quantifiable. This can be accomplished based on either formal grammars or relational graphs.

The most important component of a grammar is the string. A string is a list of symbols, the elements of an alphabet, that represent the basic components of a pattern structure.

A string produced by a class-specific grammar can be easily classified based on the following procedures:

- String matching: Pattern structure is essential, but the available number of patterns is too small. Data structure is employed to represent both prototypes and unknown input patterns, and the classification is based on matching metrics between two structural representations.
- Parsing: the goal of this method is to determine if an input string (pattern) can be generated by a given grammar. Therefore, parsers can be considered as syntax analyzers [216]. If an input string is successfully parsed, then a structural analysis of this pattern is also available, since a production sequence has been found.

In many practical applications, the grammars are not explicitly given, but have to be determined based on the available set of feature vectors. This raises the issue of grammatical inference.

Statistical pattern recognition methods rely strongly on quantitative measurements and are not well qualified for all problems. Figure 6.10 shows an example where statistical methods are not sufficient. Suppose that telocentric chromosomes of various sizes and locations in the plane are to be classified. Only the boundary is to be considered, and because of the variable size of shapes, not the metrics but the structure proves to be critical. It is evident that the structure has to be extracted in some way, and then it has to be compared to that of other objects for the purpose of classification.

This example of a telocentric chromosome [28] illustrates the concept of structural pattern recognition. Figure 6.10 shows the five curved line segments a, b, c, d, e which serve as primitives for the boundary description of the telocentric chromosome.

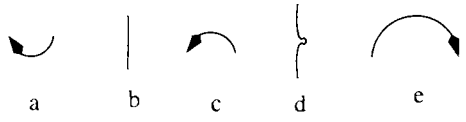
6.6.1 Elements of Formal Grammars and Languages

Syntactic pattern recognition is based on employing formal grammars for pattern class representation. The use of a grammar has some major advantages: It considers (1) the patterns recursively and (2) common substructures shared by the members of a pattern class.

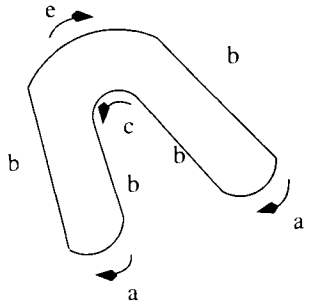
Formal grammars are based on words over finite sets of symbols. For a better understanding of this important concept, we will give some basic notations and descriptions.

An alphabet Z is defined as a finite, nonempty set of symbols such as

$$Z = \{a, b, \ldots, z\} \tag{6.56}$$



(a) Pattern primitives



(b) Telocentric chromosome

FIGURE 6.10 Boundary structure of telocentric chromosome ([28] p. 5).

A *string* over Z is defined either as a single symbol from Z or as an ordered sequence of symbols of zero or more symbols from Z. In general, a string can be produced as follows: (1) letters from Z can be concatenated; (2) any letter in Z can be used several times; (3) strings can be concatenated; and (4) the empty string $\varnothing$ has no letters but is still considered as a valid string. Strings over Z define sentences. The set of all strings over Z is described by Z^* and is also known as the closure set of Z. Two strings m and n can be concatenated to form the new string mn or nm. The string n is a substring of the string m whenever $m = inj$, where i and j are strings. i is the prefix string of and j is a suffix string of n with respect to m. The length of a string m is the number of letters it contains and is denoted by $|m|$.

EXAMPLE 6.6.1: Let $Z = \{a, b\}$, $m = abaaabba$, and $n = aab$. Then $|m| = 8$, $|n| = 3$, n is a substring of m. The strings of all concatenations of m and n are: $mn = abaaabbaaab$ and $nm = aababaaabba$.

Formal linguistics can be applied to model and describe patterns in syntactic pattern recognition. A concept borrowed from linguistics is that of the language. Languages are generated by grammars.

Grammars are used with Z to impose some structure on a subset of strings $L \subseteq Z^*$. L is called a language. A language over the alphabet Z is defined as any subset of all strings over this given alphabet. Two sets of strings GH can also be elementwise concatenated via $GH = \{mn : m \in$

$G, n \in H\}$. For any set of strings $M \subset Z^*$, we define: (a) $M^0 = \{\varnothing\}$, (b) $M^1 = M$, (c) $M^2 = MM$, (d) $M^n = MM^{n-1}$, (e) $M^* = \cup_{(n=0,\infty)} M^n$, and (f) $M^+ = \cup_{(n=1,\infty)} M^n$.

A grammar is defined as a four-tuple $\mathbf{G} = (Z, V, S, P)$, with Z describing an alphabet of letters while V is an auxiliary alphabet of letter-valued variables with $Z \cap V = \varnothing$. S denotes the start variable from V denoted while P describes a set of production rules that determines the construction of sentences. This includes the intermediate "strings" composed of letters and letter-valued variables. The production rules serve as a basis for constructing strings from letters from Z and letter-valued variables from V. S is always used to begin any such string. We would like to add that the variables in V are called nonterminals while the letters from Z are called terminals. The set of productions P in a grammar G defines a structure or syntax on the language generated by G via an implicit relation on strings.

The language L generated by grammar $\mathbf{G}$, described as $L(\mathbf{G})$, forms the set of all strings that satisfy the following conditions: (1) each string contains only terminal symbols from Z, and (2) each string was produced from S using the production rules P from the given grammar G.

EXAMPLE 6.6.2: The following grammar $G = \{Z, V, P, S\}$ is given with $V = \{S, A, B\}$, $Z = \{a, b, c\}$, and $P = \{S \rightarrow cAb, A \rightarrow aBa, B \rightarrow aBa, B \rightarrow cb\}$.

The language $L(G) = \{ca^n cba^n b | n \leq 1\}$ is generated based on the given grammar. In [28] it is shown that the word *caacbaab* can be generated by the following sequence:

$$S \rightarrow cAb \rightarrow caBab \rightarrow caaBaab \rightarrow caacbaab \tag{6.57}$$

It is evident that a grammar can generate a language of infinite words, even if all components of this grammar are finite.

6.6.2 Syntactic Recognition via Parsing and Grammars

Up to now we have considered the generation of syntactic or structural pattern description using formal grammars. Here, we will look at the "inverse" of this problem. Given the description of a pattern as a string produced by a class-specific grammar, the goal is to determine to which $L(\mathbf{G_i}), i = 1, 2, \ldots, c$ the string belongs.

It is evident that recognition of entities using syntactic descriptions can be considered as a matching procedure. Suppose that we are interested in a c-classification and therefore we develop class-specific grammars $\mathbf{G_1}, \mathbf{G_2}, \ldots, \mathbf{G_c}$. Every unknown description x can be classified by determining if $x \in L(\mathbf{G_i})$, for $i = 1, 2, \ldots, c$. Suppose that the language of each $\mathbf{G_i}$ could be generated and stored in a class-specific library of patterns. x has to be matched against each pattern in each library in order to determine its class membership. In other words, x is parsed to determine to

which class it belongs. Parsing is a basic concept for syntactic recognition and determines if an input pattern (string) is syntactically well formed in the context of one or more prespecified grammars. Parsers produce the parsing process and are known for being syntax analyzers. One possibility to parse the sentences is to use a finite-state machine $L(\mathbf{G})$ for a decision. It was shown [67, 68] that there exists a connection between grammars as generators of languages and finite-state machines as recognizers of languages. An input string is processed by such a finite-state machine, which then decides if this string is in a specific language L or not. Thus, the finite-state machines are often used as the bases of language recognition algorithms.

A finite-state machine is a memory-based machine of stored states. It then changes a state based on the current state, and the received input letter, and outputs a letter. The structure of a finite-state machine is described in Figure 6.11.

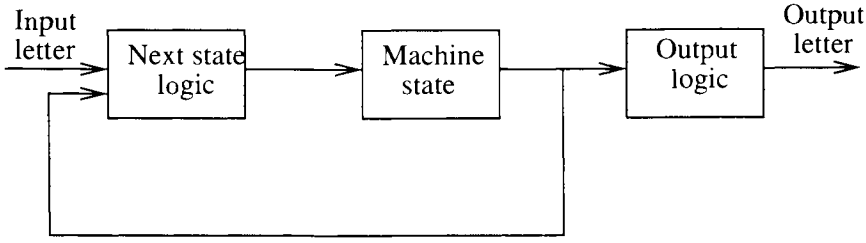


FIGURE 6.11 A finite-state machine.

The finite-state machine can be described abstractly by a tuple (Z, Q, g, h, q_0, Q_f), where Z is the alphabet of input/output letters, Q is the set of states, q_0 is the initial state, $Q_f \subset Q$ is the set of end states, g is the next state function $g : ZxQ \rightarrow Q$ that maps an input letter and current state into a new state, and h is the output function $h : QxZ \rightarrow Z$ that assigns an output letter in function of the current state and input letter.

Let us further define a finite-state machine by β and assume that it accepts a set of sentences defining a language $L(\beta)$. The existing correspondence between a regular grammar and a finite-state machine can be expressed by the following relations: (a) $M \rightarrow aN$ describes a transition from state s_M to state s_N while being in state s_M and receiving a letter a as an input; and (b) $M \rightarrow a$ describes the transition from state s_M to a final state $s_F \in Q_F$ with Q_F being the set of final states.

EXAMPLE 6.6.3: The following grammar is described by $\mathbf{G} = (Z, V, S, P)$ with $Z = \{m, n, o\}$, $V = \{S, M\}$, and $P = \{S \rightarrow mS, S \rightarrow nM, M \rightarrow nM, M \rightarrow mM, M \rightarrow o\}$. We immediately see that $L(\mathbf{G})$ defines a regular

language. Figure 6.12 describes the finite-state machine. s_G is the rejected state, while s_F is the final state.

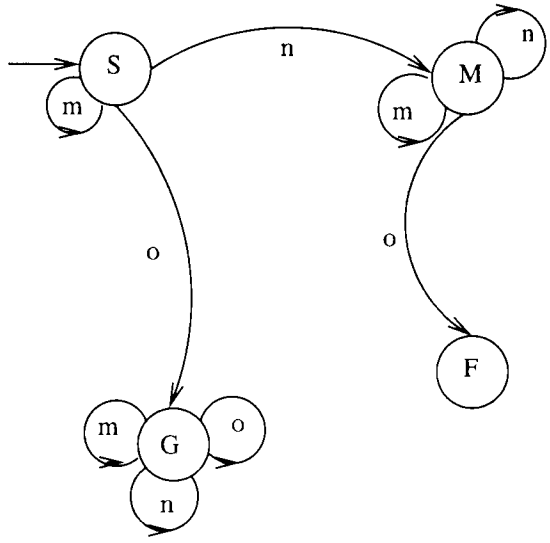


FIGURE 6.12 Finite-state machine corresponding to grammar **G**.

6.6.3 Learning Classes via Grammatical Inference

In most practical applications, it is desired that the necessary grammars be directly inferred from a set of sample patterns. In this context, inference represents a procedure for constructing a structural description of items from a class. Grammatical inference is the process of learning grammars from a training set of sample sentences. The idea comes from Chomsky [40]. The most difficult part of syntactical pattern recognition is to learn classes from a sample of patterns. To describe the inference problem, we must first consider a finite sample I of strings of letters stemming from an alphabet Z, and then find a grammar $\mathbf{G}$ that accepts I, i.e., $I \subset L(\mathbf{G})$ as shown in [68, 165].

The application of grammars in pattern recognition follows these steps: (1) The alphabet Z is used to describe elementary patterns that can be extracted from an input pattern based on preprocessing or segmenting; (2) the auxiliary alphabet V is then used to describe subpatterns of greater complexity, which are formed by production rules from the primitive elements determined at the previous step; and finally (3) the language generated by the grammar is used to represent a whole class of patterns.

For syntactic pattern recognition, we need a grammar describing each pattern class. A given input string is parsed in order to determine its

correspondence to one of the given classes. If none can be established, then the input pattern is rejected.

Since there are plenty of grammars that contain I, a major problem is caused by overlapping classes. The same sentence for instance can be accepted by several grammars. A solution to this problem is provided by introducing a distance function which determines the distance between a sentence and the regular grammars. As always in pattern recognition, the classification is then achieved by determining the minimal distance between the given sentence and the regular grammars [165]. The grammatical inference procedure is described in Figure 6.13.

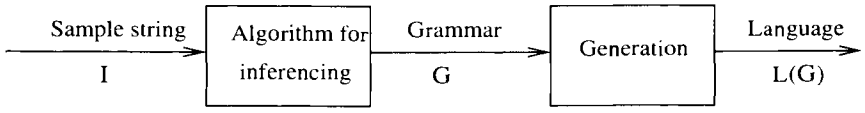


FIGURE 6.13 Grammatical inference.

EXAMPLE 6.6.4: For a given single sentence mn from $Z = \{m, n\}$, any of the following production rules are inferring a grammar whose language contains mn:

$$P_1 : \quad S \rightarrow mn$$

$$P_2 : \quad S \rightarrow AB, A \rightarrow m, B \rightarrow n$$

$$P_3 : \quad S \rightarrow An, A \rightarrow m$$

$$P_4 : \quad S \rightarrow mB, B \rightarrow n$$

We see immediately that a minimal grammar can be generated by the minimal set of productions $\{P_1\}$.

It is important to restate that there is not a unique relationship between a given language and some grammar. Thus, for a given string I there may be more than one grammar as solutions. Grammatical inference can be achieved by searching through the rule space of all context-free grammars for a grammar that can be inferred directly from a set of training patterns I.

A training pattern I is said to be structurally complete with respect to a grammar $\mathbf{G} = \{(Z, V, S, P)\}$ if (a) $I \subset L(\mathbf{G})$, (b) I is defined over

the alphabet Z, and (c) every production rule in P is employed in the generation of strings in I. Then the grammatical inference problem can be described as follows: (a) Find $\mathbf{G}$ such that $L(\mathbf{G}) \subset L(\mathbf{G})$; and (b) I is structurally complete with respect to $\mathbf{G}$ [146].

A methodological definition for grammatical inference was given in [28] as an "automatic learning of formal grammars from finite subsets of the language they generate."

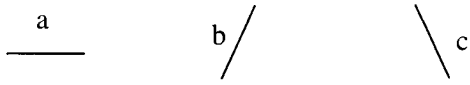
EXAMPLE 6.6.5: An example of how a grammar G can be employed for the description of a possibly infinite set of strings, i.e., the language $L(G)$, is given in [28].

Consider example 6.6.2 and its defined grammar. Figure 6.14 shows that the terminal symbols a, b, c are used to represent line segments of fixed length. A class of arrow-like shapes can be described based on these terminal symbols if one proceeds in a clockwise direction.

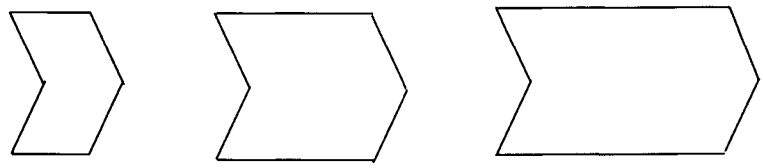
Thus, the class of patterns shown in Figure 6.14 can be represented by the set of words

$$\{cacbab, caacbaab, caaacbaaab\} \tag{6.58}$$

which is identical with $L(G)$.



Line segments as primitives



Sample patterns of a class

FIGURE 6.14 Inference of a symbolic data structure ([28] p. 37).

6.6.4 Graphical Approaches

Graph-based structural representations can be used as an approach in pattern recognition. Structural representations are replaced by graphical alternatives. They can be considered as extensions of higher dimensional

grammars, where instead of parsers a measure of graph similarity is employed.

In the following we will give some basic definitions from graph theory which is a fundamental area of discrete mathematics. A graph G is an ordered pair $G = (V, E)$, where $V = \{v_i, i = 1, \ldots, N\}$ is a set of vertices (nodes) and E is a set of edges (arcs) connecting some pairs of vertices. An edge which connects the vertices v_i and v_j is denoted either by e_{ij} or by (v_i, v_j). Often there is a significance attached to the direction of an edge, in the sense that an edge emanates from a vertice and is incident on another vertice. Therefore, $(a, b) \in R$ means there is an edge from node a to node b. It should be stated here that $(a, b) \in R$ does not imply $(b, a) \in R$. This directional significance characterizes a digraph, or directed graph. When the direction of edges is a graph is not important, this means that specification of either (a, b) or $(b, a) \in R$ is acceptable, and an undirected graph results. A relational graph is an unlabeled digraph wherein nodes represent subpatterns and primitives, and edges represent relationships between these subpatterns and primitives. To recognize structure using graphs requires that each pattern (structural) class be represented by a prototypical relational graph. An unknown input pattern has to be described by a graph, and this graph is then compared with the relational graphs for each class. We will illustrate this fact based on an example. Suppose we have four classes, and that there are four prototypical relational graphs $G_{C1}, G_{C_2}, G_{C_3}, G_{C_4}$ available. For each new pattern data, a relational graph G_U has to be extracted from the data, and then compared with $G_{C1}, G_{C_2}, G_{C_3}, G_{C_4}$. To accomplish this, a discriminant function and a measure of similarity must be established.

A directed graph G with p nodes can be converted to an adjacency matrix by the following procedure:

1. Number each node and obtain a set of numbered graph nodes indexed by $[1, \ldots, p]$.
2. The adjacency matrix describes the existence or absence of an edge between two nodes in G:

$$ADJ[i, j] = \begin{cases} 1, & \text{for an existing edge from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases} \tag{6.59}$$

In addition, if no cost is associated with the edges of a graph, the graph is called an unweighted graph. Otherwise, a weighted graph results.

Some useful graph definitions [229] are illustrated in Figure 6.15.

An undirected, unweighted graph with N nodes, each corresponding to a vector of the data set X, is called a threshold graph and is denoted by $G(a)$, where a represents the dissimilarity level. To determine if an edge

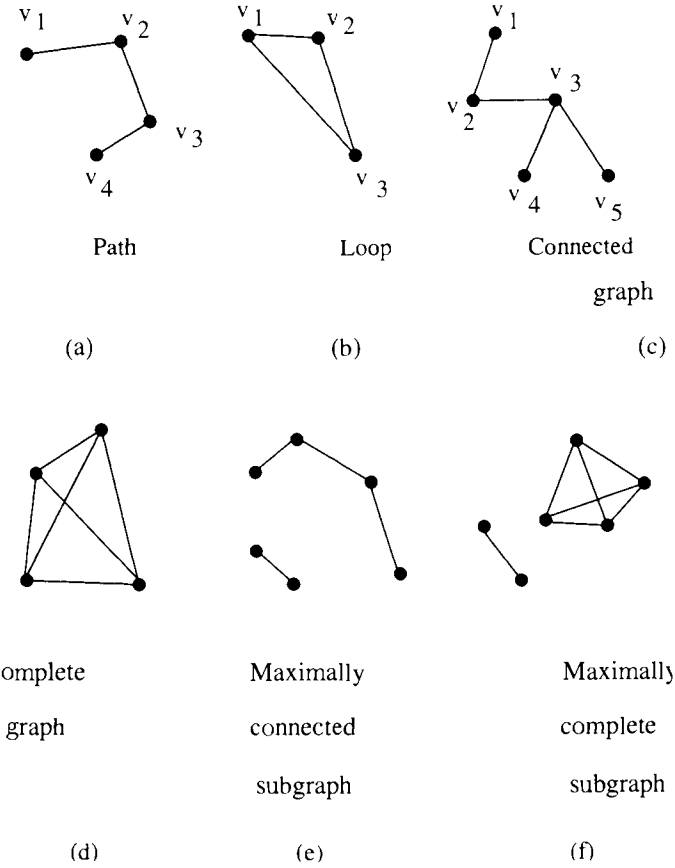


FIGURE 6.15 Basic graph types.

between two nodes i and j with $i, j = 1, \ldots, N$ of the threshold graph $G(a)$ with N nodes exist, the dissimilarity between the corresponding vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ must be less than or equal to a:

$$(v_i, v_j) \in G(a), \quad \text{if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \leq a, \quad i, j = 1, \ldots, N \tag{6.60}$$

A threshold graph $G(a)$ whose edges (u_i, u_j) are weighted with the proximity measure between $\mathbf{x}_i$ and $\mathbf{x}_j$ is named a proximity graph $G_p(a)$. The proximity between two vectors can be defined based on a dissimilarity measure, and then produces a dissimilarity graph.

EXAMPLE 6.6.6: Let $X = \{\mathbf{x}_i, i = 1, \ldots, 5\}$, with $\mathbf{x}_1 = [1, 1]^T, \mathbf{x}_2 = [2, 1]^T, \mathbf{x}_3 = [5, 4]^t, \mathbf{x}_4 = [6, 5]^T$, and $\mathbf{x}_5 = [3, 4]^T$. The pattern matrix of X is

$$D(X) = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 5 & 4 \\ 6 & 5 \\ 3 & 4 \end{bmatrix} \tag{6.61}$$

This yields a corresponding dissimilarity matrix based on using the Euclidean distance

$$P(X) = \begin{bmatrix} 0 & 1 & 5 & 6.4 & 3.6 \\ 1 & 0 & 4.2 & 5.7 & 3.1 \\ 5 & 4.2 & 0 & 1.4 & 2 \\ 6.4 & 5.7 & 1.4 & 0 & 3.1 \\ 3.6 & 3.1 & 2 & 3.1 & 0 \end{bmatrix} \tag{6.62}$$

The resulting dissimilarity graph is shown in Figure 6.16.

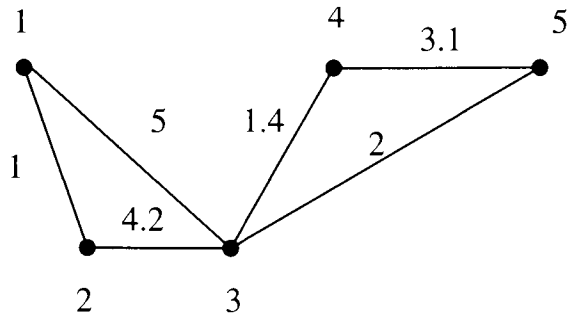


FIGURE 6.16 Dissimilarity graph defined by an Euclidean dissimilarity matrix.

Graph theory is a valuable tool for supporting agglomerative algorithms. In a graph G of N nodes, there is a correspondence between each node and vector of the data set X. Clusters can usually be built by connecting nodes together, and such forming connected subgraphs. A so-called graph property $h(k)$ imposes constraints on the subgraphs. In a very simple algorithm, the so-called single link algorithm [229], clusters C_r and C_s are merged based on the proximity function $g_{h(k)}(C_r, C_s)$ between their nodes:

$$g_{h(k)}(C_i, C_j) = \min_{\mathbf{x}_u \in C_r, \mathbf{x}_v \in C_s} \{d(\mathbf{x}_u, \mathbf{x}_v) = a\} \tag{6.63}$$

The subgraph $G(a)$ defined by $C_r \cup C_s$ is connected.

EXAMPLE 6.6.7: Consider the following dissimilarity matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 1.1 & 3 & 3.7 & 4.2 \\ 1.1 & 0 & 2.5 & 3.2 & 3.9 \\ 3 & 2.5 & 0 & 1.8 & 2.0 \\ 3.7 & 3.2 & 1.8 & 0 & 1.4 \\ 4.2 & 3.9 & 2.0 & 1.4 & 0 \end{bmatrix} \tag{6.64}$$

In the first step, $\mathcal{R}_0$, each pattern vector from X forms a single cluster. The next step, $\mathcal{R}_1$, requires the computation of $g_{h(k)}(C_r, C_s)$ for all cluster pairs. Based on the fact that $g_{h(k)}(\{\mathbf{x}_1\}, \{\mathbf{x}_2\}) = 1.1$ is the minimum value, the pattern vectors $\{\mathbf{x}_1\}$ and $\{\mathbf{x}_2\}$ are merged and we obtain

$$\mathcal{R}_1 = \{\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3\}, \{\mathbf{x}_4\}, \{\mathbf{x}\}_5\}\}$$

The next minimum we find for $g_{h(k)}(\{\mathbf{x}_4\}, \{\mathbf{x}_5\}) = 1.4$, and so we obtain

$$\mathcal{R}_2 = \{\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3\}, \{\mathbf{x}_4, \mathbf{x}_5\}\}$$

For $\mathcal{R}_3$, we get

$$\mathcal{R}_3 = \{\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}\}$$

In the end, we obtain $g_{h(k)}(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}) = 2.5$, and this leads to $\mathcal{R}_4$.

6.7 DIAGNOSTIC ACCURACY OF CLASSIFICATION MEASURED BY ROC-CURVES

Often a clinical researcher has to determine how accurate a particular laboratory test is in identifying diseased cases. The ability of a test to discriminate diseased cases from normal cases is based on receiver operating characteristic (ROC) analysis. The ROC represents an analytical technique derived from biometrics and is used for providing both the desired accuracy index and utility in terms of cost and benefit [161].

At the same time, a method to specify the performance of a classifier is in terms of its ROC curve. The ROC curve represents a plot describing the classifier's true positive detection rate versus its false positive rate. The false positive (FP) rate is the probability of incorrectly classifying a nontarget object (normal tissue region) as a target object (tumor region). On the other hand, the true positive (TP) detection rate is the probability of correctly classifying a target object as being indeed a target object. Both the TP and FP rates are specified in the interval from 0.0 to 1.0, inclusive.

In medical imaging the TP rate is commonly referred to as sensitivity, and (1.0−FP rate) is called specificity.

The schematic outcome of a particular test in two populations, one population with a disease, the other population without the disease, is summarized in Table 6.1.

TABLE 6.1 Results of a particular test in two populations, one population with a disease.

	Disease present	Disease absent	Sum
Test positive	True positive (TP)	False positive (FP)	(TP + FP)
Test negative	False negative (FN)	True positive (TN)	(FN + TN)
Sum	(TP + FN)	(FP + RN)	

In general, the sensitivity S_e and the specificity S_p of a particular test can be mathematically determined.

Sensitivity S_e is the probability that a test result will be positive when the disease is present (true positive rate, expressed as a percentage):

$$S_e = \frac{TP}{FN + TP} \tag{6.65}$$

Specificity S_p is the probability that a test result will be negative when the disease is not present (true negative rate, expressed as a percentage).

$$S_p = \frac{TN}{TN + FP} \tag{6.66}$$

Sensitivity and specificity are a function of each other and are also counterrelated.

Parameters associated with statistical classifiers can be varied in order to change the TP and FP rates. Each set of parameter values may result in a different (TP,FP) pair, or operating point. It is possible to trade a lower (higher) FP rate for a higher (lower) TP detection rate by choosing appropriate values for the parameters in question.

A typical ROC curve is shown in Figure 6.17. The area under the curve (AUC) is an accepted modality of comparing classifier performance. A perfect classifier has a TP rate of 1.0 and an FP rate of 0.0, resulting in an AUC of 1.0. Random guessing would result in an AUC of 0.5.

An ROC curve demonstrates several things:

- It shows the trade-off between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).

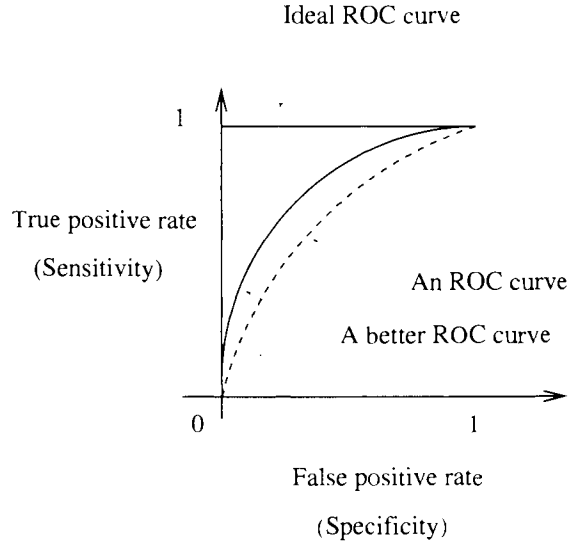


FIGURE 6.17 Typical ROC curves.

- The closer the curve approaches to the left-hand border and then the top border of the ROC space, the more accurate the test.
- The closer the curve comes to the 45° diagonal of the ROC space, the less accurate the test.
- The area under the curve is a measure of comparing classifier performance. An ideal classifier has an area of 1.

In the following it is shown how ROC curves are generated for statistical classifiers [244].

6.7.1 Bayesian Classifier

A Bayesian classifier can be trained by determining the mean vector and the covariance matrices of the discriminant functions for the abnormal and normal classes from the training data. Instead of computing the maximum of the two discriminant functions $g_{abnormal}(x)$ and $g_{normal}(x)$, the decision in [244] was based on the ratio $g_{abnormal}(x)/_{normal}(x)$. A decision threshold T was set, such that if the ratio is larger than T the unknown pattern vector is classified as abnormal, else as normal. By changing T, the sensitivity/specificity trade–off of the Bayes classifier can be altered. A larger T will result in lower TP and FP rates, while a smaller T will result in higher TP and FP rates. The procedure described in [244] is illustrated in Figure 6.18.

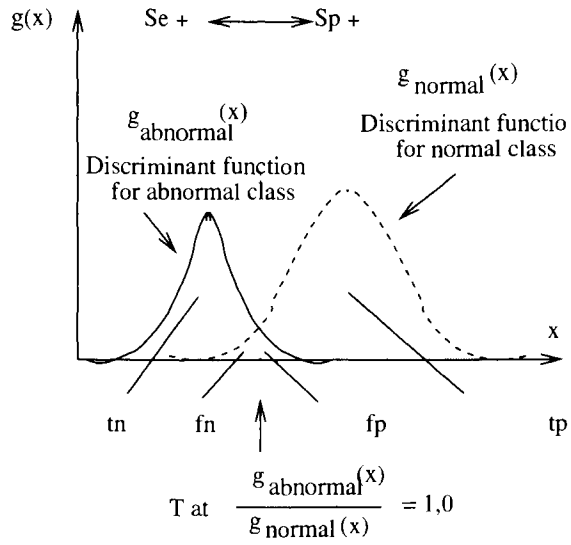


FIGURE 6.18 Discriminant functions for two populations, one with a disease, the other population without the disease. A perfect separation between the two groups is rarely given; an overlap is most often observed. The FN, FP, TP, and TN areas are denoted.

6.7.2 k-Nearest-Neighbor Classifier

The k-nearest-neighbor classifier described in [244] is based on two different parameters: k is the number of nearest neighbors to base the decision on, and a threshold T which specifies the minimum number of nearest neighbors that represent the disease before a new pattern vector is classified as an abnormality. The ROC points for a given k are determined by varying T from 1 to k and monitoring the resulting TP and FP rates. It is immediately evident that as T is increased, the TP and FP rates will decrease. In [244] k is varied from 1 to 250, and for each k an ROC curve is produced by altering T from 1 to 40. Then the AUC is computed. For a subsequent classification the k producing the maximum AUC is selected. The operating point of this classifier is selected by choosing an adequate threshold T.

6.7.3 Binary Decision Trees

In [244] it was shown that for obtaining ROC points the threshold for the abnormal class has to be varied over some range on the leaf nodes of the tree. Usually, a new pattern vector is assigned to the class with the highest probability. Dealing only with patterns being either abnormal or normal, the threshold T is at 50%. Increasing T for the abnormal class will lead to a decrease in TP and FP rates; lowering T will have a reverse effect.

6.8 APPLICATION OF STATISTICAL CLASSIFICATION METHODS IN BIOMEDICAL IMAGING

Pattern classification techniques play an important role in the detection and classification of tumors [103, 109, 110, 121, 171, 245]. The procedure is simple and follows the following scheme. After segmenting a suspicious region, a feature extraction and selection is performed in order to extract the relevant information from this region. A classification paradigm is chosen so that, based on the available features and determined tumor classes the best results are achieved.

This section describes the application of statistical or noncognitive classification methods to breast cancer detection.

A standard mammogram screening case consists of four images, two views of each breast. These are a craniocaudal (top-bottom) view, and a mediolateral (middle-to-outside) view. Sometimes, an abnormality or suspicious region can be missed by a radiologist in one view, but is easily detected in the other view.

A radiologist is looking for distinctive cues and characteristics indicative of cancer when evaluating a mammogram. Among these signs are the presence of clustered microcalcifications, spiculated or stellate lesions, circumscribed or well-defined masses, ill-defined or irregular masses, and architectural distortions. Sometimes asymmetry and developing densities are also considered.

A microcalcification is a tiny calcium deposit that has accumulated in the breast tissue and represents a small bright spot on the mammogram. A cluster is typically defined as including at least three to five microcalcifications within a 1-cm^2 region [228]. Up to 50% of malignant masses demonstrate clustered microcalcifications, and sometimes the clusters represent the only sign of malignancy. They can be embedded in dense parenchymal tissue. The size of calcifications ranges from smaller than 0.1 mm to 5 mm in diameter. Their detection is normally very difficult for a radiologist, who must usually locate them with a magnifier. When analyzing individual calcifications, two aspects are usually considered: (1) size, shape, and radiographic density for individual calcifications and (2) number and distribution of calcifications within a cluster.

There are a number of different types of masses which may be malignant. A circumscribed mass with a distinct border and circular in shape is easy to identify. High-density, radiopaque, circumscribed masses are mostly malignant; radiolucent masses are almost always benign. A detection difficulty is posed by ill–defined masses which have an irregular shape and less distinctive borders. Contour, density, shape, orientation, and size of the mass are important features for classifying a visible mass.

Spiculated lesions with a star-shaped appearance and blurred borders are far more difficult to detect. Like circumscribed and ill-defined masses,

the spiculated lesions have a radiographically dense tumor compared to the surrounding glandular tissue. The distinct star shape is the result of radially oriented spicules extending from the tumor center into the surrounding breast tissue. They are always malignant, and the density of the tumor center and the radiating structure are the most important features [228].

Architectural distortions in the connective tissue of the breast represent one of the most difficult abnormalities to detect. The breast structures are loosely arranged along the duct lines, which have the tendency to radiate in curvilinear arcs outward from the nipple to the chest wall. Both malignant and benign cancer forms are responsible for a disruption in this symmetric pattern.

The most frequent breast cancer forms are clustered microcalcifications and stellate lesions, as shown in Figure 6.19.

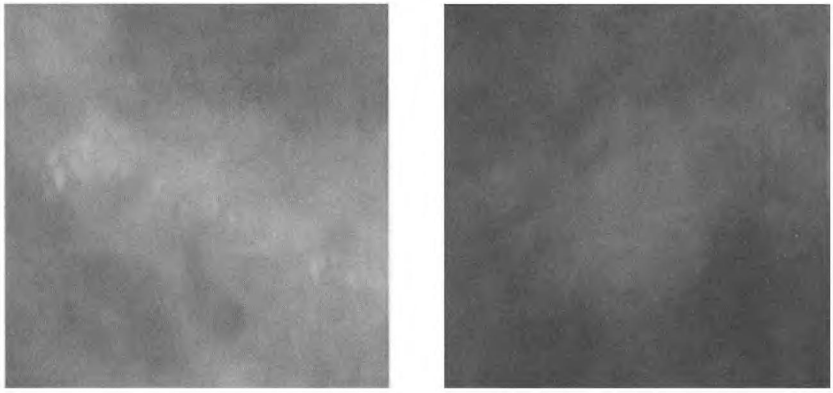


FIGURE 6.19 Example of (a) a clustered microcalcification and (b) a stellate lesion. (Images courtesy of Dr. A. Wismüller and the Dep. of Radiology, University of Munich.)

In this section, we will show how clustered microcalcifications and stellate lesions are detected based on statistical classification methods. Chapter 11 describes the computer-aided diagnosis of breast cancer based on emerging biologically inspired techniques.

6.8.1 Microcalcification Detection

The ground-truth location of each individual microcalcification is described by the center and radius of a circle. Since neither microcalcifications nor spiculated masses are perfectly round, each ground-truth circle always includes some pixels belonging to normal breast tissue.

To validate a positive calcification in an image, we must first check if the centroid of the detected object lies inside a circle denoting the location

of a known calcification in an image. When the centroid of the detected object lies within a ground-truth circle, then a TP is detected; otherwise it is an FP. To measure TP and FP cluster detections, detected microcalcifications have to be grouped together. A detected calcification is grouped with another detection if the distance between them is less than 1 cm. We have a TP detection if two or more calcifications are detected within the location of the cluster described by a circular area; otherwise we have a FP detection. This situation is shown in Figure 6.20.

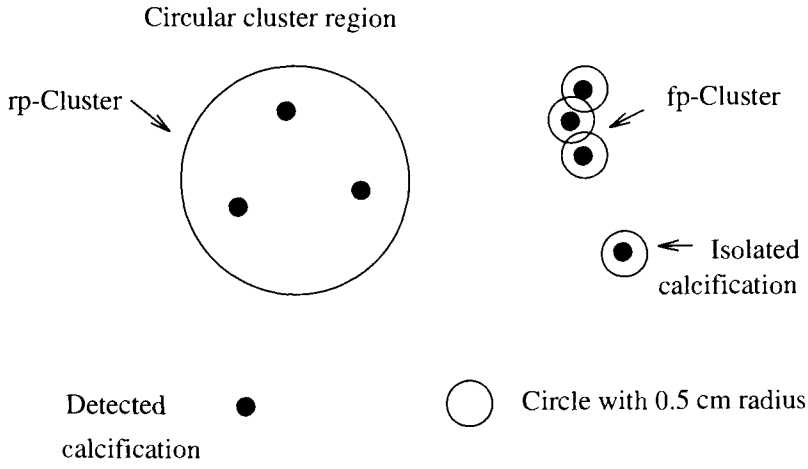


FIGURE 6.20 Examples of TP and FP microcalcification cluster detection. A TP is detected since at least two calcifications are found inside the ground-truth circle.

In [244] are described three categories of features, 42 in all extracted from a segmented suspicious region corresponding to a clustered microcalcification from the MIAS data set. These categories are:

1. Object size, contrast, and gray-level distribution within the object, which in the case of calcification detection refers to all pixels that are part of the segmented object.
2. Shape description of the segmented object.
3. Texture information from the image. Among these are the Laws' texture energy measures, modified Laws' measures, and fractal-based texture. They describe the local intensity distribution properties such as smoothness and coarseness, and also the structural properties such as regular patterns of image primitives.

The relevant features are determined based on an exhaustive search method. Figure 6.21 shows that a feature vector with seven components is optimal for both linear and quadratic Bayesian classifiers whereas a feature vector with six components is optimal for the k-nearest-neighbor classifier.

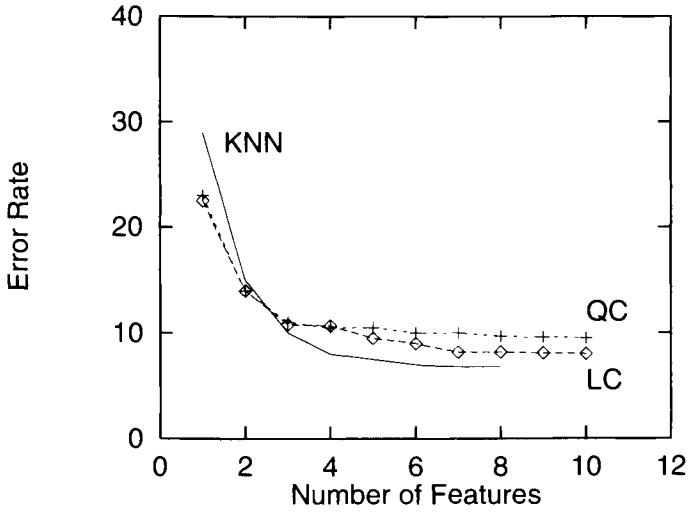


FIGURE 6.21 Feature selection results for microcalcification detection. Classification error rate versus number of features is shown for k-nearest-neighbor (k=1), linear and quadratic Bayesian classifier [244].

Table 6.2 shows the selected features for each classifier. A contrast measurement and various Law's texture energy measures are common to all three vectors. This makes sense because these features must aid in the detection of small, bright spots.

The AUC of four classifiers for microcalcification detection is shown in Table 6.3. The k-nearest-neighbor classifier is significantly better than the others [244].

6.8.2 Spiculated Lesion Detection

Breast cancer is mostly observed in the form of a mass (lesion) with margins that are well distinguished from the surrounding breast parenchyma. Cancerous lesions have a tumor center that is radiologically denser than the surrounding tissue and have spiculated (stellate) margins. A central tumor mass that is either very small or embedded in dense parenchymal tissue is difficult to perceive. Therefore all efforts are focused on detecting and diagnosing lesions when they are under 1 cm in diameter.

The ground-truth location of each individual spiculated lesion is described by the center and radius of a circle. The center of the circle is at the centroid of the object, and the radius equals the distance from the centroid to the furthest point of the object's perimeter. If the ground-truth circle overlaps, by at least 50%, the area of a circle surrounding a detected object, then the detection is considered a TP; otherwise it is an FP. Figure 6.22 visualizes this aspect.

TABLE 6.2 Feature vectors selected for microcalcification detection [244].

Classifier	Feature vector selected
Quadratic	absolute contrast, Laws $E5 * S5$, Laws $E5 * E5$, modified Laws $E5 * E5$, shape moment 3, fractal texture 1, fractal texture 2, fractal texture 2
Linear	average gray level, absolute contrast, Laws $E5 * S5$, Laws $S5 * S5$, modified Laws $L5 * E5$, average border gradient, fractal texture 2
k-nearest-neighbor	average gray level, object size, contrast, gray level moment 1, Laws $L5 * E5$, Laws $S5 * S5$

TABLE 6.3 AUC for individual classifiers applied to microcalcification detection [244].

Classifier	AUC
k-nearest-neighbor	93.0
Quadratic	91.7
Linear	90.0
Binary decision tree	85.0

Segmentation of spiculated lesions is usually performed in two steps: (1) extraction of locally bright spots of appropriate size and a (2) growing region procedure to improve the shape estimation of the regions initially extracted.

Tumor classification is usually either pixel-based or region-based. Texture is in general computed at the pixel level. In fact, the human vision system responds to texture at the "pixel" level. For example, features based on Laws' texture energy measures discriminate very well between various textured regions in an image when pixel-based classification is required.

A typical spiculated lesion has a distinct tumor center with spicules radiating outward. In [244], therefore, the discriminating power of each texture measurement was examined when the feature value was computed only for pixels located in various regions in and around the segmented objects. This is because the texture within and around different regions of a segmented object is different, and therefore the discriminating power of the texture features is lost if the computation is performed across the various regions. An object and its background are divided into three disjoint regions: middle, border, and the surrounding. This is shown in Figure 6.23.

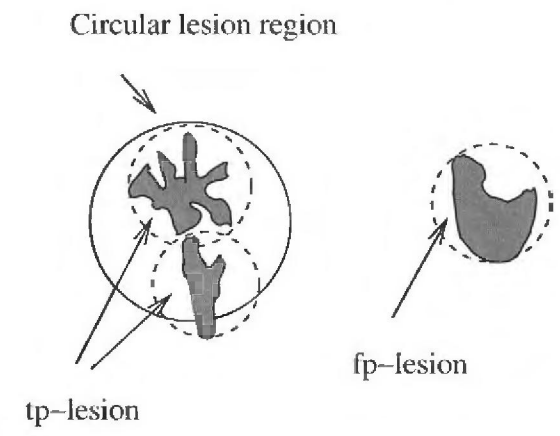


FIGURE 6.22 Examples of TP and FP spiculated lesions detections. The actual tumor is segmented into two TP detection objects.

As for microcalcifications, a total of 42 features were extracted from the segmented lesion. The three categories of features mentioned in [244] are:

1. Object size, contrast. and gray-level distribution within the object.
2. Local edge features such as the Analysis of Local Edge Orientation (ALOE) features [109].
3. Texture information from the image. Among these are the Laws' texture energy measures and fractal-based texture.

The relevant features are determined based on an exhaustive search method. Figure 6.24 shows that a feature vector with seven components is optimal for both linear Bayesian and k-nearest-neighbor classifiers while a feature vector with eight components is optimal for the quadratic Bayesian classifier.

Table 6.4 shows the selected features for each classifier. Each vector contains an ALOE feature which is representative for the radiating structure of the lesion spicules, and also two shape measurements. Also, the relative extremum density feature present in all vectors determines if spicules are present. In summary, the feature selection corresponds to the technique followed by a radiologist when detecting stellate lesions: the search for an oval or circular tumor center with a radiating structure consisting of spicules.

The AUC of four classifiers for spiculated lesion detection is shown in Table 6.5. As in the microcalcifications case, the k-nearest-neighbor classifier offers the best performance [244].

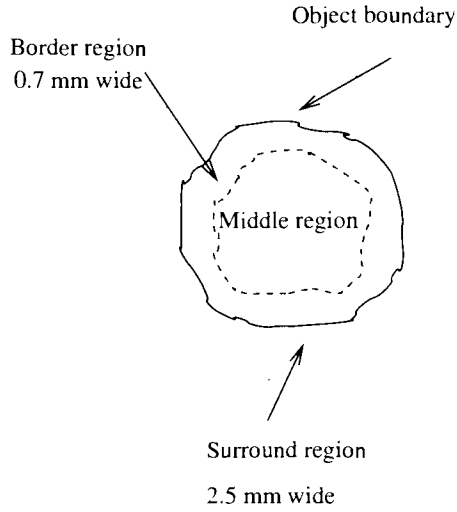


FIGURE 6.23 Regions of an object and its local background for which texture features are computed [244].

6.9 APPLICATION OF SYNTACTIC PATTERN RECOGNITION TO BIOMEDICAL IMAGING

Application of syntactic image recognition methods enables a more in-depth analysis by describing the semantic content of the image rather than just simply recognizing pathological lesions. The main importance of these methods lies in the design of intelligent visual and information diagnostic systems. These systems employ syntactic reasoning based on grammar use.

Syntactic pattern recognition based on grammatical reasoning has been widely used in medical image analysis in several application areas: (1) performing simple semantic reasoning related to their content [134], (2) morphology description of important shapes such as ECG course analysis [232], and (3) recognition of local irregularities in ureter lumens and examination of the morphology of the renal pelvis and calyxes [175, 176].

Syntactic pattern recognition is applied to produce a perceptual description of the analyzed structures and pathological signs first by defining simple shape elements, and then by building the general grammar description of the considered organ. These descriptions based on sequences or terminal symbols belong to the languages generated by the introduced grammar. Thus, the analysis and recognition of pathological signs are per-

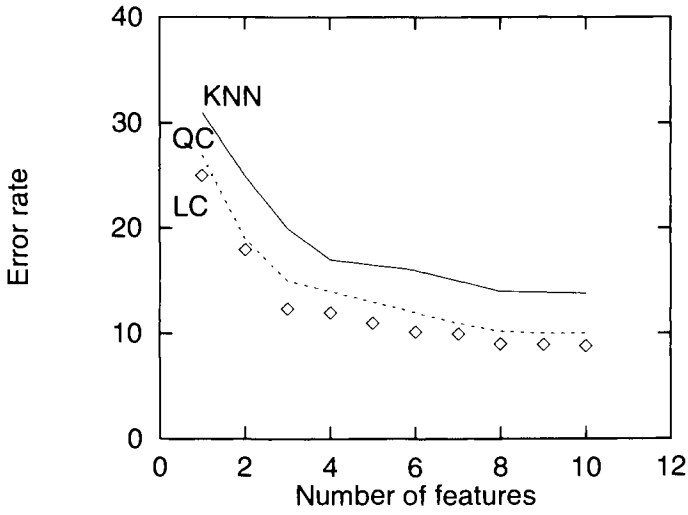


FIGURE 6.24 Feature selection results for spiculated lesion detection. Classification error rate versus number of features is shown for k-nearest-neighbor (k=1), and linear and quadratic Bayesian classifiers [244].

formed by parsing algorithms, which analyze input sequences and reduce them to one of the known categories.

This section describes the application of syntactic pattern recognition to support the diagnosis of ischemic heart disease based on coronography images [174]. The goal is to analyze and diagnose significant stenoses of the coronary lumen. Using context-free grammars enables the diagnosis of profiles of the examined artery width of both concentric stenoses and eccentric stenoses. Concentric stenoses appear on a cross-section as a uniform stricture of the whole lumen and present the characteristic of a stable disturbance of heart rhythm, whereas eccentric stenoses appear only on one vascular wall, and characterize an unstable angina. Examples of cardiac stenoses are visualized in Figure 6.25.

The goal of the structural analysis is to obtain width graphs that show the pathological changes occurring in these arteries. It is important to preserve the sequence of operations which are part of the required preprocessing. To obtain these width diagrams, a specific straightening transformation algorithm is employed that provides, besides the width diagrams of the analyzed vessels, the morphological lesions occurring in them and also the correct ramification of the diagnosed arteries. The required steps for the analysis of cardiac stenose images are [174]: (1) segmentation and skeletonization of the examined coronary artery, and (2) straightening transformation of the external contour of the examined artery from a

TABLE 6.4 Feature vectors selected for spiculated lesion detection [244].

Classifier	Feature vector selected
Quadratic	ALOE of radius, contrast, shape moment 2, shape moment 3, relative extrema density of outer region, Laws' $R5 * R5$ of object, fractal texture 3, Laws' $L5 * E5$ of angle image 2
Linear	relative extrema density of outer region, shape moment 1, compactness 1, Laws' $R5 * R5$ of object, relative extrema density of outer region of angle image 2, average edge strength, ALOE of the object and the surround
k-nearest-neighbor	ALOE of radius, contrast, shape moment 3, relative extrema density of outerregion, Laws' $E5 * E5$, Laws' $S5 * S5$, compactness 1

TABLE 6.5 AUC for individual classifiers applied to microcalcification detection [244].

Classifier	AUC
k-nearest-neighbor	95.9
Quadratic	95.0
Linear	94.6
Binary decision tree	78.9

two-dimensional space to a one-dimensional diagram, thus visualizing the profile of the straightened coronary artery.

The recognition and correct description of the degree of advancement of lesions are done in [174] by a context-free attributed grammar of the look-ahead $LR(1)$-type. The attributed grammar enables the diagnosis of coronographic X-ray examinations by defining all potential shapes of expected morphological lesions. Automata of the $LR(1)$-type analyze the input sequence from the left-hand to the right-hand side by processing its successive elements. The output represents the opposite of the right-hand derivation. At each analysis step, only one input symbol is examined. The detailed definition of the attributed grammar and of the $LR(1)$-type automate can be found in [174].

The following attributed grammar is proposed in [174] for various shape recognition of stenoses:

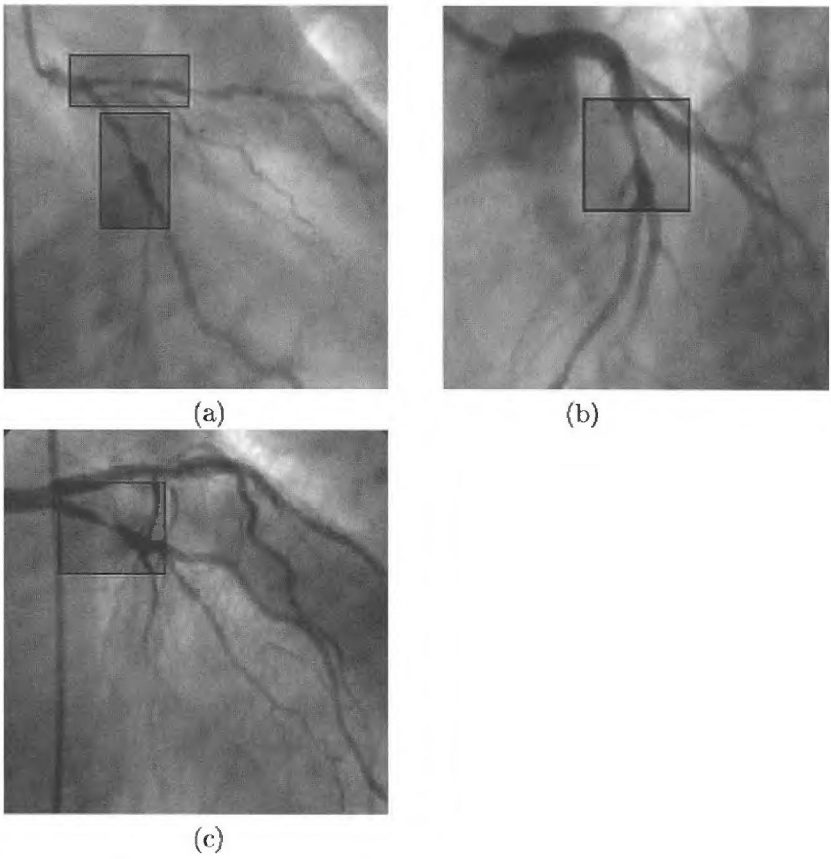


FIGURE 6.25 Examples of cardiac stenoses. (Reprinted from [174] with permission from Elsevier Science.)

$$G_{CA} = (V_N, V_T, SP, STS) \tag{6.67}$$

where V_N is the set of non-terminal symbols, V_T the set of terminal symbols, SP the production set, and STS the grammar start symbol.

$$\begin{aligned} V_N &= \{SYMPTOM, STENOSIS, H, V, NV\} \\ V_T &= \{h, \nu, n\nu\} \quad \text{for} \\ & \quad h \in (-10^o, 10^o), \nu \in (11^o, 90^o), n\nu \in (-11^o, -90^o) \end{aligned} \tag{6.68}$$

STS=SYMPTOM
SP:

1. SYMPTOM → STENOSIS	SYMPTOM:=STENOSIS	
2. STENOSIS → NVHV		
3. STENOSIS → NV $v\|NHV$		
4. V → $\nu\|V\nu$	$w_{sym} := w_{sym} + w_{nv}$;	$h_{sym} := h_{sym} + h_{nv}$
5. NV → $nv\|NVnv$	$w_{sym} := w_{sym} + w_v$;	$h_{sym} := h_{sym} + h_v$
6. H → $h\|Hh$	$w_{sym} := w_{sym} + w_h$;	$h_{sym} := h_{sym} + h_h$

The first production rule (Symptom:=Stenosis) states that a vessel stenosis was discovered in the input sequence. The second and third production rules define potential shapes of cardiac stenoses while successive rules define the upward and downward ramifications of the analyzed stenosis. Thus, the semantic variables h_e and w_e specify the altitude and length of the terminal section labeled e and make the diagnosis more specific by defining (in percentage) the degree of the cardiac stenosis.

A recognition rate of 93% was reported in [174] based on the proposed structural analysis. The recognition rate describes the percentage of accurately recognized and measured vessel stenoses versus the total number of tested images. The recognition pertains to determining the correct location of a stenosis and its type (concentric or eccentric).

In summary, the objective of the study performed in [174] was to determine the percentage of the correct recognition of artery stenosis and its size. Figure 6.26 shows examples from [174] of detected cardiac stenosis. The recognized symptoms are marked with a bold line.

EXERCISES

1. The following class-dependent pdfs are given:

$$p(\mathbf{x}|\omega_1) = \begin{cases} \frac{1}{16}, & \text{for} \quad 1 \leq x_1 \leq 5 \quad \text{and} \quad 1 \leq x_2 \leq 5 \\ 0, & \text{else} \end{cases} \tag{6.69}$$

$$p(\mathbf{x}|\omega_2) = \begin{cases} 1, & \text{for} \quad 2 \leq x_1 \leq 3 \quad \text{and} \quad 2 \leq x_2 \leq 3 \\ 0, & \text{else} \end{cases} \tag{6.70}$$

where $\mathbf{x}$ is a two-dimensional vector. Assume $P(\omega_1) = p(\omega_2) = \frac{1}{2}$.

a) Draw the decision regions.

b) Develop the classification strategy.

c) Compute the classification error.

2. Consider the following one-dimensional, two-class case where $p(x|\omega_1)$ is a Gaussian pdf with mean $\mu = 0$ and $\sigma_i^2 = 1$ and $p(x|\omega_2)$ is a uniform pdf with mean $\mu = 2$ and $\sigma_1^2 = \frac{1}{3}$. We also assume equal a priori probabilities.

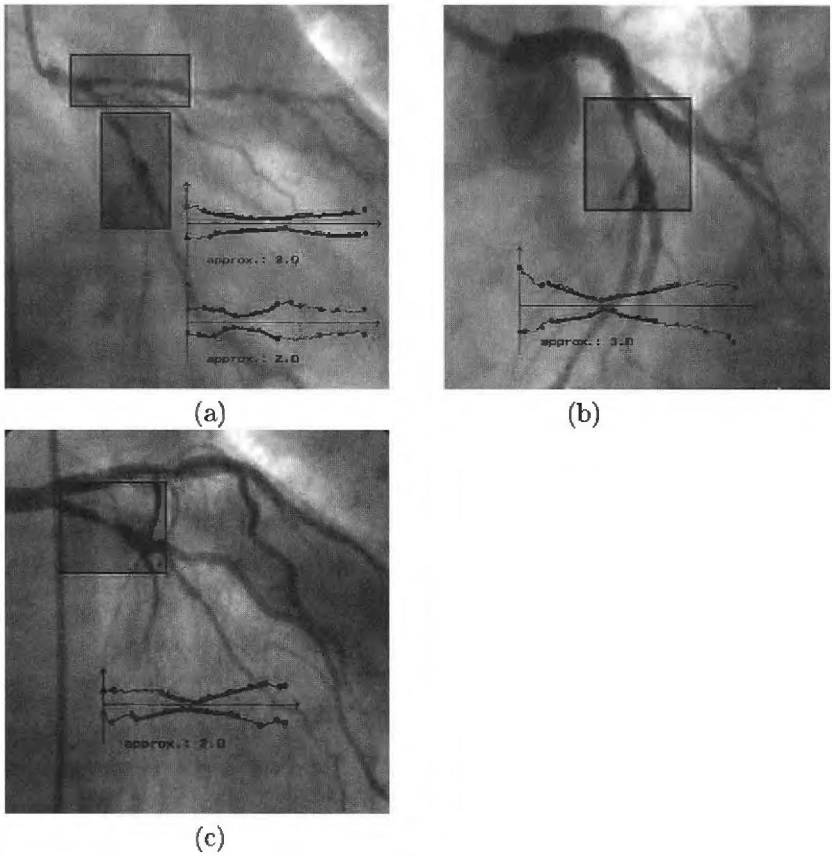


FIGURE 6.26 Results of structural pattern recognition of cardiac stenoses. (Reprinted from [174] with permission from Elsevier Science.)

a) Plot $p(x|\omega_1)$ and $p(x|\omega_2)$.

b) Determine the classification method that minimizes the classification error. What are the discriminant functions $g_1(x)$ and $g_2(x)$?

c) Compute the probability of the decision error.

3. In a three-class, two-dimensional classification problem the feature vectors in each class are normally distributed with covariance matrix

$$\Sigma = \begin{bmatrix} 1.2 & 0.5 \\ 0.5 & 1.8 \end{bmatrix} \tag{6.71}$$

The mean vectors for each class are $\mu_1 = [0.1, 0.2]^T$, $\mu_2 = [2.2, 1.9]^T$, and $\mu_3 = [-1.4, 2.1]$. Assume that the classes are equiprobable.

a) Classify the feature vector $[1.8, 1.6]^T$ according to the Bayes minimum error probability classifier.

b) Draw the curves of equal Mahalanobis distance from $[2.2, 1.8]^T$.

4. Consider the multivariate Gaussian pdf with $\sigma_{ij} = 0$ and $\sigma_{ii} = \sigma_i^2$.

a) Show that this pdf is given by

$$p(\mathbf{x}) = \frac{1}{\prod_{i=1}^{d} \sqrt{2\pi}\sigma_i} \exp\left[-\frac{1}{2}\sum_{i=1}^{d}\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2\right] \tag{6.72}$$

b) describe the contours of constant density, and write an expression for the Mahalanobis distance from $\mathbf{x}$ to μ.

5. Show that if the likelihood function is Gaussian, we get as estimates for the mean μ and the covariance matrix Σ the following values

$$\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_k \tag{6.73}$$

and

$$\widehat{\Sigma} = \frac{1}{N}\sum_{k=1}^{N}(\mathbf{x}_k - \widehat{\mu})(\mathbf{x}_k - \widehat{\mu})^T \tag{6.74}$$

6. The pdf of a random variable x is given by

$$p(x;\theta) = \theta^2 x \exp(-\theta x)u(x) \tag{6.75}$$

with $u(x)$ being the unit step function. Determine the maximum likelihood estimate for θ under the condition that N measurements, $x_1, \ldots, x_N$, are known.

7. The pdf of a random variable has the following exponential distribution:

$$p(x|\alpha) = \begin{cases} \theta \exp -\alpha x, & x \geq 0 \\ 0, & \text{else} \end{cases} \tag{6.76}$$

a) Plot $p(x|\alpha)$ versus x for a fixed value of the parameter θ.

b) Plot $p(x|\alpha)$ versus $\theta > 0$ for a fixed value of x.

c) Suppose that n samples are drawn independently according to $p(x|\alpha)$. Show that the maximum likelihood estimate for α is given by

$$\widehat{\alpha} = \frac{1}{\frac{1}{n}\sum_{k=1}^{n} x_k} \tag{6.77}$$

8. The pdf of a random variable x is given by

$$p(x) = \begin{cases} \frac{1}{3}, & 1 < x < 3 \\ 0, & \text{else} \end{cases} \tag{6.78}$$

Employ for its approximation the Parzen window method and use a Gaussian kernel with $\mu = 0$ and $\sigma = 1$. Let the smoothing parameters be

a) $h = 0.5$.

b) $h = 3$.

For (a) and (b) plot the approximation based on $N = 32, N = 256$, and $N = 5,000$ points, which are produced by a pseudorandom generator according to $p(x)$.

9. The pdf of a random variable x is given by

$$p(x) = \begin{cases} 1, & -2.5 < x < -2 \\ 0.125, & 0 < x < 4 \\ 0, & \text{else} \end{cases} \tag{6.79}$$

Employ for its approximation the Parzen window method and use a Gaussian kernel with $\mu = 0$ and $\sigma = 1$. Let the smoothing parameters be

a) $h = 0.25$.

b) $h = 1$.

For (a) and (b) plot the approximation based on $N = 32, N = 256$, and $N = 5,000$ points, which are produced by a pseudorandom generator according to $p(x)$.

10. Determine $L(\mathbf{G})$ for each of the following grammars ([216], p. 142:)

a) Context-free:

$$\begin{array}{lcl} S & \rightarrow & aAa \\ A & \rightarrow & a \\ A & \rightarrow & b \end{array} \tag{6.80}$$

b) Context-sensitive:

$$\begin{array}{lcl} S & \rightarrow & SC \\ CB & \rightarrow & Cb \\ aB & \rightarrow & aa \\ bB & \rightarrow & bb \end{array} \tag{6.81}$$

c) Finite-state:

$$\begin{array}{lcl} S & \rightarrow & aA_1 \\ S & \rightarrow & bA_1 \\ A_1 & \rightarrow & a \\ A_1 & \rightarrow & b \end{array} \tag{6.82}$$

11. Define a grammar that produces the language

$$L = \{a^n b^{n+2} | n > 0\} \tag{6.83}$$

12. For the Finite-state grammar productions shown in Exercise 6.10c:

a) Determine $L(\mathbf{G})$.

b) How does $L(\mathbf{G})$ change if the production rules are changed to

$$\begin{array}{lcl} S & \rightarrow & bA_1 \\ S & \rightarrow & aA_1 \\ A_1 & \rightarrow & b \\ A_1 & \rightarrow & a \end{array} \tag{6.84}$$

13. Let $X = \{\mathbf{x}_i, i = 1, \ldots, 5\}$, with $\mathbf{x}_1 = [1, 1]^T$, $\mathbf{x}_2 = [2, 1]^T$, $\mathbf{x}_3 = [4, 6]^T$, $\mathbf{x}_4 = [2, 7]^T$, and $\mathbf{x}_5 = [3, 4]^T$.

a) Determine the dissimilarity matrix of the data set when the Euclidean distance is in use.

b) On the data set perform agglomerative clustering based on the single-link algorithm [229] described in eq. (6.63).

VII

FOUNDATIONS OF NEURAL NETWORKS

7.1 INTRODUCTION

Advances in clinical medical imaging have brought about the routine production of vast numbers of medical images that need to be analyzed. As a result, an enormous amount of computer vision research effort has been targeted at achieving automated medical image analysis. This has proved to be an elusive goal in many cases. The complexity of the problems encountered in analyzing these images has prompted considerable interest in the use of neural networks for such applications.

Artificial neural networks are an attempt to emulate the processing capabilities of biological neural systems. The basic idea is to realize systems capable of performing complex processing tasks by interconnecting a high number of very simple processing elements which might even work in parallel. They solve cumbersome and intractable problems by learning directly from data. An artificial neural network usually consists of a large amount of simple processing units, i.e., neurons, with mutual interconnections. It learns to solve problems by adequately adjusting the strength of the interconnections according to input data. Moreover, it can be easily adapted to new environments by learning. At the same time, it can deal

with information that is noisy, inconsistent, vague, or probabilistic. These features motivate extensive research and developments in artificial neural networks.

The main features of artificial neural networks are their massive parallel processing architectures and the capabilities of learning from the presented inputs. They can be utilized to perform a specific task only by means of adequately adjusting the connection weights, i.e., by training them with the presented data. For each type of artificial neural network, there exists a corresponding learning algorithm by which we can train the network in an iterative updating manner. Those learning algorithms fit into two main categories: supervised learning and unsupervised learning.

For supervised learning, not only the input data but also the corresponding target answers are presented to the network. Learning is done by the direct comparison of the actual output of the network with known correct answers. This is also referred to as learning with a teacher. In contrast, if only input data without the corresponding target answers are presented to the network for learning, we have unsupervised learning. In fact, the learning goal is not defined at all in terms of specific correct examples. The available information is in the correlations of the input data. The network is expected to create categories from these correlations and to produce output signals corresponding to the input category.

Neural networks have been successfully employed to solve a variety of computer vision problems. They are systems of interconnected, simple processing elements. Generally speaking, a neural network aims to learn the nonlinear mapping from an input space describing the sensor information onto an output space describing the classes to which the inputs belong.

There are three basic mapping neural networks known in literature [84]:

1. Recurrent networks: They are nonlinear, fully interconnected systems and form an autoassociative memory. Stored patterns correspond to stable states of a nonlinear system. The neural network is trained via a storage prescription that forces stable states to correspond to local minima of a network "energy" function. The memory capacity, or allowed number of stored patterns, is related to the network size. The Hopfield neural network [88] is the most popular recurrent neural network. A similar model is the bidirectional associative memory (BAM) [119].

2. Multilayer feedforward neural networks: They implement a nonlinear mapping between an input and output space and are nonlinear function approximators. Also, the network is able to extract higher-order statistics. The multilayer perceptron [143], the backpropagation-type neural network [48], and the radial-basis neu-

ral network [166] are the best known multilayer feedforward neural networks.

3. Local interaction-based neural networks: These networks are based on competitive learning; the output neurons of the network compete among themselves to be activated. The output that wins the competition is called a winning neuron. One way of introducing competition among the output neurons is to use lateral inhibitory connections between them [205]. The local interaction is found in Kohonen maps [114], ART maps [77, 78], and in the von der Malsburg model [51, 242].

The classification in terms of architecture and learning paradigm of the neural networks described in this chapter is shown in Figure 7.1.

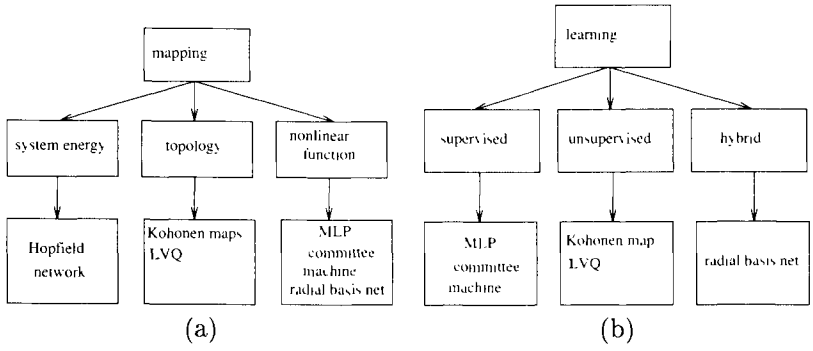


FIGURE 7.1 Classification of neural networks based on (a) architecture type and (b) learning algorithm.

There exist many types of neural networks that solve a wide range of problems in the area of image processing. There are also many types of neural networks, determined by the type of connectivity between the processing elements, the weights (synapses) of the connecting links, the processing elements' characteristics, and training or learning rules. These rules specify an initial set of weights and indicate how weights should be modified during the learning process to improve network performance.

The theory and representation of the various network types is motivated by the functionality and representation of biological neural networks. In this sense, processing units are usually referred to as neurons, while interconnections are called synaptic connections.

Although different neural models are known, all have the following basic components in common:

1. A finite set of neurons $x(1), x(2), \ldots, x(n)$ with each neuron having a specific activity at time t, which is described by $x_t(i)$

2. A finite set of neural connections $\mathbf{W} = (w_{ij})$, where w_{ij} describes the strength of the connection of neuron $x(i)$ with neuron $x(j)$
3. A propagation rule $\tau_t(i) = \sum_{j=1}^{n} x_t(j) w_{ij}$
4. An activation function f, which has τ as an input value and produces the next state of the neuron $x_{t+1}(i) = f(\tau_t(i) - \theta)$, where θ is a threshold and f is a nonlinear function such as a hard limiter, threshold logic, or sigmoidal function

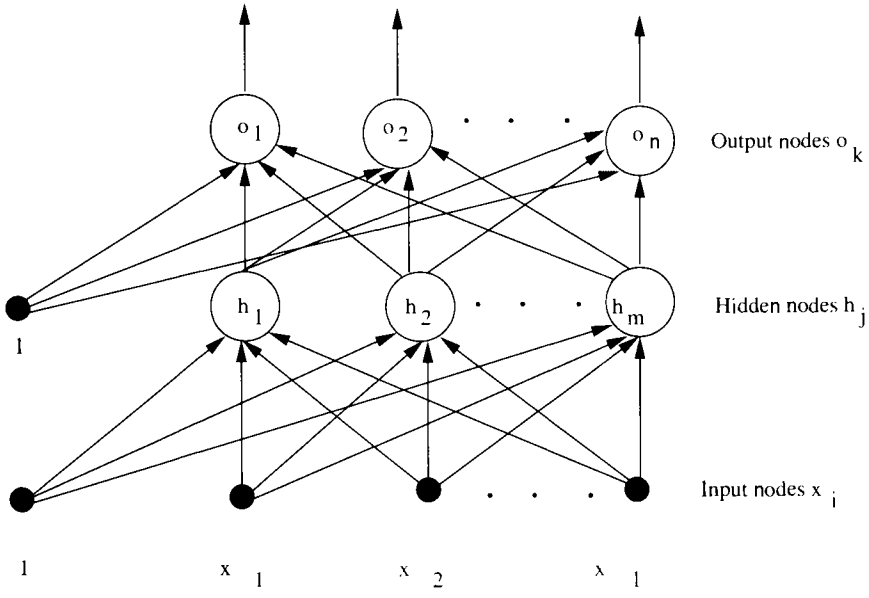


FIGURE 7.2 Two-layer perceptron.

7.2 MULTILAYER PERCEPTRON (MLP)

Multilayer perceptrons are one of the most important types of neural nets because many applications are successful implementations of MLPs. The architecture of the MLP is completely defined by an input layer, one or more hidden layers, and an output layer. Each layer consists of at least one neuron. The input vector is processed by the MLP in a forward direction, passing through each single layer. Figure 7.2 illustrates the configuration of the MLP.

A neuron in a hidden layer is connected to every neuron in the layers above and below it. In Figure 7.2 weight w_{ij} connects input neuron x_i to hidden neuron h_j and weight v_{jk} connects h_j to output neuron o_k. Classification starts by assigning the input neurons x_i, $1 \leq i \leq l$, equal

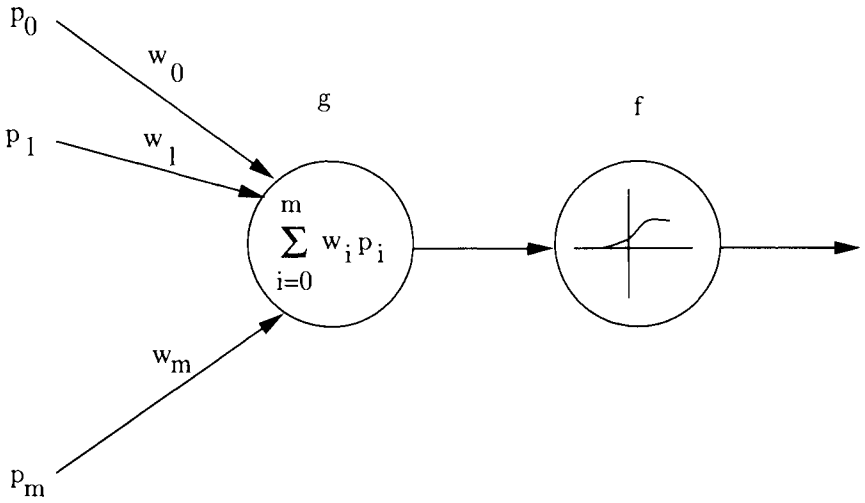


FIGURE 7.3 Propagation rule and activation function for the MLP network.

to the corresponding data vector component. Then data propagates in a forward direction through the perceptron until the output neurons o_k, $1 \leq k \leq n$, are reached. Assuming that the output neurons are equal to either 0 or 1, then the perceptron is capable of partitioning its pattern space into 2^n classes.

Before employing any pattern recognition system, we need first to collect the sensor signals that form the input data. Then the data are normalized and filtered in order to remove the noise component. Features represent only the relevant data information that needs to be extracted from the available sensor information. The subsequent classification assigns the resulting feature vector to one of the given classes. While most classification techniques operate based on an underlying statistical model, neural networks estimate the necessary discriminant functions. Mathematically, the multilayer perceptron performs a nonlinear approximation using sigmoidal kernel functions as hidden units and linear weights. During the learning of feature vectors, the weights are continuously adapted by minimizing the error between desired outputs and the computed network's outputs.

The steps that govern the data flow through the perceptron during classification are [203]:

1. Present the pattern $\mathbf{p} = [p_1, p_2, \ldots, p_l] \in \mathbf{R}^1$ to the perceptron, that is, set $x_i = p_i$ for $1 \leq i \leq l$.
2. Compute the values of the hidden-layer neurons as illustrated in Figure 7.3.

$$h_j = \frac{1}{1 + \exp\left[-\left(w_{0j} + \sum_{i=1}^{l} w_{ij} x_i\right)\right]}; \quad 1 \leq j \leq m \tag{7.1}$$

The activation function of all units in the MLP is given by the sigmoid function $f(x) = \frac{1}{1+\exp(-x)}$ and is the standard activation function in feedforward neural networks. It is defined as a monotonic increasing function representing an approximation between nonlinear and linear behavior.

3. Calculate the values of the output neurons based on

$$o_k = \frac{1}{1 + \exp\left(v_{0k} + \sum_{j=1}^{m} v_{jk} h_j\right)}: \quad 1 \leq k \leq n \tag{7.2}$$

4. The class $\mathbf{c} = [c_1, c_2, \ldots, c_n]$ that the perceptron assigns $\mathbf{p}$ must be a binary vector. So o_k must be the threshold of a certain class at some level τ and depends on the application.
5. Repeat steps 1, 2, 3, and 4 for each given input pattern.

Multilayer perceptrons are highly nonlinear interconnected systems and serve for both nonlinear function approximation and nonlinear classification tasks. A typical classification problem that can be solved only by the MLP is the XOR problem. Based on a linear classification rule, R^m can be partitioned into regions separated by a hyperplane. On the other hand, the MLP is able to construct very complex decision boundaries as depicted in Figure 7.4.

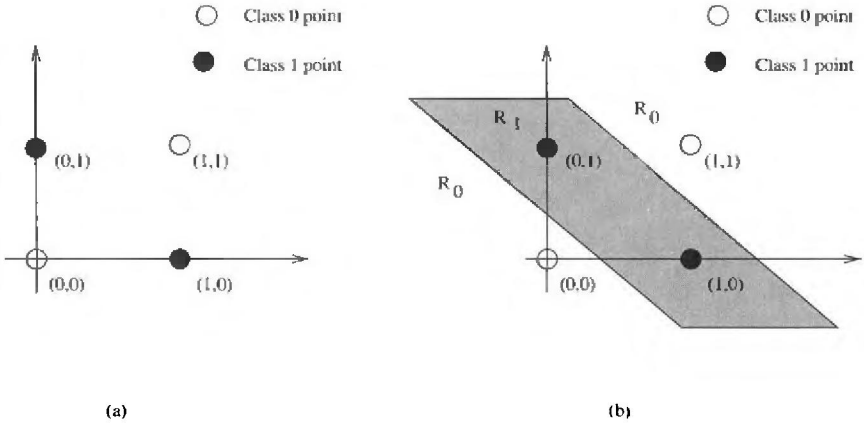


FIGURE 7.4 XOR problem and solution strategy using the MLP.

Applied to image processing the MLP has as input features extracted from images or from regions of these images. Such features can be shape,

size, or texture measures and attempt to capture the key aspects of an image.

Multilayer perceptrons have been applied to a variety of problems in image processing, including optical character recognition [208] and medical diagnosis [52, 53].

7.2.1 Backpropagation-Type Neural Networks

Multilayer perceptrons (MLPs) have been applied successfully to sophisticated classification problems. The training of the network is accomplished based on a supervised learning technique that requires given input-output data pairs. The training technique, known as the error backpropagation algorithm, is bidirectional consisting of a forward and backward direction. During the forward direction a training vector is presented to the network and classified. The backward direction consists of a recursive updating of the weights in all layers based on the computed errors. The error terms of the output layer are a function of $\mathbf{c}^t$ and output of the perceptron $(o_1, o_2, \ldots, o_n)$.

The algorithmic description of the backpropagation is as follows [48]:

1. **Initialization:** Initialize the weights of the perceptron randomly with numbers between -0.1 and 0.1; that is,

$$\begin{aligned} w_{ij} &= \text{random}([-0.1, 0.1]), \quad 0 \le i \le l, \quad 1 \le j \le m \\ v_{jk} &= \text{random}([-0.1, 0.1]), \quad 0 \le j \le m, \quad 1 \le k \le n \end{aligned} \tag{7.3}$$

2. **Presentation of training patterns:** Present $\mathbf{p}^t = [p_1^t, p_2^t, \ldots, p_l^t]$ from the training pair $(\mathbf{p}^t, \mathbf{c}^t)$ to the perceptron and apply steps 1, 2, and 3 from the perceptron classification algorithm previously described.
3. **Forward computation (output layer):** Calculate the errors $\delta_{ok}, 1 \le k \le n$, in the output layer using

$$\delta_{ok} = o_k(1 - o_k)(c_k^t - o_k) \tag{7.4}$$

where $\mathbf{c}^t = [c_1^t, c_2^t, \ldots, c_n^t]$ represents the correct class of $\mathbf{p}^t$. The vector $(o_1, o_2, \ldots, o_n)$ describes the output of the perceptron.

4. **Forward computation (hidden layer):** Calculate the errors δ_{hj}, $1 \le j \le m$, in the hidden-layer neurons based on

$$\delta_{hj} = h_j(1 - h_j) \sum_{k=1}^{n} \delta_{ok} v_{jk} \tag{7.5}$$

5. **Backward computation (output layer):** Let v_{jk} denote the value of weight v_{jk} after the tth training pattern has been presented

to the perceptron. Adjust the weights between the output layer and the hidden layer based on

$$v_{jk}(t) = v_{jk}(t-1) + \eta\delta_{ok}h_j \tag{7.6}$$

The parameter $0 \leq \eta \leq 1$ represents the learning rate.

6. **Backward computation (hidden layer):** Adjust the weights between the hidden layer and the input layer using

$$w_{ij}(t) = w_{ij}(t-1) + \eta\delta_{hj}p_i^t \tag{7.7}$$

7. **Iteration:** Repeat steps 2 through 6 for each pattern vector of the training data. One cycle through the training set defines an iteration.

7.2.2 Design Considerations

MLPs are global approximators and can be trained to implement any given nonlinear input-output mapping. In a subsequent testing phase, they prove their interpolation ability by generalizing even in sparse data space regions.

When designing a neural network, specifically deciding for a fixed architecture, performance and computational complexity considerations play a crucial role. Mathematically, it has been proved [82] that even one hidden-layer MLP is able to approximate the mapping of any continuous function.

As with all neural networks, the dimension of the input vector dictates the number of neurons in the input layer, while the number of classes to be learned dictates the number of neurons in the output layer. The number of chosen hidden layers and the number of neurons in each layer have to be empirically determined. As a rule of thumb, the neurons in the hidden layers are chosen as a fraction of those in the input layer. However, there is a trade-off regarding the number of neurons: Too many produce overtraining; too few affect generalization capabilities. Usually, after satisfactory learning is achieved, the neurons' number can be gradually reduced. Then subsequent retraining of a reduced-size network exhibits much better performance than the initial training of the more complex network.

Great care must be taken when selecting the training data set. It has to be representative of all data classes to allow a good generalization. During training, input vectors are presented in random order so that the network achieves a global learning and not class-selective learning.

7.2.3 Complexity of the Multilayer Perceptron

The complexity of a classifier is solely described by its adjustable parameters. Thus, the MLP's adaptable parameters are the number of weights and biases. Theoretically, there is always an optimal complexity for a given classification problem. If this optimal complexity for an MLP is exceeded,

then the classification outcome becomes prone to the intrinsic data noise. Although the MLP learns all training examples very well, it fails to correctly recognize new feature vectors it did not learn. This effect is called overfitting and always occurs when the number of weights exceeds the number of input patterns. Underfitting occurs when we have a lower complexity than the optimum, and thus an insufficient discrimination capability. In other words, each classification problem requires a certain optimum MLP complexity, and thus complexity control becomes an issue.

Complexity control can be obtained by the following techniques:

- Data preprocessing: feature extraction, feature selection, feature dimensionality reduction, as shown in Chapter II
- Training scheme: early stopping, cross validation
- Network structure: committee machines

7.2.3.1 Cross Validation The simplest method to avoid overfitting is to provide more training samples so that the MLP cannot model the noise of the training data set. In practical applications, it may happen that there is a shortage of labeled data available.

A solution to this dilemma poses a technique from statistics known as cross-validation [222]. The MLP is trained on a disjoint subset of the input data, and then a cross-validation subset of the data is used. Thus, the MLP is trained on the training set and then the classification error is checked on the validation set. This is repeated, and then the training is stopped when the cross-validation measure given by the classification errors stops improving. This means that continued training results in overtraining on the training data subset of input data, and this implies that the MLP loses its generalization ability on the cross-validation set.

This procedure proves to be extremely useful when we want to achieve a good generalization (optimal number of hidden neurons) and when we want to decide when the training of the MLP has to be stopped.

7.2.3.2 The Process of Training and Validation The goal of training a neural network is to achieve a good generalization. However, it is difficult to decide when it is best to stop training without ending up with overfitting. Overfitting can be easily detected by using cross-validation.

In general, the process of achieving good generalization with an MLP consists of three parts: (1) training, (2) validation, and (3) verification. This means, that we need three disjoint sets of input data: a training set that is the largest set, a validation set, and a verification set or test set. Validation has to be performed after cycles of several iterations during training to check when the classification error stops decreasing. The classification error is an ideal sensitivity measure in this respect since it

deteriorates when the training of the MLP is too specialized on the training set. The advantage of this method is that we can stop the training before the classification error starts degradating and choose the respective MLP configuration as its final. Hence the name "early stopping." Figure 7.5 visualizes this technique. The validation subset is small compared to the training set. The final step represents the testing phase when the trained MLP is verified to demonstrate that it is indeed implementing a correct classification mechanism. Usually, the test set is half the size of the training set.

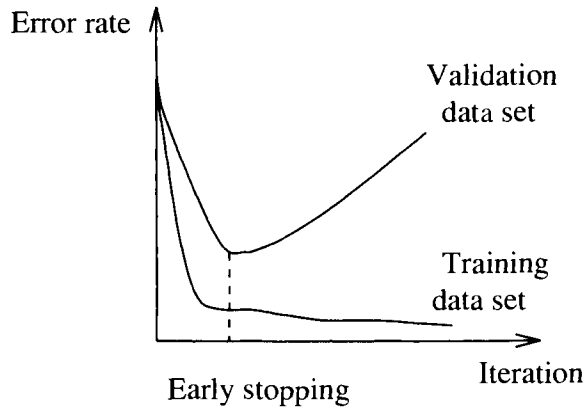


FIGURE 7.5 Early stopping method of training.

7.2.3.3 Committee Machines Complex classification problems in practice require the contribution of several neural networks for achieving an optimal solution. Thus a complex task is split into smaller ones whose contributions are necessary to provide the global complex solution. In neural network terminology, it means to allocate the learning task among a number of experts, which in turn split the input data into a set of subspaces. This idea leads to the concept of the committee machine, which is based on a combination of concurrent experts, and thus implements the famous principle of divide and conquer. In this way, it combines the knowledge of every single expert and achieves a global decision that is superior to that achieved by a single expert. The idea of the committee machine was first introduced by Nilsson [170].

Committee machines fall into the category of universal approximators and can have either a static or a dynamic structure. In the dynamic structures, the input signal directly influences the mechanism that fuses the output decisions of the individual experts into an overall output deci-

sion. There are two groups of dynamic structures: mixture of experts and hierarchical mixture of experts.

- Mixture of experts: A single gating network represents a nonlinear function of the individual responses of the experts, and thus the divide-and-conquer algorithm is applied only once.
- Hierarchical mixture of experts: a hierarchy of several gating networks forms a nonlinear function of the individual responses of the experts, and thus the divide and conquer algorithm is applied several times at the corresponding hierarchy levels.

Figure 7.6 shows a hierarchical mixture of experts [94]. The architecture is a tree in which the gating networks sit at the nonterminals of the tree. These networks receive the vector $\mathbf{x}$ as input and produce scalar outputs that are a partition of unity at each point in the input space. The expert networks sit at the leaves of the tree. They are linear with the exception of a single output nonlinearity. Expert network (i, j) produces its output μ_{ij} as a generalized function of the input vector $\mathbf{x}$ and a weight matrix $\mathbf{U}_{ij}$

$$\mu_{ij} = \mathbf{U}_{ij}\mathbf{x} \tag{7.8}$$

The neurons of the gating networks are nonlinear.

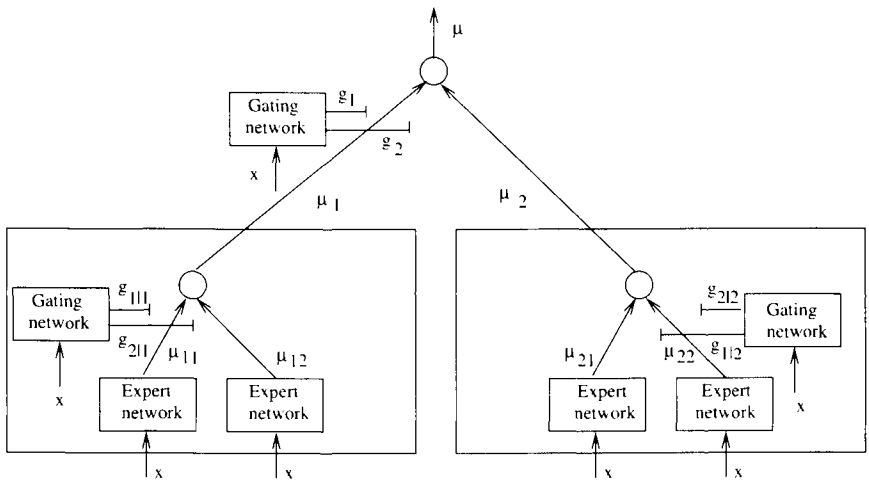


FIGURE 7.6 Hierarchical mixture of experts.

Let ξ_i be an intermediate variable

$$\xi_i = \mathbf{v}_i^T \mathbf{x} \tag{7.9}$$

$\mathbf{v}_i$ is a weight vector. Then the ith output of the top-level gating network is the "softmax" function of ξ_i given as

$$g_i = \frac{\exp(\xi_i)}{\sum_k \exp(\xi_k)} \tag{7.10}$$

Note that $g_i > 0$ and $\sum_i g_i = 1$. The g_i values can be interpreted as providing a "soft" partitioning of the input space.

Similarly, the neurons of the gating networks at lower levels are also nonlinear. Define ξ_{ij} as

$$\xi_{ij} = \mathbf{v}_{ij}^T \mathbf{x} \tag{7.11}$$

Then

$$g_{i\,j} = \frac{\exp(\xi_{ij})}{\sum_k \exp(\xi_{ik})} \tag{7.12}$$

is the output of the jth unit in the ith gating network at the second level of the architecture.

The output vector at each nonterminal of the tree is the weighted output of the experts below the nonterminal. That is, the output at the ith nonterminal in the second layer is

$$\mu_i = \sum_j g_{j\,i} \mu_{ij} \tag{7.13}$$

and the output at the top level is

$$\mu = \sum_i g_i \mu_i \tag{7.14}$$

Both g and μ depend on the input $\mathbf{x}$; thus the total output is a nonlinear function of the input.

7.3 SELF-ORGANIZING NEURAL NETWORKS

A self-organizing map is implemented by a 1D or 2D lattice of neurons. The neurons become specialized to various input patterns or classes of input patterns while the network performs an unsupervised competitive learning process. The weights of the neurons that have close locations on the lattice are trained to represent similar input vectors. We obtain a network that preserves neighborhood relations in the input space. and thus preserves the topology of the input space. This map is known as a self-organizing feature map [115].

7.3.1 Self-Organizing Feature Map

Mathematically, the self-organizing map (SOM) determines a transformation from a high-dimensional input space onto a one- or two-dimensional discrete map. The transformation takes place as an adaptive learning process such that when it converges the lattice represents a topographic map of the input patterns. The training of the SOM is based on a random presentation of several input vectors, one at a time. Typically, each input vector produces the firing of one selected neighboring group of neurons whose weights are close to this input vector.

The most important components of such a network are:

1. A 1D or 2D lattice of neurons that encodes an input pattern of an arbitrary dimension into a specific location on it, as shown in Figure 7.7a.
2. A method that selects a "winning neuron" based on the similarity between the weight vector and the input vector.
3. An interactive network that activates the winner and its neighbors simultaneously. A neighborhood $\Lambda_{i(\mathbf{x})}(n)$ which is centered on the winning neuron is a function of the discrete time n. Figure 7.7b illustrates such a neighborhood, which first includes the whole array and then shrinks gradually to only one "winning neuron", represented by the black circle.
4. An adaptive learning process that reinforces all neurons in the close neighborhood of the winning neuron, and inhibits all those that are farther away from the winner.

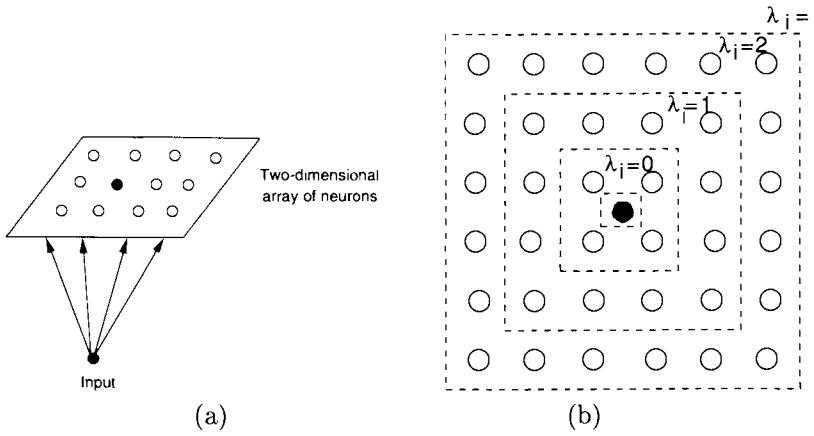


FIGURE 7.7 (a) Kohonen neural network; and (b) neighborhood Λ_i, of varying size, around "winning" neuron i, identified as a black circle.

The learning algorithm of the self-organized map is simple and is described below:

1. **Initialization:** Choose random values for the initial weight vectors $\mathbf{w}_j(0)$ to be different for $j = 1, 2, \ldots, N$, where N is the number of neurons in the lattice. The magnitude of the weights should be small.
2. **Sampling:** Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents the new pattern that is presented to the lattice.
3. **Similarity matching:** Find the "winner neuron" $i(\mathbf{x})$ at time n based on the minimum-distance Euclidean criterion:

$$i(\mathbf{x}) = \arg\min_j ||\mathbf{x}(n) - \mathbf{w}_j(n)||, \quad j = 1, 2, \ldots, N \tag{7.15}$$

4. **Adaptation:** Adjust the synaptic weight vectors of all neurons (winners or not), using the update equation

$$\mathbf{w}_j(n+1) = \begin{cases} \mathbf{w}_j(n) + \eta(n)[\mathbf{x}(n) - \mathbf{w}_j(n)], & j \in \Lambda_{i(\mathbf{x})}(n) \\ \mathbf{w}_j(n), & \text{else} \end{cases} \tag{7.16}$$

where $\eta(n)$ is the learning-rate parameter, and $\Lambda_{i(\mathbf{x})}(n)$ is the *neighborhood function* centered around the winning neuron $i(\mathbf{x})$; both $\eta(n)$ and $\Lambda_{i(\mathbf{x})}$ are a function of the discrete time n and thus continuously adapted for optimal learning.
5. **Continuation:** Go to step 2 until there are no noticeable changes in the feature map.

The presented learning algorithm has some interesting properties, which are described based on Figure 7.8.

Mathematically, the feature map represents a nonlinear transformation Φ from a continuous input space X to a spatially discrete output space A:

$$\Phi : X \to A \tag{7.17}$$

The map preserves the topological relationship that exists in the input space, if the input space is of lower or equal dimensionality compared to the output space. In all other cases, the map is said to be neighborhood preserving, in the sense that neighboring regions of the input space activate neurons that are adjacent on the lattice. Thus, in cases when the topological structure of the input space is not known apriori or is too complicated to be specified, Kohonen's algorithm necessarily fails in providing perfectly topology-preserving maps.

In the following, we will give some basic properties of self-organizing maps. The first property pertains to an approximation of the input space. The self-organizing feature map Φ, completely determined by the neural

lattice, learns the input data distribution by adjusting its synaptic weight vectors $\{\mathbf{w}_j | j = 1, 2, \ldots, N\}$ to provide a good approximation to the input space X.

The second property refers to the topological ordering achieved by the nonlinear feature map: There is a correspondence between the location of a neuron on the lattice and a certain domain or distinctive feature of the input space.

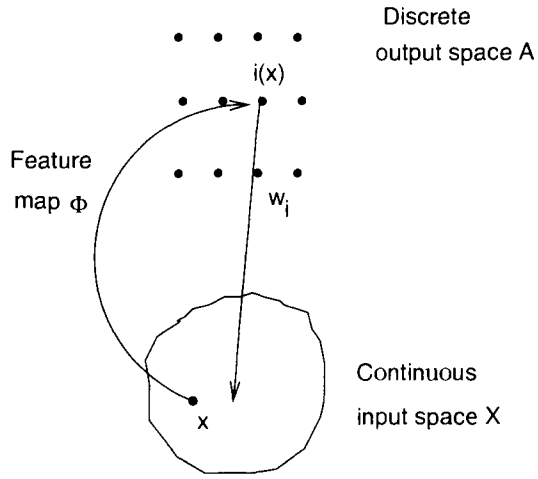


FIGURE 7.8 Mapping between input space X and output space A.

Kohonen maps have been applied to a variety of problems in image processing, including texture segmentation [179] and medical diagnosis [121].

7.3.1.1 Design Considerations A successful application of the Kohonen neural networks is highly dependent on the main parameters of the algorithm, namely, the learning-rate parameter η and the neighborhood function Λ_i. Since there is no theoretical foundation for the choice of these parameters, we have to rely on practical hints [114]: The learning-rate parameter $\eta(n)$ employed for adaptation of the synaptic vector $\mathbf{w}_j(n)$ should be time-varying. For the first 100 iterations $\eta(n)$ it should stay close to unity and decrease slowly thereafter, but remain above 0.1.

The neighborhood function Λ_i has to be wisely selected in order to ensure topological ordering of the weight vectors $\mathbf{w}_j$. Λ_i can have any geometrical form but should always include the winning neuron in the middle. In the beginning, the neighborhood function Λ_i includes all neurons in the network and then slowly shrinks with time. It takes in most cases about 1,000 iterations for the radius of the neighborhood function Λ_i to shrink linearly with time n to a small value of only a couple of neighboring neurons.

7.3.2 Learning Vector Quantization

Vector quantization (VQ) provides an efficient technique for data compression. Compression is achieved by transmitting the index of the codeword instead of the vector itself.

VQ can be defined as a mapping that assigns each vector $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})^T$ in the n–dimensional space R^n to a codeword from a finite subset of R^n. The subset $\mathbf{Y} = \{\mathbf{y}_i : i = 1, 2, \ldots, M\}$ representing the set of possible reconstruction vectors is called a codebook of size M. Its members are called the codewords. In the encoding process, a distance measure is evaluated to locate the closest codeword for each input vector $\mathbf{x}$. Then, the address corresponding to the codeword is assigned to $\mathbf{x}$ and transmitted. The distortion between the input vector and its corresponding codeword $\mathbf{y}$ is defined by the distance, $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$, where $||\mathbf{x}||$ represents the norm of $\mathbf{x}$.

A vector quantizer achieving a minimum encoding error is referred to as a Voronoi quantizer. Figure 7.9 shows an input data space partitioned into four different regions, called Voronoi cells, and the corresponding Voronoi vectors. These regions describe the collection of only those input vectors that are very close to the respective Voronoi vector.

Recent developments in neural network architectures have lead to a new VQ concept, the so-called learning vector quantization (LVQ). It represents an unsupervised learning algorithm associated with a competitive neural network consisting of one input and one output layer. The algorithm permits only the update of the winning prototype, that is, the closest prototype (Voronoi vector) of the LVQ network. If the class of the input vector and the Voronoi vector match, then the Voronoi vector is moved in the direction of the input vector $\mathbf{x}$. Otherwise the Voronoi vector $\mathbf{w}$ is moved away from this vector $\mathbf{x}$.

The LVQ algorithm is simple:

1. **Initialization:** Initialize the weight vectors $\{\mathbf{w}_j(0)|j = 1, 2, \ldots, N\}$ by setting them equal to the first N exemplar input feature vectors $\{\mathbf{x}_i|i = 1, 2, \ldots, L\}$.
2. **Sampling:** Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents the new pattern that is presented to the LVQ.
3. **Similarity Matching:** Find the best matching codeword (Voronoi vector) $\mathbf{w}_j$ at time n based on the minimum-distance Euclidean criterion:

$$\arg\min_j ||\mathbf{x}(n) - \mathbf{w}_j(n)||. \quad j = 1, 2, \ldots, N \tag{7.18}$$

4. **Adaptation:** Adjust only the best matching Voronoi vector, while the others remain unchanged. Assume that a Voronoi vector $\mathbf{w}_c$

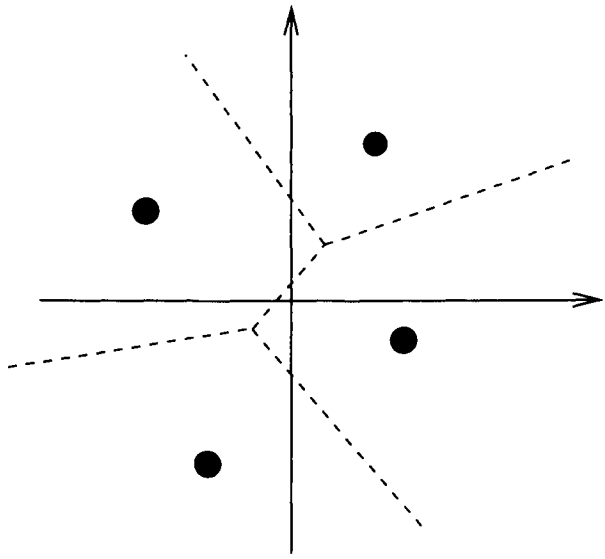


FIGURE 7.9 Voronoi diagram involving four cells. The small circles indicate the Voronoi vectors and are the different region (class) representatives.

is the closest to the input vector $\mathbf{x_i}$. By $C_{\mathbf{w_c}}$ we define the class associated with the Voronoi vector $\mathbf{w}_c$, and by $C_{\mathbf{x_i}}$ the class label associated to the input vector $\mathbf{x}_i$. The Voronoi vector $\mathbf{w}_c$ is adapted as follows:

$$\mathbf{w}_c(n+1) = \begin{cases} \mathbf{w}_c(n) + \alpha_n[\mathbf{x_i} - \mathbf{w}_c(n)], & C_{\mathbf{w_c}} = C_{\mathbf{x_i}} \\ \mathbf{w}_c(n) - \alpha_n[\mathbf{x_i} - \mathbf{w}_c(n)], & \text{otherwise} \end{cases} \tag{7.19}$$

where $0 < \alpha_n < 1$.

5. **Continuation:** Go to step 2 until there are no noticeable changes in the feature map.

The learning constant α_n is chosen as a function of the discrete time parameter n and decreases monotonically. The relative simplicity of the LVQ and its ability to work in unsupervised mode have made it a useful tool for image segmentation problems [121].

7.3.2.1 The "Neural-Gas" Algorithm The "neural-gas" algorithm [155] is an efficient approach which, applied to the task of vector quantization, (1) converges quickly to low distortion errors, (2) reaches a distortion error E lower than that from Kohonen's feature map, and (3) at the same time obeys a gradient descent on an energy surface.

Instead of using the distance $||\mathbf{x} - \mathbf{w_j}||$ or the arrangement of the $||\mathbf{w_j}||$ within an external lattice, it utilizes a neighborhood ranking of the reference vectors $\mathbf{w_i}$ for the given data vector $\mathbf{x}$. The adaptation of the reference vectors is given by

$$\Delta \mathbf{w_i} = \epsilon e^{-k_i(\mathbf{x}, \mathbf{w_i}/\lambda)}(\mathbf{x} - \mathbf{w_i}) \quad i = 1, \ldots, N \tag{7.20}$$

N is the number of units in the network. The step size $\epsilon \in [0, 1]$ describes the overall extent of the modification, and k_i is the number of the closest neighbors of the reference vector $\mathbf{w_i}$. λ is a characteristic decay constant.

In [155], it was shown that the average change of the reference vectors can be interpreted as an overdamped motion of particles in a potential that is given by the negative data point density. Added to the gradient of this potential is a "force" in the direction of the space where the particle density is low. This "force" is based on a repulsive coupling between the particles (reference vectors). In form it is similar to an entropic force and tends to uniformly distribute the particles (reference vectors) over the input space, as is the case with a diffusing gas. Hence the name "neural-gas" algorithm. It is also interesting to mention that the reference vectors are slowly adapted, and therefore pointers that are spatially close at an early stage of the adaptation procedure might not be spatially close later. Connections that have not been updated for a while die out and are removed.

Another important feature of the algorithm compared to the Kohonen algorithm is that it does not require a prespecified graph (network). In addition, it can produce topology-preserving maps, which is only possible if the topological structure of the graph matches the topological structure of the data manifold. In cases, however, where an appropriate graph cannot be determined from the beginning, such as where the topological structure of the data manifold is not known in advance or is too complex to be specified, Kohonen's algorithm always fails in providing perfectly topology-preserving maps.

To obtain perfectly topology-preserving maps we employ a powerful structure from computational geometry: the Delaunay triangulation, which is the dual of the already mentioned Voronoi diagram [199]. In a plane, the Delaunay triangulation is obtained if we connect all pairs $\mathbf{w_j}$ by an edge if their Voronoi polyhedra are adjacent. Figure 7.10 shows an example of a Delaunay triangulation. The Delaunay triangulation arises as a graph matching to the given pattern manifold.

The neural gas algorithm is simple:

1. **Initialization:** Randomly initialize the weight vectors $\{\mathbf{w}_j | j = 1, 2, \ldots, N\}$ and the training parameters $(\lambda_i, \lambda_f, \epsilon_i, \epsilon_f)$, where λ_i, ϵ_i are initial values of $\lambda(t), \epsilon(t)$ and λ_f, ϵ_f are the corresponding final values.

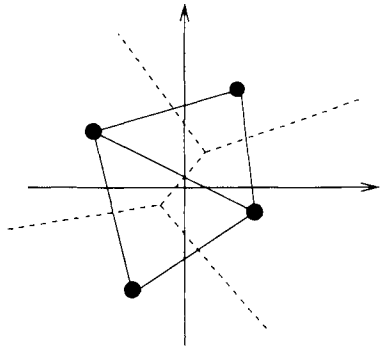


FIGURE 7.10 Delaunay triangulation.

2. **Sampling:** Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents the new pattern that is presented to the neural gas network.
3. **Distortion:** Determine the distortion set $D_{\mathbf{x}}$ between the input vector $\mathbf{x}$ and the weights $\mathbf{w}_j$ at time n based on the minimum distance Euclidean criterion:

$$D_{\mathbf{x}} = ||\mathbf{x}(n) - \mathbf{w}_j(n)||, \quad j = 1, 2, \ldots, N \tag{7.21}$$

Then order the distortion set in ascending order.

4. **Adaptation:** Adjust the weight vectors according to

$$\Delta \mathbf{w_i} = \epsilon e^{-k_i(\mathbf{x}, \mathbf{w_i}/\lambda)}(\mathbf{x} - \mathbf{w_i}) \quad i = 1, \ldots, N \tag{7.22}$$

where $i = 1, \ldots, N$. The parameters have the following time dependencies: $\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{\frac{t}{t_{max}}}$ and $\epsilon(t) = \epsilon_i(\epsilon_f/\epsilon_i)^{\frac{t}{t_{max}}}$. Increment the time parameter t by 1.

5. **Continuation:** Go to step 2 until the maximum iteration number t_{max} is reached.

7.4 RADIAL BASIS NEURAL NETWORKS (RBNN)

Radial basis neural networks describe a new concept of classification, a so-called hybrid learning mechanism. Their architecture is quite simple, based on a three-layer model having only one hidden layer. The hybrid learning mechanism combines a self-organized learning of the hidden layers' neurons with a supervised learning applied to the output layer's weight adaptation.

The design of a neural network can be viewed as a curve-fitting (nonlinear approximation) problem in a high-dimensional space. Thus, learning is equivalent to finding the surface in a multidimensional space that provides the best match to the training data. To be specific, let us consider a

system with n inputs and m outputs and let $\{x_1, \ldots, x_n\}$ be an input vector and $\{y_1, \ldots, y_m\}$ the corresponding output vector describing the system's answer to that specific input. After the training process is completed, the system has learned the input and output data distribution and is able to find for any input the correct output. In other words, we look for the "best" approximation function $\widehat{f}(x_1, \ldots, x_n)$ of the actual input-output mapping function [59, 194].

In the following, we will describe the mathematical background that is necessary to understand radial basis neural networks. We will review the concept of interpolation and show how the interpolation problem is implemented by a radial basis neural network. Since there are an infinite number of solutions for a given approximation problem, we want to choose one particular solution, and therefore we have to impose some restrictions on the solutions. This leads to a regularization approach to the approximation problem and can be easily implemented by the radial basis neural network.

7.4.1 Interpolation Networks

The interpolation problem can be very elegantly solved by a three layer feedforward neural network. The layers play entirely different roles in the learning process. The input layer is made up of source nodes that receive sensor information. The second layer, the only hidden layer, applies a nonlinear transformation from the input space to the hidden space. The nonlinear transformation is followed by a linear transformation from the hidden layer to the output layer. This leads to the theory of multivariable interpolation in the high-dimensional feature space.

The mathematical formulation of the interpolation problem is given below.

THE INTERPOLATION PROBLEM 7.4.1:
Assume that to N different points $\{\mathbf{m}_i \in \mathcal{R}^n | i = 1, \ldots N\}$ correspond N real numbers $\{d_i \in \mathcal{R} | i = 1, \ldots, N\}$. Then find a function $F : \mathcal{R}^n \rightarrow \mathcal{R}$ that satisfies the interpolation condition:

$$F(\mathbf{m}_i) = d_i \quad \textit{for } i = 1, \ldots, N \tag{7.23}$$

The radial-basis function technique consists of choosing a function F that has the following form [194]:

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h(||\mathbf{x} - \mathbf{m}_i||) + \sum_{i=1}^{M} d_i p_i(\mathbf{x}), \quad M \leq N \tag{7.24}$$

h is a smooth function, known as a radial-basis function; $||.||$ is the Euclidian norm in $\mathcal{R}^n$; $\{p_i | i = 1, \ldots, m\} : \mathcal{R}^n \rightarrow \mathcal{R}$ represents a basis of algebraic

polynomials in the linear space $\pi_{k-1}(\mathcal{R}^n)$ of maximum order $k-1$ for a given k.

Let the radial basis function $h(r)$ be continuous on $[0, \infty)$ and also its derivatives on $[0, \infty)$ strict monotone. Then we get a simplified form of the preceding equation:

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h(||\mathbf{x} - \mathbf{m}_i||) \tag{7.25}$$

It now becomes evident that we can implement the approximation problem with a three layer neural network as shown in Figure 7.11. The number of input neurons corresponds to the dimension of the feature space; the number of hidden neurons is problem-specific and is determined during the clustering process. The output neurons perform the superposition of the weighted basis functions and thus solve the approximation problem. Figure 7.11 depicts the architecture of the network for a single output. Clearly, such an architecture can be readily extended to accomodate any desired number of network outputs.

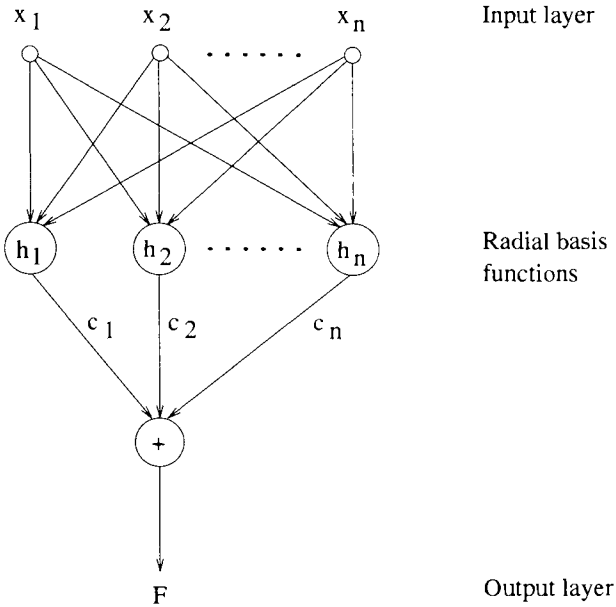


FIGURE 7.11 Approximation network.

The network architecture is solely determined by the input-output mapping to be achieved. Moreover, from the viewpoint of approximation theory, the network has three desirable properties:

1. The network is a universal approximator that approximates arbitrarily well any multivariate continuous function, given a sufficiently large number of hidden units. This can be shown based on the Stone-Weierstrass theorem [195].
2. The network has the best-approximation property. This means that given an unknown nonlinear function f, there always exists a choice of coefficients that approximate f better than other possible choices. This property holds for all other approximation schemes based on Green's functions as basis functions. It is not the case for a three-layer perceptron having sigmoid functions as activation functions. Sigmoid functions are not Green's functions, being additionally translation and rotation invariant.
3. By taking a closer look at the approximation approach,

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h(||\mathbf{x} - \mathbf{m}_i||) \tag{7.26}$$

we can obtain a simplified form if we specify for each of the N radial-basis functions a width σ_i,

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i g\left(\frac{||\mathbf{x} - \mathbf{m}_i||}{\sigma_i}\right) \tag{7.27}$$

In [187] it was shown that even then we can solve the approximation problem optimally, if all radial basis functions have the same width $\sigma_i = \sigma$. In neural network terminology, it means that neurons having the same Gaussian function as an activation function for the hidden nodes can approximate any given function.

7.4.2 Regularization Networks

The previous section dealt with the interpolation problem, and we showed how a function $F(\mathbf{x})$ can be determined to satisfy the interpolation condition. Assuming that the data are noise-corrupted, we immediately see that the problem is ill-posed since it has an infinite number of solutions. To be able to choose one particular solution, we need to include some a priori knowledge of the function $F(\mathbf{x})$. This can be some prior smoothness information about the function, in the sense that similar inputs yield similar outputs. In [197], smoothness is defined as a measure describing the "oscillatory" behavior of a function. Thus, a smooth function exhibits few oscillations. Translated into the frequency domain, this means that a smooth function has less energy in the high-frequency range and therefore has a smaller bandwidth.

The regularization theory arose as a new approach for solving ill-posed problems [230]. The main idea of this method is to stabilize the solution

by means of some auxiliary nonnegative functional that includes a priori information about the solution. Thus, the solution to this ill-posed problem can be obtained from a variational principle, which includes both the data and prior smoothness information. Smoothness is considered by introducing a smoothness functional $\phi[F]$. A small value of $\phi[F]$ corresponds to smoother functions.

Let $\{\mathbf{m}_i \in \mathcal{R}^n | i = 1, \ldots N\}$ be N input vectors and $\{d_i \in \mathcal{R} | i = 1, \ldots, N\}$ be N real numbers and represent the desired outputs. The solution to the regularization problem is denoted by $F(\mathbf{x})$, and is required to approximate the data well and at the same time to be smooth. Thus, $F[\mathbf{x}]$ should minimize the following functional:

$$H[F] = \sum_{i=1}^{N} [d_i - F(\mathbf{m}_i)]^2 + \lambda \phi[F] \tag{7.28}$$

where λ is a positive number, called the regularization term. The first term in the equation enforces closeness to the data and measures the distance between the data and the desired solution F. The second term enforces smoothness and measures the cost associated with the deviation from smoothness. The regularization parameter $\lambda > 0$ controls the trade-off between these two terms.

THE PRINCIPLE OF REGULARIZATION 7.4.1:

Determine the function $F(\mathbf{x})$ that minimizes the Tikhonov functional defined by

$$H[F] = \sum_{i=1}^{N} [d_i - F(\mathbf{m}_i)]^2 + \lambda \phi[F] \tag{7.29}$$

with λ being the regularization parameter.

The solution for the regularization problem is derived in [197] and is given here in its simplified form

$$F(\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^{N} [d_i - F(\mathbf{m}_i)] G(\mathbf{x}, \mathbf{m}_i) \tag{7.30}$$

The minimizing solution $F(\mathbf{x})$ to the regularization problem is thus given by a linear superposition of N Green's functions. The $\mathbf{m}_i$s represent the centers of the expansion and the weights $w_i = [d_i - F(\mathbf{m}_i)]\lambda$ represent the coefficients of the expansion.

Equation (7.30) can be similarly expressed as

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\mathbf{x}_j, \mathbf{m}_i) \tag{7.31}$$

The Green's function is a symmetric function with $G(\mathbf{x}, \mathbf{m}_i) = G(\mathbf{m}_i, \mathbf{x})$. If the smoothness functional $\phi[F]$ is both translationally and rotationally invariant, then the Green's function $G(\mathbf{x}, \mathbf{m}_i)$ will depend only on $||\mathbf{x} - \mathbf{m}_i||$. Under these conditions, the Green's function has the same invariance properties, and eq. (7.31) takes a special form:

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(||\mathbf{x} - \mathbf{m}_i||) \tag{7.32}$$

This solution is a strict interpolation, because all training data points are used to generate the interpolating function. The approximation scheme of eq. (7.31) can be easily interpreted as a network with only one hidden layer of units, which is referred to as a regularization network as shown in Figure 7.11.

Let's take a closer look at the regularization parameter. It can be interpreted as an indicator of the sufficiency of the available data to specify the solution $F(\mathbf{x})$. If $\lambda \to 0$, the solution $F(\mathbf{x})$ is sufficiently determined from the available examples; if $\lambda = 0$, then this corresponds to pure interpolation. If $\lambda \to \infty$, this means the examples are unreliable. In practice, λ is between these two limits in order to enable both the sample data and the prior information to contribute to the solution $F(\mathbf{x})$.

The regularization problem represents an alleviation to a well-known and frequent problem in training neural networks. the so-called overtraining. If noise is added to input data, the neural network learns all the details of the input-output pairs but not their general form. This phenomenon is overtraining. A smart choice of the regularization parameter can prevent this.

EXAMPLE 7.4.1: Figure 7.12 shows the effect of overtraining based on the example of learning an input-output mapping.

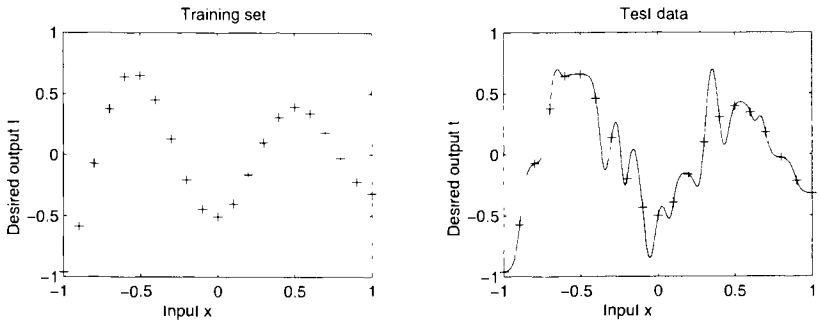


FIGURE 7.12 Overtraining. (a) Learning data and (b) testing data.

In practical applications, we have two choices for the Tikhonov functional in order to avoid overtraining:

- Zero-order regularization pertains to the weights of the network. The minimizer of the Tikhonov functional has a penalty term for the weights and in that way avoids overtraining:

$$\mathcal{E}(F) = \frac{1}{2}\sum_{p}\sum_{i}(d_i - F(\mathbf{m}_i))^2 + \lambda\sum_{i}\sum_{j} w_{ij}^2 \tag{7.33}$$

- Accelerated regularization pertains to the neurons of the hidden layer. This is a fast regularization network, since it tries to determine the centers and the widths of the radial basis functions.

7.4.3 Data Processing in Radial Basis Function Networks

Data processing in a radial basis function neural network is quite different from standard supervised or unsupervised learning techniques. The MLP's backpropagation algorithm represented an optimization method known in statistics as stochastic approximation, while the "winner takes all" concept is neurobiologically inspired strategy used for training self-organizing feature maps or in LVQ.

In this section we present a different method by looking into the design of a neural network as an approximation problem in a high-dimensional space. In neural network parlance, the hidden units are kernel functions that constitute an arbitrary "basis" for the input patterns (vectors) when they are mapped onto the hidden-unit space; these functions are known as radial basis functions. However, there is a neurobiological equivalent to the mathematical concept of kernel or radial basis functions: the receptive field. It describes the local interaction between neurons in a cortical layer.

Fundamental contributions to the theory, design, and application of radial-basis function networks are presented in an examplary manner in [166] and [196].

The construction of a radial-basis function (RBF) network in its most basic form involves three different layers. For a network with N hidden neurons, the output of the ith output node $f_i(\mathbf{x})$ when the n-dimensional input vector $\mathbf{x}$ is presented is given by

$$f_i(\mathbf{x}) = \sum_{j=1}^{N} w_{ij}\Psi_j(\mathbf{x}) \tag{7.34}$$

where $\Psi_j(\mathbf{x}) = \Psi(||\mathbf{x} - \mathbf{m}_j||/\sigma_j)$ is a suitable radially symmetric function that defines the output of the jth hidden node. Often $\Psi(.)$ is chosen to be the Gaussian function where the width parameter σ_j is the standard deviation. $\mathbf{m}_j$ is the location of the jth centroid, where each centroid is represented by a kernel/hidden node, and w_{ij} is the weight connecting the

jth kernel/hidden node to the ith output node. Figure 7.13a illustrates the configuration of the network.

The steps that govern the data flow through the radial basis function network during classification and represent a typical RBF network learning algorithm are described below:

1. **Initialization:** Choose random values for the initial weights of the RBF network. The magnitude of the weights should be small. Choose the centers $\mathbf{m}_i$ and the shape matrices $\mathbf{K}_i$ of the N given radial basis functions.
2. **Sampling:** Draw randomly a pattern $\mathbf{x}$ from the input data. This pattern represents the input to the neural network.
3. **Forward computation of hidden layer's activations:** Compute the values of the hidden-layer nodes as illustrated in Figure 7.13b:

$$\psi_i = \exp\left(-d(\mathbf{x}, \mathbf{m}_i, \mathbf{K}_i)/2\right) \tag{7.35}$$

$d(\mathbf{x}, \mathbf{m}_i) = (\mathbf{x} - \mathbf{m}_i)^T \mathbf{K}_i (\mathbf{x} - \mathbf{m}_i)$ is a metric norm and is known as the Mahalanobis distance. The *shape matrix* $\mathbf{K}_i$ is positive definite and its elements k_{jk}^i;

$$\mathbf{K}_i jk = \frac{h_{jk}}{\sigma_j * \sigma_k} \tag{7.36}$$

represent the correlation coefficients h_{jk} and σ_j the standard deviation of the ith shape matrix.

We choose for h_{jk}: $h_{jk} = 1$ for $j = k$ and $|h_{jk}| \leq 1$ otherwise.

4. **Forward computation of output layer's activations:** Calculate the values of the output nodes according to

$$f_{oj} = \phi_j = \sum_i w_{ji} \psi_i \tag{7.37}$$

5. **Updating:** Adjust weights of all neurons in the output layer based on a steepest descent rule.
6. **Continuation:** Continue with step 2 until no noticeable changes in the error function are observed.

The algorithm is based on the a priori knowledge of the radial basis centers and its shape matrices. In the following section, we will review some design strategies for RBF networks and show how we can determine the centers and widths of the radial basis functions.

RBF networks have been applied to a variety of problems in image processing, such as image coding and analysis [209], and also in medical diagnosis [255].

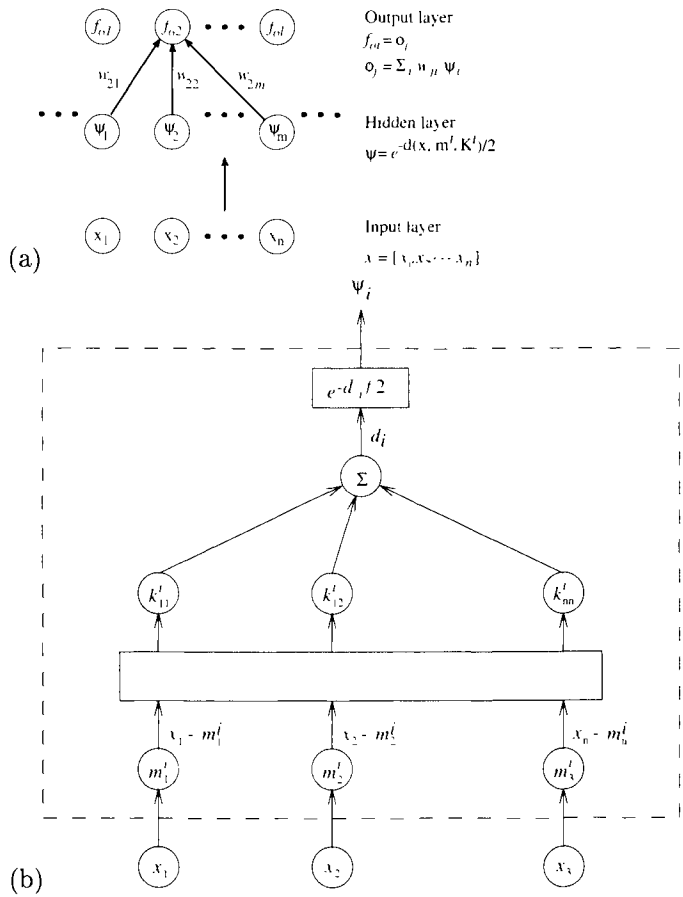


FIGURE 7.13 RBF network: (a) three-layer model; and (b) the connection between the input layer and the hidden-layer neuron.

7.4.3.1 Design Considerations RBF networks represent, in contrast to the MLP, local approximators to nonlinear input-output mapping. Their main advantages are a short training phase and a reduced sensitivity to the order of presentation of training data. In many cases, however, we find that a smooth mapping is only achieved if the number of radial basis functions required to span the input space becomes very large. This hampers the feasibility of many practical applications.

The RBF network has only one hidden layer, and the number of basis functions and their shape is problem-oriented and can be determined online during the learning process [135, 193]. The number of neurons in the input layer equals the dimension of the feature vector. Likewise, the number of nodes in the output layer corresponds to the number of classes.

7.4.4 Training Techniques for RBF Networks

As stated in the beginning, the learning process of the RBF networks is considered hybrid. We distinguish two different dynamics: the adjustment of the linear weights of the output neurons and the nonlinear activation functions of the hidden neurons. The weight adjustment of the output layer's neurons represents a linear process compared to the nonlinear parameter adjustment of the hidden-layer neurons. Since the hidden and output layers of an RBF network perform different tasks, we must separate the optimization of the hidden and output layers, and thus employ different learning techniques. The output layer's synapses are updated according to a simple delta rule as shown in the MLP case.

We have some degrees of freedom in choosing the kernel functions of an RBF network: in the simple case they are fixed, or they can be adjusted during the training phase.

There are several techniques to design an RBF network, depending on how the centers of the radial-basis functions of the network are determined. The best known strategies chosen for practical applications are reviewed in the following.

7.4.4.1 Fixed Centers of the RBF Selected at Random The simplest case is to assume fixed kernel functions with a center location chosen at random from the training set.

Specifically, a radial basis function centered at $\mathbf{m}_i$ is defined as

$$G(||\mathbf{x} - \mathbf{m}_i||^2) = \exp\left(-\frac{M}{d_{max}^2}||\mathbf{x} - \mathbf{m}_i||^2\right), \quad i = 1, \ldots, M \tag{7.38}$$

M represents the number of centers, and d_{max}^2 is the maximum distance between the selected centers. In fact, the standard deviation of all neurons is fixed at

$$\sigma = \frac{d_{max}}{\sqrt{2M}} \tag{7.39}$$

This guarantees that they are neither too spiky nor too flat. The only parameters that have to be adjusted here are the linear weights in the output layer of the network. They can be determined based on the pseudoinverse method [26]. Specifically, we have

$$\mathbf{w} = \mathbf{G}^+\mathbf{y} \tag{7.40}$$

where $\mathbf{y}$ is the desired response vector in the training set.

$\mathbf{G}^+$ is the pseudoinverse of the matrix $\mathbf{G} = \{g_{ji}\}$, which is itself defined as $\mathbf{G} = \{g_{ji}\}$, where

$$g_{ji} = \exp\left(-\frac{M}{d_{max}^2}||\mathbf{x_j} - \mathbf{m}_i||^2\right), \quad j = 1, \ldots, N; \quad i = 1, \ldots, M \quad (7.41)$$

$\mathbf{x_j}$ is the jth input vector of the training sample.

7.4.4.2 Self-organized Selection of the Centers of the RBFs The locations of the centers of the kernel functions (hidden units) are adapted based on a self-organizing learning strategy, whereas the linear weights associated with the output neurons are adapted based on a simple supervised delta rule. It's important to mention that this procedure allocates hidden units only in preponderantly input data regions.

The self-organized learning process is based on a clustering algorithm that partitions the given data set into homogeneous subsets. The k-means clustering algorithm [59] sets the centers of the radial basis functions only in those regions where significant input data are present. The optimal number M of radial basis functions is determined empirically. Let $\{\mathbf{m}_i(n)\}_{i=1}^{M}$ be the centers of the radial basis functions at iteration n of the algorithm.

Then, the k-means algorithm applied to center selection in RBF neural networks proceeds as follows:

1. **Initialization:** Choose randomly the initial centers $\{\mathbf{m}_i(0)\}_{i=1}^{M}$ of the radial basis functions.
2. **Sampling:** Draw a sample $\mathbf{x}$ from the data space with a certain probability.
3. **Similarity matching:** Determine the best-matching (winning) neuron $k(\mathbf{x})$ at iteration n using the Euclidean distance:

$$k(\mathbf{x}) = \overset{argmin}{k} ||\mathbf{x}(n) - \mathbf{m}_k(n)||, \quad k = 1, 2, \ldots M \quad (7.42)$$

   $\mathbf{m}_k(n)$ is the center of the kth radial basis function at iteration n.
4. **Updating:** Adjust the centers of the radial basis functions, using the update equation

$$\mathbf{x}_k(n+1) = \begin{cases} \mathbf{x}_k(n) + \eta(n)[\mathbf{x}(n) - \mathbf{x}_k(n)], & k = k(\mathbf{x}) \\ \mathbf{x}_k(n), & \text{else} \end{cases} \quad (7.43)$$

   η is the learning rate that lies in the range $0 < \eta < 1$.
5. **Continuation:** Go back to step 2 and continue the procedure until no noticeable changes are observed in the radial basis centers.

After determining the centers of the radial basis functions, the next and final stage of the hybrid learning process is to estimate the weights of the output layer based, for example, on the least-mean-square (LMS) algorithm [186].

7.4.4.3 Supervised Selection of Centers All free parameters of the kernel functions including centers are adapted based on a supervised learning strategy. In such a case, we deal with an RBF network in its most generalized form. This can be realized by an error-correction learning, which requests an implementation based on a gradient-descent procedure.

In the first step, we define a cost function

$$E = \frac{1}{2}\sum_{j=1}^{P} e_j^2 \tag{7.44}$$

where P is the size of the training sample and e_j is the error defined by

$$e_j = d_j - \sum_{i=1}^{M} w_i G(||\mathbf{x_j} - \mathbf{m}_i||_{C_i}) \tag{7.45}$$

The requirement is to find the free parameters w_i, $\mathbf{m}_i$, and Σ_i^{-1} such that the error E is minimized.

The results of this minimization [84] are summarized in Table 7.1. From the table we can see that the update equations for w_i, $\mathbf{x_i}$ and Σ_i^{-1} have different learning rates, thus showing the different time-scales. Unlike the back-propagation algorithm, the gradient-descent procedure described in Table 7.1 for an RBF network does not employ error back-propagation.

TABLE 7.1 Adaptation formulas for the linear weights and the position and widths of centers for an RBF network [84].

1.	**Linear weights of the output layer**
	$\frac{\partial \mathcal{E}(n)}{\partial w_i(n)} = \sum_{j=1}^{N} e_j(n) G(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|)$ $w_i(n+1) = w_i(n) - \eta_1 \frac{\partial \mathcal{E}(n)}{\partial w_i(n)}, \quad i = 1, \ldots, M$
2.	**Position of the centers of the hidden layer**
	$\frac{\partial \mathcal{E}(n)}{\partial \mathbf{m}_i(n)} = 2w_i(n)\sum_{j=1}^{N} e_j(n) G'(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|)\mathbf{K^i}[\mathbf{x_j} - \mathbf{m}_i(n)]$ $\mathbf{m}_i(n+1) = \mathbf{m}_i(n) - \eta_2 \frac{\partial \mathcal{E}(n)}{\partial \mathbf{m}_i(n)}, \quad i = 1, \ldots, M$
3.	**Widths of the centers of the hidden layer**
	$\frac{\partial \mathcal{E}(n)}{\partial \mathbf{k^i}(n)} = -w_i(n)\sum_{j=1}^{N} e_j(n) G'(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|)\mathbf{Q}_{ji}(n)$ $\mathbf{Q}_{ji}(n) = [\mathbf{x_j} - \mathbf{m}_i(n)][\mathbf{x_j} - \mathbf{m}_i(n)]^T$ $\mathbf{K^i}(n+1) = \mathbf{K^i}(n) - \eta_3 \frac{\partial \mathcal{E}(n)}{\partial \mathbf{K^i}(n)}$

7.5 TRANSFORMATION RADIAL BASIS NEURAL NETWORKS (TRBNNS)

The selection of appropriate features is an important precursor to most statistical pattern recognition methods. A good feature selection mechanism helps to facilitate classification by eliminating noisy or nonrepresentative

features that can impede recognition. Even features that provide some useful information can reduce the accuracy of a classifier when the amount of training data is limited. This so-called "curse of dimensionality," along with the expense of measuring and including features, demonstrates the utility of obtaining a minimum-sized set of features that allow a classifier to discern pattern classes well. Well-known methods in literature applied to feature selection are the floating search methods [200] and genetic algorithms [219].

Radial basis neural networks are excellent candidates for feature selection. It is necessary to add an additional layer to the traditional architecture to obtain a representation of relevant features. The new paradigm is based on an explicit definition of the relevance of a feature and realizes a linear transformation of the feature space.

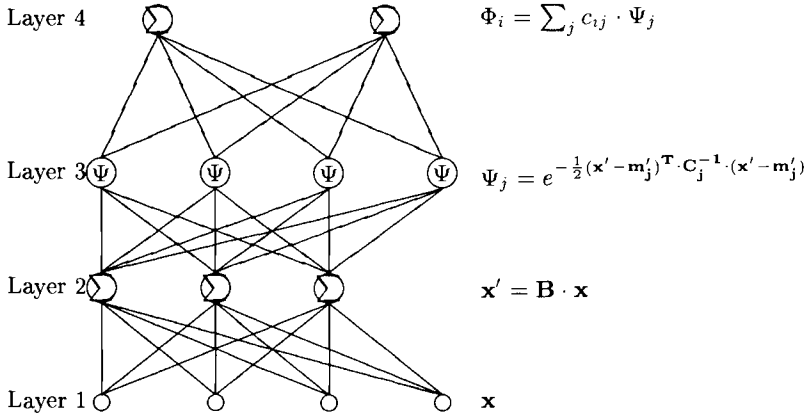


FIGURE 7.14 Linear transformation radial basis neural network.

Figure 7.14 shows the structure of a radial basis neural network with the additional layer 2, which transforms the feature space linearly by multiplying the input vector and the center of the nodes by the matrix $\mathbf{B}$. The covariance matrices of the input vector remain unmodified:

$$\mathbf{x}' = \mathbf{Bx}, \quad \mathbf{m}' = \mathbf{Bm}, \quad \mathbf{C}' = \mathbf{C} \tag{7.46}$$

The neurons in layer 3 evaluate a kernel function for the incoming input while the neurons in the output layer perform a weighted linear summation of the kernel functions:

$$\mathbf{y(x)} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{w_i} \exp\left(-\mathbf{d}(\mathbf{x}', \mathbf{m}'_\mathbf{i})/\mathbf{2}\right) \tag{7.47}$$

with

$$d(\mathbf{x}', \mathbf{m}_\mathbf{i}') = (\mathbf{x}' - \mathbf{m}_\mathbf{i}')^T \mathbf{C}_\mathbf{i}^{-1} (\mathbf{x}' - \mathbf{m}_\mathbf{i}') \tag{7.48}$$

Here, N is the number of neurons in the second hidden layer, $\mathbf{x}$ is the n-dimensional input pattern vector, $\mathbf{x}'$ is the transformed input pattern vector, $\mathbf{m}_\mathbf{i}'$ is the center of a node, w_i are the output weights, and $\mathbf{y}$ represents the m-dimensional output of the network. The $n \times n$ covariance matrix $\mathbf{C_i}$ is of the form

$$C_{jk}^i = \begin{cases} \frac{1}{\sigma_{jk}^2} & \text{if} \quad m = n \\ 0 & \text{otherwise} \end{cases} \tag{7.49}$$

where σ^{jk} is the standard deviation. Because the centers of the Gaussian potential function units (GPFUs) are defined in the feature space, they will be subject to transformation by $\mathbf{B}$, as well. Therefore, the exponent of a GPFU can be rewritten as

$$d(\mathbf{x}, \mathbf{m}_\mathbf{i}') = (\mathbf{x} - \mathbf{m_i})^\mathbf{T} \mathbf{B}^T \mathbf{C}_\mathbf{i}^{-1} \mathbf{B} (\mathbf{x} - \mathbf{m_i}) \tag{7.50}$$

and is in this form similar to eq. (7.48).

For the moment, we will regard $\mathbf{B}$ as the identity matrix. The network models the distribution of input vectors in the feature space by the weighted summation of Gaussian normal distributions, which are provided by the Gaussian Potential Function Units (GPFU) Ψ_j. To measure the difference between these distributions, we define the relevance ρ_n for each feature x_n:

$$\rho_n = \frac{1}{PJ} \sum_p \sum_j \frac{(x_{pn} - m_{jn})^2}{2\sigma_{jn}^2} \tag{7.51}$$

where P is the size of the training set and J is the number of the GPFUs. If ρ_n falls below the threshold ρ_{th}, one will decide to discard feature x_n. This criterion will not identify every irrelevant feature: If two features are correlated, one of them will be irrelevant, but this cannot be indicated by the criterion.

7.5.1 Learning Paradigm for the Transformation Radial Basis Neural Network

We follow the idea of [135] for the implementation of the neuron allocation and learning rules for the TRBNN. The network generation process starts initially without any neuron.

The mutual dependency of correlated features can often be approximated by a linear function, which means that a linear transformation of the input space can render features irrelevant.

First we assume that layers 3 and 4 have been trained so that they comprise a model of the pattern-generating process while $\mathbf{B}$ is the identity

matrix. Then the coefficients B_{nr} can be adapted by gradient descent with the relevance ρ_n' of the transformed feature x_n' as the target function. Modifying B_{nr} means changing the relevance of x_n by adding x_r to it with some weight B_{nr}. This can be done online, that is, for every training vector $\mathbf{x_p}$ without storing the whole training set. The diagonal elements B_{nn} are constrained to be constant 1, because a feature must not be rendered irrelevant by scaling itself. This in turn guarantees that no information will be lost. B_{nr} will only be adapted under the condition that $\rho_n < \rho_p$, so that the relevance of a feature can be decreased only by some more relevant feature. The coefficients are adapted by the learning rule

$$B_{nr}^{new} = B_{nr}^{old} - \mu \frac{\partial \rho_n}{\partial B_{nr}} \tag{7.52}$$

with the learning rate μ and the partial derivative

$$\frac{\partial \rho_n}{\partial B_{nr}} = \frac{1}{PJ} \sum_p \sum_j \frac{(x_{pn}' - m_{jn}')}{\sigma_{jn}^2} (x_{pr}' - m_{jr}') \tag{7.53}$$

In the learning procedure, which is based on, for example, [135], we minimize according to the LMS criterion the target function:

$$E = \frac{1}{2} \sum_{p=0}^{P} |y(\mathbf{x}) - \Phi(\mathbf{x})|^2 \tag{7.54}$$

where P is the size of the training set. The neural network has some useful features, such as automatic allocation of neurons, discarding of degenerated and inactive neurons, and variation of the learning rate depending on the number of allocated neurons.

The relevance of a feature is optimized by gradient descent:

$$\rho_i^{new} = \rho_i^{old} - \eta \frac{\partial E}{\partial \rho_i} \tag{7.55}$$

Based on the newly introduced relevance measure and the change in the architecture we get the following correction equations for the neural network:

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}} &= -(y_i - \Phi_i)\Psi_j \\ \frac{\partial E}{\partial m_{jn}} &= -\sum_i (y_i - \Phi_i) w_{ij} \Psi_j \sum_k (x_k' - m_{jk}') \frac{B_{kn}}{\sigma_{jk}^2} \\ \frac{\partial E}{\partial \sigma_{jn}} &= -\sum_i (y_i - \Phi_i) w_{ij} \Psi_j \frac{(x_n' - m_{jn}')^2}{\sigma_{jn}^3} \end{aligned} \tag{7.56}$$

In the transformed space the hyperellipses have the same orientation as in the original feature space. Hence they do not represent the same

distribution as before. To overcome this problem, layers 3 and 4 will be adapted at the same time as **B**. If these layers converge fast enough, they can be adapted to represent the transformed training data, providing a model on which the adaptation of **B** can be based. The adaptation with two different target functions (E and ρ) may become unstable if **B** is adapted too fast, because layers 3 and 4 must follow the transformation of the input space. Thus μ must be chosen $\ll \eta$. A large gradient has been observed, causing instability when a feature of extreme high-relevance is added to another. This effect can be avoided by dividing the learning rate by the relevance, that is, $\mu = \mu_0/\rho_r$.

7.6 HOPFIELD NEURAL NETWORKS

Hopfield neural networks represent a new neural computational paradigm by implementing an autoassociative memory. They are recurrent or fully interconnected neural networks. There are two versions of Hopfield neural networks: in the binary version all neurons are connected to each other but there is no connection from a neuron to itself, and in the continuous case all connections including self-connections are allowed.

A pattern, in N-node Hopfield neural network parlance, is an N-dimensional vector $\mathbf{p} = [p_1, p_2, \ldots, p_N]$ from the space $\mathbf{P} = \{-1, 1\}^N$. A special subset of **P** represents the set of stored or reference patterns $\mathbf{E} = \{\mathbf{e^k} : 1 \leq k \leq K\}$, where $\mathbf{e^k} = [e_1^k, e_2^k, \ldots, e_N^k]$. The Hopfield net associates a vector from **P** with a certain stored (reference) pattern in **E**. The neural net splits the binary space **P** into classes whose members bear in some way similarity to the reference pattern that represents the class. The Hopfield network finds a broad application area in image restoration and segmentation.

As already stated in the introduction, neural networks have four common components.

For the Hopfield net we have the following:

Neurons: The Hopfield network has a finite set of neurons $\mathbf{x}(i), 1 \leq i \leq N$, which serve as processing units. Each neuron has a value (or state) at time t described by $\mathbf{x}_t(i)$. A neuron in the Hopfield net has one of the two states, either -1 or $+1$; that is, $\mathbf{x}_t(i) \in \{-1, +1\}$.

Synaptic connections: The learned information of a neural net resides within the interconnections between its neurons. For each pair of neurons, $\mathbf{x}(i)$ and $\mathbf{x}(j)$, there is a connection w_{ij} called the synapse between $\mathbf{x}(i)$ and $\mathbf{x}(j)$. The design of the Hopfield net requires that $w_{ij} = w_{ji}$ and $w_{ii} = 0$. Figure 7.15a illustrates a three-node network.

Propagation rule: This defines how states and synapses influence the input of a neuron. The propagation rule $\tau_t(i)$ is defined by

$$\tau_t(i) = \sum_{j=1}^{N} \mathbf{x}_t(j) w_{ij} + b_i \tag{7.57}$$

b_i is the externally applied bias to the neuron.

Activation function: The activation function f determines the next state of the neuron $\mathbf{x}_{t+1}(i)$ based on the value $\tau_t(i)$ computed by the propagation rule and the current value $\mathbf{x}_t(i)$. Figure 7.15b illustrates this fact. The activation function for the Hopfield net is the hard limiter defined here:

$$\mathbf{x}_{t+1}(i) = f(\tau_t(i), \mathbf{x}_t(i)) = \begin{cases} 1, & \text{if } \tau_t(i) > 0 \\ -1, & \text{if } \tau_t(i) < 0 \end{cases} \tag{7.58}$$

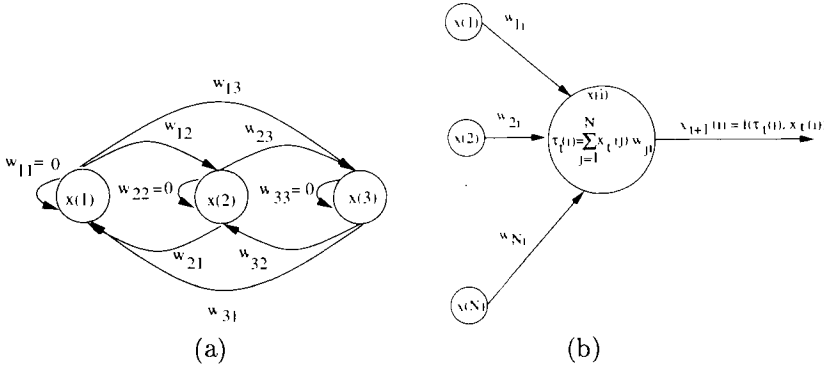


FIGURE 7.15 (a) Hopfield neural network; and (b) propagation rule and activation function for the Hopfield network.

The network learns patterns that are N-dimensional vectors from the space $\mathbf{P} = \{-1, 1\}^N$. Let $\mathbf{e^k} = [e_1^k, e_2^k, \ldots, e_n^k]$ define the kth exemplar pattern where $1 \leq k \leq K$. The dimensionality of the pattern space is reflected in the number of nodes in the net, such that the net will have N nodes $\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(N)$.

The training algorithm of the Hopfield neural network is simple and is outlined below:

1. **Learning:** Assign weights w_{ij} to the synaptic connections:

$$w_{ij} = \begin{cases} \sum_{k=1}^{K} e_i^k e_j^k, & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases} \tag{7.59}$$

Keep in mind that $w_{ij} = w_{ji}$, so it is necessary to perform the preceding computation only for $i < j$.

2. **Initialization:** Draw an unknown pattern. The pattern to be learned is now presented to the net. If $\mathbf{p} = [p_1, p_2, \ldots, p_N]$ is the unknown pattern, set

$$\mathbf{x}_0(i) = p_i, \quad 1 \leq i \leq N \tag{7.60}$$

3. **Adaptation:** Iterate until convergence. Using the propagation rule and the activation function we get for the next state,

$$\mathbf{x}_{t+1}(i) = f\left(\sum_{j=1}^{N} \mathbf{x}_t(j) w_{ij}, \mathbf{x}_t(i)\right) \tag{7.61}$$

This process should be continued until any further iteration will produce no state change at any node.

4. **Continuation:** For learning a new pattern. repeat steps 2 and 3.

In case of the continuous version of the Hopfield neural network, we have to consider neural self-connections $w_{ij} \neq 0$ and choose as an activation function a sigmoid function. With these new adjustments, the training algorithm operates in the same way.

The convergence property of Hopfield's network depends on the structure of $\mathbf{W}$ (the matrix with elements w_{ij}) and the updating mode. An important property of the Hopfield model is that if it operates in a sequential mode and $\mathbf{W}$ is symmetric with nonnegative diagonal elements, then the energy function

$$\begin{aligned} E_{hs}(t) &= \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i(t) x_j(t) - \sum_{i=1}^{n} b_i x_i(t) \\ &= -\tfrac{1}{2} \mathbf{x}^T(t) \mathbf{W} \mathbf{x}(t) - \mathbf{b}^T \mathbf{x}(t) \end{aligned} \tag{7.62}$$

is nonincreasing [87]. The network always converges to a fixed point.

Hopfield neural networks are applied to solve many optimization problems. In medical image processing, they are applied in the continuous mode to image restoration, and in the binary mode to image segmentation and boundary detection.

The continuous version will be extensively described in Chapter VIII as a subclass of additive activation dynamics.

7.7 COMPARING STATISTICAL, SYNTACTIC AND NEURAL PATTERN RECOGNITION METHODS

The delimitations between statistical, syntactic, and neural pattern recognition approaches are not necessarily clear. All these approaches share common features and have a correct classification result as a common goal.

The decision to choose a particular approach over another is based on analysis of underlying statistical components, or grammatical structure or on the suitability of neural network solution.

Table 7.2 and Figure 7.16 summarize the differences between the three pattern recognition approaches [216].

TABLE 7.2 Comparing statistical, syntactic, and neural pattern recognition approaches.

	Statistical	**Syntactic**	**Neural**
Pattern generation basis	Probabilistic models	Formal grammars	Stable state or weight matrix
Pattern classification basis	Estimation or decision theory	Parsing	Neural network properties
Feature organization	Input vector	Structural relations	Input vector
Training mechanism			
Supervised	Density estimation	Forming grammars	Determining neural network parameters
Unsupervised	Clustering	Clustering	Clustering
Limitations	Structural information	Learning structural rules	Semantic information

For statistical pattern recognition, the pattern structure is considered insignificant. If structural information from pattern is available, we choose the syntactic approach. The neural network approach, sometimes considered an alternative technique, provides useful methods for processing patterns derived from biological systems and can emulate the computational paradigm of the human brain.

A more sophisticated approach is the newly introduced concept of biologically-oriented neural networks [154]. They are robust and powerful classification techniques which disambiguate suspicious feature identification during a computer-aided diagnosis. They are derived from the newest biological discoveries aiming to imitate decision-making and sensory processing in biological systems. They have an advantage over other artificial intelligence methods since they directly incorporate brain-based mechanisms, and therefore make CAD systems more powerful.

There are also some drawbacks with the described methods: The statistical method cannot include structural information, the syntactic method does not operate based on adaptation rules, and the neural network approach doesn't contain semantic information in its architecture.

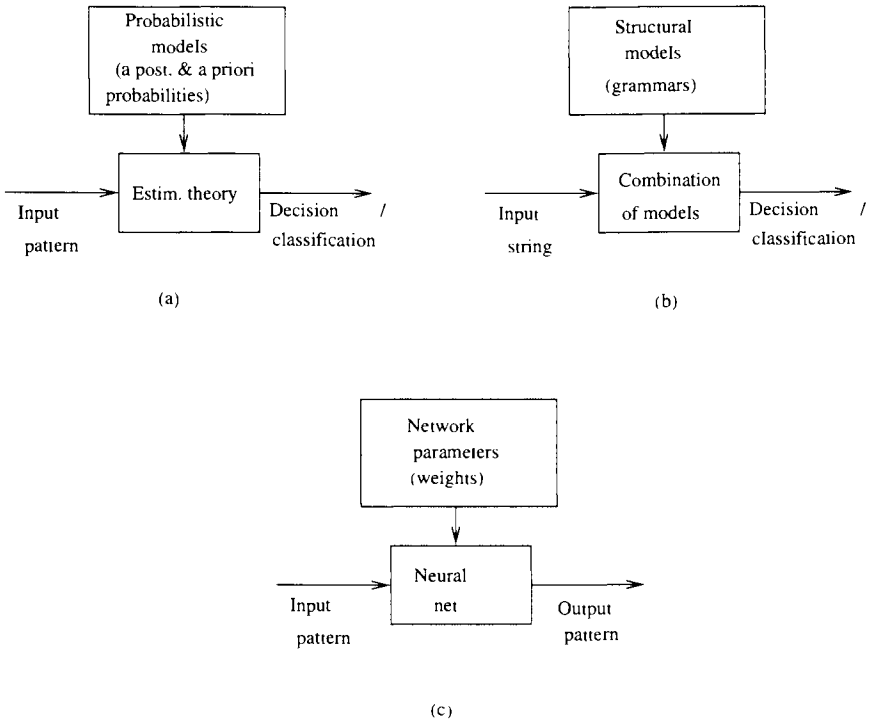


FIGURE 7.16 Pattern recognition approaches. (a) Statistical approach. (b) Syntactic approach. (c) Neural approach.

7.8 PIXEL LABELING USING NEURAL NETWORKS

Pixel labeling is a data-driven pixel classification approach leading to image segmentation. The goal is to assign a label to every pixel in the image. For a supervised classifier, the labels reflect the tissue types of the pixels. For an MRI image, appropriate labels will be gray matter, white matter, cerebrospinal fluid, bone, and fat. For an unsupervised classifier, the labels are arbitrarily assigned to pixels. A subsequent process will determine the correspondence between labels and tissue types.

7.8.1 Noncontextual Pixel Labeling

For this type of labeling, no pixel neighborhood (context information) is considered. Instead data obtained from several sources, so-called multispectral data, are used. For MRI, multispectral data are T_1, T_2, and proton density weighted images. All or only a subset of the multispectral data can be taken to generate a feature vector for each pixel location. For example, if T_1 and T_2 weighted images are used, then the vector $\mathbf{x}$ generated for a pixel at position (m, n) is

$$\mathbf{x}_{m,n} = (T_{1_{mn}}, T_{2_{mn}}) \tag{7.63}$$

Figure 7.17 shows an example of a feature vector for a single pixel.

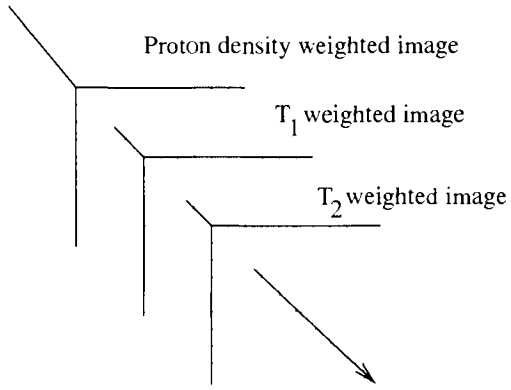


FIGURE 7.17 Multispectral data describing a feature vector for a pixel in an image.

The resulting feature vectors are classified by assigning a single label to each pixel position. This leads to cluster of points described by pixel feature vectors from a single tissue type. It has been shown [23] that even using multispectral data from MRI images, the characteristics of different tissue types overlap. This represents a problem in MRI segmentation.

7.8.2 Contextual Pixel Labeling

Noncontextual pixel labeling often leads to noisy results. An alternative technique involves the context information of pixels and uses the data in the spatial neighborhood of the pixel to be classified. Figure 7.18 visualizes this procedure. The feature vector may include both textural and contextual information. Usually, pixel intensity information across scales or measurements based on pixel intensity are used as features. In mammography, this pixel labeling is quite often used for tumor segmentation [52, 255]. A multiscale approach for MRI segmentation was proposed in [81]. Each pixel in an image was assigned a feature vector containing a limited number of differential geometric invariant features evaluated across arange of different scales at the pixel location. The features measured at different scales are products of various first- and second-order image derivatives.

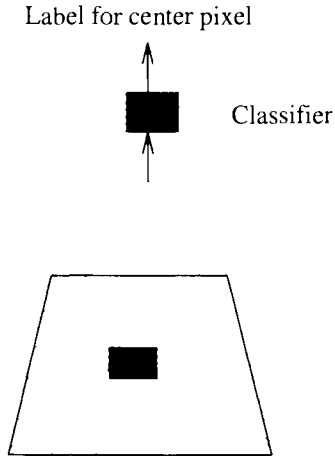


FIGURE 7.18 Contextual pixel labeling.

7.9 CLASSIFICATION STRATEGIES FOR MEDICAL IMAGES

There are two different strategies for classifying individual regions in medical images: region-based and pixel-based classification. For the region-based classification, the feature vector contains measurements of a set of features from a region in an image, while for the pixel-based classification, contextual or noncontextual information about every single pixel is obtained.

7.9.1 Pixel-Based Classification

Some features have been designed specifically for pixel-based classification in which a feature value is computed for each pixel. Each pixel in the image is then classified individually. Laws' texture energy measures are features which have been shown to discriminate well between various textured regions in an image when individual pixels are classified. These measures were successfully employed for spiculated lesion detection [244]. A pixel-level feature is considered a feature which is computed in the neighborhood of a pixel and is associated with that pixel. The neighborhood pixel will have a separate, and possibly different, feature value.

As stated before, classification at pixel level includes either contextual or noncontextual information of each image pixel. This technique is illustrated for MRI corresponding to acoustic neuromas [55]. A 7×7 pixel patch size, with a central square size of 5×5 pixels, was employed as the scheme

to encode contextual information. The classification result achieved by an MLP for each pixel was a real value in the continuous range [0,1]. The result values were used to construct results images whereby each pixel in the image being processed was replaced by its classification result value in the results image constructed. This is shown in Figure 7.19.

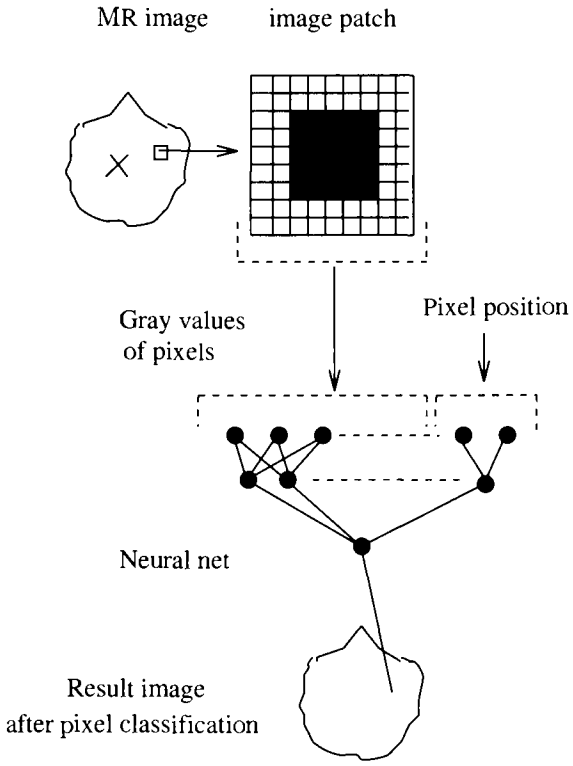


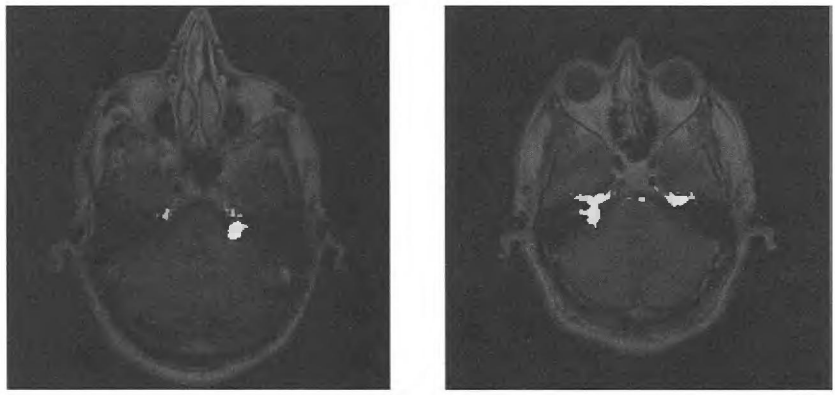
FIGURE 7.19 Pixel-based classification. An MLP is applied to the classification of all pixels in an image.

The classification results achieved based on this method are shown in Figure 7.20.

7.9.2 Region-Based Classification

In a region-based classification scheme, an object is first segmented from an image. The object is then classified based on features that are usually determined to describe the whole object.

Extracted features from regions are shape, texture, intensity, size, position within the image, contextual features, and fractal characteristics. In other words, an object-level feature is computed over an entire object.



(a) (b)

FIGURE 7.20 Pixel-level classification for detecting acoustic neuromas in MRI. (a) Typical classification result. (b) fp-tumor adjacent to the acoustic neuroma [55]. (Images courtesy of Prof. B. Thomas, University of Bristol.)

Subsequent to image segmentation, the individual detected regions have to be classified in terms of anatomical structures. A set of features from each region is extracted, and then the regions are classified based on the generated feature vectors. This procedure is explained in Figure 7.21.

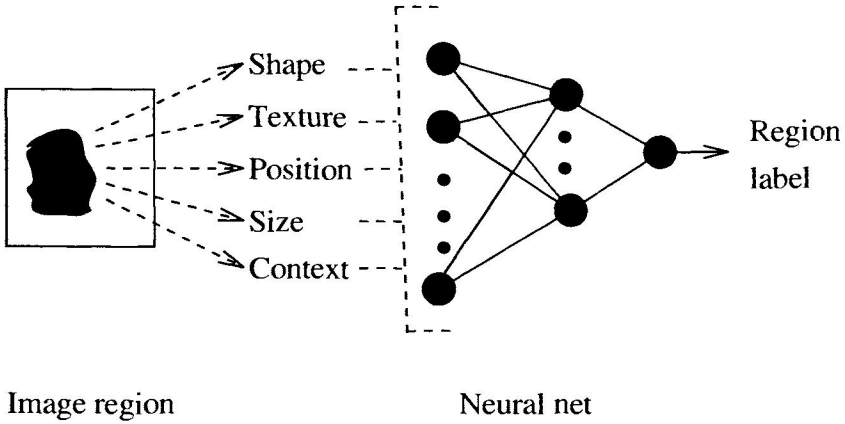
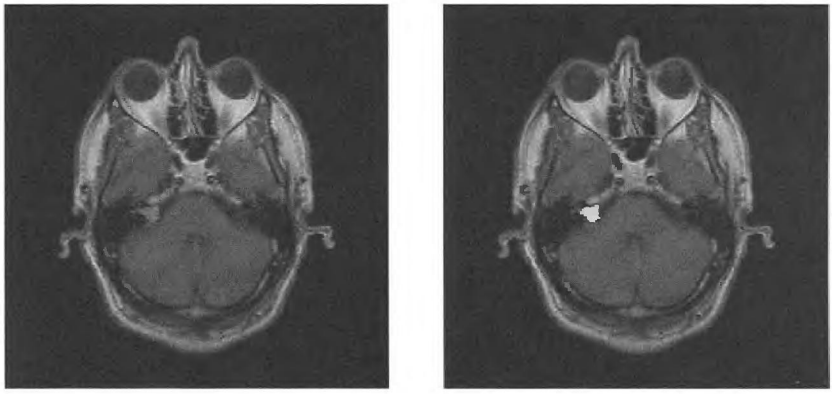


FIGURE 7.21 Region-based classification.

A common procedure is to first extract a larger feature set and then refine this, such that all classes of anatomical structures are correctly represented. An application of this method is visualized in Figure 7.22. It shows a classification error whose structure is similar to that of an acoustic neuroma.



(a) (b)

FIGURE 7.22 Region-level classification. A candidate tumor region cluster was erroneously classified as acoustic neuroma. (a) Original image and (b) classification result showing fp-tumor. (Images courtesy of Prof. B. Thomas, University of Bristol.)

7.10 CLASSIFIER EVALUATION TECHNIQUES

There are several techniques for evaluating the classification performance in medical imaging problems. The most known are the confusion matrix, ranking order curves, and ROC-curves.

7.10.1 Confusion Matrix

When conducting classification experiments, one possibility of evaluating the performance of a system is to determine the number of correctly and wrongly classified data. This gives a first impression of how correctly the classification was performed. In order to get a better understanding of the achieved classification, it is necessary to know which classes of data were most often misplaced. A convenient tool when analyzing results of classifier systems in general is the confusion matrix, which is a matrix containing information about the actual and predicted classes. The matrix is two-dimensional and has as many rows and columns as there are classes. The columns represent the true classifications and the rows represent the system classifications. If the system performs perfectly, there will be scores only in the diagonal positions. If the system has any misclassifications, these are placed in the off-diagonal cells. Table 7.3 shows a sample confusion matrix. Based on the confusion matrix, we can easily determine which classes are being confused with each other.

7.10.2 Ranking Order Curves

Ranking order curves represent a useful technique for estimating the appropriate feature dimension. They combine in a single plot several feature

TABLE 7.3 Confusion matrix for a classification of three-classes A, B, C.

	Output		
Input	A	B	C
A	90%	0%	10%
B	0%	99%	1%
C	0%	90%	10%

selection results for different classifiers where each of them visualized the function of error rate over the number of features.

Usually, we first see an improvement in classification performance when considering additional features. But after some time a saturation degree is reached, and the selection of new features leads to a deterioration of the classification performance because of overtraining. Based on ranking order curves, it is possible to determine the feature dependence. Some features alone do not necessarily deteriorate the classification performance. However, in combination with others they have a negative effect on the performance. The classifier type plays an important role as well. In other words, an optimal performance can be achieved only as a combination of an appropriate classifier architecture and the best selected feature subset.

Ranking order curves are an important tool when it comes to determining potentially useful features, monitoring overtraining, and excluding redundant and computationally expensive features. This requires a vast number of simulations such that the ranking order of the available features becomes evident. Based on the determined histograms, it becomes possible to detect the redundant features so that in subsequent simulations they can be excluded. Figure 7.23 visualizes three feature ranking order curves for different classifiers.

7.10.3 ROC Curves

Receiver operating characteristics (ROC) curves have their origin in signal detection theory. Since the outcome of a particular condition in a yes-no signal detection experiment can be represented as an ordered pair of values (the hit and false-alarm rates), it is useful to have a way to graphically present and interpret them. This is usually done by plotting hit rate against false-alarm rate. Such a plot is called a receiver operating characteristic or ROC.

In pattern recognition ROC curves play an important role in providing information about the overlap between two classes. In many cases, the mean values of distinct classes may differ significantly, yet their variances are large enough to impede a correct class distinction.

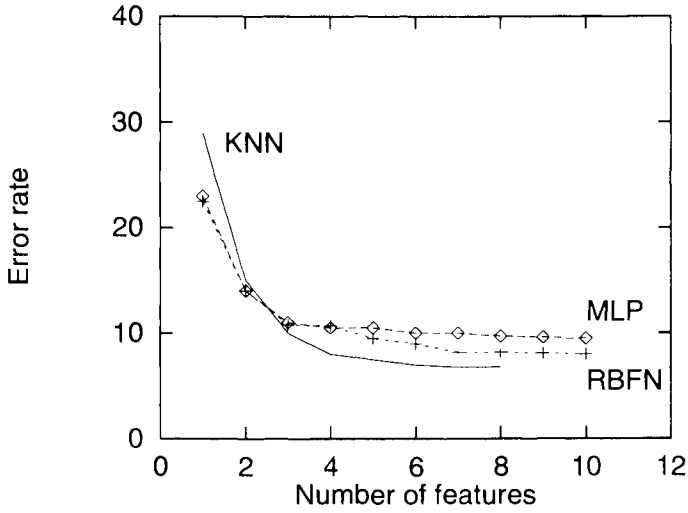
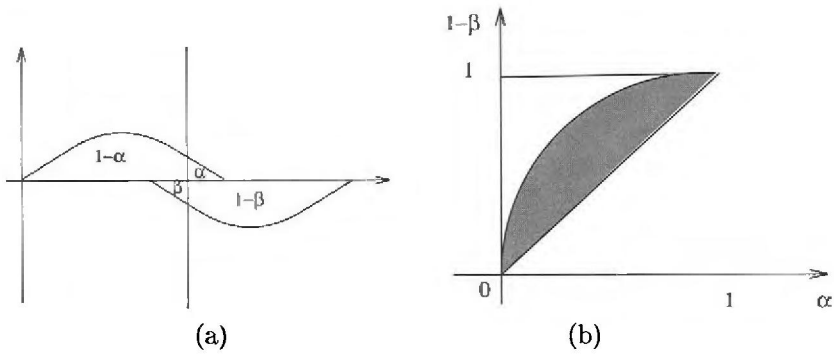


FIGURE 7.23 Example of ranking order curves showing feature selection results using three different classifiers (KNN, MLP, RBFN).

Figure 7.24a illustrates the overlapping of two pdfs describing the distribution of a feature in two classes. The threshold in the figure is given by a perpendicular line. For a better understanding, one pdf is inverted as suggested in [229]. We will assume that all values on the left of the threshold belong to class ω_1, while those on the right belong to class ω_2. However, this decision has an error probability of α while the probability of a correct decision is given by $1-\alpha$. Let β be the probability of a wrong decision concerning class ω_2 while $1-\beta$ is the probability of a correct decision. The threshold can be moved in both directions, and this will lead for every position to different values for α and β. In case of perfect overlap of the two curves, for every single threshold position $\alpha = 1-\beta$ holds. This corresponds to the bisecting line in Figure 7.24b where the two axes are α and $1-\beta$. If the overlap gets smaller, the corresponding curve in Figure 7.24b moves apart from the bisecting line. In the ideal case, there is no overlap and we obtain $1-\beta = 1$. In other words, the less the class overlap, the larger the area between the curve and the bisecting line.

We immediately see that the area varies between zero for complete overlap and 1/2 (area of the upper triangle) for complete separation. This means we have been provided with a measure of the class separability of a distinct feature.

In practical problems, the ROC curve can be easily determined by moving the threshold and computing the correct and false classification rate over the available training vectors.



(a) (b)

FIGURE 7.24 Example of (a) overlapping pdfs of the same feature in two classes and (b) the resulting ROC curve.

EXERCISES

1. Consider a biased input of the form

$$\tau_t(i) = \sum_k a_t(i) w_{ik} + b \tag{7.64}$$

and a sigmoidal activation function. What bias b is necessary for $f(0) = 0$? Is this acceptable? Are there any other possibilities?

2. A more general output activation is given by

$$o_j = f(\tau_j) = \frac{1}{1 + \exp - \{\frac{\tau_j - \theta_j}{\theta_0}\}} \tag{7.65}$$

a) Explain the contribution of the parameter θ_j.
b) Explain the role of θ_0.

3. For $f(\tau_j)$ given as in Exercise 7.2:
a) Determine and plot $f'(\tau_j)$ for $\tau_j = 0$ and $\theta_0 = 5$.
b) Repeat this for $\tau_j = 0$ and $\theta_0 = 50$ and $\theta_0 = 0.5$.

4. Show that if the output activation is given by

$$o_j = f(\tau_j) = \frac{1}{1 + \exp \tau_j} \tag{7.66}$$

then we obtain for its derivative

$$\frac{\partial f(\tau_j)}{\partial \tau_j} = f'(\tau_j) = o_j(1 - o_j) \tag{7.67}$$

Is it possible to have a τ_j such that we obtain $f(\tau_j) = 0$?

5. Design a feedforward network such that for given binary inputs A and B it implements
   a) The logical NAND function.
   b) The logical NOR function.
6. The necessary number of neurons in a hidden layer is always problem-oriented. Develop and discuss the advantages of removing or adding iteratively neurons from the hidden layer.
7. A method to increase the rate of learning yet to avoid the instability is to modify the weight updating rule

$$w_{ij}(n) = w_{ij}(n-1) + \eta \delta_{hj} p_i^t \tag{7.68}$$

   by including a momentum term as described in [48]

$$\Delta w_{ij}(n) = \alpha \Delta w_{ij}(n-1) + \eta \delta_{hj} p_i^t \tag{7.69}$$

   where α is a positive constant called the momentum constant. Describe why this is the case.
8. The momentum constant is in most cases a small number with $0 \leq \alpha < 1$. Discuss the effect of choosing a small negative constant with $-1 < \alpha \leq 0$ for the modified weight updating rule given in Exercise 7.7.
9. Set up two data sets, one set for training an MLP, and the other one for testing the MLP. Use a single-layer MLP and train it with the given data set. Use two possible nonlinearities: $f(x) = \sin x$ and $f(x) = e^{-x}$. Determine for each of the given nonlinearities:
   a) The computational accuracy of the network by using the test data.
   b) The effect on network performance of varying the size of the hidden layer.
10. Consider the updating rule for the Kohonen map given by eq. (7.16). Assume that the initial weight vectors $\mathbf{w}_j$ are not different. Comment on the following two distinct cases:
    a) They are nonrandom but distinct from each other.
    b) They are random but some weights can be the same.
11. In [114] it is stated that if the input patterns $\mathbf{x}$ have as a pdf a given $p(\mathbf{x})$, then the point density function of the resulting weight vectors $\mathbf{w}_j$ approximates $p(\mathbf{x})$. Describe based on eq. (7.16) why this is the case.
12. Comment on the differences and similarities between the Kohonen map and the "neural gas" network.
13. When is the Kohonen map preserving the topology of the input space? What condition must the neural lattice fulfill?

14. Consider a Kohonen map performing a mapping from a 2D input onto a 1D neural lattice of 60 neurons. The input data are random points uniformly distributed inside a circle of radius 1 centered at the origin. Compute the map produced by the neural network after 0, 25, 100, 1,000 and 10,000 iterations.

15. Consider a Kohonen map performing a mapping from a 3D input onto a 2D neural lattice of 1,000 neurons. The input data are random points uniformly distributed inside a cube defined by $\{(0 < x_1 < 1), (0 < x_2 < 1), (0 < x_3 < 1)\}$. Compute the map produced by the neural network after 0, 100, 1,000, and 10,000 iterations.

16. Radial basis neural networks are often referred to as "hybrid" neural networks. Comment on this property by taking into account the architectural and algorithmic similarities among radial basis neural networks, Kohonen maps, and MLPs.

17. Find a solution for the XOR problem using an RBF network with two hidden units where the two radial basis function centers are given by $\mathbf{m}_1 = [1, 1]^T$ and $\mathbf{m}_2 = [0, 0]^T$. Determine the output weight matrix $\mathbf{W}$.

18. Find a solution for the XOR problem using an RBF network with four hidden units where four radial basis function centers are given by $\mathbf{m}_1 = [1, 1]^T$, $\mathbf{m}_2 = [1, 0]^T$, $\mathbf{m}_3 = [0, 1]^T$, and $\mathbf{m}_4 = [0, 0]^T$. Determine the output weight matrix $\mathbf{W}$.

VIII

TRANSFORMATION AND SIGNAL-SEPARATION NEURAL NETWORKS

8.1 INTRODUCTION

Neural networks are excellent candidates for feature extraction and selection, and also for signal separation. The underlying architectures are mostly employing unsupervised learning algorithms and are viewed as nonlinear dynamical systems.

The material in this chapter is organized in three parts: the first part describes the neurodynamical aspects of neural networks, the second part deals with principal component analysis (PCA) and with related neural networks, and part 3 deals with independent component analysis (ICA) and neural architectures performing signal separation.

Several neural network models such as the generalized Hebbian algorithm, adaptive principal component extraction, and the linear and nonlinear Oja's algorithms are reviewed. As an application for principal component analysis, the emerging area of medical image coding is chosen.

ICA is gaining importance in artifact separation in medical imaging. This chapter reviews the most important algorithms for ICA, such as the Infomax, FastICA, and topographic ICA. Imaging brain dynamics is becoming the key to the understanding of cognitive processes associated with

the human brain. This chapter describes artifact separation based on ICA for two modalities of imaging brain dynamics, magnetoencephalographic (MEG) recordings and the functional magnetic resonance imaging (fMRI).

8.2 NEURODYNAMICAL ASPECTS OF NEURAL NETWORKS

In neural networks we deal with fields of neurons. Neurons within a field are topologically ordered, mostly based on proximity. The fields are related only by synaptic connections between them [51, 114, 242]. In mammalian brains, we find the topological ordering in volume proximity packing. An input field of neurons is defined with $\mathbf{F_X}$. We consider here only two-field neural networks and define with $\mathbf{F_Y}$ the output field. Let us assume that field $\mathbf{F_X}$ has n neurons and field $\mathbf{F_Y}$ has p neurons. There is a mapping defined from the input to the output field and described as $\mathbf{F_X} \rightarrow \mathbf{F_Y}$. Based on synaptic connections, we assume that there are m pairs of connected neurons $(\mathbf{x_i}, \mathbf{y_i})$. In this way, the function $f : \mathbf{R^n} \rightarrow \mathbf{R^P}$ generates the following associated pairs: $(\mathbf{x_1}, \mathbf{y_1})$,..., $(\mathbf{x_m}, \mathbf{y_m})$. The overall system behaves as an adaptive filter enabling a data flow from the input to the output layer and vice versa. The feature data changes the network parameters. Let m_{ij} describe the feedforward connection between the ith neuron from field $\mathbf{F_X}$ and the jth neuron from field $\mathbf{F_Y}$. m_{ij} can be positive (excitatory), negative (inhibitory), or zero. The synaptic connections between the two fields can described by a $n \times p$ synaptic matrix $\mathbf{M}$. A similar matrix denoted by $\mathbf{N}$ and having $p \times n$ elements describes the feedback connections between the two layers. Figure 8.1 shows the structure of an interconnected two-layer field.

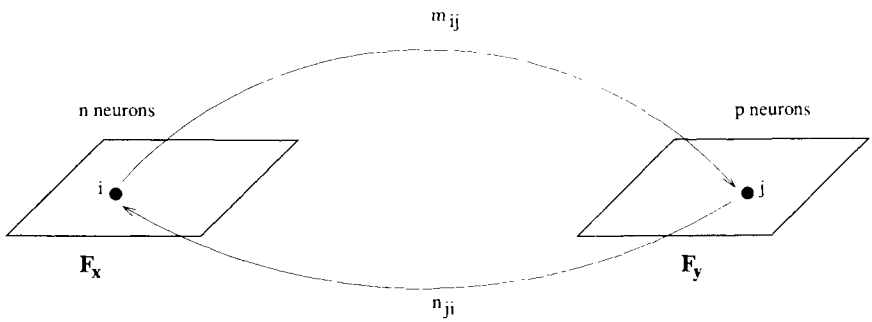


FIGURE 8.1 Neuronal structure between two neural fields.

In field terminology, a neural network can be very conveniently described by the quadruple $(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{N})$. $\mathbf{F_X}$ and $\mathbf{F_Y}$ represent not only the collection of topological neurons, but also their activation and signal computational characteristics. A two-layer neural network is called heteroassociative, while one-layer neural networks are called autoassociative

[114]. In general $\mathbf{M}$ and $\mathbf{N}$ are of different structures. In biological networks, $\mathbf{M}$ outnumbers $\mathbf{N}$, making such networks more feedforward networks. Artificial neural networks adopted the same concept, as can be seen from backpropagation-type neural networks and radial-basis neural networks. Besides the bidirectional topologies, there also are unidirectional topologies where a neuron field synaptically intraconnects to itself as shown in Figure 8.2.

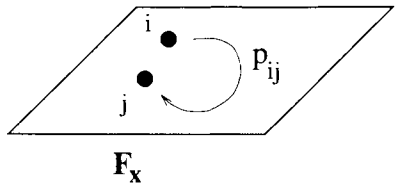


FIGURE 8.2 Unidirectional neural network.

The matrices $\mathbf{P}$ and $\mathbf{Q}$ intraconnect $\mathbf{F_X}$ and $\mathbf{F_Y}$. $\mathbf{P}$ is an $n \times n$ matrix and $\mathbf{Q}$ is a $p \times p$ matrix. In biological networks, $\mathbf{P}$ and $\mathbf{Q}$ are often symmetric and this symmetry reflects a lateral inhibition or competitive connection topology. $\mathbf{P}$ and $\mathbf{Q}$ are in most cases diagonal matrices with positive diagonal elements and negative or zero-off nondiagonal elements. An intra- and interconnected structure of neural fields is described mathematically as $(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{N}, \mathbf{P}, \mathbf{Q})$ and shown in Figure 8.3.

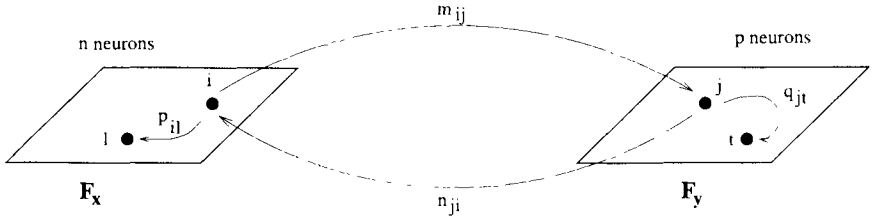


FIGURE 8.3 Intra- and interconnected neural fields.

The neural activity and the synaptic connections change over time, and this implies the existence of a neuronal dynamical system. Such a system is described by a set of first-order differential equations:

$$\dot{x}_i = g_i(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.1}$$

$$\dot{m}_{ij} = l_i(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.2}$$

$$\dot{y}_i = h_i(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.3}$$

or, in vector notation,

$$\dot{\mathbf{x}} = \mathbf{g}(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.4}$$

$$\dot{\mathbf{M}} = \mathbf{l}(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.5}$$

$$\dot{\mathbf{y}} = \mathbf{h}(\mathbf{F_X}, \mathbf{F_Y}, \mathbf{M}, \mathbf{P}, \mathbf{Q}) \tag{8.6}$$

It is assumed that $\mathbf{N} = \mathbf{0}$ and that the intraconnection matrices $\mathbf{P}$ and $\mathbf{Q}$ are not time-dependent.

Time plays a critical role in neuronal dynamics: time is "fast" at the neural level and "slow" at the synaptic level. In mammalian brains, membrane fluctuations occur at the millisecond level, while synaptic fluctuations occur at the second or minute level. Therefore, synapses encode long-term memory (LTM) pattern information, while membrane fluctuations encode short-term memory information (STM).

The state of the neuronal dynamical system at time t with activation and synaptic time functions described by eqs. (8.4), (8.5), and (8.6) is defined as

$$\mathbf{x}(t) = (x_1(t), ..., x_n(t)) \tag{8.7}$$

$$\mathbf{m}(t) = (m_{11}(t), ..., m_{np}(t)) \tag{8.8}$$

$$\mathbf{y}(t) = (y_1(t), ..., y_p(t)) \tag{8.9}$$

The state space of field $\mathbf{F_X}$ is the extended real vector space $\mathbf{R^n}$, that of $\mathbf{F_Y}$ is $\mathbf{R^p}$, and of the two-layer neural network is $\mathbf{R^n} \times \mathbf{R^p}$. A point in the state space specifies a snapshot of all neural behavior. A trajectory defines the time evolution of the network activity.

Based on the fact that there is either the LTM or STM or even both that change over time, we consider different types of dynamical systems: neuronal dynamics (only the activation fluctuates over time) defined by eq. (8.8), synaptic dynamics (only synaptic changes are considered) defined by eq. (8.7), and neuro-synaptic dynamics (both activation and synapses change) defined by eqs. (8.7) and (8.8).

While most dynamical analysis has been focused on either neuronal dynamics [41, 44, 88] or synaptic dynamics [85, 119, 198], little is known about the dynamical properties of neuro-synaptic dynamics [7, 162, 163]. The dynamics of coupled systems with different time-scales as found in neuro-synaptic dynamical systems is one of the most challenging research topics in the dynamics of neural systems.

To analyze the dynamical properties of the neural system, we adopt the same definitions as in [120]. Equilibrium is steady state (for fixed-point attractors). Convergence means synaptic equilibrium:

$$\dot{\mathbf{M}} = \mathbf{0} \tag{8.10}$$

Stability means neuronal equilibrium:

$$\dot{\mathbf{x}} = \mathbf{0} \tag{8.11}$$

And total stability is joint neuronal-synaptic steady state:

$$\dot{\mathbf{x}} = \dot{\mathbf{M}} = \mathbf{0} \tag{8.12}$$

In biological systems both neurons and synapses change as the feedback system samples fresh environmental stimuli. The neuronal and synaptic dynamical systems ceaselessly approach equilibrium and may never achieve it.

Neurons fluctuate faster than synapses fluctuate. In feedback systems this dynamical asymmetry creates the famous **stability convergence dilemma**.

8.2.1 Neuronal Dynamics

STM neural systems are systems that have a fluctuating neural activity over time but a time-constant synaptic fluctuation. Here, we consider a symmetric autoassociative neural network with $\mathbf{F_X} = \mathbf{F_Y}$ and a time-constant $\mathbf{M} = \mathbf{M^T}$. This network can be described based on the Cohen-Grossberg [44] activity dynamics:

$$\dot{x}_i = -a_i(x_i)[b_i(x_i) - \sum_{j=1}^{n} f_j(x_j)m_{ij}] \tag{8.13}$$

where $a_j > 0$ describes an amplification function. b_i are essentially arbitrary, and the matrix m_{ij} is symmetric. The function f is nonlinear and increasing. We can choose a sigmoid function for f, $f_j(x_j) = \tanh x_j$.

From eq. (8.13) we can derive two subsystems, an additive and a multiplicative system. These systems are explained in the following.

8.2.1.1 Additive Activation Dynamics The additive associative neural network is derived from eq. (8.13) by assuming $a_i(x_i)$ is a constant a_i and $b_i(x_i)$ is proportional to x_i,

$$\dot{x}_i = -a_i x_i + \sum_{j=1}^{n} f_j(x_j)m_{ij} + I_i \tag{8.14}$$

with

$$a_i(x_i) = 1 \tag{8.15}$$

$$b_i(x_i) = a_i x_i - I_i \tag{8.16}$$

I_i is an input term. Neurobiologically a_i measures the inverse cell membrane's resistance, and I_i the current flowing through the resistive membrane. Since the synaptic changes for the additive model are assumed nonexistent, the only way to achieve an excitatory and inhibitory effect is through the weighted contribution of the neighboring neurons outputs. The most famous representative of this group are the Hopfield neural network [87] and the cellular neural network [41].

8.2.1.2 Multiplicative Activation Dynamics The multiplicative or shunting neural network is derived from eq. (8.13) by assuming $a_i(x_i)$ is linear a_i and $b_i(x_i)$ is nonlinear:

$$\dot{x}_i = -a_i x_i - x_i(\sum_{j=1}^{n} f_j(x_j)m_{ij}) + I_i \tag{8.17}$$

The product term $x_i(\sum_{j=1}^{n} f_j(x_j)m_{ij})$ gave the network its name [167].

8.2.2 Synaptic Dynamics

A neural network learns a pattern if the system encodes the pattern in its structure. Also, the network connections change as it learns the information.

For example, the neural network has learned the stimulus-response pair (x_i, y_i) if it responds with y_i when x_i is the stimulus (input). The stimulus-response pair (x_i, y_i) is a sample from the mapping function $f : \mathbf{R^n} \rightarrow \mathbf{R^p}$. The system has learned the function f, if it responds to every single stimulus x_i with its correct y_i. However, in most practical cases, only partial or approximative learning is possible. Learning involves a change in synapses and also quantization. Only a subset of all patterns in the sampled pattern environment is learned. Since memory capacity is limited, an adaptive system such as a neural network has to learn efficiently by replacing old stored patterns with new patterns.

Learning can be either supervised or unsupervised. Supervised learning uses class-membership information while unsupervised learning does not. Biological synapses learn locally and without supervision on a single pass of noisy data. Local information is information available physically and briefly to the synapse. It involves synaptic properties or neuronal signal properties. Locally unsupervised synapses associate signals with signals. This leads to conjunctive, or correlation, learning laws constrained by locality.

Here, we will examine two unsupervised learning laws: signal Hebbian learning law, and competitive learning law or Grossberg law [76].

8.2.2.1 Signal Hebbian Learning Law The deterministic Hebbian learning law correlates local neuronal signals:

$$\dot{m}_{ij} = -m_{ij} + f_i^{\mathbf{X}}(x_i) f_j^{\mathbf{Y}}(y_j) \tag{8.18}$$

The field notation $\mathbf{X}$ and $\mathbf{Y}$ can be omitted and we obtain

$$\dot{m}_{ij} = -m_{ij} + f_i(x_i) f_j(y_j) \tag{8.19}$$

m_{ij} is the synaptic efficacy along the axon connecting the ith neuron in field $\mathbf{F_X}$ with the jth neuron in field $\mathbf{F_Y}$. If $m_{ij} \geq 0$ then the synaptic injunction is excitatory, and it is inhibitory if $m_{ij} \leq 0$.

8.2.2.2 Competitive Learning Law This law modulates the output signal $f_j(y_j)$ with the signal–synaptic difference $f_i(x_i) - m_{ij}$

$$\dot{m}_{ij} = f_j(y_j)[f_i(x_i) - m_{ij}] \tag{8.20}$$

The jth neuron in $\mathbf{F_Y}$ wins the competition at time t if $f_j(y_j(t)) = 1$, and loses it if $f_j(y_j(t)) = 0$. Competitive learning means that synapses learn only if their postsynaptic neurons win. In other words, postsynaptic neurons code for presynaptic signal patterns [120]. The neurons in $\mathbf{F_Y}$ compete for the activation induced by signal patterns from $\mathbf{F_X}$. They excite themselves and inhibit one another.

8.2.3 Neuro-Synaptic Dynamics

Biologically, neural networks model both the dynamics of neural activity levels, the short-term memory (STM), and the dynamics of synaptic modifications, the long-term memory (LTM). The actual network models under consideration may be considered extensions of Grossberg's shunting network [77] or Amari's model for primitive neuronal competition [6]. These earlier networks are considered pools of mutually inhibitory neurons with fixed synaptic connections. Here, we are looking at systems where the synapses can be modified by external stimuli.

The dynamics of competitive systems may be extremely complex, exhibiting convergence to point attractors and periodic attractors. For networks that model only the dynamics of the neural activity levels, Cohen and Grossberg [44] found a Lyapunov function as a necessary condition for the convergence behavior to point attractors.

Networks where both LTM and STM states are dynamic variables cannot be placed in this form since the Cohen-Grossberg equation (8.13) does not model synaptic dynamics. However, a large class of competitive systems have been identified as being "generally" convergent to point attractors even though no Lyapunov functions have been found for their flows. The emergent global properties of a network, rather than the behavior of the individual units and the local computation performed by them, describe the network's behavior. Global stability analysis techniques, such

as Lyapunov energy functions, show the conditions under which a system approaches an equilibrium point in response to an input pattern. The equilibrium point is then the stored representation of the input. This property is termed the content addressable memory (CAM) property. Local stability, by contrast, involves the analysis of network behavior around individual equilibrium points.

Such a neuro-synaptic system is a laterally inhibited network with a deterministic signal Hebbian learning law [85] that is similar to the spatiotemporal system of Amari [7].

The general neural network equations describing the temporal evolution of the STM and LTM states for the jth neuron of an N-neuron network are

$$\dot{x}_j = -a_j x_j + \sum_{i=1}^{N} D_{ij} f(x_i) + B_j \sum_{i=1}^{p} m_{ij} y_i \tag{8.21}$$

$$\dot{m}_{ij} = -m_{ij} + y_i f(x_j) \tag{8.22}$$

where x_j is the current activity level, a_j is the time constant of the neuron, B_j is the contribution of the external stimulus term, $f(x_i)$ is the neuron's output, y_i is the external stimulus, and m_{ij} is the synaptic efficiency. The neural network is modeled by a system of deterministic equations with a time-dependent input vector rather than a source emitting input signals with a prescribed probability distribution. Our interest is to store patterns as equilibrium points in N-dimensional space. In fact, the formation of stable one-dimensional cortical maps under the aspect of topological correspondence and under the restriction of a constant probability of the input signal is demonstrated in [6].

In [163] it was shown that competitive neural networks with a combined activity and weight dynamics can be interpreted as nonlinear singularly perturbed systems [111, 207]. A quadratic-type Lyapunov function was found for the coupled system, and the global stability of an equilibrium point representing a stored pattern was proven.

8.3 PCA-TYPE NEURAL NETWORKS

As previously stated, principal component analysis (PCA) represents an important method for redundant feature extraction and also for data compression.

PCA has several applications in nuclear medical imaging [11, 188], X-ray computed tomography [96] and other X-ray fields, and magnetic resonance imaging (MRI) [14, 17, 56, 157, 158].

In the following we will present neural network architectures which extract the principal components in a self-organized manner. In other words, unsupervised learning applied to a one-layer neural network performs PCA.

8.3.1 Generalized Hebbian Algorithm (GHA)

A simple feedforward neural network with a single layer of linear neurons as shown in Figure 8.4 can compute the principal components of a data vector $\mathbf{x}$.

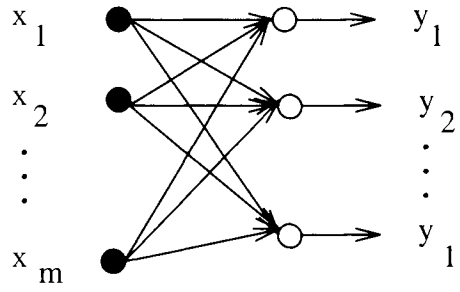


FIGURE 8.4 Feedforward neural network with a single layer of neurons.

Two basic assumptions regarding the network architecture are necessary:

- The neurons of the output layer are linear.
- The network has m inputs and l outputs with $l < m$.

Only the feedforward connections $\{w_{ij}\}$ with $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, l$ are subject to training for determining the principal components based on the generalized Hebbian algorithm (GHA).

The jth neuron output $y_i(n)$ at time t is given by

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n) \quad j = 1, 2, \ldots, l \tag{8.23}$$

The learning is performed based on a generalized Hebbian learning as shown in [214]:

$$\Delta w_{ji}(n) = \eta \left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right] \tag{8.24}$$

$$i = 1, 2, \ldots, m \quad j = 1, 2, \ldots, l \tag{8.25}$$

$\Delta w_{ji}(n)$ is the change applied to the connection weight $w_{ji}(n)$ at time n while η represents the learning rate.

Let us assume as shown in [36, 214] that in the limit we have

$$\Delta \mathbf{w}_j(n) \to \mathbf{0} \quad \text{and} \quad \mathbf{w}_j(n) \to \mathbf{q}_j \quad \text{as} \quad n \to \infty \quad \text{for all} \quad j = 1, 2 \ldots, l \tag{8.26}$$

and

$$||\mathbf{w}_j(n)|| = 1 \quad \text{for all j} \tag{8.27}$$

Then the limiting values $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_l$ of the synapse vectors of the neurons represent the normalized eigenvectors of the l dominant eigenvalues of the correlation matrix $\mathbf{R}$. The dominant eigenvalues are ranked based on their decreasing values. For the steady state, we obtain

$$\mathbf{q}_j^T \mathbf{R} \mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \tag{8.28}$$

with $\lambda_1 > \lambda_2 > \ldots > \lambda_l$.

In the limit, for the jth neuron output we write

$$\lim_{n \to \infty} y_j(n) = \mathbf{x}^T(n)\mathbf{q}_j = \mathbf{q}_j^T \mathbf{x}(n) \tag{8.29}$$

This shows that in the limit the GHA acts as an eigen analyzer of the input data.

We can also give a matrix representation of the GHA. Let $\mathbf{W}$ denote the $l \times p$ connection matrix of the feedforward network shown in Figure 8.4. Additionally, we will assume that the learning-rate parameter of the GHA from eq. (8.24) is time-varying, such that in the limit it holds that

$$\lim_{n \to \infty} \eta(n) = 0 \quad \text{and} \quad \sum_{n=0}^{\infty} \eta(n) = \infty \tag{8.30}$$

Then the GHA can be rewritten in matrix form:

$$\Delta \mathbf{W}(n) = \eta(n)\{\mathbf{y}(n)\mathbf{x}^T(n) - \mathrm{LT}[\mathbf{y}(n)\mathbf{y}^T(n)]\mathbf{W}(n)\} \tag{8.31}$$

where LT[.] is a lower triangular matrix operator. In [214] it was shown that, by randomly initializing the weight matrix $\mathbf{W}(n)$, the GHA will converge, such that $\mathbf{W}^T(n)$ will be approximately equal to a matrix whose columns are the first l eigenvectors of the $m \times m$ correlation matrix $\mathbf{R}$ of the m-dimensional input vector. The l eigenvalues are ordered by decreasing values.

The practical value of the GHA is that it helps in finding the first l eigenvalues of correlation matrix $\mathbf{R}$, assuming that they are distinct. This is accomplished without even computing $\mathbf{R}$, while the l eigenvalues are computed by the algorithm directly from the input vector $\mathbf{x}$.

A description of the generalized Hebbian algorithm is given below:

1. **Initialization:** Initialize the synaptic weights of the network, w_{ij}, with small random numbers. Set η equal to a small positive value.

2. **Adaptation:** For $n = 1$ and $j = 1, 2, \ldots, l$, and $i = 1, 2, \ldots, m$, compute

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n)$$

$$\Delta w_{ji}(n) = \eta \left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right]$$

where $x_i(n)$ is the ith component of the m-dimensional input vector $\mathbf{x}(n)$ and l the prespecified number of principal components.

3. **Continuation:** Increment n by 1, and go to step 2 until the weights w_{ij} reach their steady-state values. Finally, the weight vectors will converge to the normalized eigenvectors of the correlation matrix $\mathbf{R}$ of the input vector $\mathbf{x}$.

8.3.2 Adaptive Principal Component Extraction (APEX)

The GHA is based only on variation of the feedforward synapses to determine the principal components. The adaptive principal component extraction (APEX) algorithm [54] described in this section employs both feedforward and feedback synapses for PCA. APEX is an iterative algorithm that determines the jth principal component from the $(j-1)$ previous ones. Figure 8.5 shows the neural architecture employed for derivation of the APEX algorithm. The input vector $\mathbf{x}$ is m-dimensional, and the neuron output functions are considered linear.

There are two types of synapses involved in the training:

- Feedforward weights: connect the input nodes with each of the neurons $1, 2, \ldots, j$, with $j < m$. A feedforward weight vector for the jth neuron is given by

$$\mathbf{w}_j = [w_{j1}(n), w_{j2}(n), \ldots, w_{jm}(n)]^T$$

  The feedforward weights are trained based on a Hebbian learning algorithm as presented in eq. (8.19), are excitatory, and provide self-amplification.

- Lateral weights: connect the jth neuron and the other $1, 2, \ldots, j-1$ neurons. A lateral weight vector for the jth neuron is given by

$$\mathbf{a}_j(n) = [a_{j1}(n), a_{j2}(n), \ldots, a_{j,j-1}(n)]^T$$

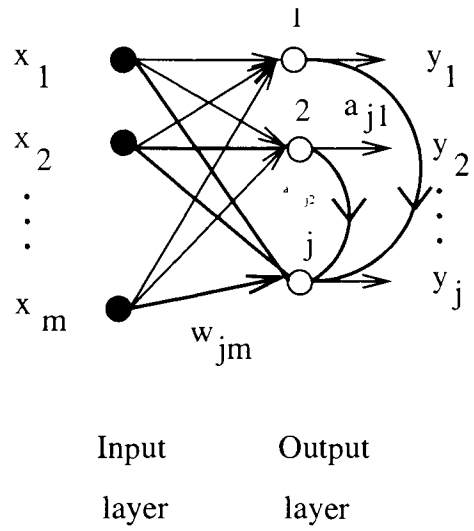


FIGURE 8.5 Adaptive principal component extraction (APEX) neural network architecture.

The lateral weights are trained based on an anti-Hebbian learning algorithm and provide an inhibitory contribution.

The jth neuron output $y_j(n)$ is given by

$$y_j(n) = \mathbf{w}_j^T(n)\mathbf{x}(n) + \mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n) \tag{8.32}$$

where $\mathbf{w}_j^T(n)\mathbf{x}(n)$ describes the contribution of the feedforward synapses, and $\mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n)$ that of the lateral synapses.

The feedback vector $\mathbf{y}_{j-1}(n)$ is defined by the outputs of neurons $1, 2, \ldots, j-1$:

$$\mathbf{y}_{j-1}(n) = [y_1(n), y_2(n), \ldots, y_{j-1}(n)]^T \tag{8.33}$$

Assume that the eigenvalues of the correlation matrix $\mathbf{R}$ are distinct and arranged in decreasing order:

$$\lambda_1 > \lambda_2 > \ldots > \lambda_m \tag{8.34}$$

Additionally, we assume that the neurons have already reached their stable steady states

$$\mathbf{w}_k(0) = \mathbf{q}_k, \quad k = 1, 2, \ldots, j-1 \tag{8.35}$$

and

$$\mathbf{a}_k(0) = \mathbf{0}. \quad k = 1.2, \ldots . j - 1 \tag{8.36}$$

where $\mathbf{q}_k$ corresponds to the kth eigenvalue of matrix $\mathbf{R}$. $n = 0$ denotes the iteration start. Based on eqs. (8.32), (8.33), (8.35), and (8.36), we obtain the following result:

$$\mathbf{y}_{j-1}(n) = [\mathbf{q}_1^T\mathbf{x}(n), \mathbf{q}_2^T\mathbf{x}(n), \ldots , \mathbf{q}_{j-1}^T\mathbf{x}(n)] = \mathbf{Q}\mathbf{x}(n) \tag{8.37}$$

$\mathbf{Q}$ is the $(j-1) \times m$ eigenvector matrix. This matrix has $j-1$ eigenvectors corresponding to the dominant eigenvalues $\lambda_1, \lambda_2, \ldots , \lambda_{j-1}$ of the correlation matrix $\mathbf{R}$:

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots , \mathbf{q}_{j-1}]^T \tag{8.38}$$

It is required to use the jth neuron to determine the next largest eigenvalue λ_j of the correlation matrix $\mathbf{R}$.

The adaptation synaptic equations for the feedforward synaptic vector and lateral inhibition synaptic vector are defined as

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta[y_j(n)\mathbf{x}(n) - y_j^2(n)\mathbf{w}_j(n)] \tag{8.39}$$

and

$$\mathbf{a}_j(n+1) = \mathbf{a}_j(n) - \eta[y_j(n)\mathbf{y}_{j-1}(n) + y_j^2(n)\mathbf{a}_j(n)] \tag{8.40}$$

η represents the learning rate that is the same for both adaptation equations since they implement the same type of dynamics. $y_j(n)\mathbf{x}(n)$ denotes Hebbian learning as shown in eq. (8.19), while $y_j(n)\mathbf{y}_{j-1}(n)$ denotes anti-Hebbian learning. The two residual terms, $y_j^2(n)\mathbf{w}_j(n)$ and $y_j^2(n)\mathbf{a}_j(n)$, impose stability on the algorithm.

In [54] it was shown that by choosing a sufficiently small eigenvalue that the weight vectors do not adjust too fast, and given the eigenvectors $\mathbf{q}_1, \mathbf{q}_2, \ldots , \mathbf{q}_{j-1}$, then neuron j in Figure 8.5 computes the next largest eigenvalue λ_j and the corresponding eigenvector $\mathbf{q}_j$.

In practical applications, the learning rate η_j is set equal to

$$\eta_j = \frac{1}{\lambda_{j-1}} \tag{8.41}$$

λ_{j-1} is available, since it was computed by neuron $j-1$.

An algorithmic description of the APEX algorithm is given below:

1. **Initialization:** Initialize the synaptic weight vectors of the network, $\mathbf{w}_j$ and $\mathbf{a}_j$, with small random numbers. Set η equal to a small positive value.

2. **Adaptation, part I:** Consider the first neuron by setting $j = 1$, and compute for $n = 1, 2, \ldots$

$$\begin{aligned} y_1(n) &= \mathbf{w}_1^T(n)\mathbf{x}(n) \\ \mathbf{w}_1(n+1) &= \mathbf{w}_1(n) + \eta[y_1(n)\mathbf{x}(n) - y_1^2(n)\mathbf{w}_1(n)] \end{aligned}$$

$\mathbf{x}(n)$ is the input vector. For large n, we have $\mathbf{w}_1(n) \to \mathbf{q}_1$, where $\mathbf{q}_1$ represents the eigenvector associated with the largest eigenvalue λ_1 of the correlation matrix $\mathbf{R}$.

3. **Adaptation, part II:** Consider now the second neuron by setting $j = 2$, and compute for $n = 1, 2, \ldots$

$$\begin{aligned} \mathbf{y}_{j-1}(n) &= [y_1(n), y_2(n), \ldots, y_{j-1}(n)]^T \\ y_i(n) &= \mathbf{w}_j^T(n)\mathbf{x}(n) + \mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n) \\ \mathbf{w}_j(n+1) &= \mathbf{w}_j(n) + \eta[y_j(n)\mathbf{x}(n) - y_j^2(n)\mathbf{w}_j(n)] \\ \mathbf{a}_j(n+1) &= \mathbf{a}_j(n) - \eta[y_j(n)\mathbf{y}_{j-1}(n) + y_j^2(n)\mathbf{a}_j(n)] \end{aligned}$$

4. **Continuation:** Consider the next neuron, increment j by 1. and go to step 3 until $j = m$. where m is the desired number of principal components. For large n, we obtain $\mathbf{w}_j(n) \to \mathbf{q}_j$ and $\mathbf{a}_j(n) \to \mathbf{0}$, where $\mathbf{q}_j$ is the jth eigenvector of the correlation matrix $\mathbf{R}$.

8.3.3 Linear and Nonlinear Oja's Algorithm

There is an amazing correspondence between the self-organization principle in neurocomputation and principal component analysis in statistics. Oja showed in [177] that it is possible with a single linear neuron to find the first principal component of a stationary input. The weights of this neuron must be trained based on a Hebbian-type learning rule.

Let us assume that the input vector $\mathbf{x}$ and also the weight vector $\mathbf{w}$ are m-dimensional, and the only neuron is linear and its output y is given as

$$y = \sum_{i=1}^{m} w_i x_i \tag{8.42}$$

For the synaptic adaptation we consider a Hebbian-type learning

$$w_i(n+1) = w_i(n) + \eta y(n) x_i(n) \tag{8.43}$$

where n is the discrete time and η the learning rate. Described by the above equation, the learning rule has the tendency to grow too much. Therefore, it is necessary to find a form of saturation for the learning rule, and thus

ensure convergence. In [177], it was proposed to use the following normalization term:

$$w_i(n+1) = \frac{w_i(n) + \eta y(n) x_i(n)}{(\sum_{i=1}^{m} [w_i(n) + \eta y(n) x_i(n)]^2)^{\frac{1}{2}}} \tag{8.44}$$

If the learning rate η is small, then we can expand eq. (8.44) as a power series in η:

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n) w_i(n)] + O(\eta^2) \tag{8.45}$$

where $O(\eta^2)$ corresponds to the higher-order effects in η. For small η they can be ignored, and we get a new representation of the learning rule:

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n) w_i(n)] \tag{8.46}$$

$y(n)x_i(n)$ is the known Hebbian modification while $y(n)w_i(n)$ is for stabilization purposes.

Equation (8.46) representing the learning algorithm can also be expressed in matrix form. Let

$$\mathbf{x} = [x_1(n), x_2(n), \ldots, x_m(n)]^T$$

be an input vector and

$$\mathbf{w} = [w_1(n), w_2(n), \ldots, w_m(n)]^T$$

the synaptic weight vector. The output y can be represented as following

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{w}^T(n)\mathbf{x}(n) \tag{8.47}$$

Then eq. (8.46) can be rewritten as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n)[\mathbf{x}(n) - y(n)\mathbf{w}(n)] \tag{8.48}$$

By substituting eq. (8.47) into eq. (8.46), we obtain a new representation of the learning rule:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n)\mathbf{w}(n)] \tag{8.49}$$

Let us assume that the correlation matrix $\mathbf{R}$ is positive definite with the largest eigenvalue λ_1 having multiplicity 1. In [177] it was shown that a Hebbian-based linear neuron has the following two important properties:

1. The variance of the model output approaches the largest eigenvalue of the correlation matrix $\mathbf{R}$:

$$\lim_{n\to\infty} \sigma^2(n) = \lambda_1 \tag{8.50}$$

2. The synaptic weight vector of the model approaches the associated eigenvector

$$\lim_{n\to\infty} \mathbf{w}(n) = \mathbf{q}_1 \tag{8.51}$$

with

$$\lim_{n\to\infty} ||\mathbf{w}(n)|| = 1 \tag{8.52}$$

Oja's single neuron learning can be generalized to networks having several neurons in the output layer. The proposed learning rule consists of Hebbian learning of the connection weights and a nonlinear feedback term that is used to stabilize the weights. Specifically, Oja's learning equation is of the form

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta[\mathbf{x}(n)\mathbf{x}^{\mathbf{T}}\mathbf{W}(n) - \mathbf{W}(n)(\mathbf{W}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{W}(n))] \tag{8.53}$$

When the input data are assumed to arrive as samples from some stationary pattern class distribution with autocorrelation matrix $\mathbf{R}$, eq. (8.53) can be averaged to the more convenient equation

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta[\mathbf{I} - \mathbf{W}(n)\mathbf{W}^T(n)]\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{W}(n) \tag{8.54}$$

In the matrix case, Oja has conjectured similar convergence properties as in the one-unit case. Those properties have been proved in [247] based on a mathematical analysis of the learning rule.

Let us assume that $\mathbf{R}$ has eigenvalues arranged in decreasing order $\lambda_1 > \lambda_2 > \ldots > \lambda_m$ and let $\mathbf{q}_i$ be the corresponding eigenvalue to the ith eigenvector. Let $\mathbf{Q}$ be an $m \times m$ eigenvector matrix

$$\mathbf{Q} = [\mathbf{q}_1, \ldots . \mathbf{q}_m] \tag{8.55}$$

Let $\mathbf{W}$ denote the $l \times m$ weight matrix of the feedforward network

$$\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_l]^T \tag{8.56}$$

Then Oja's algorithm shown in eq. (8.54) converges to a fixed point, which is characterized as follows:

1. The product of $\mathbf{W}^T(n)\mathbf{W}(n)$ tends in the limit to the identity matrix:

$$\lim_{n\to\infty} \mathbf{W}^T(n)\mathbf{W}(n) = \mathbf{I} \tag{8.57}$$

2. The columns of $\mathbf{W}(n)$ converge to a dominant eigenspace of the associated autocorrelation matrix:

$$\lim_{n\to\infty} \mathbf{w_i}(n) = \mathbf{q}_i \tag{8.58}$$

An algorithmic description of Oja's learning rule is given below:

1. **Initialization:** Initialize the synaptic weights of the network, w_{ij}, with small random numbers. Set η equal to a small positive value.
2. **Adaptation:** For $n = 1$ compute the weight matrix $\mathbf{W}$ based on

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta[\mathbf{I} - \mathbf{W}(n)\mathbf{W}^T(n)]\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{W}(n) \tag{8.59}$$

   where $\mathbf{x}$ is an m-dimensional input vector.
3. **Continuation:** Increment n by 1, and go to step 2 until the weight matrix $\mathbf{W}$ reaches the steady-state value. Finally, the weight vectors will converge to the eigenvectors of the correlation matrix $\mathbf{R}$ of the input vector $\mathbf{x}$.

PCA has some limitations that make it less attractive from a neural network point of view. These limitations are: (1) PCA networks implement only linear input-output mappings, (2) slow convergence makes it not useful in large problems, (3) second-order statistics describe only Gaussian data completely, and (4) outputs of PCA networks are mutually uncorrelated but not independent.

For this reason, it is necessary to study nonlinear generalizations of PCA learning algorithms and networks. In [180], some nonlinear generalizations of Oja's learning rule in equation (8.54) have been proposed:

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta[\mathbf{I} - \mathbf{W}(n)\mathbf{W}^T(n)]\mathbf{x}(n)g(\mathbf{x}^T(n)\mathbf{W}(n)) \tag{8.60}$$

$$\begin{aligned}\mathbf{W}(n+1) &= \mathbf{W}(n) + \eta[\mathbf{x}(n)g(\mathbf{x}^T(n)\mathbf{W}(n)) \\ &\quad -\mathbf{W}(n)g(\mathbf{W}^T(n)\mathbf{x}(n))g(\mathbf{x}^T(n)\mathbf{W}(n))]\end{aligned} \tag{8.61}$$

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{W}(n) - \mathbf{W}(n)\mathbf{W}^T(n)\mathbf{x}(n)g(\mathbf{x}^T(n)\mathbf{W}(n))] \tag{8.62}$$

g is usually a monotone odd nonlinear function. Here the function $g(t)$ is applied separately to each component of its argument vector. Generally, we choose for $g(t)$ the function $g(y(t)) = \tanh(\alpha y(t))$, where α is a scalar parameter. The advantage of using nonlinearities is that the neurons become more selective during the learning phase. They also introduce higher-order statistics into the computations in an implicit way and also increase the independence of the outputs, so that the original signals can sometimes be roughly separated from their mixtures.

The nonlinear Hebbian learning rules proposed by Oja are not only related to PCA. Provided that all directions of the input space have equal variance, and the nonlinearity is $\tanh(y(t))$, then the learning rule minimizes the kurtosis in several directions. and it tends to find those directions along which the input data projections are statistically independent. Therefore these algorithms can be used for independent component analysis (ICA). Minimizing the kurtosis means, as we will see in section 8.4.2, to find the directions of the weight vectors in the input space, on which the projection of the input data is heavily clustered and deviates strongly from the Gaussian distribution.

8.3.4 Medical Image Coding

In this section we complete the discussion of PCA-type neural networks by examining their use for solving medical image coding.

A set of four mammograms were selected from the publicly available MIAS database at http://skye.icr.ac.uk/misdb/miasdb.html. All 1024×1024 images are monochrome with 256 gray levels. The images were coded using a linear feedforward neural network with L input neurons and M linear neurons, where $M << L$ is performing the PCA based on three different algorithms. The output of the network $\mathbf{y}$ is defined by $\mathbf{y} = \mathbf{W}^{\mathbf{T}}\mathbf{x}$. $\mathbf{x}$ is the input vector and $\mathbf{W}$ is the weight matrix of the network.

Depending on the type of the activation function (linear or nonlinear) three different PCA algorithms, generalized Hebbian algorithm (GHA) [214], Oja's symmetric algorithm [178], and nonlinear PCA (NLPCA) [101], were implemented.

Oja's symmetric algorithm will provide the convergence of the weight matrix to a subspace spanned by the most significant eigenvectors of the input data covariance matrix. The outputs of the network will then be decorrelated and their variance maximized, so the information content of the L-dimensional vectors will be maximally transferred by the M-dimensional output vectors in the mean square sense.

The weight adaptation equations for the feedforward weight matrix $\mathbf{W}$ for Oja's algorithm are given by eq. (8.54) and for the NLPCA algorithm are given by eq. (8.61).

The synaptic weights w_{ij} connecting the neuron i of the input layer (consisting of L neurons) with the neuron j of the output layer (consisting of

M neurons) of the network are updated by the GHA algorithm as described in eq. (8.24).

To train the network, different block sizes, from 2×2, 4×4, and 8×8 pixels, were used.

In the following the convergence properties of the three PCAs are shown, along with the compression results achieved based on the PCAs without and with overlapping. To reduce the blocking effects associated with any transform-based algorithm, overlapping of neighboring image blocks by one pixel was allowed.

The convergence speed of the three algorithms is shown in Figure 8.6. They were trained with 320,000 8×8 sample blocks. The graph 8.6 shows the number of iterations versus the output error. The results in the same figure show that the nonlinear PCA type learning algorithm converges very fast (dashed curve), the GHA algorithm converges more slowly (solid curve) but more steadily, and Oja's symmetric algorithm takes longer than the others two but decreases monotonously.

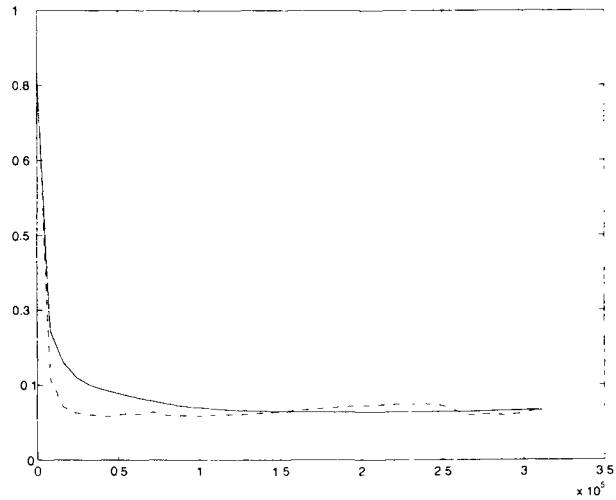
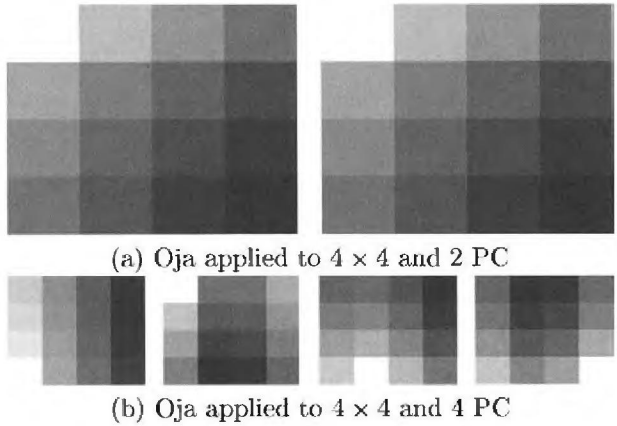


FIGURE 8.6 The three different implementations of the PCA algorithm: solid, GHA; dotted, Oja's symmetric algorithm; dashed, nonlinear PCA type learning. The x-axis shows the number of the training step; the corresponding value on the y-axis shows the normalized error.

Figures 8.7 and 8.8 show the 2×2, 4×4, and 8×8 masks representing the synaptic weights learned by the network. Figure 8.7 shows the 4×4 weight matrices for two and four largest principal components for Oja's algorithm, which is the fastest of the three algorithms.

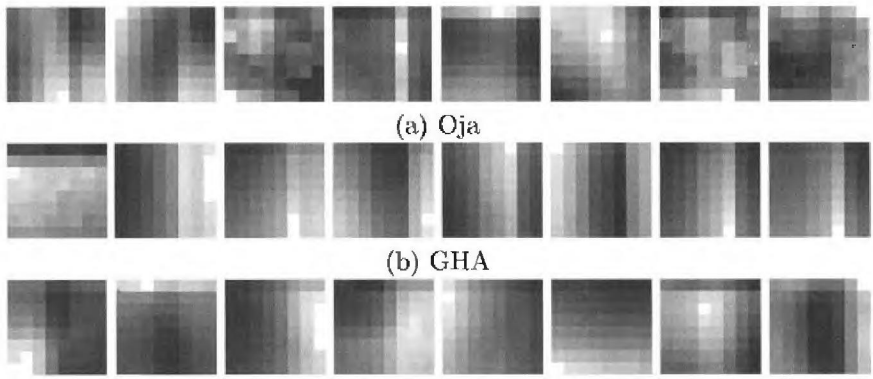


(a) Oja applied to 4 × 4 and 2 PC

(b) Oja applied to 4 × 4 and 4 PC

FIGURE 8.7 Weight matrices based on Oja's algorithm for (a) two and (b) four principal components.

In Figure 8.8 a comparison between the 8 × 8 weight matrices for Oja, GHA, and nonlinear PCA for the eight largest principal components is illustrated.



(a) Oja

(b) GHA

(c) Nonlinear PCA

FIGURE 8.8 8×8 weight matrices based on Oja, GHA, and nonlinear PCA, and eight largest principal components.

A mammogram from the MIAS database, which will serve as an original image for compression purposes, is shown in Figure 8.9.

The compression results based on the three PCAs are shown in Figure 8.10(a), (c) and (e). The nonlinear PCA shows a strong blocking effect as a result of the compression process. The best image quality subjectively rated is achieved by Oja's algorithm.

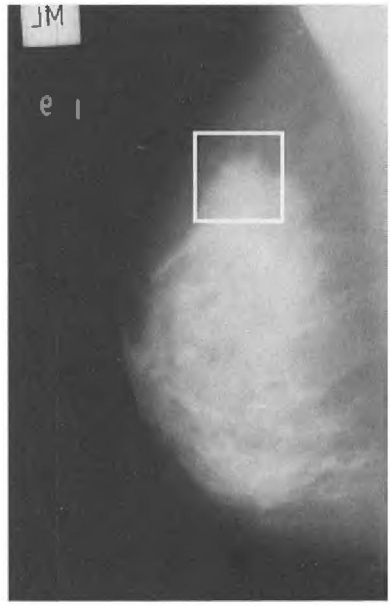
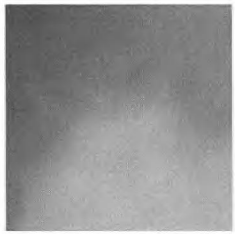


FIGURE 8.9 Original digitized mammogram from MIAS database. The white square indicates the area, which is subject to the compression operation.

8.4 ICA-TYPE NEURAL NETWORKS

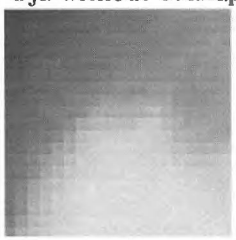
8.4.1 Introduction

Independent component analysis (ICA) is viewed as an extension of the widely used statistical technique principal component analysis (PCA). ICA has been developed in the context of blind separation of independent sources from their linear mixtures, and it has been applied to many areas such as speech, radar, medical instrumentation, mobile telephone communication, and hearing-aid devices. The problem is defined as the estimation of original source signals from a known and measurable sensor output when the sensor receives an unknown mixture of the source signals. ICA is closely related to blind source separation in communication, which consists of recovering unobserved signals or "sources" from several observed mixtures. The observed signals typically represent the sensors' outputs where each sensor receives a different combination (mixture) of the source signals. The term "blind" refers to the fact that (1) the source signals are not observed, and (2) there is no information available about the mixture. The missing mixture information is not considered a disadvantage since at the same time there is an underlying assumption about the statistical independence of the source signals. Thus "blindness" represents a strength, since it captures the spatial diversity provided by an array of sensors.

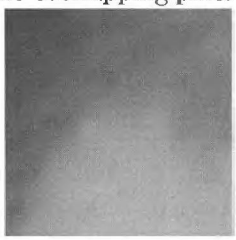


(a) Oja without overlapping

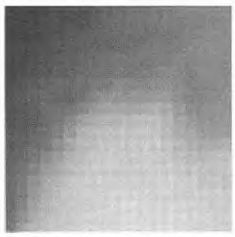
(b) Oja with one overlapping pixel



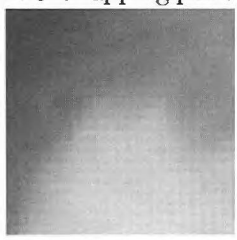
(c) GHA without overlapping



(d) GHA with one overlapping pixel



(e) Nonlinear without overlapping pixel



(f) Nonlinear with one overlapping

FIGURE 8.10 Cut-outs of the compressed test image. Compression results with and without overlapping for the three PCA algorithms.

An interesting form of blind source separation is found in cognitive processing, the so-called "cocktail party problem," where a person can single out a specific speaker from a group speaking simultaneously [72].

Blind source separation represents an underdetermined problem for which a close solution does not exist, since the single source signal statistics, the mixing, and the transfer channels are all unknown. It goes beyond decorrelation, which is based on second-order statistics, and thus requires higher order statistics. To achieve this, the learning process must employ some suitable nonlinearities. However, the input-output function remains linear.

Jutten and Herault [95] were the first to develop a neural architecture and learning algorithm for blind separation; since then several variants of it have appeared in the literature. Bell and Sejnowski [19] have devel-

oped a feedforward network and learning rule which minimizes the mutual information at the output nodes. Karhunen and Joutsensalo [101] have proposed several nonlinear variants of principal component analysis learning and have demonstrated their utility in sinusoidal frequency estimation. Oja [102] and Girolami and Fyfe [73] have justified theoretically the use of the nonlinear PCA algorithm in blind separation of source signals. In general, source separation is obtained by minimizing an appropriate contrast function, i.e., a function of the distribution of the estimated sources. Usually, the contrast functions are based on maximum likelihood [13, 192, 32], the infomax principle [136, 20, 206], and mutual information [45, 8, 90]. Although each model stems from different theoretical aspects, it is possible to generalize them under the maximum likelihood principle, leading to simple and efficient algorithms [32].

Summarizing, ICA can be viewed as an improvement of PCA with a high practical value. The coefficients of a linear expansion of the data vectors are no longer required to be uncorrelated for ICA, but mutually independent. However, although ICA provides in many cases a more meaningful representation of the data than PCA, it still sometimes employs PCA as a preprocessing and simplifying step.

8.4.1.1 The Data Model Suppose that the sensor signals $x_i(t)(i = 1, \ldots, N)$ are generated by linearly weighting a set of N statistically independent random sources $s_j(t)$ with time-independent coefficients a_{ij}:

$$x_i(t) = \sum_{j=1}^{N} a_{ij} s_j(t) \tag{8.63}$$

This equation can be equivalently expressed in vector notation as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{8.64}$$

where $\mathbf{s}(t) = [s_1(t), \ldots, s_N(t)]^T$, $\mathbf{x}(t) = [x_1(t), \ldots, x_N(t)]$, and $\mathbf{A} = [a_{ij}]$.

The objective is to recover the source signals $s_j(t)(j = 1, \ldots, N)$ from the sensor signals $x_i(t)(i = 1, \ldots, N)$ in the absence of any special information about the properties of $\mathbf{A}$ and $\mathbf{x}(t)$. This definition of signal separation can be somewhat confusing: If $s_j(t)(j = 1, \ldots, N)$ are source signals, then $d_1 s_{p1}(t), \ldots, d_N s_{pN}(t)$ may as well be considered source signals, where $\{p_1, \ldots, p_N\}$ is an arbitrary permutation of $\{1, \ldots, N\}$ and $d_1, \ldots, d_N$ are any constants unequal to zero. This is because $d_1 s_{p1}(t), \ldots, d_N s_{pN}(t)$ are statistically independent and $x_i(t)$ can be expressed by their linear combination. Since the sensor signals might be possibly recalled in a different order and rescaled from the resulting vector $\bar{\mathbf{s}}(t)$, we define blind source separation as any procedure providing the following type of signal:

$$\bar{\mathbf{s}}(t) = \mathbf{D}\mathbf{P}\mathbf{s}(t) \tag{8.65}$$

where $\mathbf{P}$ is a permutation matrix and $\mathbf{D}$ is a diagonal matrix with nonzero elements.

The problem of estimating the data model of ICA is to estimate the mixing matrix $\mathbf{A}$ using only the information contained in the sensor vector $\mathbf{x}$. In this perspective, the statistical model has two components: the mixing $\mathbf{A}$ and the probability distribution of the source vector $\mathbf{s}$.

The following assumptions [102] are necessary for the blind separation process:

1. Matrix $\mathbf{A}$ is constant and nonsingular. The mixing matrix is the parameter of interest, and its columns are assumed to be linearly independent such that it is invertible.
2. The source signals $s_j(t)(j = 1, \ldots, N)$ are mutually statistically independent signals. Although nothing is specified with regard to the distribution of each source, the assumption of mutual source independence states a great deal about their joint distribution.
3. Each source signal $s_j(t)$ is a stationary zero-mean stochastic process. Only one source signal can be Gaussian. Since the linear combination of Gaussian source signals is Gaussian, it becomes impossible to separate the initial sources from each other.

Under assumptions 1–3 it can be proved that the source signals can be uniquely determined from the sensor signals except the arbitrariness of the permutation matrix $\mathbf{P}$ and the diagonal matrix $\mathbf{D}$.

Here it is useful to compare ICA to standard PCA. Although in PCA, the data model has the same form as in eq. (8.63), PCA decorrelates only the signals by constraining the basis vectors $\mathbf{a}(t)$ to be mutually orthonormal. On the other hand, ICA separates the signals. Usually, the basis vectors $\mathbf{a}(t)$ are not mutually orthogonal. Figure 8.11 illustrates the differences between ICA and PCA [102].

In the figure, it is assumed that the data vectors are uniformly distributed within the sketched parallelogram. The solid lines correspond to the two basis vectors of ICA, and the dashed lines to the orthogonal basis vectors of PCA. It is obvious that the basis vectors of ICA capture the data much better.

The problem of estimating the data model of ICA is to estimate the mixing matrix $\mathbf{A}$ using only the information contained in the sensor vector $\mathbf{x}$. In the following sections we take a closer look at these aspects.

8.4.1.2 Definition of Independence As we have seen so far, the basic assumption of mutual source independence represents the building block for blind source separation.

For a better understanding of the theory behind ICA estimation, we give in the following the mathematical definitions of independence and uncorrelatedness.

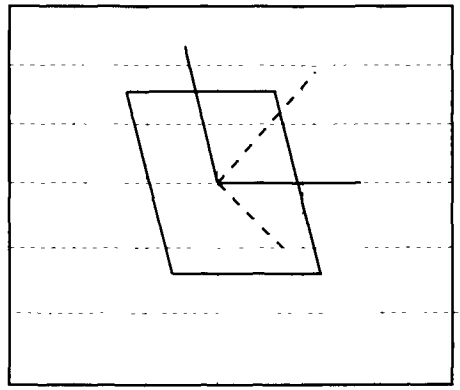


FIGURE 8.11 The basis vectors of ICA (solid lines) and PCA (dashed lines). The data vectors are distributed uniformly inside the parallelogram.

Given are two random variables y_1 and y_2. They are said to be independent if the occurrence or nonoccurrence of y_1 does not influence the occurrence or nonoccurrence of y_2, and vice versa.

Independence can be defined in terms of probability densities. Let $p(y_1, y_2)$ denote the joint probability density function (pdf) of y_1 and y_2. Also, let $p_1(y_1)$ denote the marginal pdf of y_1:

$$p_1(y_1) = \int p(y_1, y_2) dy_2 \tag{8.66}$$

and similarly for y_2. The two random variables y_1 and y_2 are statistically independent if and only if the joint pdf is factorizable:

$$p(y_1, y_2) = p_1(y_1) p_2(y_2) \tag{8.67}$$

This definition allows us to derive an important result pertaining to the product of two functions of these two independent random variables y_1 and y_2

$$E\{h_1(y_1)h_2(y_2)\} = E\{h_1(y_1)\}E\{h_2(y_2)\} \tag{8.68}$$

An interesting implication emerges from the definition of independence: Statistically independent variables are also uncorrelated, but the reverse is not necessarily true. It is said that two random variables y_1 and y_2 are uncorrelated if their covariance equals zero:

$$E\{y_1 y_2\} = E\{y_1\}E\{y_2\} = 0 \tag{8.69}$$

In summary, if the variables are independent, then they are uncorrelated. This follows directly from eq. (8.68) by just taking $h_1(y_1) = y_1$ and $h_2(y_2) =$

y_2. It is very important to mention that uncorrelatedness does not imply independence.

Because uncorrelatedness results from independence, many ICA methods constrain the estimation procedure such that it always first determines the uncorrelated estimates of the independent components. This simplifying preprocessing step reduces the number of free parameters. In other words, blind separation can be based on independence; but independence cannot be reduced to simple decorrelation. This is evident from the fact that there are only $N(N-1)/2$ such conditions while N^2 parameters are unknown.

The fundamental restriction for ICA is that only non-Gaussian independent components can be estimated. At most one independent component is allowed to be gaussian. To understand this restriction, we have to take a close look at the mixing of Gaussian source signals. If the mixing matrix is orthogonal, it leads to sensor signals that are also Gaussian, uncorrelated, and of unit variance. Their joint pdf is completely symmetric, and so it does not contain any information on the directions of the columns of the mixing matrix. In other words, the mixing matrix $\mathbf{A}$ cannot be estimated. In conclusion, the mixing remains unidentifiable for Gaussian independent components.

8.4.1.3 Preprocessing for ICA In order to make the ICA estimation simpler and more feasible for practical applications, it is useful to preprocess the signals. Second-order statistics (decorrelation) is employed to remove redundancy, and thus make ICA estimation simpler. The most used techniques are centering, and the whitening or sphering of a random signal.

By centering a signal $\mathbf{x}$, the mean vector $\mathbf{m} = E\{\mathbf{x}\}$ is subtracted from $\mathbf{x}$, in order to make both $\mathbf{x}$ and $\mathbf{s}$ zero-mean variables.

The other preprocessing technique is whitening, and it is applied after the centering and before the ICA estimation. By whitening we transform the observed vector $\mathbf{x}$ linearly into a new vector $\mathbf{z}$, which has as a covariance matrix the identity matrix:

$$E\{\mathbf{z}\mathbf{z}^T\} = \mathbf{I} \tag{8.70}$$

In other words, the components of $\mathbf{z}$ are uncorrelated and their variances equal unity. The whitening transformation can be performed based on the eigenvalue decomposition [59] of the covariance matrix $E\{\mathbf{x}\mathbf{x}^\mathbf{T}\} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, where $\mathbf{Q}$ is the orthonormal matrix of the eigenvectors of $E\{\mathbf{x}\mathbf{x}^\mathbf{T}\}$ and $\mathbf{\Lambda}$ is the diagonal matrix of its eigenvalues, $\mathbf{\Lambda} = \text{diag}\{\lambda_i\}$. The transformation for whitening is thus given by

$$\mathbf{z} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T\mathbf{x} \tag{8.71}$$

Whitening leads to a new orthogonal mixing matrix $\tilde{\mathbf{A}}$,

$$\mathbf{z} = \mathbf{Q}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{Q}^T\mathbf{A}\mathbf{s} = \tilde{\mathbf{A}}\mathbf{s} \tag{8.72}$$

with $E\{\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T\} = \tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \mathbf{I}$. But $\tilde{\mathbf{A}}$ is at the same time a rotation matrix since it connects to spatially white vectors $\mathbf{z}$ and $\mathbf{s}$. In other words, "whitening" or "sphering" the data reduces the mixture to a rotation matrix.

The whitening also reduces the number of parameters to be estimated. The new orthogonal matrix $\tilde{\mathbf{A}}$ has about half the number of parameters of an arbitrary matrix. Additionally, whitening provides a dimension reduction. By employing PCA, the small eigenvalues can be discarded. The benefit lies not only in dimension reduction, it also prevents overlearning of the ICA algorithm and reducing of noise.

8.4.2 Techniques for ICA Estimation

As stated in the previous section, ICA estimation requires non-Gaussianity. This is different from classical statistical theory where random variables are assumed to have Gaussian distributions. Furthermore, the central limit theorem [186] states that the sum of independent random variables tends to a normal distribution. This represents a very important consequence for ICA estimation, as will be seen in the following.

We will now state the ICA estimation problem. Given N independent realizations of the sensor signals $\mathbf{x}$, determine an estimate of the inverse of the mixing matrix $\mathbf{A}$. In other words, find a demixing matrix $\mathbf{W}$, whose individual rows are a rescaling and permutation of those of the mixing matrix $\mathbf{A}$. Thus, the solution to the ICA estimation problem may be expressed in the form

$$\mathbf{y} = \mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{A}\mathbf{s} \rightarrow \mathbf{D}\mathbf{P}\mathbf{s} \tag{8.73}$$

where $\mathbf{D}$ is a nonsingular diagonal matrix and $\mathbf{P}$ a permutation matrix. The block diagram of the ICA estimation problem is illustrated in Figure 8.12.

We immediately see that y_i is a linear combination of the s_i. Based on the central limit theorem, y_i is more Gaussian than any of the s_i, and becomes least Gaussian when it in fact equals one of the s_i.

For the ICA estimation problem, we could take as $\mathbf{W}$ a matrix that maximizes the non-Gaussianity of $\mathbf{W}\mathbf{x}$.

We see that the following aspects are important:

- ICA must go beyond second-order statistics. A cost function with a penalty term involving only pairwise decorrelation would not lead to signal separation.
- Source separation can be achieved by optimizing a "contrast function" [32]. This contrast function represents a scalar measure describing the distributional property of the output $\mathbf{y}$ such as entropy,

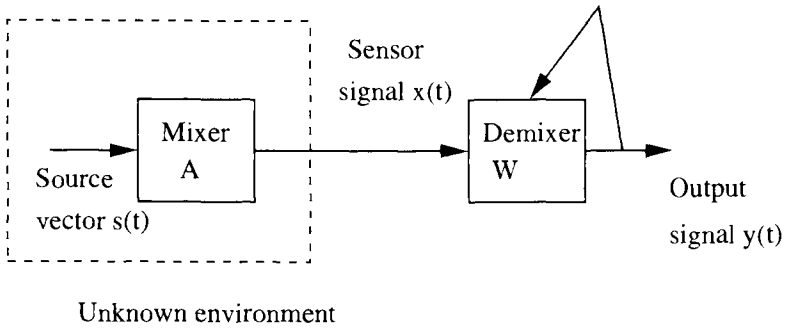


FIGURE 8.12 Block diagram for the blind source separation problem.

mutual independence, high-order decorrelations, and divergence between the joint distribution of $\mathbf{y}$ and a given model.

- Fast adaptation is feasible, even based on simple algorithms, and requires a small number of samples.

We see that "contrast functions" are objective functions for source separation. They are based either on high-order correlations or on information theory. They are a real function of the probability distribution of the output $\mathbf{y} = \mathbf{W}\mathbf{x}$, and they serve as objectives: They have to be designed such that the source separation is achieved when they reach their minimum. Contrast functions for source separation are usually denoted as $\phi[\mathbf{y}]$.

In [32] several such "contrast functions" are proposed. The most important groups are:

1. High-order approximations: kurtosis and negentropy
2. Information theoretic contrasts: matching distributions such as likelihood and infomax, and matching the structure such as mutual information

For simplification reasons, we need to assume that y is a random variable of zero-mean (centered) and of unit variance (sphered):

$$E\{y\} = 0 \quad \text{and} \quad E\{y^2\} = 1 \tag{8.74}$$

8.4.2.1 Kurtosis As a measure for non-Gaussianity, we can use simple higher order statistics called kurtosis, which represents a fourth-order cumulant with zero time lags. The kurtosis of y is classically defined by

$$\text{kurt(y)} = E\{y^4\} - 3(E\{y^2\})^2 \tag{8.75}$$

If eq. (8.74) holds, then we get for the kurtosis $\text{kurt}(y) = E\{y^4\} - 3$, which represents a shifted version of the fourth moment $E\{y^4\}$. For a normal random variable the kurtosis is equal to zero. whereas for most non-Gaussian random variables it is nonzero, being either positive or negative. Negative kurtosis is typical for sub-Gaussian variables having a "flat" pdf that exhibits rather constant values around zero and otherwise is very small. An example of such a sub-Gaussian random variable is the uniform distribution

$$p(y) = \begin{cases} \frac{1}{2\sqrt{5}}, & \text{if } |y| \leq \sqrt{5} \\ 0, & \text{otherwise} \end{cases} \tag{8.76}$$

Positive kurtosis is typical for super-Gaussian variables having a "spiky" pdf with long tails such as the Laplace distribution:

$$p(y) = \exp(-|y|) \tag{8.77}$$

Kurtosis is a linear operation satisfying the superposition principle. Thus, for any two random variables y_1 and y_2 and a scalar α it holds that

$$\text{kurt}(y_1 + y_2) = \text{kurt}(y_1) + \text{kurt}(y_2) \tag{8.78}$$

$$\text{kurt}(\alpha y_1) = \alpha^4 \text{kurt}(y_1) \tag{8.79}$$

To use kurtosis in practical problems means to start from some weight matrix $\mathbf{W}$ and compute the direction the kurtosis is growing most strongly (positive kurtosis) or decreasing most strongly (negative kurtosis). However, it was shown in [89] that kurtosis is sensitive to outliers and therefore is not considered a robust measure of non-Gaussianity.

An interesting interpretation to kurtosis is given in terms of contrast functions. High-order statistics can be employed to define contrast functions which are simple approximations to those derived from the maximum likelihood approach, as we will see in the next sections.

8.4.2.2 **Negentropy** Negentropy is usually taken as another non-Gaussian measure and describes the information-theoretic quantity of (differential) entropy. The entropy of a random variable is viewed as a measure of the uncertainty of an event. The maximal entropy is achieved by a distribution that exhibits the highest probable randomness. The differential entropy H of a random vector $\mathbf{y}$ with a given pdf $f(\mathbf{y})$ is defined as [186]

$$H(\mathbf{y}) = -\int f(\mathbf{y}) \log f(\mathbf{y}) d\mathbf{y} \tag{8.80}$$

In [186] it was proved that the Gaussian variable has the largest entropy among all random variables of equal variance. The implication of

this fundamental result is that the entropy can serve as a measure of non-Gaussianity. The entropy is small for "spiky" pdfs or those pdfs being centered around certain values.

The goal is to obtain a measure of non-Gaussianity that is zero for a Gaussian variable and in general is nonnegative. Such a measure is the so-called negentropy J, defined as follows:

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{8.81}$$

where $\mathbf{y}_{gauss}$ has a normal pdf and has the same mean and variance as $\mathbf{y}$. The negentropy has the following important properties: (1) It is always non-negative, and it is zero if and only if $\mathbf{y}$ has a normal pdf, and (2) it is invariant to invertible linear transformations [45]. Its main disadvantage is that it is computationally expensive. This makes it necessary to approximate the negentropy, and the most useful approximation is given in [93]. By using a nonquadratic function G, we get the following approximation for J:

$$J(y) \propto [E\{G(y)\} - E\{G(\nu)\}]^2 \tag{8.82}$$

where ν is a Gaussian variable of zero mean and unit variance. y is a random variable of zero mean and unit variance. This holds for practically any nonquadratic function G. The choice of G is important: Only a slowly growing G ensures a robust estimation. The following choices of G are common [93]:

$$G_1(u) = \frac{1}{a_1} \log \cosh a_1 u, \quad G_2(u) = -\exp(-u^2/2) \tag{8.83}$$

where $1 \leq a_1 \leq 2$ is some suitable constant.

The proposed approximation of negentropy is thus simple, easy to compute, and robust.

8.4.2.3 Minimization of Mutual Information The maximum likelihood principle leads to several contrast functions which can be derived from the Kullback divergence. The Kullback divergence is a measure of the distance between two probability density functions $p(\mathbf{x})$ and $\widehat{p}(\mathbf{x})$ on R^N and is given by

$$K(p|\widehat{p}) = \int_{\mathbf{x}} p(\mathbf{x}) \log\{\frac{p(\mathbf{x})}{\widehat{p}(\mathbf{x})}\} d\mathbf{x} \tag{8.84}$$

assuming the existence of the integral. The divergence between two random vectors $\mathbf{w}$ and $\mathbf{z}$ is represented by $K[\mathbf{w}|\mathbf{z}] \geq 0$. The equality holds if and only if $\mathbf{w}$ and $\mathbf{z}$ have the same distribution.

Although K is not symmetric and cannot be considered as a distance, it represents a statistical measure for quantifying the closeness of two distributions.

The minimization of mutual information is another approach for ICA estimation and leads to finding most non-Gaussian directions [93]. The mutual information I between m (scalar) random variables, y_i, $i = 1, \ldots, m$ is given by

$$I(y_1, y_2, \ldots, y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y}) \tag{8.85}$$

Mutual information determines the independence between random variables and has the following properties: (1) It is always non-negative, and (2) it is zero if and only if the variables are statistically independent. Thus, mutual information gives the whole statistical picture.

A very important property of mutual information [186] is given for an invertible linear transformation $\mathbf{y} = \mathbf{W}\mathbf{x}$:

$$I(y_1, y_2, \ldots, y_n) = \sum_{i} H(y_i) - H(\mathbf{x}) - \log|\det(\mathbf{W})| \tag{8.86}$$

In general, we write $H(\mathbf{y}) = H(\mathbf{W}\mathbf{x}) = H(\mathbf{x}) + \log|\det(\mathbf{W})|$ where $\det(\mathbf{W})$ is the determinant of $\mathbf{W}$.

We can easily deduce an important relation between negentropy and mutual information:

$$I(y_1, y_2, \ldots, y_n) = C - \sum_{i} J(y_i) \tag{8.87}$$

The constant C is not a function of $\mathbf{W}$ if the y_i are uncorrelated.

This leads to an alternative ICA estimation as an invertible transformation based on the fact that we have to determine $\mathbf{W}$ from eq. (8.73) by minimizing the mutual information (statistical dependence) of the transformed components s_i. Or, from eq. (8.87), we see that ICA estimation by minimization of mutual information corresponds to maximizing the sum of non-Gaussianities of the estimates, when the estimates are uncorrelated.

An interesting interpretation of mutual information is given in terms of contrast functions. The idea is to look for a contrast function that also matches the unknown system structure. In other words, the observed data should be modeled by adjusting both the unknown system and the source distribution. In other words, the divergence $K[\mathbf{y}|\mathbf{s}]$ should be minimized with respect to $\mathbf{A}$ based on the distribution of $\mathbf{y} = \mathbf{W}\mathbf{x}$ and with respect to the model distribution of $\mathbf{s}$. A basic property of the Kullback divergence is

$$K[\mathbf{y}|\mathbf{s}] = K[\mathbf{y}|\tilde{\mathbf{y}}] + K[\tilde{\mathbf{y}}|\mathbf{s}] \tag{8.88}$$

where $\tilde{\mathbf{y}}$ is a random vector with independent entries and with the same distribution as $\mathbf{y}$. $\mathbf{s}$ is a vector with independent entries. Minimization of

the above relationship is only given by the minimization of the first term $K[\mathbf{y}|\tilde{\mathbf{y}}]$ since the second term vanishes if $\mathbf{s} = \tilde{\mathbf{y}}$. In other words, the contrast function

$$\phi_{MI}[\mathbf{y}] = K[\mathbf{y}|\tilde{\mathbf{y}}] \tag{8.89}$$

has to be minimized. The Kullback divergence $K[\mathbf{y}|\tilde{\mathbf{y}}]$ between a distribution and the closest distribution with independent entries is defined as the mutual information between the entries of $\mathbf{y}$. We always have $\phi_{MI}[\mathbf{y}] \geq 0$. The equality holds, if and only if $\mathbf{y} = \mathbf{W}\mathbf{x}$ is distributed as $\tilde{\mathbf{y}}$.

8.4.2.4 Maximum Likelihood Estimation ICA estimation is also possible based on another information-theoretic principle, the maximum likelihood. From the derivation in Chapter 6, we recall that the likelihood function represents the probability density function of a data set in a given model, but is considered as a function of the unknown parameters of the model. In [192] it was shown that the log-likelihood for a noiseless ICA model is given by

$$L = \sum_{t=1}^{T} \sum_{i=1}^{n} \log p_i(\mathbf{w}_i^T \mathbf{x}(t)) + T \log |\det(\mathbf{W})| \tag{8.90}$$

where f_i are the density functions of the s_i, and $\mathbf{x}(t), t = 1, \ldots, T$, are the realizations of $\mathbf{x}$. We recall, that the pdf of $\mathbf{y} = \mathbf{W}\mathbf{x}$ is given by $p_{\mathbf{y}} = p_{\mathbf{x}}(\mathbf{W}^{-1}\mathbf{y})|\det(\mathbf{W}^{-1})|$, and based on this the above log-likelihood equation can be explained.

In [32] it was shown that the maximum likelihood principle is associated with a contrast function,

$$\phi_{ML}[\mathbf{y}] = K[\mathbf{y}|\mathbf{s}] \tag{8.91}$$

The maximum likelihood principle when applied to blind source separation states: Find a matrix $\mathbf{A}$ such that the distribution of $\mathbf{W}\mathbf{x}$ is as close as possible in terms of Kullback divergence to the hypothesized distribution of sources. From eqs. (8.88), (8.89), and (8.91), we obtain the following fundamental relationship showing the interconnection between maximum likelihood and mutual information [32]:

$$\phi_{ML}[\mathbf{y}] = \phi_{MI}[\mathbf{y}] + \sum_{i=1}^{n} K[y_i|s_i] \tag{8.92}$$

This equation states that in order to maximize the likelihood two components need to be minimized: The first component describes the deviation from independence, while the second component represents the mismatch between the source signals s_i and their estimation y_i.

8.4.2.5 The Information Maximization Estimation In [186] it was shown that the mutual information represents a measure of the uncertainty about the output of a system that is resolved by observing the system input. The Infomax principle is based on the maximization of the mutual information and produces a reduction in redundancy in the output compared to that in the input.

This idea was captured in a nonlinear neural network proposed in [20], and it was shown that the network is able to separate independent signals.

By assuming $\mathbf{x}$ is the input to the neural network whose output vector $\mathbf{y} = \mathbf{g}(\mathbf{W}^{\mathrm{T}}\mathbf{x})$ is nonlinear, and $\mathbf{W}$ is the weight matrix, ICA estimation is based on the maximization of the output vector entropy

$$H[\mathbf{g}(\mathbf{y})] = H(g_1(\mathbf{w}_1^T\mathbf{x}), \ldots, g_n(\mathbf{w}_n^T\mathbf{x})) \tag{8.93}$$

where the nonlinearities g_i are chosen as the cumulative distribution functions corresponding to the densities p_i:

$$g_i^{'}(\cdot) = p_i(\cdot) \tag{8.94}$$

This ICA estimation based on the principle of network entropy maximization, or "infomax," corresponds to maximum likelihood estimation, as we will prove by introducing a contrast function.

In [32] was determined an associated contrast function for the infomax:

$$\phi_{IM}[\mathbf{y}] = -H[\mathbf{g}(\mathbf{y})] \tag{8.95}$$

The infomax idea yields the same contrast function as the likelihood:

$$\phi_{IM}[\mathbf{y}] = \phi_{ML}[\mathbf{y}] \tag{8.96}$$

This proves the connection between maximum likelihood and infomax [32].

8.4.3 Fast ICA

The previous sections dealt with the theoretical aspects of ICA estimation. In this section, we will focus on ICA implementation issues and present a practical algorithm for maximizing the contrast function. The data must be preprocessed by centering and whitening as described in the preceding section.

8.4.3.1 Fast ICA for One Unit or One Independent Component We first consider the one-unit version. The unit can be an artificial neuron having a weight vector $\mathbf{w}$ that is updated based on a learning algorithm, such that the projection $\mathbf{w}^T\mathbf{x}$ maximizes non-Gaussianity. As a measure for non-Gaussianity the approximation of the negentropy as given by eq. (8.82) can be chosen [93].

The fast ICA algorithm is a fixed-point iteration scheme for finding a maximum of the non-gaussianity of $\mathbf{w}^T\mathbf{x}$. Let g be the derivative of the non-quadratic function used in eq. (8.82). Choices of such nonquadratic functions were given in eq. (8.83).

An algorithmic description of the fast ICA algorithm for estimating one independent component is given below:

1. **Preprocessing:** Whiten the data to give x.
2. **Initialization:** Initialize the weight vector $\mathbf{w}$ of unit norm with random numbers.
3. **Adaptation:** Compute the change of the weight vector $\mathbf{w}$ according to

$$\mathbf{w} \leftarrow E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w} \tag{8.97}$$

   where g is defined as in eq. (8.83).
4. **Normalization:** Normalize the weight vector $\mathbf{w}$

$$\mathbf{w} \leftarrow \frac{\mathbf{w}}{||\mathbf{w}||} \tag{8.98}$$

5. **Continuation:** If the weight vector $\mathbf{w}$ is not converged, go back to step 3.

The algorithm converged only if the old and new values of $\mathbf{w}$ point in the same direction, i.e., their dot product is close to 1.

The derivation of this algorithm is simple and was given in [93]. The maxima of the approximation of the negentropy of $\mathbf{w}^T\mathbf{x}$ are obtained at certain optima of $E\{G(\mathbf{w}^T\mathbf{x})\}$. The Kuhn-Tucker conditions determine the optima of $E\{G(\mathbf{w}^T\mathbf{x})\}$, under the constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = ||\mathbf{w}||^2 = 1$, as points where

$$E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w} = 0 = F(\mathbf{w}) \tag{8.99}$$

This equation can be solved by Newton's method, and we need the Jacobian matrix $JF(\mathbf{w})$:

$$JF(\mathbf{w}) = E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{I} \tag{8.100}$$

An approximation of the above is given by

$$E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} \approx E\{\mathbf{x}\mathbf{x}\}E\{g'(\mathbf{w}^T\mathbf{x})\} = E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{I} \tag{8.101}$$

It was assumed that the data are whitened.

The Jacobian matrix becomes diagonal and can be easily inverted. The resulting Newton iteration is

$$\mathbf{w} \leftarrow -\frac{E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}}{E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta} \tag{8.102}$$

This completes the proof of the fast ICA algorithm.

8.4.3.2 Fast ICA for Several Units or Several Independent Components To estimate several independent components, the one-unit fast ICA is extended for several units (neurons) with weight vectors $\mathbf{w}, \ldots, \mathbf{w}_n$. One important fact needs to be mentioned: some of the weight vectors might converge to the same maxima. To avoid this, the outputs $\mathbf{w}_1^T\mathbf{x}, \ldots, \mathbf{w}_n^T\mathbf{x}$ have to be decorrelated after every iteration. For a whitened $\mathbf{x}$ this is equivalent to orthogonalization. There are several known methods to achieve this [93]. Here, only the symmetric decorrelation is considered. It has several benefits over other techniques: (1) the weight vectors $\mathbf{w}_i$ are estimated in parallel and not sequentially and (2) it does not perpetuate the errors from one weight vector to the next.

The symmetric orthogonalization of $\mathbf{W}$ can be accomplished by involving matrix square roots

$$\mathbf{W} = (\mathbf{W}\mathbf{W}^T)^{\frac{-1}{2}}\mathbf{W} \tag{8.103}$$

Numerical simplifications of this equation are given in [93].

An algorithmic description of the Fast ICA algorithm for estimating several independent components is given below:

1. **Preprocessing:** Whiten the data to give $\mathbf{x}$.
2. **Number of independent components:** Choose m as number of independent components to estimate.
3. **Initialization:** Initialize each weight vector $\mathbf{w}_i, i = 1, \ldots, m$, of unit norm with random numbers.
4. **Adaptation:** Compute the change of every weight vector $\mathbf{w}_i, i = 1, \ldots, m$, according to

$$\mathbf{w}_i \leftarrow E\{\mathbf{x}g(\mathbf{w}_i^T\mathbf{x})\} - E\{g'(\mathbf{w}_i^T\mathbf{x})\}\mathbf{w}_i \tag{8.104}$$

   where g is defined as in eq. (8.83).
5. **Symmetric orthogonalization:** Do a symmetric orthogonalization of the matrix $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)^T$ by

$$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^T)^{\frac{-1}{2}}\mathbf{W} \tag{8.105}$$

6. **Continuation:** If the weight matrix $\mathbf{W}$ is not converged, go back to step 4.

8.4.3.3 Fast ICA and Maximum Likelihood It is interesting and useful to reveal the connection between fast ICA and the well-known infomax and maximum likelihood algorithms presented in [19].

By writing $\mathbf{y} = \mathbf{W}\mathbf{x}$, $\beta_i = E\{y_i g(y_i)\}$, and $\Gamma = \text{diag}\{\beta_i - E\{g'(y_i)\}\}^{-1}$, the fast ICA equation (8.102) can be equivalently written as

$$\mathbf{W} \leftarrow \Gamma[\text{diag}\{-\beta_i\} + E\{g(\mathbf{y})\mathbf{y}^T\}]\mathbf{W} \tag{8.106}$$

The matrix $\mathbf{W}$ needs to be orthogonalized after every step.

We can easily see the analogy between eq. (8.106) and the stochastic gradient method for maximizing likelihood [93]:

$$\mathbf{W} \leftarrow \mu[\mathbf{I} + g(\mathbf{y})\mathbf{y}^T]\mathbf{W} \tag{8.107}$$

where μ is the learning rate. Fast ICA can be considered as a fixed-point algorithm for maximum likelihood estimation of the ICA data model. The convergence of eq. (8.106) can be controlled by the adequate choice of the matrices Γ and $\text{diag}\{-\beta_i\}$. Another advantage over the ordinary maximum likelihood is the estimation of both super- and sub-Gaussian independent components.

8.4.3.4 Properties of the Fast ICA Algorithm The presented fast ICA and the chosen contrast functions enjoy several practical properties [93] compared with other ICA estimation methods:

- Fast convergence: Convergence is cubic compared to ordinary ICA methods where it is linear.
- Simplicity: No learning parameter is needed.
- Generalization: Finds, based on nonlinearity of choice, g directly independent components of any non-Gaussian distribution.
- Optimized performance: Based on the choice of a suitable nonlinearity g.
- Parallel processing: Learning algorithm is parallel, distributed, computationally simple, and requires little memory space.

8.4.4 The Infomax Neural Network

In [20] a self-organizing learning algorithm is described that maximizes the information transferred in a network of nonlinear units. It was shown that the neural network is able to perform ICA estimation and that higher order statistics are introduced by the nonlinearities in the transfer function.

The network architecture is shown in Figure 8.13.

The network has N input and output neurons, and an $N \times N$ weight matrix $\mathbf{W}$ connecting the input layer neurons with the output layer neurons. Assuming sigmoidal units, the neuron outputs are given by

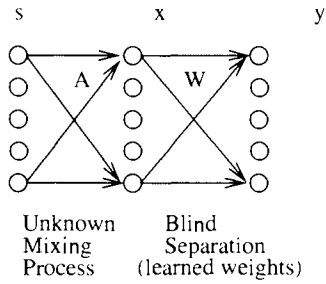


FIGURE 8.13 Feedforward neural network for ICA [20].

$$\mathbf{y} = g(\mathbf{u}), \quad \text{with} \quad \mathbf{u} = \mathbf{W}\mathbf{x} \tag{8.108}$$

where g is a logistic function $g(u_i) = \frac{1}{1+\exp(-u_i)}$.

The idea of this algorithm is to find an optimal weight matrix $\mathbf{W}$ such that the output entropy $H(\mathbf{y})$ is maximized. The algorithm initializes $\mathbf{W}$ to the identity matrix $\mathbf{I}$. The elements of $\mathbf{W}$ are updated based on the following rule

$$\mathbf{W} \leftarrow -\eta(\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}})\mathbf{W}^T\mathbf{W} = -\eta(\mathbf{I} + f(\mathbf{u})\mathbf{u}^T)\mathbf{W} \tag{8.109}$$

where η is the learning rate. The term $\mathbf{W}^T\mathbf{W}$ in eq. (8.109) was first proposed in [8] to avoid avoid matrix inversions and to improve convergence. The vector-function f has the elements

$$f_i(u_i) = \frac{\partial}{\partial u_i} \ln g_i'(u_i) = (1 - 2y_i) \tag{8.110}$$

During learning, the learning rate is reduced gradually until the weight matrix $\mathbf{W}$ stops changing considerably. Equation (8.109) represents the so-called "infomax" algorithm.

An algorithmic description of the infomax ICA algorithm is given below:

1. **Preprocessing:** Whiten the data to give $\mathbf{x}$.
2. **Number of independent components:** Choose m as the number of independent components to estimate.
3. **Initialization:** Initialize each the weight matrix $\mathbf{W}$ to the identity matrix $\mathbf{I}$.
4. **Adaptation:** Compute the change of the weight matrix $\mathbf{W}$ according to

$$\mathbf{W} \leftarrow -\eta(\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}})\mathbf{W}^T\mathbf{W} = -\eta(\mathbf{I} + f(\mathbf{u})\mathbf{u}^T)\mathbf{W} \tag{8.111}$$

where f is defined as in eq. (8.110). Reduce the learning rate gradually.

5. **Continuation:** If the weight matrix $\mathbf{W}$ is not converged, go back to step 4.

8.4.5 Topographic Independent Component Analysis

The topographic independent component analysis [92] represents a unifying model which combines topographic mapping with ICA.

Achieved by a slight modification of the ICA model, it can at the same time be used to define a topographic order between the components, and thus has the usual computational advantages associated with topographic maps.

The paradigm of topographic ICA has its roots in [91] where a combination of invariant feature subspaces [116] and independent subspaces [33] is proposed. In the following, we will describe these two parts, which substantially reflect the concept of topographic ICA.

8.4.5.1 Invariant Feature Subspaces The principle of invariant-feature subspaces was developed by Kohonen [116] with the intention of representing features with some invariances. This principle states that an invariant feature is given by a linear subspace in a feature space. The value of the invariant feature is given by the squared norm of the projection of the given data point on that subspace.

A feature subspace can be described by a set of orthogonal basis vectors $\mathbf{w_j}, j = 1, \ldots, n$, where n is the dimension of the subspace. Then the value $G(\mathbf{x})$ of the feature G with the input vector $\mathbf{x}$ is given by

$$G(\mathbf{x}) = \sum_{j=1}^{n} < \mathbf{w_j}, \mathbf{x} >^2 \tag{8.112}$$

In other words, this describes the distance between the input vector $\mathbf{x}$ and a general linear combination of the basis vectors $\mathbf{w_j}$ of the feature subspace [116].

8.4.5.2 Independent Subspaces Traditional ICA works under the assumption that the observed signals $x_i(t), (i = 1, \ldots, n)$ are generated by a linear weighting of a set of n statistically independent random sources $s_j(t)$ with time-independent coefficients a_{ij}. In matrix form, this can be expressed as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{8.113}$$

where $\mathbf{x}(t) = [x_1(t), \ldots, x_n(t)]^T, \mathbf{s}(t) = [s_1(t), \ldots, s_n(t)]$, and $\mathbf{A} = [a_{ij}]$.

In multidimensional ICA [33], the sources s_i are not assumed to be all mutually independent. Instead, it is assumed that they can be grouped in n-tuples, such that within these tuples they are dependent on each other,

but are independent outside. This newly introduced assumption was observed in several image processing applications. Each n-tuple of sources s_i corresponds to n basis vectors given by the rows of matrix $\mathbf{A}$. A subspace spanned by a set of n such basis vectors is defined as an independent subspace. In [33] two simplifying assumptions are made: (1) Although s_i are not at all independent, they are chosen to be uncorrelated and of unit variance, and (2) the data are preprocessed by whitening (sphering) them. This means the $\mathbf{w_j}$ are orthonormal.

Let J be the number of independent feature subspaces and $S_j, j = 1, \ldots, J$ the set of indices that belong to the subspace of index j. Assume that we have T given observations $\mathbf{x}(t), t = 1 \ldots, T$. Then the likelihood L of the data based on the model is given by

$$L(\mathbf{w_i}, i = 1, \ldots, n) = \prod_{t=1}^{T} [|\det \mathbf{W}| \prod_{j=1}^{J} p_j(< \mathbf{w_i}, \mathbf{x}(t) >, i \in S_j)] \quad (8.114)$$

with $p_j(\cdot)$ being the probability density inside the jth n-tuple of s_i. The expression $|\det \mathbf{W}|$ is due to the linear transformation of the pdf. As always with ICA, $p_j(\cdot)$ need not be known in advance.

8.4.5.3 Fusion of Invariant Feature and Independent Subspaces In [91] it is shown that a fusion between the concepts of invariant and independent subspaces can be achieved by considering probability distributions for the n-tuples of s_i being spherically symmetric, that is, depending on the norm. In other words, the pdf $p_j(\cdot)$ has to be expressed as a function of the sum of the squares of the $s_i, i \in S_j$ only. Additionally, it is assumed that the pdfs are equal for all subspaces.

The log-likelihood of this new data model is given by

$$\log L(\mathbf{w_i}, i = 1, \ldots, n) = \sum_{t=1}^{T} \sum_{j=1}^{J} \log p(\sum_{i \in S_j} < \mathbf{w_i}, \mathbf{x}(t) >^2) + T \log |\det \mathbf{W}| \quad (8.115)$$

$p(\sum_{i \in S_j} s_i^2) = p_j(s_i, i \in S_j)$ gives the pdf inside the jth n-tuple of s_i. Based on the prewhitening, we have $\log |\det \mathbf{W}| = 0$.

For computational simplification, set

$$G(\sum_{i \in S_j} s_i^2) = \log p(\sum_{i \in S_j} < \mathbf{w_i}, \mathbf{x}(t) >^2) \quad (8.116)$$

Since it is known that the projection of visual data on any subspace has a super-Gaussian distribution, the pdf has to be chosen to be sparse. Thus, we will choose $G(u) = \alpha\sqrt{u} + \beta$ yielding a multidimensional version of an exponential distribution. α and β are constants and enforce that s_i is of unit variance.

8.4.5.4 The Topographic ICA Architecture Based on the concepts introduced in the preliminary subsections, this section describes the topographic ICA.

To introduce a topographic representation in the ICA model, it is necessary to relax the assumption of independence among neighboring components s_i. This makes it necessary to adopt an idea from self-organized neural networks, that of a lattice. It was shown in [92] that a representation which models topographic correlation of energies is an adequate approach for introducing dependencies between neighboring components.

In other words, the variances corresponding to neighboring components are positively correlated while the other variances are in a broad sense independent. The architecture of this new approach is shown in Figure 8.14.

This idea leads to the following representation of the source signals:

$$s_i = \sigma_i z_i \tag{8.117}$$

where z_i is a random variable having the same distribution as s_i, and the variance σ_i is fixed to unity.

The variance σ_i is further modeled by a nonlinearity:

$$\sigma_i = \phi\left(\sum_{k=1}^{n} h(i,k)u_k\right) \tag{8.118}$$

where u_i are the higher order independent components used to generate the variances, while ϕ describes some nonlinearity. The neighborhood function $h(i,j)$ can either be a two-dimensional grid or have a ring-like structure. Further u_i and z_i are all mutually independent.

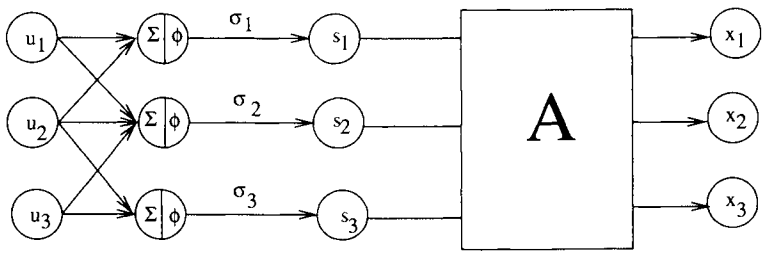


FIGURE 8.14 Topographic ICA model [92]. The variance generated variables u_i are randomly generated and mixed linearly inside their topographic neighborhoods. This forms the input to nonlinearity ϕ, thus giving the local variance σ_i. Components s_i are generated with variances σ_i. The observed variables x_i are obtained as with standard ICA from the linear mixture of the components s_i.

The learning rule is based on the maximization of the likelihood. First, it is assumed that the data are preprocessed by whitening and that the

estimates of the components are uncorrelated. The log-likelihood is given by:

$$\log L(\mathbf{w_i}, i = 1, \ldots, n) = \sum_{t=1}^{T} \sum_{j=1}^{n} G(\sum_{i=1}^{n} (\mathbf{w_i}^T \mathbf{x}(t)^2) + T \log |\det \mathbf{W}| \quad (8.119)$$

The update rule for the weight vector $\mathbf{w}_i$ is derived from a gradient algorithm based on the log-likelihood assuming $\log |\det \mathbf{W}| = 0$:

$$\Delta \mathbf{w}_i \propto E\{\mathbf{x}(\mathbf{w}_i^T \mathbf{x}) r_i\} \quad (8.120)$$

where

$$r_i = \sum_{k=1}^{n} h(i,k) g(\sum_{j=1}^{n} h(k,j)(\mathbf{w}_j^T \mathbf{x})^2) \quad (8.121)$$

The function g is the derivative of $G = -\alpha_1 \sqrt{u} + \beta_1$. After every iteration, the vectors $\mathbf{w}_i$ in eq. (8.120) are normalized to unit variance and orthogonalized. This equation represents a modulated learning rule, where the learning term is modulated by the term r_i.

The classic ICA results from the topographic ICA by setting $h(i,j) = \delta_{ij}$.

8.4.6 Imaging Brain Dynamics

The analysis of magnetoencephalographic (MEG) recordings is important both for basic brain research and for medical diagnosis and treatment. ICA is an effective method for removing artifacts and separating sources of the brain signals from these recordings [236]. A similar approach is proving useful for analyzing functional magnetic resonance brain imaging (fMRI) data [157, 158].

8.4.6.1 Separation of Artifacts in MEG Data MEG is a technique to measure the activity of the cortical neurons. However, it is difficult to extract the essential features of the neuromagnetic signals from that of artifacts generally resembling pathological signals. In [236], a new method for separating brain activity from artifacts based on ICA is presented. The approach is based on the assumption that the brain activity and the artifacts, e.g., eye movements or blinks, are anatomically and physiologically separate processes and statistically independent signals.

Figure 8.15 shows a subset of 12 spontaneous MEG signals $x_i(t)$ from the frontal, temporal, and occipital areas. The figure also shows the helmet positions of the sensors. The considered artifacts were ocular (blinking and horizontal saccades) and muscular (biting) in nature.

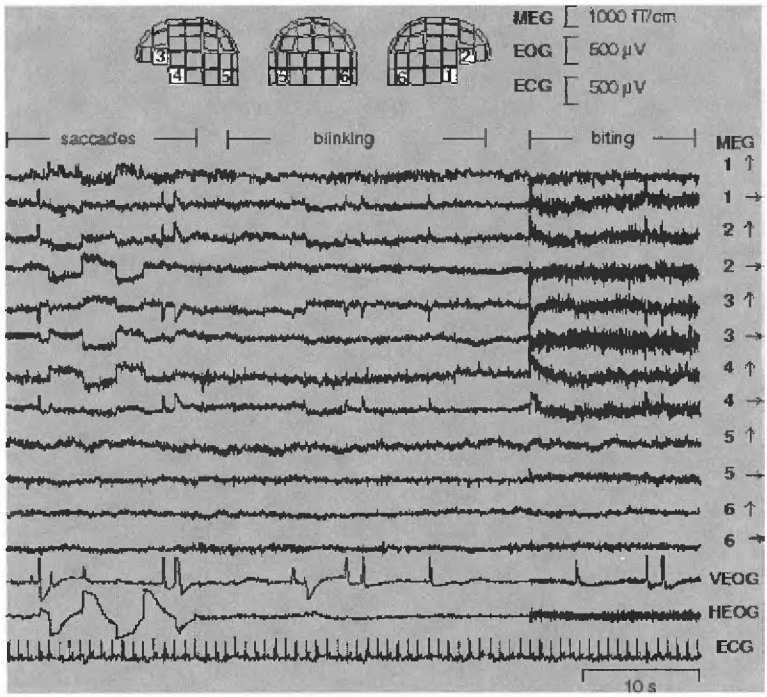


FIGURE 8.15 Samples of MEG signals, showing artifacts produced by blinking, saccades, biting, and cardiac cycle. For each of the six positions shown, the two orthogonal directions of the sensors are plotted. (Image from [236] reprinted with permission from The MIT Press.)

The signal vector $\mathbf{x}$ in the ICA model (8.63) consists of the amplitude $x_i(t)$ of the recorded signals at a certain time point. The $\mathbf{x}(t)$ vectors were whitened using PCA and the dimensionality was reduced. Using fast ICA, it became possible to determine the independent components. Figure 8.16 shows sections of nine independent components (IC) $s_i(t), i = 1, \ldots, 9$ found from the recorded data together with the corresponding field pattern. The following ICs were determined: muscular activity (IC1,IC2), ocular activity (IC3,IC5), cardiac (IC4), and others (IC8,IC9).

The results from Figure 8.16 show that based on the fast ICA algorithm the artifacts can be reliably separated from other MEG signals.

The separation results achieved based on topographic ICA are very interesting. Figure 8.17 shows the resulting separated signals from an original 122-dimensional input data reduced to 20 dimensions by PCA. A one-dimensional ring-shaped topography was employed. For the topographic organization, we can see that the signals corresponding to bites (IC9-IC15) and also those corresponding to eye artifacts (IC18, IC19) are adjacent. In

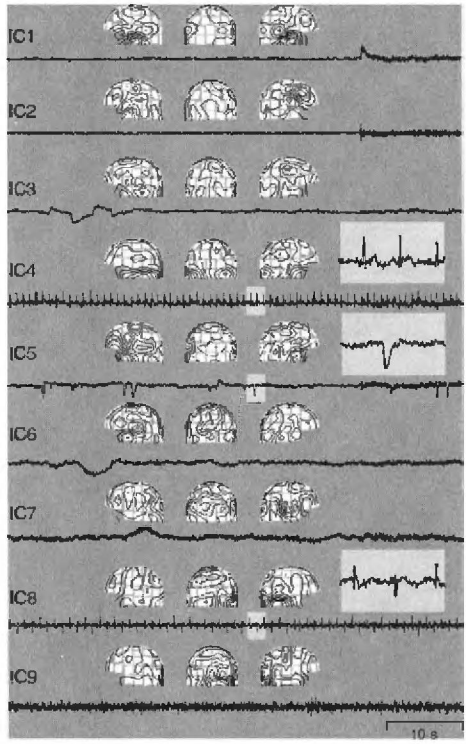


FIGURE 8.16 Nine independent components found from MEG data. For each component the left, back, and right views of the field patterns generated by these components are shown: the solid line stands for magnetic flux coming out of the head, and the dotted line for the flux inward. (Image from [236] reprinted with permission from The MIT Press.)

other words, topographic ICA orders the signals mainly into two clusters, one created by the signals coming from the muscle artifact and the other by eye muscle activity.

8.4.6.2 Separation of Artifacts in fMRI Data Functional magnetic resonance imaging with high temporal and spatial resolution represents a powerful technique for visualizing rapid and fine activation patterns of the human brain [15, 66, 123, 124, 173]. As is known from both theoretical estimations and experimental results [24, 124, 172], an activated signal variation appears very low on a clinical scanner. This motivates the application of analysis methods to determine the response waveforms and associated activated regions. Generally, these techniques can be divided into two groups: Model-based techniques require prior knowledge about activation patterns, whereas model-free techniques do not. However, model-

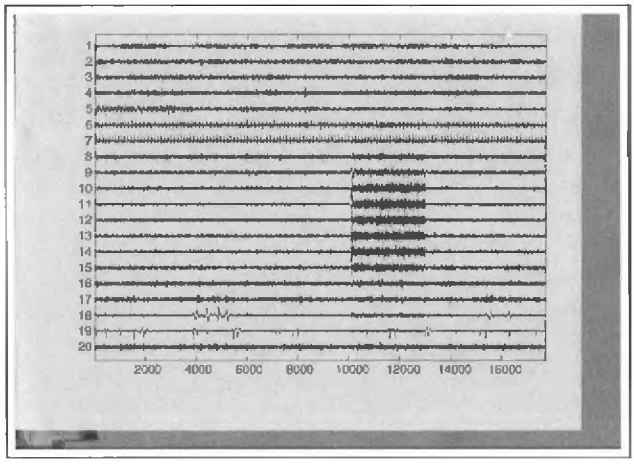


FIGURE 8.17 Twenty independent components found from MEG data. The separated source signals are found by topographic ICA. (Reprinted from [92] with permission from Springer-Verlag.)

based analysis methods impose some limitations on data analysis under complicated experimental conditions. Therefore, analysis methods that do not rely on any assumed model of functional response are considered more powerful and relevant. We distinguish two groups of model-free methods: transformation-based and clustering-based. The first method, principal component analysis (PCA) [227, 14] or independent component analysis (ICA) [158, 157], transforms raw data into high-dimensional vector space to separate functional response and artifacts from each other. The major problem of PCA is that it only decorrelates the components. ICA can separate the components, but is still limited by stationary distribution and the linear mixture assumption [157].

The second method, fuzzy clustering analysis (FCA) [42, 215] or self-organizing map (SOM) [169, 64], attempts to group time signals of the brain into several clusters based on the temporal similarity among them.

Functional organization of the brain is based on two complementary principles, localization and connectionism. Localization means that each visual function is performed mainly by a small set of the cortex. Connectionism, on the other hand, means that the brain regions involved in a certain visual-cortex function are widely distributed, and thus the brain activity necessary to perform a given task may be the functional integration of activity in distinct brain systems. It is important to stress that in neurobiology the term "connectionism" is used in a different sense than in neural network terminology.

According to the principle of functional organization of the brain, it was suggested for the first time in [157] that the multifocal brain areas acti-

vated by performance of a visual task should be unrelated to the brain areas whose signals are affected by artifacts of physiological nature, head movements, or scanner noise related to fMRI experiments. Every single process mentioned earlier can be described by one or more spatially independent components, each associated with a single time course of a voxel and a component map. It is assumed that the component maps, each described by a spatial distribution of fixed values, represent overlapping, multifocal brain areas of statistically dependent fMRI signals. This aspect is visualized in Figure 8.18. In addition, it is considered that the distributions of the component maps are spatially independent. and in this sense uniquely specified.

It was shown in [157] that these maps are independent if the active voxels in the maps are sparse and mostly nonoverlapping. Additionally it is assumed that the observed fMRI signals are the superposition of the individual component processes at each voxel. Based on these assumptions, ICA can be applied to fMRI time series to spatially localize and temporally characterize the sources of BOLD activation.

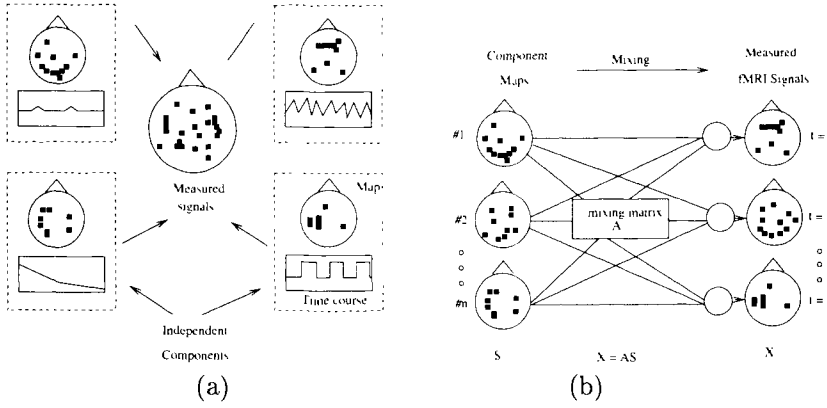


FIGURE 8.18 Visualization of ICA applied to fMRI data. (a) Scheme of fMRI data decomposed into independent components, and (b) fMRI data as a mixture of independent components where the mixing matrix **M** specifies the relative contribution of each component at each time point [157].

In [157] a new method for analyzing fMRI data based on the infomax ICA algorithm [20] is described. The fMRI data were acquired while a subject performed a Stroop color-naming task, and the data were then decomposed into spatially independent components. Each component consisted of voxel values at fixed three-dimensional locations (component "map"), and a unique associated time course of activation.

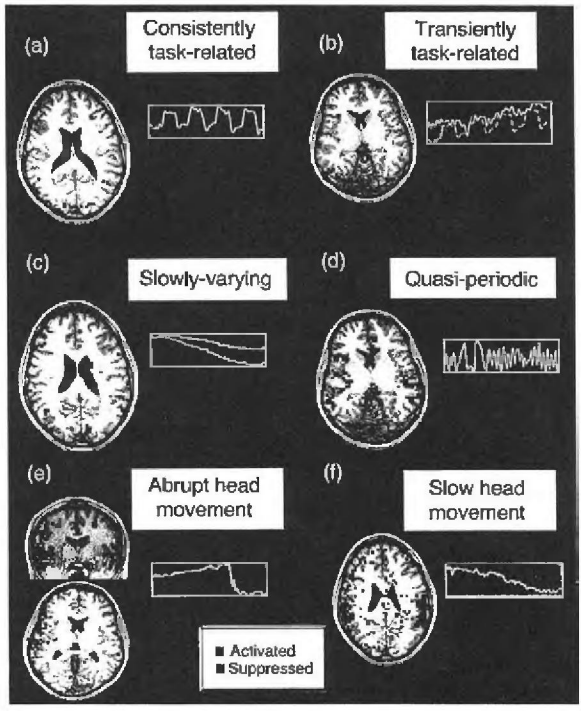


FIGURE 8.19 Different classes of components detected by ICA decomposition of Stroop task fMRI data. (Reprinted from [158] with permission from National Academy of Sciences.)

ICA derived one and only one component with a time course closely matching the time course of alternations between experimental and control tasks.

Time courses of other IC were transiently task-related, quasiperiodic, or slowly varying. Figure 8.19 illustrates the different classes of IC components.

The results in Figure 8.19 show that ICA can be used to reliably separate fMRI data sets into meaningful constituent components.

In [181] a comparison is shown between different ICA techniques and PCA when applied to fMRI. For a visual task, unique task-related activation maps and associated time courses were obtained by different ICA techniques and PCA.

Figure 8.20 shows, for eight ICs, the component time course most closely associated with the visual task for three ICA techniques (infomax, fastICA, topographic ICA) and PCA. The best results are achieved by fastICA and topographic ICA, yielding almost identical reference functions,

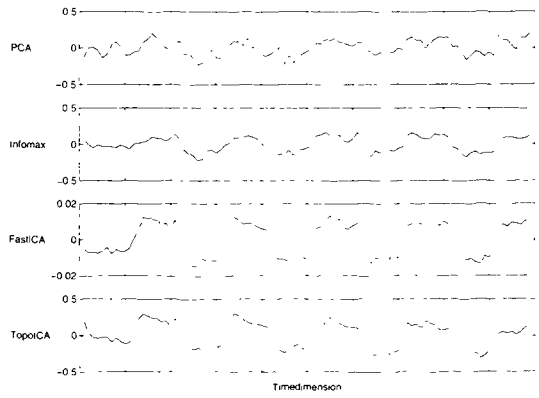


FIGURE 8.20 Computed reference functions for the four techniques, PCA, infoma x, fastICA, and topographic ICA, for 16 ICs. The correlation coefficients found are r=0.69 for PCA, r=0.85 for infomax, r=0.94 for fast ICA, and r=0.85 for topographic ICA.

and a correlation coefficient of $r = 0.92$ between those two. PCA has a much lower correlation as does infomax.

However, the activation maps do not reveal much of a difference between the ICA techniques. Figure 8.21 shows the activation maps as a comparison of results obtained by the four techniques.

It has been shown that both topographic ICA and fast ICA outperform, in terms of component time course found and activation maps, principal component analysis and infomax neural networks when applied to fMRI studies. This supports the theory that different methods for performing ICA decompositions employing different objective functions in conjunction with different criteria of optimization of these functions produce different results.

EXERCISES

1. Find the singular value decomposition of the matrix

$$X = \begin{bmatrix} 4 & 4 \\ 0 & 2 \\ 3 & 0 \\ 0 & 5 \end{bmatrix} \tag{8.122}$$

2. Determine the eigenvalues of the matrix

$$X = \begin{bmatrix} 1 & 3 \\ 4 & 2 \\ 1 & 3 \end{bmatrix} \tag{8.123}$$

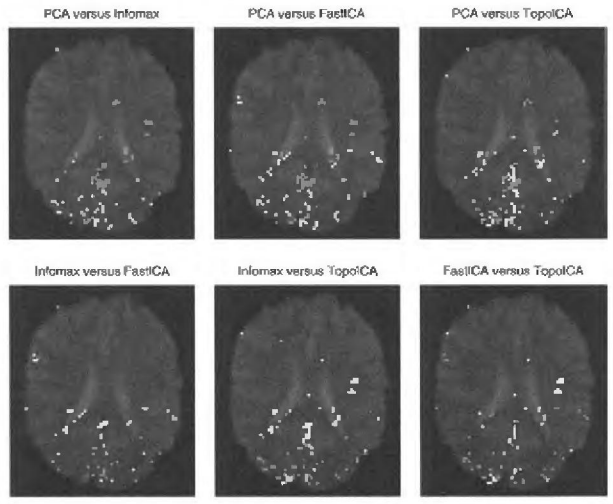


FIGURE 8.21 Comparison of the results for the four techniques, PCA, infomax, FastICA, and topographic ICA. The activation maps show only the pairwise differences. The darker pixels show active voxels for the first technique: while the lighter pixels show the active voxels for the second technique.

Give the singular value decomposition of X.

3. Comment on the capability of PCA as an image compression mechanism. What are the strengths and weaknesses of each of the following robust PCA algorithms: Oja, NLPCA, and GHA?
4. Discuss the differences between GHA and the APEX algorithm. Why is the asymptotic convergence of APEX in theory exactly the same as that of GHA? Explain this intuitively!
5. Show that the following equality $\Phi_{MI}[\mathbf{y}] > 0$ holds, if and only if $\mathbf{y} = \mathbf{W}\mathbf{x}$ is statistically distributed as $\tilde{\mathbf{y}}$. $\Phi_{MI}[\mathbf{y}] > 0$ is given by eq. (8.68).
6. Show that the maximum likelihood principle is associated with the contrast function $\Phi_{[}ML] = K[\mathbf{y}|\mathbf{s}]$; see eq. (8.74).
7. Show that the infomax and the maximum likelihood contrast function have the same contrast function $\Phi_{IM}[\mathbf{y}] = \Phi_{ML}[\mathbf{y}]$, as shown in eq. (8.96).
8. Discuss the algorithmic differences between infomax and fast ICA.
9. Compare topographical ICA with other topographic mappings.

IX

NEURO-FUZZY CLASSIFICATION

9.1 INTRODUCTION

In a medical imaging system, uncertainties can be present at any point resulting from incomplete or imprecise input information, ambiguity in medical images, ill-defined or overlapping boundaries among the disease classes or regions, and indefiniteness in extracting features and relations among them. Any decision taken at a particular point will heavily influence the following stages. Therefore, an image analysis system must have sufficient possibilities to capture the uncertainties involved at every stage, such that the system's output results can be described with minimal uncertainty. In pattern recognition parlance, a pattern can be assigned to more than one class. Or in medical diagnosis, a patient can have a set of symptoms, which can be attributed to several diseases. The symptoms need not be strictly numerical. Thus, fuzzy variables can be both linguistic and/or set variables. An example of a fuzzy variable is a temperature ranging from 97°F to 103°F, which can be described as very high, or more or less high. This gives an idea of the concept of fuzziness.

The main difference between fuzzy and neural paradigms is that fuzzy set theory tries to mimic the human reasoning and thought process, whereas

neural networks attempt to emulate the architecture and information representation scheme of the human brain. It is therefore meaningful to integrate these two distinct paradigms by enhancing their individual capabilities in order to build a more intelligent processing system. This new processing paradigm is known under the name of neuro-fuzzy computing. Compared to standard neural networks or simple fuzzy classifiers, it is a powerful computational paradigm: A neuro-fuzzy system can process linguistic variables compared to a classical neural network, or it learns overlapping classes better than a fuzzy classifier.

This chapter reviews the basic concepts of fuzzy sets and the definitions needed for fuzzy clustering, and it presents several of the best-known fuzzy clustering algorithms and fuzzy learning vector quantization. Applications of neuro-fuzzy classification to medical image compression and exploratory data analysis are also shown.

9.2 FUZZY SETS

Fuzzy sets represent a suitable mathematical technique for the description of imprecision and vagueness. Usually, vagueness describes the difficulty of obtaining concise affirmations regarding a given domain. On the other hand, in fuzzy set theory, the crisp alternative yes-no can be endlessly expanded. Thus, fuzzy set theory not only deals with ambiguity and vagueness, but represents a theory of nuance reasoning [60].

This chapter will review some of the basic notions and results in fuzzy set theory.

The two basic components of fuzzy systems are fuzzy sets and operations on fuzzy sets. Fuzzy logic defines rules, based on combinations of fuzzy sets by these operations. The notion of fuzzy sets has been introduced by Zadeh [250].

DEFINITION 9.2.1: Let X be a non-empty set considered to be the universe of discourse. A crisp set A is defined by enumerating all elements $x \in X$

$$A = \{x_1, x_2, \ldots, x_n\} \tag{9.1}$$

that belong to A.

The membership function can be expressed by a function u_A, mapping X on a binary value described by the set $I = \{0, 1\}$:

$$u_A : X \to I, \quad u_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases} \tag{9.2}$$

$u_A(x)$ represents the membership degree of x to A. Thus, an arbitrary x either belongs to A, or it does not; partial membership is not allowed.

For two sets A and B, combinations can be defined by the following operations:

$$\{A \cup B = \{x | x \in A \quad \text{or} \quad x \in B\} \tag{9.3}$$

$$A \cap B = \{x | x \in A \quad \text{and} \quad x \in B\} \tag{9.4}$$

$$\bar{A} = \{x | x \notin A, x \in X\} \tag{9.5}$$

Additionally, the following rules have to be satisfied:

$$A \cup \bar{A} = \varnothing, \text{and} \quad A \cap \bar{A} = X \tag{9.6}$$

DEFINITION 9.2.2: Let X be a non-empty set considered to be the universe of discourse. A fuzzy set is a pair $(X. A)$, where $u_A : X \to I$ and $I = [0, 1]$.

Figure 9.1 illustrates an example of a possible membership function.

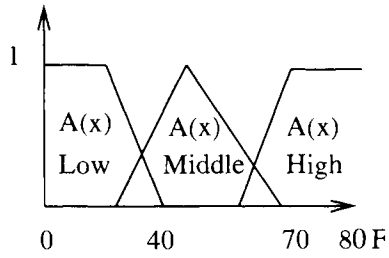


FIGURE 9.1 Example of a membership function of temperature.

The family of all fuzzy sets on the universe x will be denoted by $L(X)$. Thus

$$L(X) = \{u_A | u_A : X \to I\} \tag{9.7}$$

$u_A(x)$ is the membership degree of x to A. It can be also interpreted as the plausibility degree of the affirmation "x belongs to A." If $u_A(x) = 0$, x is definitely not in A, and if $u_A(x) = 1$, x is definitely in A. The intermediate cases are fuzzy.

DEFINITION 9.2.3: The fuzzy set A is called non-ambiguous or crisp, if $u_A(x) \in \{0, 1\}$.

DEFINITION 9.2.4: If A is from $L(X)$, the complement of A is the fuzzy set $\bar{A}$ defined as

$$u_{\bar{A}}(x) = 1 - u_A(x), \forall x \in X \tag{9.8}$$

In order to manipulate fuzzy sets, we need to have operations that enable us to combine them. As fuzzy sets are defined by membership functions, the classical set-theoretic operations have to be replaced by function-theoretic operations.

For two fuzzy sets A and B on X, the following operations can be defined.

DEFINITION 9.2.5:

$$\begin{array}{lcl} \text{Equality } A & = & B \Longleftrightarrow u_A(x) = u_B(x), \forall x \in X \\ \text{Inclusion } A & \subseteq & B \Longleftrightarrow u_A(x) \leq u_B(x), \forall x \in X \\ \text{Product } u_{(AB)}(x) & = & u_A(x) \cdot u_B(x), \forall x \in X \\ \text{Difference } u_{(A-B)}(x) & = & \max(u_A(x) - u_B(x), 0), \forall x \in X \\ \text{Intersection } u_{(A \cap B)}(x) & = & \min(u_A(x), u_B(x)), \forall x \in X \\ \text{Union } u_{(A \cup B)}(x) & = & \max(u_A(x), u_B(x)), \forall x \in X \end{array} \tag{9.9}$$

DEFINITION 9.2.6: The family $A_1, \ldots, A_n, n \geq 2$, of fuzzy sets is a fuzzy partition of the universe X if and only if the condition

$$\sum_{i=1}^{n} u_{A_i}(x) = 1 \tag{9.10}$$

holds for every x from X.

The above condition can be generalized for a fuzzy partition of a fuzzy set. By C we define a fuzzy set on X. We may require that the family $A_1, \ldots, A_n$ of fuzzy sets be a fuzzy partition of C if and only if the condition

$$\sum_{i=1}^{n} u_{A_i}(x) = u_C(x) \tag{9.11}$$

is satisfied for every x from X.

9.3 NEURO-FUZZY INTEGRATION

Neural networks and fuzzy systems are both learning systems aiming to estimate input-output relationships. They are not based on a mathematical model, and learning incorporates experience with sample data. The fuzzy system infers adaptively and modifies its fuzzy associations from representative numerical samples. Neural networks generate blindly and change fuzzy rules based on training data [120]. Fuzzy systems estimate functions with fuzzy-set samples (A_i, B_i), whereas neural networks use numerical-point samples (x_i, y_i). Both kinds of samples are given by the input-output product space $X \times Y$. Figure 9.2 illustrates the geometry of fuzzy-set and numerical-point samples taken from the function $f : X \to Y$ [120].

Thus, neuro-fuzzy algorithms maintain the basic properties and architectures of neural networks and simply fuzzify some of their components.

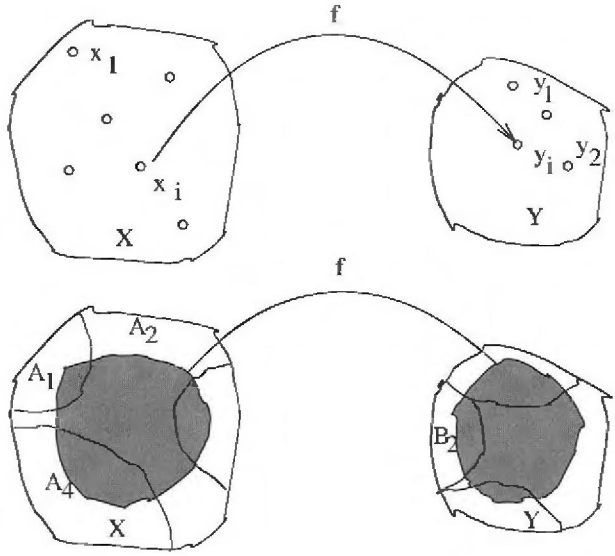


FIGURE 9.2 Function f maps X to Y. In the first illustration several numerical-point samples (x_i, y_i) are used for the function estimation f: $X \rightarrow Y$. In the second illustration only a few fuzzy subsets A_i of X and B_i of Y are used. The fuzzy association (A_i, B_i) describes the system structure as an adaptive clustering might infer. In practice, there are usually fewer different output associants or "rule" consequents B_i than input associants or antecedents A_i [120].

As shown in [185], there are several known techniques of combining neuro-fuzzy concepts. Neuro-fuzzy hybridization is possible in many ways as shown in Figures 9.3 to 9.7:

- Incorporating fuzziness into the neural net framework: for example, building fuzzy neural network classifiers
- Designing neural networks guided by fuzzy logic formalism: for example a computational task within the framework of a preexisting fuzzy model
- Changing the basic characteristics of the neurons: for example, when each node performs a fuzzy aggregation operation (fuzzy union, intersection)
- Using measures of fuzziness as the error or instability of a network: for example, incorporating various fuzziness measures in a multilayer network, thus performing unsupervised self-organization

- Making the individual neurons fuzzy: the input and output of the neurons are fuzzy sets, and the activity involving the fuzzy neurons is also a fuzzy process

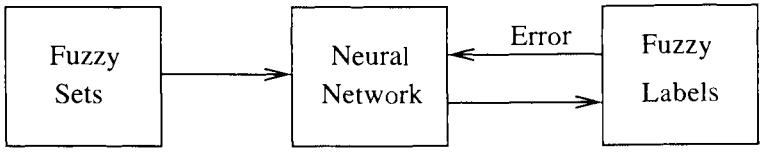


FIGURE 9.3 Neuronal network implementing fuzzy classifier.

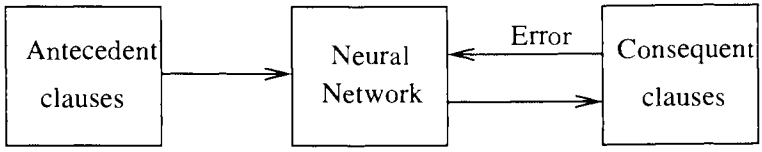


FIGURE 9.4 Neuronal network implementing fuzzy logic.

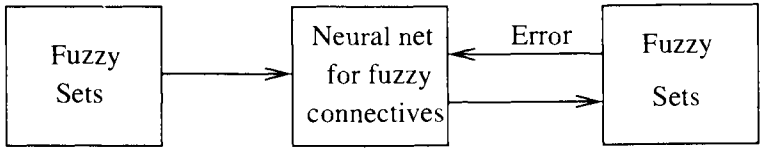


FIGURE 9.5 Neuronal network implementing fuzzy connectives.

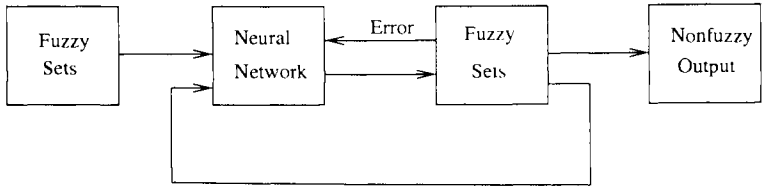


FIGURE 9.6 Neuronal network implementing self-organization.

9.4 MATHEMATICAL FORMULATION OF A FUZZY NEURAL NETWORK

A fuzzy neural network can include fuzziness in many different places such as at the input-output level, in the synapses, in the confluence operation, or

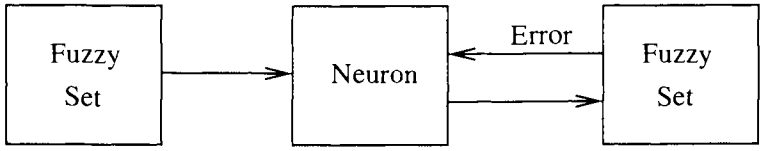


FIGURE 9.7 Block diagram of a fuzzy neuron.

even in the activation function. To give a precise mathematical formulation of the fuzzy neural network, we will adopt here the terminology used in [185]. $\mathbf{x}$ is the fuzzy input and $\mathbf{y}$ is the fuzzy output vector both being fuzzy numbers or intervals. The connection weight vector is denoted by $\mathbf{W}$.

Mathematically, we can define the following mapping from the n-dimensional input space to the l-dimensional space:

$$\mathbf{x}(t) \in \mathbf{R}^{\mathbf{n}} \rightarrow \mathbf{y}(t) \in \mathbf{R}^{\mathbf{l}} \tag{9.12}$$

A confluence operation $\otimes$ determines the similarity between the fuzzy input vector $\mathbf{x}(t)$ and the connection weight vector $\mathbf{W}(t)$. For neural networks, the confluence operation represents a summation or product operation, whereas for a fuzzy neural network it describes an arithmetic operation such as fuzzy addition and fuzzy multiplication.

The output neurons implement the following nonlinear operation:

$$\mathbf{y}(t) = \psi[\mathbf{W}(t) \otimes \mathbf{x}(t)] \tag{9.13}$$

Based on the given training data $\{(\mathbf{x}(t), \mathbf{d}(t)), \mathbf{x}(t) \in \mathbf{R}^{\mathbf{n}}, \mathbf{d}(t) \in \mathbf{R}^{\mathbf{l}}, t = 1, \ldots, N\}$ the cost function can be optimized:

$$E_N = \sum_{t=1}^{N} d(\mathbf{y}(t), \mathbf{d}(t)) \tag{9.14}$$

where $d(\cdot)$ defines a distance in $\mathbf{R}^{\mathbf{l}}$.

The learning algorithm of the fuzzy neural network is given by

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \epsilon \Delta \mathbf{W}(t) \tag{9.15}$$

and thus adjusts $N^{\mathbf{W}}$ connection weights of the fuzzy neural network.

9.5 FUZZY CLUSTERING

Traditional statistical classifiers assume that the pdf for each class is known or must somehow be estimated. Another problem is posed by the fact that sometimes clusters are not compact but shell-shaped. A solution to

this problem is given by fuzzy clustering algorithms, a new classification paradigm intensively studied during the past three decades. The main difference between traditional statistical classification techniques and fuzzy clustering techniques is that in the fuzzy approaches an input vector belongs *simultaneously* to more than one cluster, while in statistical approaches it belongs *exclusively* to only one cluster.

Usually, clustering techniques are based on the optimization of a cost or objective function J. This predefined measure J is a function of the input data and of an unknown parameter vector set $\mathbf{L}$. Throughout this chapter, we will assume that the number of clusters n is predefined and fixed.

A successful classification is based on estimating the parameter $\mathbf{L}$ such that the cluster structure of the input data is as good as possible determined. It is evident that this parameter depends on the cluster's geometry. We distinguish two different types of geometry: compact and spherical, as shown in Figure 9.8.

FIGURE 9.8 Different cluster shapes: (a) compact, and (b) spherical.

Compact clusters are pretty well described by a set of n points $\mathbf{L}_i \in \mathbf{L}$ where each point describes such a cluster. Spherical clusters have two distinct parameters describing the center and the radius of the cluster. Thus the parameter vector set $\mathbf{L}$ is replaced by two new parameter vector sets, $\mathbf{V}$ describing the centers of the clusters, and $\mathbf{R}$ describing the radii of the clusters.

In the following, we will review the most important fuzzy clustering techniques, and show their relationship to nonfuzzy approaches.

9.5.1 Metric Concepts for Fuzzy Classes

Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\}, \mathbf{x}_j \in \mathbf{R}^s$. be a data set. Suppose the optimal number of clusters in $\mathbf{X}$ is given and that the cluster structure of $\mathbf{X}$ may be described by disjunct fuzzy sets which combined yield $\mathbf{X}$.

Also, let C be a fuzzy set associated with a class of objects from $\mathbf{X}$ and $F_n(C)$ be the family of all n-member fuzzy partitions of C. Let n be the

given number of subclusters in C. The cluster family of C can be appropriately described by a fuzzy partition P from $F_n(C)$, $P = \{A_1, \ldots, A_n\}$.

Every class A_i is described by a prototype or a cluster center $\mathbf{L}_i$ which represents a point in an s-dimensional Euclidean space $\mathbf{R}^s$. The clusters' form can be either spherical ellipsoidal. $\mathbf{L}_i$ represents the center of gravity of the fuzzy class A_i.

The fuzzy partition is typically described by a $n \times p$ membership matrix $\mathbf{U} = [u_{ij}]_{n \times p}$. Thus, the membership u_{ij} represents the degree of belonging of the pattern $\mathbf{x}_j$ to the ith class. For a crisp partition, the membership matrix $\mathbf{U}$ is binary-valued, whereas for a fuzzy partition its elements have values between 0 and 1. Here, we see the main difference between crisp and fuzzy partition: Given a fuzzy partition, a given data point $\mathbf{x}_j$ can belong to several classes with the degree of belonging specified by membership grades between 0 and 1. Given a crisp partition, this data point belongs to exactly one class. We will use the following notation hereafter: $u_{ij} = u_i(\mathbf{x}_j)$.

In the following, we will give the definition of a weighted Euclidean distance.

DEFINITION 9.5.1: The norm-induced distance d between two data $\mathbf{x}$ and $\mathbf{y}$ from $\mathbf{R}^s$ is given by

$$d^2(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}|| = (\mathbf{x} - \mathbf{y})^T \mathbf{M} (\mathbf{x} - \mathbf{y}) \tag{9.16}$$

where $\mathbf{M}$ is a symmetric positive definite matrix.

The distance with respect to a fuzzy class is given by the following definition.

DEFINITION 9.5.2:

The distance d_i between $\mathbf{x}$ and $\mathbf{y}$ with respect to the fuzzy class A_i is given by

$$d_i(\mathbf{x}, \mathbf{y}) = \min(u_{A_i}(\mathbf{x}), u_{A_i}(\mathbf{y})) d(\mathbf{x}, \mathbf{y}), \quad \forall \mathbf{x}, \mathbf{y} \in \mathbf{X} \tag{9.17}$$

The distance d_i can be easily extended to the entire metric space Y. A_i^* denotes the extension of A_i to Y, where

$$u_{A_i^*}(\mathbf{x}) = \begin{cases} u_{A_i}(\mathbf{x}), & \text{if } \mathbf{x} \in \mathbf{X} \\ 1, & \text{else} \end{cases} \tag{9.18}$$

DEFINITION 9.5.3: The extended distance is the distance $d_i^* : Y \times Y \to \mathbf{R}$, defined as

$$d_i^*(\mathbf{x}, \mathbf{y}) = \begin{cases} \min(u_{A_i}(\mathbf{x}), u_{A_i}(\mathbf{y}) d(\mathbf{x}, \mathbf{y}), & \text{if } \mathbf{x}, \mathbf{y} \in \mathbf{X} \\ u_{A_i}(\mathbf{x}) d(\mathbf{x}, \mathbf{y}), & \text{if } \mathbf{x} \in \mathbf{X}, \mathbf{y} \notin \mathbf{X} \\ d(\mathbf{x}, \mathbf{y}), & \text{if } \mathbf{x}, \mathbf{y} \notin \mathbf{X} \end{cases} \tag{9.19}$$

9.5.2 Mean Value of a Fuzzy Class

The mean vector of all vectors from $\mathbf{X}$ is defined as the only point $\mathbf{R}^s$ for which the sum of the squared distances from the points in $\mathbf{X}$ is minimal.

Let us assume that $\mathbf{v}$ is a point in $\mathbf{R}^s$ and $\mathbf{v}$ is not from $\mathbf{X}$. The sum of the squared distances in A_i between $\mathbf{v}$ and the points in $\mathbf{X}$ is given by

$$F_i(\mathbf{v}) = \sum_{j=1}^{p} d_i^{*2}(\mathbf{x}_j, \mathbf{v}) = \sum_{j=1}^{p} u_{ij}^2 d^2(\mathbf{x}_j, \mathbf{v}) \tag{9.20}$$

The minimum point of the function F_i can be interpreted as the mean point of the fuzzy class A_i.

THEOREM 9.5.1:
The function $F_i : (\mathbf{R}^s - \mathbf{X}) \to \mathbf{R}$

$$F_i(\mathbf{v}) = \sum_{j=1}^{p} u_{ij}^2 ||\mathbf{v} - \mathbf{x}_j||^2 \tag{9.21}$$

has a unique minimum $\mathbf{m}_i$ *given by*

$$\mathbf{m}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j \tag{9.22}$$

9.5.3 Alternating Optimization Technique

Fuzzy clustering algorithms are based on finding an adequate prototype for each fuzzy cluster and suitable membership degrees for the data to each cluster. In general, the cluster algorithm attempts to minimize an objective function which is based on either an intra-class similarity measure or a dissimilarity measure.

Let the cluster substructure of the fuzzy class C be described by the fuzzy partition $P = \{A_1, \ldots, A_n\}$ of C. In Section 9.2 it was shown that this is equivalent to

$$\sum_{j=1}^{p} u_{ij} = u_C(\mathbf{x}_j), \quad j = 1, \ldots, p \tag{9.23}$$

Further, let $\mathbf{L}_i \in \mathbf{R}^s$ be the prototype of the fuzzy class A_i and a point from the data set $\mathbf{X}$. We then obtain

$$u_{A_i}(\mathbf{L}_i) = \max_j u_{ij} \tag{9.24}$$

The dissimilarity between a data point and a prototype $\mathbf{L}_i$ is given by

$$D_i(\mathbf{x}_j, \mathbf{L}_i) = u_{ij}^2 d^2(\mathbf{x}_j, \mathbf{L}_i) \tag{9.25}$$

The inadequacy $I(A_i, \mathbf{L}_i)$ between the fuzzy class A_i and its prototype is defined as

$$I(A_i, \mathbf{L}_i) = \sum_{j=1}^{p} D_i(\mathbf{x}_j, \mathbf{L}_i) \tag{9.26}$$

Assume $\mathbf{L} = (\mathbf{L}_1, \ldots, \mathbf{L}_n)$ is the set of cluster centers and describes a representation of the fuzzy partition P.

The inadequacy $J(P, \mathbf{L})$ between the partition P and its representation $\mathbf{L}$ is defined as

$$J(P, \mathbf{L}) = \sum_{i=1}^{n} I(A_i, \mathbf{L}_i) \tag{9.27}$$

Thus the following objective function $J : F_n(C) \times \mathbf{R}^{sn} \to R$ is obtained:

$$J(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d^2(\mathbf{x}_j, \mathbf{L}_i) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 ||\mathbf{x}_j - \mathbf{L}_i||^2 \tag{9.28}$$

The objective function is of the least squared error type, and the optimal fuzzy partition and its representation can be found as the local solution of the minimization problem

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \\ P \in F_n(C) \\ \mathbf{L} \in \mathbf{R}^{sn} \end{cases} \tag{9.29}$$

Since an exact solution to the above problem is difficult to find, an adequate approximation based on an iterative method, the so-called alternating optimization technique [22], provides the solution by minimizing the functions $J(P, \cdot)$ and $J(\cdot, \mathbf{L})$.

In other words, the minimization problem from eq. (9.29) is replaced by two separate problems:

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \to \min \\ P \in F_n(C) \\ \mathbf{L} \quad \text{is fixed} \end{cases} \tag{9.30}$$

and

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \to \min \\ \mathbf{L} \in \mathbf{R}^{sn} \\ P \quad \text{is fixed} \end{cases} \tag{9.31}$$

For the solution to the first optimization problem we need to introduce the following notation:

$$I_j = \{i | 1 \leq i \leq n, \quad d(\mathbf{x}_j, \mathbf{L}_i) = 0\} \tag{9.32}$$

and

$$\bar{I}_j = \{1, 2, \ldots, n\} - I_j \tag{9.33}$$

We give without proof the following two theorems regarding the minimization of the functions $J(P, \cdot)$ or $J(\cdot, \mathbf{L})$ in eqs. (9.30) and (9.31).

THEOREM 9.5.2:
$P \in F_n(C)$ represents a minimum of the function $J(\cdot, \mathbf{L})$ only if

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{L}_i)}{d^2(\mathbf{x}_j, \mathbf{L}_k)}}, \quad \forall 1 \leq i \leq n: \quad 1 \leq j \leq p \tag{9.34}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.35}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

THEOREM 9.5.3:
If $\mathbf{L} \in \mathbf{R}^{sn}$ is a local minimum of the function $J(P, \cdot)$, then $\mathbf{L}_i$ is the cluster center (mean vector) of the fuzzy class A_i, for every $i = 1, \ldots, n$

$$\mathbf{L}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j \tag{9.36}$$

The alternating optimization (AO) technique is based on the Picard iteration of equations (9.34), (9.35), and (9.36). This technique can also be used to obtain a local solution to problem that will be addressed in equation (9.29). The iterative technique is also known as the generalized fuzzy n-means (GFNM) algorithm.

9.5.4 Family of Fuzzy n-Means Objective Functions

Let us assume that $P = \{A_1, \ldots, A_n\}$ is a fuzzy partition of the data set $\mathbf{X}$ and $\mathbf{L}_i \in \mathbf{R}^s$ is the prototype of the fuzzy class A_i. In [22] a more general objective function is considered:

$$J_m(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^m d^2(\mathbf{x}_j, \mathbf{L}_i) \tag{9.37}$$

with $m > 1$ being a weighting exponent, sometimes known as a fuzzifier, and d the norm-induced distance.

The prototypes are obtained as a solution to the optimization problem

$$J_m(P, \cdot) \to \min \tag{9.38}$$

and are given by

$$\mathbf{L}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^m} \sum_{j=1}^{p} u_{ij}^m \mathbf{x}_j \tag{9.39}$$

while the fuzzy partition is the solution of another optimization problem

$$J_m(\cdot, \mathbf{L}) \to \min \tag{9.40}$$

and is given by

$$u_{ij} = \frac{1}{\sum_{k=1}^{n} \left[\frac{d(\mathbf{x}_j, \mathbf{L}_i)}{d(\mathbf{x}_j, \mathbf{L}_k)}\right]^{\frac{2}{m-1}}}, \forall 1 \le i \le n; \quad 1 \le j \le p \tag{9.41}$$

Since the parameter m can take infinite values, an infinite family of fuzzy clustering algorithms is obtained.

In case $m \to 1$, the fuzzy n-means algorithm converges to a hard n-means solution. The membership degrees take either 0 or 1, thus describing a crisp representation. In this case, each input vector or data point $\mathbf{x}_j$ belongs exclusively to a single cluster. Thus, the fuzzy n-means algorithm is an extension of the hard n-means clustering algorithm, which is based on a crisp clustering criterion.

The integer m works to eliminate noises, and as m becomes larger, more data with small degrees of membership are neglected.

If $m \to \infty$, we get for the membership degrees the maximum fuzziness degrees

$$u_{ij} \to \frac{1}{n}, i = 1, \ldots, n; \quad j = 1, \ldots, p \tag{9.42}$$

and all prototypes converge to the mean vector of $\mathbf{X}$,

$$\mathbf{L}_i \to m(\mathbf{X}) = \frac{\sum_{j=1}^{p} \mathbf{x}_j}{p}, \quad i = 1, \ldots, n \tag{9.43}$$

These ideas can be visualized by the following example, which illustrates the differences between crisp and fuzzy n-means clustering.

EXAMPLE 9.5.1: Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$, where $\mathbf{x}_1 = [0,0]^T$, $\mathbf{x}_2 = [4,0]^T, \mathbf{x}_3 = [0,6]^T, \mathbf{x}_4 = [4,6]^T$. Let $\mathbf{L}_1 = [2,0]^T$ and $\mathbf{L}_2 = [2,6]^T$ be the cluster representatives. Assume that the distortion is based on the Euclidean distance between an input vector and a cluster representative. The membership matrix $\mathbf{U}$ for crisp n-means clustering is given by

$$\mathbf{U}_{hard} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \tag{9.44}$$

Based on eq. (9.37), we obtain $J_{hard}(P, L) = 8$. Assume now that $m = 1$ and that $u_{ij} \in [0, 1]$. Then we obtain for the objective function

$$J_1(P, L) = \sqrt{40}(u_{12} + u_{21} + u_{31} + u_{41}) + 2(u_{11} + u_{22} + u_{32} + u_{44}) > 8 \tag{9.45}$$

This can be easily obtained from $u_{i1} + u_{i2} = 1$. The conclusion we can draw is that hard clustering always yields better results than fuzzy clustering for $m = 1$. However, for $m \geq 2$, the resulting fuzzy objective function yields better results (< 8) than that for hard clustering.

A reformulation of the clustering objective functions is given in [83] and directly incorporates the necessary optimum conditions satisfied by P.

The following objective function is considered

$$J_m(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^m d^2(\mathbf{x}_j, \mathbf{L}_i) \tag{9.46}$$

The minimization of the function $J_m(\cdot, \mathbf{L}) \rightarrow \min$ leads to the membership values

$$u_{ij} = \frac{1}{\sum_{k=1}^{n} \left[\frac{d^2(\mathbf{x}_j, \mathbf{L}_i)}{d^2(\mathbf{x}_j, \mathbf{L}_k)}\right]^{\frac{1}{m-1}}}, \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{9.47}$$

For u_{ij}^{m-1} we obtain

$$u_{ij}^{m-1} = \frac{1}{d^2(\mathbf{x}_j, \mathbf{L}_i)} \left[\sum_{k=1}^{n} \left(\frac{1}{d^2(\mathbf{x}_j, \mathbf{L}_k)}\right)^{\frac{1}{m-1}}\right]^{-(m-1)} \tag{9.48}$$

By using the above expression, we obtain a new expression for $J_m(P, \mathbf{L})$

$$J_m(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij} u_{ij}^{m-1} d^2(\mathbf{x}_j, \mathbf{L}_i)$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{p} u_{ij} \frac{1}{[\sum_{k=1}^{n} (\frac{1}{d^2(\mathbf{x}_j, \mathbf{L}_k)})^{\frac{1}{m-1}}]^{(m-1)}}$$

Taking into account that $\sum_{j=1}^{p} u_{ij} = 1, \forall j = 1, \ldots, p$, we thus obtain a new expression for $J_m(P, \mathbf{L})$

$$J_m(P, \mathbf{L}) = \sum_{j=1}^{p} \frac{1}{[\sum_{k=1}^{n} (\frac{1}{d^2(\mathbf{x}_j, \mathbf{L}_k)})^{\frac{1}{m-1}}]^{(m-1)}} \tag{9.49}$$

This adapted expression depends only on the $\mathbf{L}$, and it is described by a new function $R_m(\mathbf{L})$. We are now in position to give the reformulated version of the fuzzy n-means criterion:

$$R_m(\mathbf{L}) = \sum_{j=1}^{p} [\sum_{k=1}^{n} (\frac{1}{d^2(\mathbf{x}_j, \mathbf{L}_k)})^{\frac{1}{m-1}}]^{1-m} \tag{9.50}$$

In [83] it was shown that the original and reformulated versions of fuzzy n-means objective functions are completely equivalent.

9.5.5 Generalized Fuzzy n-Means Algorithm

In this section we describe the GFNM algorithm based on the theoretical results derived in the previous section. We consider here compact clusters completely described by a point representative $\mathbf{L}_i$.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$ define the data set and C be a fuzzy set on $\mathbf{X}$. The following assumptions are made:

- C represents a cluster of points from $\mathbf{X}$.
- C has a cluster substructure described by the fuzzy partition $P = \{A_1, \ldots, A_n\}$.
- n is the number of known subclusters in C.

The basic GFNM requires a random initialization of the fuzzy partition. The prototypes $\mathbf{L}_1, \ldots, \mathbf{L}_n$ are computed based on eq. (9.36), while the new fuzzy partition of C is computed using eqs. (9.34) and (9.35). These equations are continuously updated until two successive fuzzy partitions do not change substantially.

In order to monitor the convergence of the algorithm, the $n \times p$ partition matrix $\mathbf{Q}^i$ is introduced to describe each fuzzy partition P^i at the ith iteration and is used to determine the distance between two fuzzy partitions. The matrix $\mathbf{Q}^i$ is defined as

$$\mathbf{Q}^i = \mathbf{U} \quad \text{at} \quad \text{iteration} \quad \text{i} \tag{9.51}$$

The termination criterion for iteration m is given by

$$d(P^m, P^{m-1}) = ||\mathbf{Q}^m - \mathbf{Q}^{m-1}|| < \epsilon \tag{9.52}$$

where ϵ defines the admissible error and $||\cdot||$ is any vector norm.

An algorithmic description of the GFNM is given below:

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ϵ. P^1 is selected as a random fuzzy partition of C having n atoms. Determine $\mathbf{Q}^1$, the matrix representation of P^1.
2. **Adaptation, part I:** Determine the cluster prototypes $\mathbf{L}_i, i = 1, \ldots, n$ using

$$\mathbf{L}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j \tag{9.53}$$

3. **Adaptation, part II:** Determine a new fuzzy partition P^2 of C using the following rules:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{L}_i)}{d^2(\mathbf{x}_j, \mathbf{L}_k)}}, \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{9.54}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.55}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

Determine $\mathbf{Q}^2$, the matrix representation of the fuzzy partition P^2.

4. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold $||\mathbf{Q}^1 - \mathbf{Q}^2|| < \epsilon$, then stop. Else set $P^1 = P^2, \mathbf{Q}^1 = \mathbf{Q}^2$ and go to step 2.

If we obtain $C = \mathbf{X}$, the GFNM algorithm converts to the fuzzy n-means (FNM) algorithm. Then the membership degrees in eqs. (9.54) and (9.55) are determined by the following new rules:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{1}{\sum_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{L}_i)}{d^2(\mathbf{x}_j, \mathbf{L}_k)}}, \quad \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{9.56}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.57}$$

and arbitrarily assign $\sum_{i \in I_j} u_{ij} = 1$.

9.5.6 Generalized Adaptive Fuzzy n-Means Algorithm

In this section we describe an adaptive fuzzy technique based on using different metrics to allow the detection of cluster shapes ranging from spherical to ellipsoidal clusters.

To achieve this an adaptive metric is used. We define a new distance metric $d(\mathbf{x}_j, \mathbf{L}_i)$ from the data point $\mathbf{x}_j$ to the cluster prototype $\mathbf{L}_i$ as

$$d^2(\mathbf{x}_j, \mathbf{L}_i) = (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i) \tag{9.58}$$

where $\mathbf{M}_i$ is a symmetric and positive definite shape matrix and adapts to the clusters' shape variations. The growth of the shape matrix is monitored by a bound:

$$|\mathbf{M}_i| = \rho_i, \quad \rho_i > 0, \quad i = 1, \ldots, n \tag{9.59}$$

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$, $\mathbf{x}_j \in \mathbf{R}^s$ be a data set. Let C be a fuzzy set on $\mathbf{X}$ describing a fuzzy cluster of points in $\mathbf{X}$, and having a cluster substructure which is described by a fuzzy partition $P = \{A_1, \ldots, A_n\}$ of C. Each fuzzy class A_i is described by the point prototype $\mathbf{L}_i \in \mathbf{R}^s$. The local distance with respect to A_i is given by

$$d_i^2(\mathbf{x}_j, \mathbf{L}_i) = u_{ij}^2 (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i) \tag{9.60}$$

As an objective function we choose

$$J(P, \mathbf{L}, M) = \sum_{i=1}^{n} \sum_{j=1}^{p} d^2(\mathbf{x}_j, \mathbf{L}_i) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i) \tag{9.61}$$

where $M = (\mathbf{M}_1, \ldots, \mathbf{M}_n)$.

The objective function is again chosen to be of the least squared error type. We can find the optimal fuzzy partition and its representation as the local solution of the minimization problem:

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}, M) \\ \sum_{i=1}^{n} u_{ij} = u_C(\mathbf{x}_j), \quad j = 1, \ldots, p \\ |\mathbf{M}_i| = \rho_i, \quad \rho_i > 0, \quad i = 1, \ldots, n \\ \mathbf{L} \in \mathbf{R}^{sn} \end{cases} \tag{9.62}$$

The following theorem regarding the minimization of the functions $J(P, \mathbf{L}, \cdot)$ is given without proof. It is known as the adaptive norm theorem.

THEOREM 9.5.4:

Assuming the point prototype $\mathbf{L}_i$ *of the fuzzy class* A_i *equals the cluster center of this class,* $\mathbf{L}_i = \mathbf{m}_i$, *and the determinant of the shape matrix* $\mathbf{M}_i$ *is bounded,* $|\mathbf{M}_i| = \rho_i, \rho_i > 0, i = 1, \ldots, n$, *then* $\mathbf{M}_i$ *is a local minimum of the function* $J(P, \mathbf{L}, \cdot)$ *only if*

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_i|]^{\frac{1}{s}} \mathbf{S}_i^{-1} \tag{9.63}$$

where $\mathbf{S}_i$ *is the within-class scatter matrix of the fuzzy class* A_i*:*

$$\mathbf{S}_i = \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{m}_i)(\mathbf{x}_j - \mathbf{m}_i)^T. \tag{9.64}$$

The above theorem can be employed as part of an alternating optimization technique. The resulting iterative procedure is known as the generalized adaptive fuzzy n-means (GAFNM) algorithm.

An algorithmic description of the GAFNM is given below:

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ϵ. P^1 is selected as a random fuzzy partition of C having n atoms. Set iteration counter $l = 1$.
2. **Adaptation, part I:** Determine the cluster prototypes $\mathbf{L}_i = \mathbf{m}_i, i = 1, \ldots, n$ using

$$\mathbf{L}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j \tag{9.65}$$

3. **Adaptation, part II:** Determine the within class scatter matrix $\mathbf{S}_i$ using

$$\mathbf{S}_i = \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{m}_i)(\mathbf{x}_j - \mathbf{m}_i)^T \tag{9.66}$$

Determine the shape matrix $\mathbf{M}_i$ using

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_i|]^{\frac{1}{s}} \mathbf{S}_i^{-1} \tag{9.67}$$

and compute the distance $d^2(\mathbf{x}_j, \mathbf{m}_i)$ using

$$d^2(\mathbf{x}_j, \mathbf{m}_i) = (\mathbf{x}_j - \mathbf{m}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{m}_i) \tag{9.68}$$

4. **Adaptation part III:** Compute a new fuzzy partition P^l of C using the following rules

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{m}_i)}{d^2(\mathbf{x}_j, \mathbf{m}_k)}}, \quad \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{9.69}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.70}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

The standard notation is used

$$I_j = \{i | 1 \leq i \leq n. \quad d(\mathbf{x}_j, \mathbf{L}_i) = 0\} \tag{9.71}$$

and

$$\bar{I}_j = \{1.2, \ldots, n\} - I_j \tag{9.72}$$

5. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold, $||P^l - P^{l-1}|| < \epsilon$, then stop. Else go to step 2.

An important issue for the GAFNM algorithm is the selection of the bounds of the shape matrix $\mathbf{M}_i$. They can be chosen as:

$$\rho_i = 1. \quad i = 1, \ldots, n \tag{9.73}$$

If we choose $C = \mathbf{X}$, we obtain $u_C(\mathbf{x}_j) = 1$ and thus we get the following membership degrees:

$$u_{ij} = \frac{1}{\sum_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{m}_i)}{d^2(\mathbf{x}_j, \mathbf{m}_k)}}, \quad \forall 1 \leq i \leq n: \quad 1 \leq j \leq p \tag{9.74}$$

The resulting iterative procedure is known as the adaptive fuzzy n-means (AFNM) algorithm.

9.5.7 Generalized Fuzzy n-Shells Algorithm

So far, we have considered clustering algorithms that use point prototypes as cluster prototypes. Therefore, the previous algorithms cannot detect clusters that can be described by shells. hyperspheres, or hyperellipsoids. The generalized fuzzy n-shells algorithm [49, 50] is able to detect such clusters. The cluster prototypes used are s-dimensional hyperspherical shells,

and the distances of data points are measured from the hyperspherical surfaces. Since the prototypes contain no interiors, they are referred to as shells. They are completely described by a set of centers $\mathbf{V}$ and a set of radii $\mathbf{R}$.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$, $\mathbf{x}_j \in \mathbf{R}^s$ be a data set. Let C be a fuzzy set on $\mathbf{X}$ describing a fuzzy cluster of points in $\mathbf{X}$. C has a cluster substructure that is described by a fuzzy partition $P = \{A_1, \ldots, A_n\}$ of C. Each fuzzy class A_i is described by the point prototype $\mathbf{L}_i \in \mathbf{R}^s$. It is assumed that the cluster resembles a hyperspherical shell, and thus the prototype $\mathbf{L}_i$ of A_i is determined by two parameters $(\mathbf{v}_i, \mathbf{r}_i)$, where $\mathbf{v}_i$ is the center of the hypersphere and r_i is the radius.

The hypershperical shell prototype $\mathbf{L}_i$ with center $\mathbf{v}_i \in \mathbf{R}^s$ and radius $r_i \in R^+$, is defined by the set

$$\mathbf{L}_i(\mathbf{v}_i, r_i) = \{\mathbf{x} \in \mathbf{R}^s | (\mathbf{x} - \mathbf{v}_i)^T (\mathbf{x} - \mathbf{v}_i) = r_i^2\} \tag{9.75}$$

The distance d_{ij} between the point $\mathbf{x}_j$ and the prototype $\mathbf{L}_i(\mathbf{v}_i, r_i)$ is defined as

$$d_{ij}^2 = d^2(\mathbf{x}_j, \mathbf{L}_i) = [||\mathbf{x}_j - \mathbf{v}_i|| - r_i]^2 \tag{9.76}$$

The corresponding objective function $J(P, \mathbf{L})$ describing the deviation between the fuzzy partition and its representation $\mathbf{L}$ can be written as

$$J(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} D_i(\mathbf{x}_j, \mathbf{L}_i) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d_{ij}^2 \tag{9.77}$$

$\mathbf{L}$ represents the set of centers

$$\mathbf{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n | \mathbf{v}_i \in \mathbf{R}^s\} \tag{9.78}$$

while the set of radii is given by

$$R = \{r_1, \ldots, r_n | r_i \in R^+\} \tag{9.79}$$

This means that the objective function $J(P, \mathbf{L})$ is a function $J : F_n(\mathbf{X}) \times \mathbf{R}^{sn} \times \mathbf{R}^{+n} \to R$ and is defined as

$$J(P, \mathbf{L}) = J(P, V, R) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 [||\mathbf{x}_j - \mathbf{v}_i|| - r_i]^2 \tag{9.80}$$

The generalized shell clustering algorithm is based on an iterative procedure to solve the following minimization problem:

$$\begin{cases} \text{minimize} \quad J(P, V, R) \\ \sum_{i=1}^{n} u_{ij} = u_C(\mathbf{x}_j), \quad j = 1, \ldots, p \\ V \in \mathbf{R}^{sn}, R \in \mathbf{R}^{+n} \end{cases} \tag{9.81}$$

To solve this problem two theorems are needed. The first theorem is the theorem for optimal fuzzy partition.

THEOREM 9.5.5:

Assume that $\mathbf{X}$ *has at least* $n < p$ *distinct points, and define the following sets :*

$$I_j = \{i | 1 \leq i \leq n, \quad d(\mathbf{x}_j, \mathbf{L}_i) = 0\} \tag{9.82}$$

and

$$\bar{I}_j = \{1, 2, \ldots, n\} - I_j \tag{9.83}$$

A fuzzy partition P *represents the minimum of the objective function* $J(\cdot, \mathbf{V}, \mathbf{R})$ *only if*

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum\limits_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.84}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \quad \forall i \in I_j \tag{9.85}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

The next theorem gives the conditions for the optimal prototypes. The conditions describe a system of nonlinear equations to determine the radii and centers of the cluster prototypes.

THEOREM 9.5.6:

Let P *be a fixed fuzzy partition. Then* $\mathbf{V}$ *and* $\mathbf{R}$ *minimize the objective function* $J(P, \cdot, \cdot)$ *only if the centers and the radii equations*

$$\begin{cases} \sum_{j=1}^{p} u_{ij}^2 \frac{\|\mathbf{x}_j - \mathbf{v}_i\| - r_i}{\|\mathbf{x}_j - \mathbf{v}_i\|} (\mathbf{x}_j - \mathbf{v}_i) = 0 \\ \sum_{j=1}^{p} u_{ij}^2 [\|\mathbf{x}_j - \mathbf{v}_i\| - r_i] = 0 \end{cases} \tag{9.86}$$

hold, where $i = 1, \ldots, n$.

The preceding theorems can be used as the basis of an alternating optimization technique. The resulting iterative procedure is known as the generalized fuzzy n-shells (GFNS) algorithm.

An algorithmic description of the GFNS is given below:

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ϵ. P^1 is selected as a random fuzzy partition of C having n atoms. Set iteration counter $l = 1$.

2. **Adaptation, part I:** Determine the centers $\mathbf{v}_i$ and radii r_i by solving the system of equations

$$\sum_{j=1}^{p} u_{ij}^2 \frac{||\mathbf{x}_j - \mathbf{v}_i|| - r_i}{||\mathbf{x}_j - \mathbf{v}_i||}(\mathbf{x}_j - \mathbf{v}_i) = 0 \tag{9.87}$$

$$\sum_{j=1}^{p} u_{ij}^2 [||\mathbf{x}_j - \mathbf{v}_i|| - r_i] = 0 \tag{9.88}$$

where $i = 1, \ldots, n$.

3. **Adaptation, part II:** Determine the distance d_{ij} of the point $\mathbf{x}_j$ from the prototype $\mathbf{L}_i(\mathbf{v}_i, r_i)$ defined as

$$d_{ij}^2 = d^2(\mathbf{x}_j, \mathbf{L}_i) = [||\mathbf{x}_j - \mathbf{v}_i|| - r_i]^2 \tag{9.89}$$

4. **Adaptation, part III:** Determine a new fuzzy partition P^l of C using the following rules

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.90}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.91}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

Set $l = l + 1$.

5. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold, $||P^l - P^{l-1}|| < \epsilon$, then stop. Else go to step 2.

If we choose $C = \mathbf{X}$, we obtain $u_C(\mathbf{x}_j) = 1$ and thus we get the following fuzzy partition

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{1}{\sum_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.92}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.93}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = 1$.

The resulting iterative procedure is known as the of fuzzy n-shells (FNS) algorithm.

9.5.8 Generalized Adaptive Fuzzy n-Shells Algorithm

In this section we describe an adaptive procedure similar to the generalized adaptive fuzzy n-means algorithm, but for shell prototypes. As described in [50] hyperellipsoidal shell prototypes will be considered.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$, $\mathbf{x}_j \in \mathbf{R}^s$ be a data set. Let C be a fuzzy set on $\mathbf{X}$ describing a fuzzy cluster of points in $\mathbf{X}$. C has a cluster substructure, which is described by a fuzzy partition $P = \{A_1, \ldots, A_n\}$ of C. Each fuzzy class A_i is described by the point prototype $\mathbf{L}_i \in \mathbf{R}^s$. Let us assume that each cluster resembles a hyperellipsoidal shell.

The hyperellipsoidal shell prototype $\mathbf{L}_i(\mathbf{v}_i, r_i, \mathbf{M}_i)$ of the fuzzy class A_i is given by the set

$$\mathbf{L}_i(\mathbf{v}_i, r_i, \mathbf{M}_i) = \{\mathbf{x} \in \mathbf{R}^s | (\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i) = r_i^2\} \tag{9.94}$$

with $\mathbf{M}_i$ representing a symmetric and positive definite matrix.
Distance d_{ij} between the point $\mathbf{x}_j$ and the cluster center $\mathbf{v}^i$ is defined as

$$d_{ij}^2 = d^2(\mathbf{x}_j, \mathbf{v}_i) = [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - r_i \tag{9.95}$$

Thus a slightly changed objective function is obtained:

$$J(P, V, R, M) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d_{ij}^2 = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - r_i]^2 \tag{9.96}$$

For optimization purposes, we need to determine the minimum of the functions $J(\cdot, V, R, M)$, $J(P, \cdot, R, M)$, and $J(P, V, \cdot, M)$. It can be shown that they are given by the following propositions [60].

The following is the proposition for optimal partition.

PROPOSITION 9.5.1:

The minimum of the function $J(\cdot, \mathbf{V}, \mathbf{R}, \mathbf{M})$ is given by the fuzzy partition P only if

$$I_j = \varnothing \Rightarrow u_{ij}) = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.97}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.98}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

The next is the proposition for optimal prototype centers.

PROPOSITION 9.5.2:

The optimal value of $\mathbf{V}$ *with respect to the function* $J(P, \cdot, R, M)$ *is given by*

$$\sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i) = 0, \quad i = 1 \ldots, n \tag{9.99}$$

where q_{ij} *is given by*

$$q_{ij} = (\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i) \tag{9.100}$$

The following is the proposition for optimal prototype radii.

PROPOSITION 9.5.3:

The optimal value of R *with respect to the function* $J(P, V, \cdot, M)$ *is given by*

$$\sum_{j=1}^{p} u_{ij}^2 d_{ij} = 0, \quad i = 1, \ldots, n \tag{9.101}$$

Thus, we have given the necessary propositions to determine the minimum of the functions $J(\cdot, V, R, M)$, $J(P, \cdot, R, M)$. and $J(P, V, \cdot, M)$.

To ensure that the adaptive norm is bounded, we impose the constraint

$$|\mathbf{M}_i| = \rho_i, \quad \text{where} \quad \rho_i > 0, \quad i = 1, \ldots, n \tag{9.102}$$

The norm is given by the following theorem, the so-called adaptive norm theorem [60].

THEOREM 9.5.7:

Let $\mathbf{X} \subset \mathbf{R}^s$. *Suppose the objective function* J *already contains the optimal* P, V *and* R. *If the determinant of the shape matrix* $\mathbf{M}_i$ *is bounded,* $|\mathbf{M}_i| = \rho_i$, $\rho_i > 0$, $i = 1, \ldots, n$, *then* $\mathbf{M}_i$ *is a local minimum of the function* $J(P, V, R, \cdot)$ *only if*

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_{si}|]^{\frac{1}{s}} \mathbf{S}_{si}^{-1} \tag{9.103}$$

where $\mathbf{S}_{si}$ *represents the nonsingular shell scatter matrix of the fuzzy class* A_i

$$\mathbf{S}_{si} = \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T. \tag{9.104}$$

In practice, the bound is chosen as $\rho_i = 1$, $i = 1, \ldots, n$.

The preceding theorems can be used as the basis of an alternating optimization technique. The resulting iterative procedure is known as the generalized adaptive fuzzy n-shells (GAFNS) algorithm.

An algorithmic description of the GAFNS is given below:

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ϵ. P^1 is selected as a random fuzzy partition of C having n atoms. Initialize $\mathbf{M}_i = \mathbf{I}, i = 1, \ldots, n$ where $\mathbf{I}$ is an $s \times s$ unity matrix. Set iteration counter $l = 1$.
2. **Adaptation, part I:** Determine the centers $\mathbf{v}_i$ and radii r_i by solving the system of equations

$$\begin{cases} \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i) = 0 \\ \sum_{j=1}^{p} u_{ij}^2 d_{ij} = 0 \end{cases} \tag{9.105}$$

   where $i = 1, \ldots, n$ and $q_{ij} = (\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i)$.
3. **Adaptation, part II:** Determine the shell scatter matrix $\mathbf{S}_{si}$ of the fuzzy class A_i

$$\mathbf{S}_{si} = \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T \tag{9.106}$$

   where the distance d_{ij} is given by

$$d_{ij}^2 = [(\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i)]^{1/2} - r_i \tag{9.107}$$

4. **Adaptation, part III:** Determine the approximate value of $\mathbf{M}_i$:

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_{si}|]^{\frac{1}{s}} \mathbf{S}_{si}^{-1}, \quad i = 1, \ldots, n \tag{9.108}$$

   $\rho_i = 1$ or ρ_i is equal to the determinant of the previous $\mathbf{M}_i$.
5. **Adaptation, part IV:** Compute a new fuzzy partition P^l of C using the following rules:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.109}$$

   and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \quad \forall i \in I_j \tag{9.110}$$

   and arbitrarily $\sum_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

   Set $l = l + 1$.

6. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold, $||P^l - P^{l-1}|| < \epsilon$, then stop. Else go to step 2.

If we choose $u_C = \mathbf{X}$, we obtain $u_C(\mathbf{x}_j) = 1$ and thus we get the following fuzzy partition:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{1}{\sum_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{9.111}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{9.112}$$

and arbitrarily $\sum_{i \in I_j} u_{ij} = 1$.

The resulting iterative procedure is known as adaptive fuzzy n-shells (AFNS) algorithm. This technique enables one to identify the elliptical data substructure, and even to detect overlapping between clusters to some degree.

9.5.8.1 Simplified AFNS Algorithm Newton's method can be employed to determine $\mathbf{v}_i$ and r_i from the system of equations (9.99) and (9.101). The drawback of the AFNS is its computational complexity. Therefore, some simplifications are necessary for a better practical handling. In [50] a simplification of the algorithm was proposed based on the fact that the radius r_i can be absorbed into the matrix $\mathbf{M}_i$ such that the hyperellipsoidal prototype $\mathbf{L}_i$ becomes the surface of the equation

$$(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i) = 1 \tag{9.113}$$

This means that eq. (9.101) is no longer necessary. The distance d_{ij} of the point $\mathbf{x}_j$ from the prototype $\mathbf{L}_i$ is now given by

$$d_{ij} = [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - 1 \tag{9.114}$$

This changes the objective function into a new one:

$$J(P, V, M) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d_{ij}^2 = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - 1]^2 \tag{9.115}$$

for $i = 1, \ldots, n$. The centers are obtained as a solution of the system of equations

$$\sum_{j=1}^{p} u_{ij}^2 \frac{[(\mathbf{x}-\mathbf{v}_i)^T\mathbf{M}_i(\mathbf{x}-\mathbf{v}_i)]^{\frac{1}{2}}-1}{d_{ij}}(\mathbf{x}_j-\mathbf{v}_i)=0 \tag{9.116}$$

9.5.8.2 Infinite AFNS Family By introducing a weighting exponent $m>1$ as a fuzzifier, we obtain the infinite family J_m of objective functions

$$J(P,V,R,M)=\sum_{i=1}^{n}\sum_{j=1}^{p} u_{ij}^m d_{ij}^2 \tag{9.117}$$

Assume, that the prototype $\mathbf{L}_i$ is given by

$$(\mathbf{x}-\mathbf{v}_i)^T\mathbf{M}_i(\mathbf{x}-\mathbf{v}_i)=r_i \tag{9.118}$$

and the distance d_{ij} between the point $\mathbf{x}_j$ and the prototype $\mathbf{L}_i$ is now given by

$$d_{ij}=[(\mathbf{x}-\mathbf{v}_i)^T\mathbf{M}_i(\mathbf{x}-\mathbf{v}_i)]^{\frac{1}{2}}-r_i \tag{9.119}$$

Thus, the optimal fuzzy partition becomes

$$u_{ij}=\frac{1}{\sum_{k=1}^{n}\left[\frac{d_{ij}^2}{d_{kj}^2}\right]^{\frac{1}{m-1}}} \tag{9.120}$$

for $j=1,\ldots,p$ and $i=1,\ldots,n$.

The centers $\mathbf{v}_i$ and radii r_i are obtained as the solutions of the system of equations

$$\sum_{j=1}^{p} u_{ij}^m \frac{d_{ij}}{q_{ij}}(\mathbf{x}_j-\mathbf{v}_i)=0 \tag{9.121}$$

$$\sum_{j=1}^{p} u_{ij}^m d_{ij}=0 \tag{9.122}$$

where $i=1,\ldots,n$ and $q_{ij}=(\mathbf{x}_j-\mathbf{v}_i)^T\mathbf{M}_i(\mathbf{x}_j-\mathbf{v}_i)$.
The shell scatter matrix $\mathbf{S}_{si}$ of the fuzzy class A_i becomes

$$\mathbf{S}_{si}=\sum_{j=1}^{p} u_{ij}^m \frac{d_{ij}}{q_{ij}}(\mathbf{x}_j-\mathbf{v}_i)(\mathbf{x}_j-\mathbf{v}_i)^T \tag{9.123}$$

and the matrix $\mathbf{M}_i, i=1,\ldots,n$ is given by

$$\mathbf{M}_i=[\rho_i|\mathbf{S}_{si}|]^{\frac{1}{s}}\mathbf{S}_{si}^{-1}, \quad i=1,\ldots,n \tag{9.124}$$

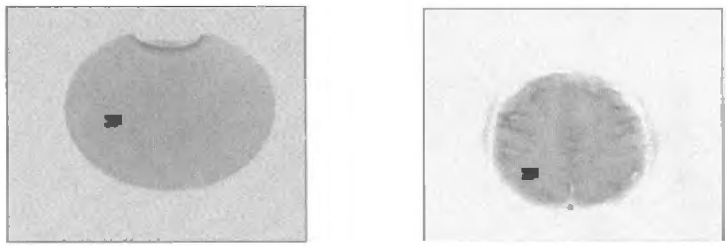
The constraint $|\mathbf{M}_i|=\rho_i, \rho_i>0, i=1,\ldots,n$, is imposed to achieve this.

The simplifications used for the AFNS can also be applied to the infinite AFNS family. This means that the prototypes can have equal radii $r_i = 1, i = 1, \ldots, n$, and the distance d_{ij} has the form

$$d_{ij} = [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - 1 \tag{9.125}$$

9.6 COMPARISON OF FUZZY CLUSTERING VERSUS PCA FOR FMRI

Recent work in fMRI [17] compared two paradigm-free (data-driven) methods: fuzzy clustering analysis (FCA) based on the fuzzy n-means algorithm versus principal component analysis (PCA) for two types of fMRI signal data. These are a water phantom with scanner noise contributions only and in vivo data acquired under null hypothesis conditions as shown in Figure 9.9.



(a) Water phantom (b) In vivo data

FIGURE 9.9 Test images with activation region overlaid: water phantom (a) and (b) in vivo data acquired under null hypothesis conditions. (Images from [17] reprinted with permission from Elesevier Science.)

Figure 9.10a shows the activation map determined by the FNM algorithm and added to the mean image of the time series acquired for the water phantom, while Figure 9.10b shows that of the in vivo data. Figure 9.10c shows the activation map determined by PCA, added on the mean image for the same data set. The similarity between the activity maps calculated by the FNM algorithm and PCA is visible. Figure 9.10d, on the other hand, shows that PCA did not identify the activation focus, and produced well-scattered positives. In 9.10e the time-courses obtained for water phantom data by both PCA and FNM algorithm are shown. Both exhibit a high correlation coefficient with respect to the simulated time-course. For the in vivo data, only the FNM algorithm achieves a high correlation as shown in Figure 9.10f, while the time-course extracted by PCA is corrupted and fails to detect the simulated activation.

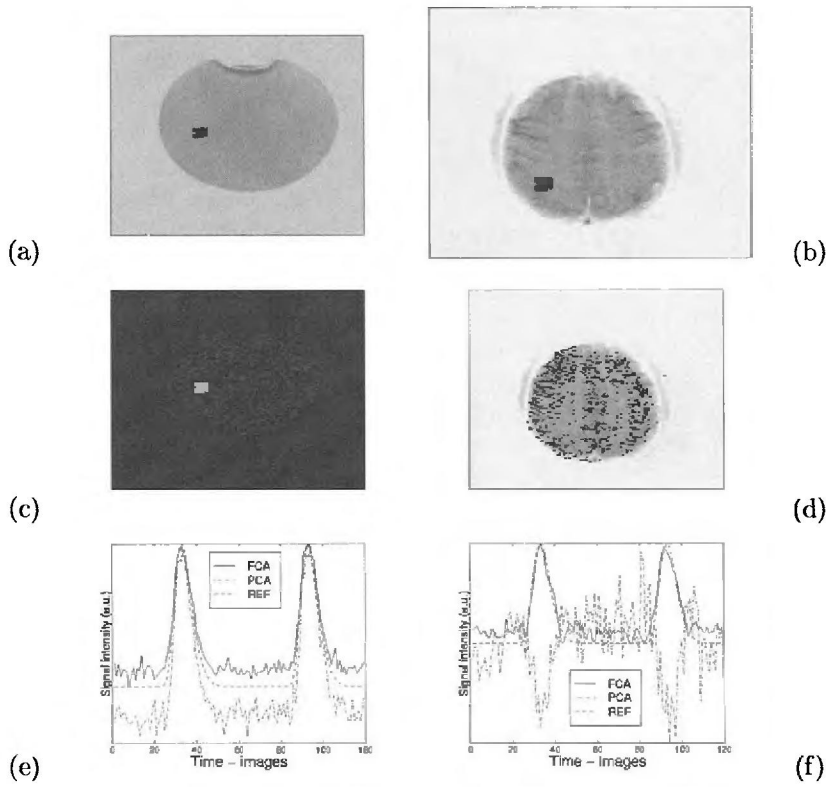


FIGURE 9.10 Results of applying FNM algorithm and PCA to water phantom and in vivo data. (a) FNM algorithm: Water phantom with the region extracted overlaid. (b) FNM: In vivo anatomy with the region extracted by FNM overlaid. (c) PCA: Activation map plus T_2^* anatomy. (d) PCA: Activation map plus T_2^* anatomy. (e) Time-courses corresponding to FNM and PCA for water phantom: good agreement for both methods with the reference time-course. (f) Time-courses corresponding to FNM and PCA for in vivo data: only with FNM good agreement. (Images from [17] reprinted with permission from Elesevier Science.)

The experiments performed in [17] comparing the performance of FNM algorithm versus PCA showed the following:

- In the presence of scanner noise only, both methods show comparable performance.
- In the presence of other noise sources (artifacts), PCA fails to detect activation at lower contrast-to-noise ratios (CNR). This could be critical in fMRI. FNM outperforms PCA for all chosen CNR values.
- PCA globally fails to detect both positively and negatively correlated time-courses, i.e., activation and deactivation. This is because

of the arbitrary sign of the eigenvectors obtained from the correlation matrix decomposition. Only FNM preserves the original shapes of the time-courses and yields immediately interpretable results.

9.7 FUZZY ALGORITHMS FOR LVQ

Let us consider the set $\mathbf{X}$ of samples from an n-dimensional Euclidean space and let $f(\mathbf{x})$ be the probability distribution function of $\mathbf{x} \in \mathbf{X} \in \mathbf{R}^n$. Learning vector quantization is based on the minimization of the functional [183]

$$D(\mathbf{L}_1, \ldots, \mathbf{L}_c) = \int \cdots \int_{\mathbf{R}^n} \sum_{r=1}^{c} u_r(\mathbf{x}) ||\mathbf{x} - \mathbf{L}_r||^2 f(\mathbf{x}) d\mathbf{x} \tag{9.126}$$

with $D_{\mathbf{x}} = D_{\mathbf{x}}(\mathbf{L}_1, \ldots, \mathbf{L}_c)$ being the expectation of the loss function, defined as

$$D_{\mathbf{x}}(\mathbf{L}_1, \ldots, \mathbf{L}_c) = \sum_{r=1}^{c} u_r(\mathbf{x}) ||\mathbf{x} - \mathbf{L}_r||^2 \tag{9.127}$$

$u_r = u_r(\mathbf{x}), 1 \leq r \leq c$, all membership functions that describe competitions between the prototypes for the input $\mathbf{x}$. Supposing that $\mathbf{L}_i$ is the winning prototype that belongs to the input vector $\mathbf{x}$, that is, the closest prototype to $\mathbf{x}$ in the Euclidean sense the memberships $u_{ir} = u_r(\mathbf{x}), 1 \leq r \leq c$ are given by

$$u_{ir} = \begin{cases} 1, & \text{if} \quad r = i \\ u(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}), & \text{if} \quad r \neq i \end{cases} \tag{9.128}$$

In the given context, the loss function measures the locally weighted error of each input vector with respect to the winning prototype.

Choosing the gradient descent method as a minimization technique for eq. (9.126) proves to be difficult, since the winning prototype involved in the definition of the loss function $D_{\mathbf{x}}$ is a function of the input vector $\mathbf{x} \in \mathbf{X}$. In [183], an alternative is proposed, that is, to use the gradient of the instantaneous loss function (9.127), if the pdf $f(\mathbf{x})$ is not known. This means that the prototypes will be updated not at the same time but one after another. If $u_{ir} = 0, \forall r \neq i$, then minimization of the loss function (9.127) based on the gradient descent is similar to Kohonen's (unlabeled) LVQ [114], which generates hard c-partitions of the input data. In LVQ, only the winning prototype is updated during learning in order to match the input vector. However, there are disadvantages associated with this technique: heavy dependence on the initial set of prototypes and susceptibility to local minima.

In order to employ fuzzy techniques for learning vector quantization, membership functions have to be assigned to the prototypes [99, 100]. The membership function assigned to each prototype has to fulfill the following criteria to ensure a fair competition:

- Invariance under uniform scaling of the entire data set.
- Equal to 1 if the prototype is determined to be the winner.
- If the prototype is not the winner, it takes any value between 0 and 1.
- If it is not the winner, then it is close to 0, and its distance from the input vector approaches infinity.

Based on eq. (9.128) we can rewrite the loss function from eq. (9.127) in the form

$$D_{\mathbf{x}}(\mathbf{L}_1, \ldots, \mathbf{L}_c) = \sum_{r=1}^{c} u_r(\mathbf{x})||\mathbf{x} - \mathbf{L}_r||^2 = ||\mathbf{x} - \mathbf{L}_i||^2 + \sum_{r \neq i}^{c} u_{ir}||\mathbf{x} - \mathbf{L}_r||^2 \tag{9.129}$$

The loss function is determined by the weighted contribution between membership functions and distance between input data and cluster prototypes. There is a relative contribution of the nonwinning prototype $\mathbf{L}_r$ with respect to the winning prototype $\mathbf{L}_i$ given by the ratio $u_{ir}\frac{||\mathbf{x}-\mathbf{L}_r||^2}{||\mathbf{x}-\mathbf{L}_i||^2}$.

To determine an adequate membership function, we have to require that the relative contribution of the nonwinning prototype $\mathbf{L}_r$ with respect to the winning prototype $\mathbf{L}_i$ be a function of the ratio $\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}$, that is,

$$u_{ir}\frac{||\mathbf{x} - \mathbf{L}_r||^2}{||\mathbf{x} - \mathbf{L}_i||^2} = p\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_r||^2}\right) \tag{9.130}$$

It can be seen immediately that the corresponding function $u(\cdot)$ is of the form $u(z) = zp(z)$. Thus, for eq. (9.127) we obtain the following result:

$$D_{\mathbf{x}} = ||\mathbf{x} - \mathbf{L}_i||^2 \left[1 + \sum_{r \neq i}^{c} p\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_r||^2}\right)\right] \tag{9.131}$$

The selection of membership functions of the form $u(z) = zp(z)$ implies that the winning prototype $\mathbf{L}_i$ is updated with respect to the input $\mathbf{x}$ by minimizing a weighted version of the squared Euclidean distance $||\mathbf{x}-\mathbf{L}_i||^2$. To satisfy the admissibility conditions for the membership function, the function $p(\cdot)$ has to satisfy the following properties [99, 100]:

- $0 < p(z) < 1, \forall z \in (0, 1)$.
- $p(z)$ approaches 1 as z approaches 0.
- $p(z)$ is a monotonically decreasing function in the interval $(0, 1)$.

- $p(z)$ attains its minimum value at $z = 1$.

Several fuzzy learning vector quantization (FALVQ) algorithms can be determined based on minimizing the loss function (9.131).

The gradient of $D_{\mathbf{x}}$ with respect to the winning prototype $\mathbf{L}_i$ is

$$\begin{aligned}\frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_i} &= \frac{\partial}{\partial \mathbf{L}_i}\left[||\mathbf{x}-\mathbf{L}_i||^2 + \sum_{r\neq i}^{c}||\mathbf{x}-\mathbf{L}_i||^2 p\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right)\right] \\ &= -2(\mathbf{x}-\mathbf{L}_i)\{1+\sum_{r\neq i}^{c} p\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) \\ &\quad + \frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2} p'\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right)\} \end{aligned} \tag{9.132}$$

The winning prototype $\mathbf{L}_i$ is adapted iteratively based on the following rule:

$$\Delta \mathbf{L}_i = -\eta' \frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_i} = \eta(\mathbf{x}-\mathbf{L}_i)\left(1+\sum_{i\neq r}^{c} w_{ir}\right) \tag{9.133}$$

where $\eta = 2\eta'$ and

$$w_{ir} = p\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) + \frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2} p'\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) \tag{9.134}$$

Since $u(x) = xp(x)$, we obtain $u' = p(x) + xp'(x)$ and also $w(x) = p(x) + xp'(x)$. Equation (9.134) can be similarly expressed as

$$w_{ir} = u'\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) = w\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) \tag{9.135}$$

For the nonwinning prototypes we obtain based on the minimization of $D_{\mathbf{x}}$

$$\begin{aligned}\frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_j} &= \frac{\partial}{\partial \mathbf{L}_j}\left[||\mathbf{x}-\mathbf{L}_i||^2 + \sum_{r\neq i}^{c}||\mathbf{x}-\mathbf{L}_i||^2 p\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right)\right] \\ &= -2(\mathbf{x}-\mathbf{L}_j)\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_j||^2}\right)^2 p'\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_j||^2}\right) \end{aligned} \tag{9.136}$$

The nonwinning prototype $\mathbf{L}_j \neq \mathbf{L}_i$ is adapted iteratively based on the following rule:

$$\Delta \mathbf{L}_j = -\eta' \frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_j} = \eta(\mathbf{x} - \mathbf{L}_j) n_{ij} \tag{9.137}$$

where $\eta = 2\eta'$ and

$$n_{ij} = -\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2}\right)^2 p'\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2}\right) \tag{9.138}$$

Since $u'(x) = p(x) + xp'(x)$, $x^2 p'(x) = xu'(x) - xp(x) = xu'(x) - u(x)$. Thus we get for eq. (9.138) the following expression

$$\begin{aligned} n_{ij} &= u\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2}\right) - \frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2} u'\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2}\right) \\ &= n\left(\frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2}\right) = u_{ij} - \frac{||\mathbf{x} - \mathbf{L}_i||^2}{||\mathbf{x} - \mathbf{L}_j||^2} w_{ij} \end{aligned}$$

We also obtain $n(x) = -x^2 p'(x) = u(x) - xu'(x)$.

The above strategy represents the framework for the derivation of fuzzy learning vector quantization algorithms as given in [100]. Table 9.1 shows the membership functions and interference functions $w(\cdot)$ and $n(\cdot)$ that generated three distinct fuzzy LVQ algorithms.

TABLE 9.1 Membership functions and interference functions for the FALVQ1, FALVQ2, and FALVQ3 families of algorithms.

Algorithm	$u(z)$	$w(z)$	$n(z)$
FALVQ1 $(0 < \alpha < \infty)$	$z(1+\alpha z)^{-1}$	$(1+\alpha z)^{-2}$	$\alpha z^2 (1+\alpha z)^{-2}$
FALVQ2 $(0 < \beta < \infty$	$z \exp(-\beta z)$	$(1-\beta z)\exp(-\beta z)$	$\beta z^2 \exp(-\beta z)$
FALVQ3 $(0 < \gamma < 1)$	$z(1-\gamma z)$	$1 - 2\gamma z$	γz^2

An algorithmic description of the FALVQ is given below:

1. **Initialization:** Choose the number c of prototypes, a fixed learning rate η_0, and the maximum number of iterations N. Set the iteration counter equal to zero, $\nu = 0$. Generate randomly an initial codebook $\mathbf{L} = \{\mathbf{L}_{1,0}, \ldots, \mathbf{L}_{c,0}\}$.
2. **Adaptation, part I:** Compute the updated learning rate $\eta = \eta_0 \left(1 - \frac{\nu}{N}\right)$. Also set $\nu = \nu + 1$.
3. **Adaptation, part II:** For each input vector $\mathbf{x}$ find the winning prototype based on the equation

$$||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2 < ||\mathbf{x} - \mathbf{L}_{j,\nu-1}||^2, \quad \forall j \neq i \tag{9.139}$$

Determine the membership functions $u_{ir,\nu}$ using

$$u_{ir,\nu} = u\left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right), \quad \forall r \neq i \tag{9.140}$$

Determine $w_{ir,\nu}$ using

$$w_{ir,\nu} = u'\left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right), \quad \forall r \neq i \tag{9.141}$$

Determine $n_{ir,\nu}$ using

$$n_{ir,\nu} = u_{ir,\nu} - \left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right) w_{ir,\nu}, \quad \forall r \neq i \tag{9.142}$$

4. **Adaptation, part III:** Determine the update of the winning prototype $\mathbf{L}_i$ using

$$\mathbf{L}_{i,\nu} = \mathbf{L}_{i,\nu-1} + \eta(\mathbf{x} - \mathbf{L}_{i.\nu-1})(1 + \sum_{r \neq i}^{c} w_{ir,\nu}) \tag{9.143}$$

Determine the update of the nonwinning prototype $\mathbf{L}_j \neq \mathbf{L}_i$ using

$$\mathbf{L}_{j,\nu} = \mathbf{L}_{j,\nu-1} + \eta(\mathbf{x} - \mathbf{L}_{j,\nu-1})n_{ij,\nu} \tag{9.144}$$

5. **Continuation:** If $\nu = N$ stop, else go to step 2.

9.7.1 Medical Image Coding

In [184] is presented a novel image-compression system designed for digital mammograms using wavelet-based image decomposition and vector quantization.

Codebook design was performed in the experiments by a family of FALVQ and the Linde-Buzo-Gray algorithm [142]. The FALVQ algorithms were tested against the LBG algorithm because of its implementation simplicity and satisfactory performance. The advantage of the FALVQ algorithms over the traditional LBG is that the initial codebook is generated randomly. However, the adaptation of both the winning and nonwinning prototypes are computationally extensive for the FALVQ algorithms. But this aspect can be controlled by the user, because the learning process terminates after a specified number of iterations. On average, the FALVQ and LBG algorithms are comparable in terms of their computational complexity and implementation simplicity. In [184] it was shown that the codebook design based on FALVQ algorithms outperformed the design based on the LBG algorithm as the length of the quantized vectors increased.

Figures 9.11 and 9.12 show an original mammogram and its reconstructed images based on a wavelet filter (Daubechies' 20 coefficient filter) and FALVQ1, FALVQ2, FALVQ3, and LBG used for codebook design. The

original mammogram was compressed at 1.16 bpp. In the original mammogram, there is a very small black dot embedded inside a microcalcification called a sebaceous calcification. In each of these three images, the same area appeared to be identical.

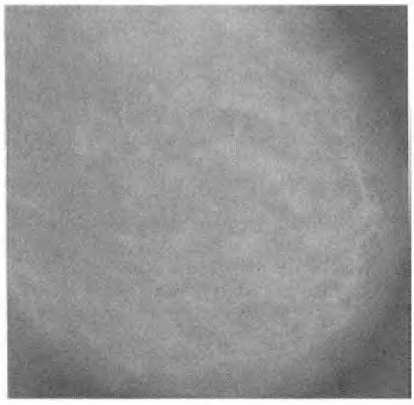


FIGURE 9.11 Original mammogram. (Image from [184] reprinted with permission from Springer-Verlag.)

EXERCISES

1. Prove eq. (9.22).
2. Prove the adaptive norm theorem for the generalized adaptive fuzzy n-means algorithm.
3. Prove for the generalized fuzzy n-shells algorithm both the theorem for optimal fuzzy partition and the theorem for optimal prototypes.
4. Prove the adaptive norm theorem for the generalized adaptive fuzzy n-shells algorithm.
5. Prove for the generalized adaptive fuzzy n-shells algorithm the propositions for optimal partition, for optimal prototype centers, and for optimal prototype radii.

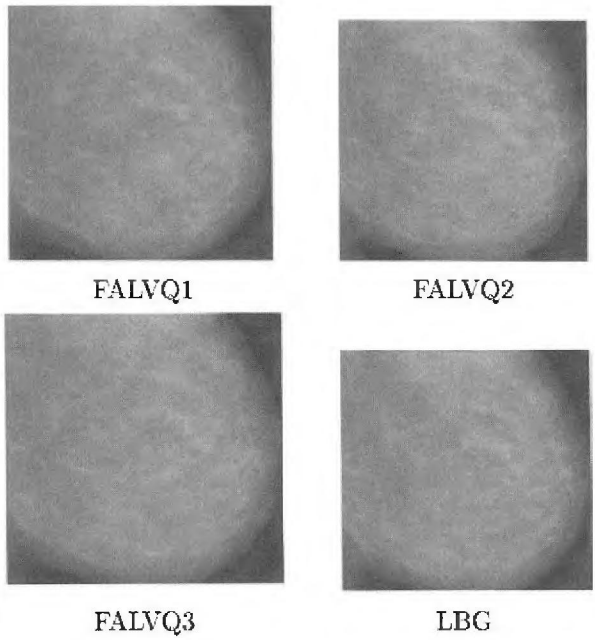


FIGURE 9.12 Reconstructed mammogram using FALVQ1, FALVQ2, FALVQ3, and LBG for codebook design. (Images from [184] reprinted with permission from Springer-Verlag.)

X

SPECIALIZED NEURAL NETWORKS RELEVANT TO BIOIMAGING

10.1 INTRODUCTION

Neural networks have demonstrated a growing importance in the area of biomedical image processing and have been increasingly used for a variety of biomedical imaging tasks. The applications scan a wide spectrum: detection and characterization of disease patterns, analysis (quantification and segmentation), compression, modeling, motion estimation, and restoration of images from a variety of imaging modalities: magnetic resonance, positron emission tomography, ultrasound, radiography, mammography, and nuclear medicine. At the same time, traditional artificial neural networks and nonbiological image processing have demonstrated limited capabilities when applied to integrated medical imaging systems or to data analysis [53, 64, 98, 109, 121, 171, 215, 245, 246]. A better understanding of brain-based mechanisms of visual processing would potentially enable dramatic progress in clinical medical imaging. This fact has led to the development of new neural architectures derived from the basic architectures but with emphasis on incorporating visual processing mechanisms and cognitive decision making [154].

In Chapter VII, we described the foundations of neural networks. This chapter presents some of the most important neural architectures relevant to bioimaging and their applications to several bioimaging tasks. In general, medical image patterns either possess a circular symmetric shape (e.g., nodules) or appear as small objects with a variety of geometric patterns (e.g., calcifications).

The applications in bioimaging considered most important and their underlying required neural architectures are as follows:

- Disease pattern classification: Based on convolutional neural networks [144, 210], hierarchical pyramid neural networks [213], factorization MLPs, and radial basis neural networks [233]
- Medical image restoration: Based on a modified Hopfield neural network [108, 182]
- Medical image segmentation: Based on RBF neural networks [255] and Hopfield neural networks with a priori image models [75]
- Tumor boundary detection: Based on a modified Hopfield neural network [254]
- Medical image coding and compression: Based on robust principal component analysis [97] and combined neural architectures

10.2 BASIC ASPECTS OF SPECIALIZED NEURAL NETWORK ARCHITECTURES

Specialized neural network architectures implement processing techniques relevant to most bioimaging applications, such as invariant and context-based classification, optimization, and elastic contour models.

10.2.1 Invariant Neural Networks

Since most medical image patterns either possess a circular symmetric shape or appear as small objects with a variety of geometric patterns, image pattern recognition does not call on top-down or left-right geometry as classification criteria. Therefore, one can take advantage of this characteristic as an invariance. In other words, we can rotate and/or shift the input vector two-dimensionally and maintain the same output assignments for the training. This method has two fundamental effects on the neural network:

- It trains the neural network that the rotation and shift of the input vector would receive the same classification result.
- It increases the training data and thus improves the performance of the neural network.

Neural networks in bioimaging operate directly on images without extracting features out of these images. It is known that the neighborhood

correlation is usually higher than the long-distance correlation between two pixels on an image. It is conceivable that features associated with nearby pixels should be emphasized. In neural network terms, the local interactions should be included in the training algorithm rather than the nonlocal interactions. A good example of this type of approach is found in [144, 210, 253].

10.2.2 Context-Based Neural Networks

An important problem in image analysis is finding small objects in large images. This problem is challenging for bioimaging, because:

- Searching a large image is computationally intensive.
- Small objects, in most cases on the order of a few pixels in size, have relatively few distinctive features which enable them to be distinguished from nontargets.

By embedding multiresolution pyramid processing in a neural network architecture, we gain several advantages over traditional neural processing: (1) Both training and testing can be performed at a reduced computational cost based on a coarse-to-fine paradigm, and (2) learning of low-resolution contextual information facilitates the detection of small objects. A good example of this type of approach is found in [213].

10.2.3 Optimization Neural Networks

Image segmentation can also be achieved using unsupervised clustering algorithms [79, 23]. A typical algorithm, such as c–means [120], determines a class partitioning by minimizing the following objective function:

$$J = \sum_{i=1}^{d} ||\mathbf{x_i} - \mathbf{m}_{c(\mathbf{x_i})}||^2 \tag{10.1}$$

where $\mathbf{x_i}$ is the pattern vector, d the number of pattern vectors, $\mathbf{m}_{c(\mathbf{x_i})}$ the closest cluster centroid to $\mathbf{x_i}$, and C the number of classes. Using this type of algorithm, image segmentation can be formulated as an optimization problem. The Hopfield neural network model and its variants operate by minimizing an energy function. The topology of the Hopfield network employed in medical image segmentation can be visualized as a cube of nodes with width, depth, and height as shown in Figure 10.1. The cube can be visualized as a three-dimensional grid of nodes where the number of nodes in the plane described by the width and depth corresponds to the pixels in the image, while the height of the cube, in terms of nodes, corresponds to the predefined number of clusters necessary for segmentation of this image. In order to minimize the energy function of the neural network, only one of the M neurons has to be active for each pixel. When convergence of the

neural networks is achieved, the label represented by each pixel's active node was assigned to the pixels to achieve a segmentation of the image. A good example of this type of approach is found in [9, 75].

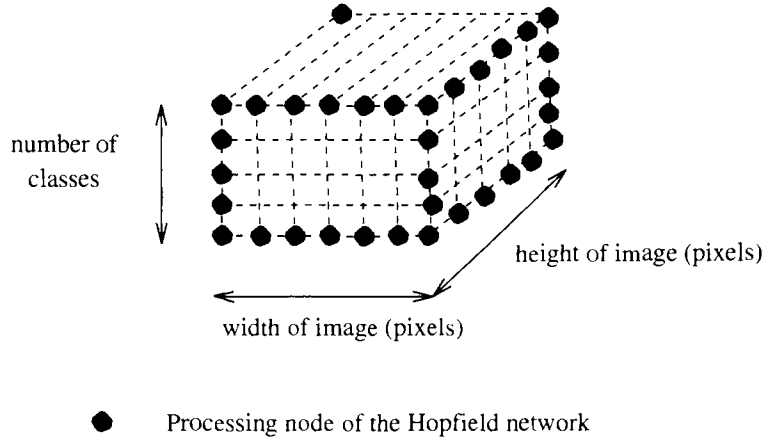


FIGURE 10.1 Image segmentation based on Hopfield network. The width and depth of the cube, in terms of nodes, corresponds to the number of pixels in a given image, while the height of the cube, in terms of nodes, corresponds to the prespecified number of clusters into which the image must be segmented.

10.2.4 Active Contour Models

An important problem in bioimaging is to extract the boundary of an area of interest. The boundary detection problem can be formulated as an optimization problem based on an active contour model. The boundary is detected by seeking an active contour that minimizes an energy functional. The basic active contour model is an energy-minimizing spline [107] which is influenced by internal contour forces, image forces, and external forces. The internal forces serve to impose a piecewise smoothness constraint. The image forces push the contour toward salient image features such as lines, edges, and subjective contours such that it becomes a good fit of the image data features. The external forces are responsible for ensuring that the contour reaches the desired local minimum. The energy functional has to be carefully chosen so that it captures the relevant image properties. The goal is to achieve an optimized model such that a set of particular features in an image can be located. Two major factors dramatically influence the success of the elastic model: the initial position and the external energy. The assumed initial position is critical, since it has a dramatic impact on the convergence properties. Thus if it is not close to the desired features, the contour converges to the wrong solution. A robust solution can be achieved by employing a Hopfield neural network as shown in [254]. The

most relevant applications for active contour models can be found in PET and MRI [98, 110].

10.3 CONVOLUTION NEURAL NETWORKS (CNNS)

Convolution neural networks (CNN) represent a well-established method in medical image processing [144, 210]. Although it has a multilayer architecture like the MLP, when applied to image classification the CNN works directly with images and not with extracted features like the MLP. The CNN approach is based on the idea of neighborhood correlation: The correlation of adjacent pixels is usually higher than of nonadjacent. Therefore, it makes sense to emphasize features associated with adjacent pixels. In neural network terms, the local interactions should be included in the training algorithm rather than the nonlocal interactions. The basic structure of a CNN is shown in Figure 10.2, which represents a four-layer CNN with two input images, three image groups in the first hidden layer, two groups in the second hidden layer, and a real-valued output [210]. The number of layers and the number of groups in each layer are implementation-oriented. The more complicated the disease patterns, the more layers are required to distinguish high-order information of image structures. The idea behind the convolution kernels is to emphasize important image characteristics rather than those less correlated values obtained from feature spaces for input. These characteristics are: (1) the horizontal versus vertical information; (2) local versus nonlocal information: and (3) the image processing (filtering) versus signal propagation.

The image propagates through the neural network from input layer to output layer by means of convolution with trainable weight kernels. In the following, we will describe the forward and backward propagation through the CNN.

10.3.1 Forward Propagation

Forward propagation pertains to the image propagation in the CNN from the input layer ($l = 1$) to the output layer ($l = L$) [210]. Let $H_{l,g}$ define the gth image group at layer l, and let $N(l)$ describe the number of such groups. The image $H_{l,g}(l \geq 2)$ is determined by applying a pointwise sigmoid nonlinearity to an intermediate image $I_{l,g}$, that is,

$$H_{l,g}(i,j) = \frac{1}{1 + \exp(-I_{l,g}(i,j))}, \quad g = 1, \ldots, N(l) \tag{10.2}$$

The intermediate image $I_{l,g}$ represents a superposition of the images obtained from the convolution of $H_{l-1,g'}$ at layer $l-1$ with trainable kernel of weights $w_{l-1,g,g'}$. Specifically, we obtain $I_{l,g}$ from

$$I_{l,g} = \sum_{g'=1}^{N(l-1)} H_{l-1,g'} * *w_{l-1,g,g'} \tag{10.3}$$

where $**$ defines a 2D convolution with the 2D kernel $w_{l-1,g,g'}$ of weights connecting the $g'th$ group in the $(l-1)$th layer with the gth group in the lth layer.

The spatial width $S_w(l-1)$ of the weight kernel $w_{l-1,g,g'}$ defines the receptive field for the layer l. The spatial width $S_H(l)$ of an image at layer l can be expressed as a function of the image width at the layer $l-1$ as

$$S_H(l) = S_H(l-1) - S_w(l-1) + 1 \tag{10.4}$$

It is evident that the image width becomes smaller as the layer number increases. However convolution introduces an edge effect which can be easily avoided by defining the width of the receptive field of a given node in the lth layer as the sum of the kernel widths of the preceding layers minus $(l-2)$. The spatial width of the image at the output layer $(l = L)$ is 1, and thus the output of the CNN, defined as $O(g) \equiv H_{L,g}(0,0)$, becomes a real number.

The MLP represents a special case of a CNN, both having similar fundamental equations. By replacing the weight kernels and the image groups by real numbers, the CNN architecture turns into a standard MLP architecture. The weight kernels become ordinary weights, while the images become nodes. If for the weight kernels and image groups in a CNN we substitute real numbers, then we get ordinary MLP weights for the weight kernels and nodes for the images.

10.3.2 Backpropagation

For the training of the CNN, we employ the backpropagation algorithm. To each training image p (or set p of training images in case the input layer processes more than one image) there is a correspondent desired-output value $O_d^{(p)}(g)$, where $g = 1, \ldots, N(L)$ describes the output node number. As in case of the MLP, for each iteration t, training images are presented in a random fashion to the CNN and the resulting CNN outputs $O_a^{(p)}[t]$ are determined based on eqs. (10.2) and (10.3). The resulting CNN output error for a given training image p at iteration t is given by

$$E^p[t] = \frac{1}{2} \sum_{g=1}^{N(L)} (O_d^{(p)}(g) - O_a^{(p)}(g)[t])^2 \tag{10.5}$$

while the total CNN error at iteration t is given by

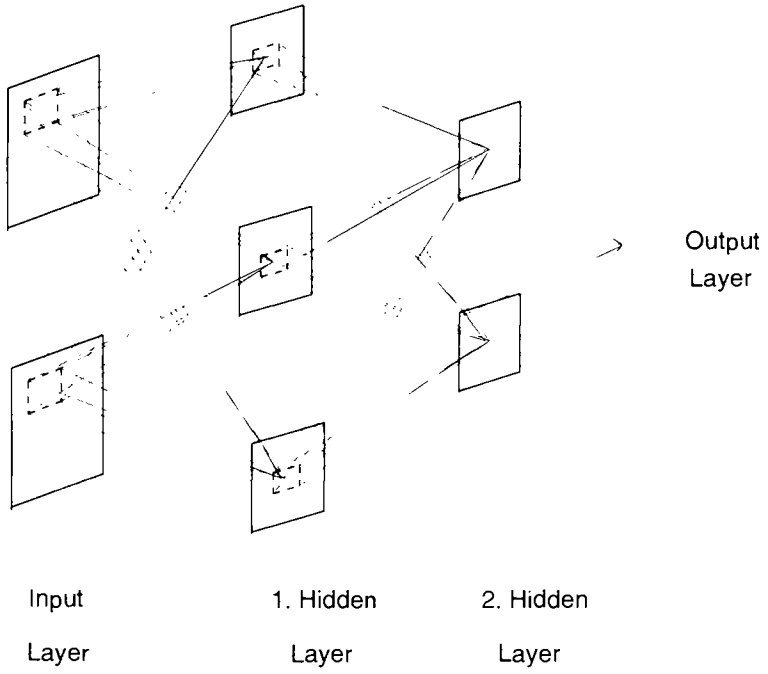


FIGURE 10.2 Convolution neural network.

$$E[t] = \sum_{p=1}^{P} E^{(p)}[t] \tag{10.6}$$

where p represents the total number of training samples.

It is easy to see that for a CNN, the weight adjustment is based on a backpropagation process. [210] gives the derivation of the backpropagation algorithm for the CNN.

The kernel weights $w_{l.g.g'}$ are derived by the generalized delta rule

$$w_{l,g,g'}(i,j)[t+1] = w_{l.g.g'}(i,j)[t] - \eta \Delta w_{l.g,g'}(i,j)[t] \tag{10.7}$$

with

$$\Delta w_{l,g.g'}(i,j)[t] = \frac{\partial E[t]}{\partial w_{l,g,g'}(i,j)[t]} \tag{10.8}$$

The initialization process for the CNN is based on assigning the kernel weights and the other weighting factors a normalized random number. For the simulations, it is necessary to normalize the pixel values of the training

and test images such that a 1.0 is assigned to the highest pixel value, and a 0 to the lowest pixel value.

The two-dimensional convolution operation represents an emulation of the radiologists' viewing of a suspected area, while the output side models their decision-making process. The neural network is trained based on a backpropagation algorithm such that it extracts from the center and the surroundings of an image block relevant information describing local features.

EXAMPLE 10.3.1: An application of a CNN to mammograms is shown in [145]. The mammograms were digitized with a computer format of 2048×2500×12 bits per image. The study pertains only to microcalcification detection and utilizes only the central region of 16×16 pixels as an input to evaluate the performance of four different neural networks: dynamic stable associate learning (DYSTAL) [5], conventional backpropagation neural network with (BP/1H) and without (BP/0H) a hidden layer, and the CNN. The DYSTAL uses the winner-take-all approach of propagating maximum similarity and employs a threshold value for deciding if the maximum similarity of a new pattern vector is less than it. If so, the new pattern becomes the representative of a false or true disease class.

Thirty-eight digital mammograms including 220 true and 1132 false subtle microcalcifications were considered for the study. The mammograms were divided into two sets: A_m, 19 images (containing 108 true and 583 false image blocks), and B_m, another set of 19 images (containing 112 true and 549 false image blocks). The central region of 16×16 pixels was considered as an input for the CNN, while the convolution kernel had the size of 5×5. This is perfectly sufficient since the microcalcifications are very small compared to other diseases. Tables 10.1 and 10.2 show the performance resulting from the three neural systems [145].

TABLE 10.1 Performance of neural networks in the detection of clustered microcalcifications using group A_m as training set and group B_m as testing set [145].

Neural networks	DYSTAL	BP/0H	BP/1H	CNN
A_Z (area under the ROC curve)	0.78	0.75	0.86	0.97
Detection accuracy % true-positive detection # false-positive per image	$\frac{70}{4.3}$	$\frac{70}{4.5}$	$\frac{75}{3.5}$	$\frac{90}{0.5}$

The best A_Z of 0.97 was achieved by the CNN, thus demonstrating the superiority of CNN over the other classifiers.

10.4 HIERARCHICAL PYRAMID NEURAL NETWORKS

One of the most challenging tasks in biomedical imaging is to find small disease patterns in large images. A special architecture, the so-called hier-

TABLE 10.2 Performance of neural networks in the detection of clustered microcalcifications using group B_m as training set and group A_m as testing set [145].

Neural networks	DYSTAL	BP/0H	BP/1H	CNN
A_Z (area under the ROC curve)	0.76	0.77	0.84	0.90
Detection accuracy % true-positive detection # false-positive per image	$\frac{70}{4.3}$	$\frac{70}{4.2}$	$\frac{75}{3.7}$	$\frac{90}{0.5}$

archical pyramid neural network, has proved to be superior to both conventional neural network architectures and standard statistical classification techniques [213]. By embedding multiresolution pyramid processing in a neural network architecture, several advantages over traditional neural processing can be gained: (1) both training and testing can be performed at a reduced computational cost based on a coarse-to-fine paradigm and (2) learning of low-resolution contextual information enables the detection of small objects.

The hierarchical pyramid neural network architecture is shown in Figure 10.3.

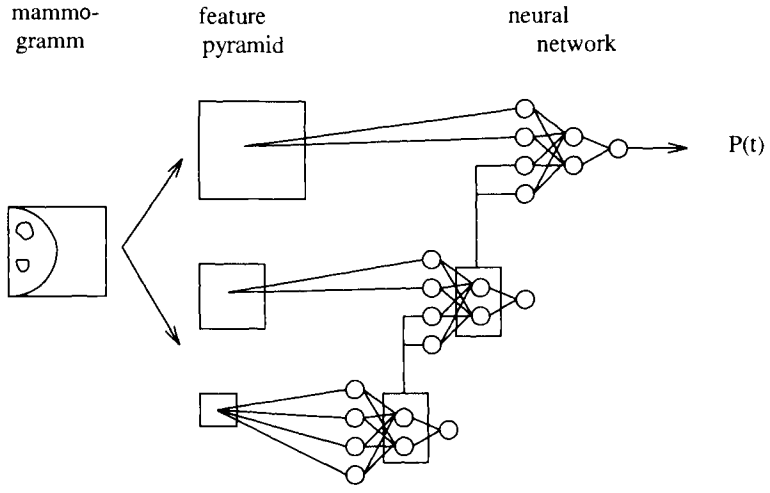


FIGURE 10.3 Hierarchical pyramid neural network architecture for learning context. The hidden units of low-resolution networks propagate hierarchically the context, such that the output of the highest resolution network is a probability estimator for the presence of a target [213].

A hierarchy of networks is trained in a coarse–to–fine paradigm on a feature set derived from a pyramid decomposition of the image. Low-

resolution networks are first trained to detect small objects, which might be absent at this level of the pyramid. In order to detect small objects, low-resolution networks must learn the context in which these small objects exist. To include the context information with the feature information of the small objects, the outputs of the hidden units from the low-resolution networks are propagated hierarchically as inputs to the corresponding higher resolution neural networks. The neural nets in the HPNN are multilayer perceptrons, having one hidden layer with four hidden units. The inputs to the networks are features at different levels of an image pyramid with outputs $P(t)$, representing the probability that a target is present at a given location in the image.

The error function used for the training is chosen as a cross-entropy error:

$$E = -\sum_i d_i \log y_i - (1 - d_i) \log(1 - y_i) \tag{10.9}$$

where $d \in \{0, 1\}$ represents the desired output. To avoid overtraining, a regularization term of the form $r = \frac{\lambda}{2} \sum_i w_i^2$ is added to the total error on the training examples. λ is adjusted to minimize the cross-validation error.

EXAMPLE 10.4.1: The HPNN was successfully applied for detection of microcalcifications [213]. It outperforms the Chicago neural network [252] in terms of its ability to reduce the number of false-positive ROIs generated by the computer.

Figure 10.4 [213] shows how the system takes advantage of the context information by looking at the representations developed by various hidden units in the network. Two classes of hidden units were found. The first one shown in Figure 10.4b describes pointlike structure similar to that of individual microcalcifications. The second class of hidden unit in Figure 10.4c represents ductal location and appears to be tuned for longer, extended, and oriented structures. This example proves the HPNN's ability to automatically extract and process context information.

10.5 PROBLEM FACTORIZATION

The complexity of training can be dramatically reduced by using the problem factorization approach. This is a collection of small interconnected MLPs such that a factorization of the classification task is achieved by considering separate classification stages. The information is thus processed hierarchically. Also an overtraining can be avoided based on this method.

EXAMPLE 10.5.1: The problem factorization approach can be used for the classification of candidate tumor regions. In [55] they are employed to classify clusters of regions that correspond to acoustic neuromas. Figure 10.5 shows a collection of three small interconnected MLPs that reflect a factorization of the classification task into three separate stages. The first

(a) (b) (c)

FIGURE 10.4 (a) Digitized mammogram, (b) hidden unit representing pointlike structure, and (c) hidden unit representing elongated structure. (Reprinted from [213] with permission from SPIE.)

two stages capture only the shape and the intercluster position (ICP), with each of them having four distinct features. The final stage combines the results from the previous two stages with two additional features, namely compactness and global angle.

By factorizing into the three stages, the classifier was reduced to three MLPs that were of lower complexity that a single MLP. Each of the three MLPs was trained separately. The data available constrain each of the three MLPs to a greater extent than single MLPs are constrained. Therefore, overfitting using a hierarchy of MLPs is highly reduced compared to use of a single larger MLP. For the simulations reported in [55], the data available were split into only two data sets, one for the training and the other for testing. The training data set consisted of 35 tumor regions and 70 nontumor. The testing data set contained 15 tumor regions and 35 nontumors. Table 10.3 shows a performance comparison among three different types of MLP architectures: single MLP versus committee of MLPs and versus factorized MLPs. The classifier that achieved the best results used the problem factorization approach.

10.6 MODIFIED HOPFIELD NEURAL NETWORK

The problem of restoring noisy-blurred images is important for many medical image applications, especially in confocal microscopy [63, 108, 226].

The goal of image restoration is to "undo" the blurring and noise artifacts that are imposed on the image during image acquisition. In other words, image restoration algorithms try to restore the "original" image from the acquired image by deconvolving the blurring imposed by the point spread function (PSF) and by reducing the noise imposed by the image recording. Linear restoration methods have a major drawback: They are incapable of restoring frequencies for which the PSF has a zero response. Furthermore they cannot restrict the domain in which the solution should

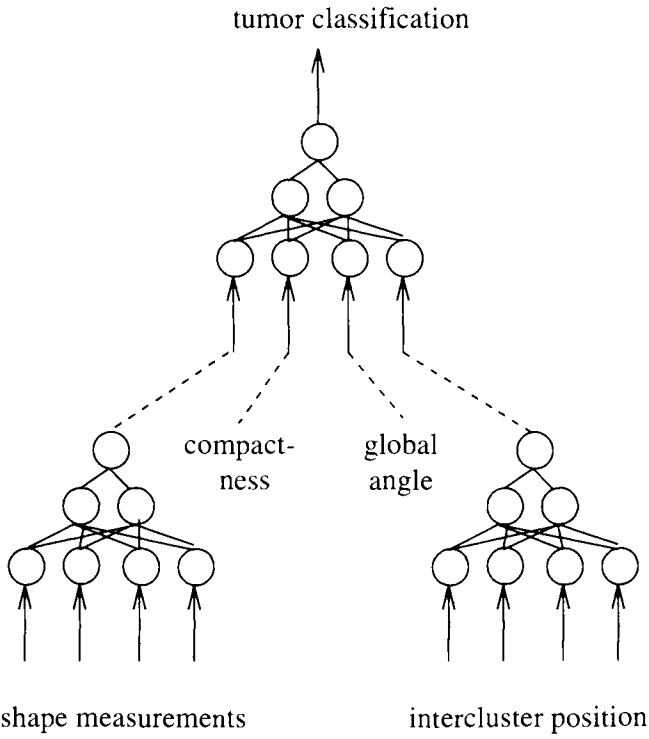


FIGURE 10.5 Classifier structure of the problem factorization approach to candidate tumor region combination classification.

be found. This property is a major drawback as the intensity of an imaged object represents light energy, which is nonnegative.

For these reasons, nonlinear iterative image-restoration algorithms are the best candidates for restoring images contaminated by Poisson noise. Nonlinear iterative algorithms mostly follow two approaches: either the constrained Tikhonov restoration approach or the maximum likelihood restoration approach. They all tackle the above-mentioned problems in exchange for a considerable increase in computational complexity. To overcome both the increase in complexity and the linear methods' drawbacks, neural-network-based image restoration algorithms were proposed and successfully applied to image restoration problems in many application fields.

Often, the image degradation can be adequately modeled by a linear blur and an additive white Gaussian process. Then the degradation model is given by

$$\mathbf{z} = \mathbf{D}\mathbf{x} + \eta \tag{10.10}$$

TABLE 10.3 The best performance of the three types of classifier against the testing data set. The training scheme employed was both early stopping as well as no early stopping. Values not in parantheses are the error rates against tumors regions that correspond to acoustic neuromas, and values in parantheses are the error rates against tumor regions that did not correspond to acoustic neuromas [55].

Classifier structure	No Early Stopping	Early Stopping
Single MLP	6.67% (5.71%)	0% (17.14%)
Committee of MLPs	13.33% (5.71%)	6.67% (14.29%)
Factorized MLPs:		
... shape	0% (20.00%)	6.67% (17.14%)
... Intercluster Position	0% (17.14%)	0% (22.86%)
... Final	0% (5.71%)	0% (5.71%)

where $\mathbf{x}, \mathbf{z}$, and η represent the original and degraded images and the additive noise. The matrix $\mathbf{D}$ describes the linear spatially invariant or spatially varying distortion. Specifically in confocal microscopy, the blurring matrix represents the PSF of the microscope.

The purpose of digital image restoration is to invert the degradation process by turning the degraded image $\mathbf{z}$ into one that is as close to the original image $\mathbf{x}$ as possible, subject to a suitable optimality criterion. A common optimization problem is:

$$\text{minimize} \quad \mathbf{f}(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{T}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{x} \tag{10.11}$$

$$\text{subject to} \quad 0 \leq x_i \leq 255, \quad i = 1, \ldots, n \tag{10.12}$$

where x_i denotes the ith element of the vector $\mathbf{x}$, $\mathbf{b} = \mathbf{D}^{\mathrm{T}}\mathbf{z}$, and $\mathbf{T}$ is a symmetric, positive semidefinite matrix equal to

$$\mathbf{T} = \mathbf{D}^{\mathrm{T}}\mathbf{D} + \lambda \mathbf{C}^{\mathrm{T}}\mathbf{C} \tag{10.13}$$

In eq. (10.13), $\mathbf{C}$ represents a highpass filter and λ the regularization parameter, describing the interplay between deconvolution and noise smoothing.

Comparing eqs. (10.11) and (10.13), it becomes evident that looking for the minimum of the function $f(\mathbf{x})$ for the restoration problem corresponds to finding the minimum of the energy function E_{hs} of the Hopfield neural network, if we choose $\mathbf{W} = -\mathbf{T}$ and $\mathbf{x} = \mathbf{v}$.

The Hopfield neural network operates with binary patterns, and in order to employ it for image restoration, we have to transform the image gray values into binary state variables. A solution proposed in [108] is to represent them as a simple sum of the binary neuron state variables. The main disadvantage of this strategy is the resulting tremendous storage and number of interconnections. This problem can be overcome by defining a

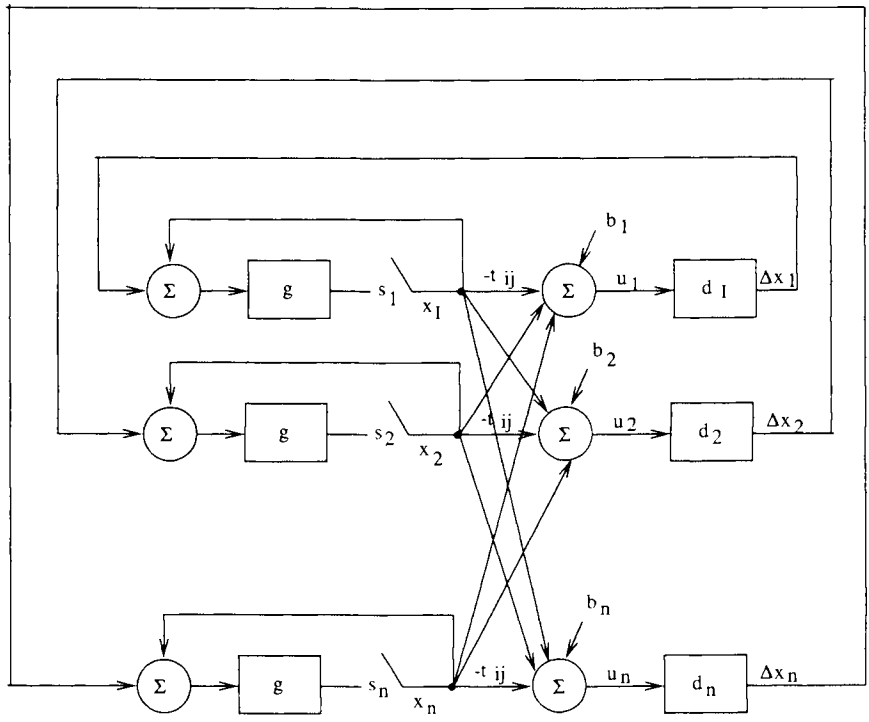


FIGURE 10.6 Block diagram of the modified Hopfield network model applied to image restoration.

nonelementary neuron that takes discrete values between 0 and 255, instead of binary values [182]. In this case, the interconnections are based on pixel locations and not on gray values.

10.6.1 Updating rule:

The modified Hopfield network for image restoration that was proposed in [182] is shown in Figure 10.6 and is given by the following equations:

$$x_i(t+1) = g(x_i(t) + \Delta x_i), \quad i = 1, \ldots, n \tag{10.14}$$

where

$$g(v) = \begin{cases} 0, & v < 0 \\ v, & 0 \leq v \leq 255 \\ 255, & v > 255 \end{cases} \tag{10.15}$$

$$\Delta x_i = d_i(u_i) = \begin{cases} -1, & u_i < -\theta_i \\ 0, & -\theta_i \leq u_i \leq \theta_i \\ 1, & u_i > \theta_i \end{cases} \tag{10.16}$$

with

$$\theta_i = \frac{1}{2} t_{ii} > 0 \quad \text{and} \quad u_i = b_i - \sum_{j=1}^{n} t_{ij} x_j(t) \tag{10.17}$$

The degraded image $\mathbf{z}$ is used as the initial condition for $\mathbf{x}$. x_i are the states of neuron that take discrete values between 0 and 255, instead of binary values. The symbol s_i denotes a switch, which describes either a sequential or asynchronous updating mode.

In the following, an algorithm is presented that sequentially updates each pixel value according to the updating rule. Let $l(t)$ denote a partition of the set $\{1, \ldots, n\}$. The algorithm has the following form:

ALGORITHM 2.

1. $\mathbf{x}(0) = \mathbf{D}^{\mathbf{T}}\mathbf{z}; t := 0$ and $i := 1$.
2. Check termination.
3. Choose $l(t) = \{i\}$.
4. $temp = g(x(t) + \Delta x_i e_i)$ where Δx_i is given by eq. (10.16).
5. If $temp \neq x(t)$ then $x(t+1) := temp$ and $t := t + 1$.
6. $i; = i + 1$ (if $i > n, i = i - n$) and go to step 1.

In step 3 of the preceding algorithm, the function $g(.)$ is applied separately to each component of its argument vector. In this case $g(\mathbf{x}) = [g(x_1), \ldots, g(x_n)]$, where $g(x_i)$ is defined by eq. (10.15.)

EXAMPLE 10.6.1: Modified Hopfield neural networks are often applied in nuclear medicine image restoration [201]. Figure 10.7 shows an application of the algorithm to the image restoration problem by using an artificial phantom source image study. An image of 50×50 pixels in size represents a cylindrical object with a diameter of 7 pixels as reported in [201]. This image simulates a cylindrical source of activity. The blurred image is restored by a modified Hopfield neural network.

10.7 HOPFIELD NEURAL NETWORK USING A PRIORI IMAGE INFORMATION

The Hopfield neural network can also be applied to image segmentation, if the segmentation is posed as an optimization problem. In [75] a general framework for the design of the optimization criterion is proposed consisting of two main parts: one based on the observed image, and another based on an a priori image model. The approach exploits a well-known fact in statistical image segmentation technique, namely that neighboring pixels in an image are "similar" in some sense. Therefore, by not only using information about observed data but also including the local spatial interaction between pixel classes, a gain in segmentation results can be achieved.

The pixel classification problem can be viewed as an optimization problem: A feature vector, containing the gray values and additional data such

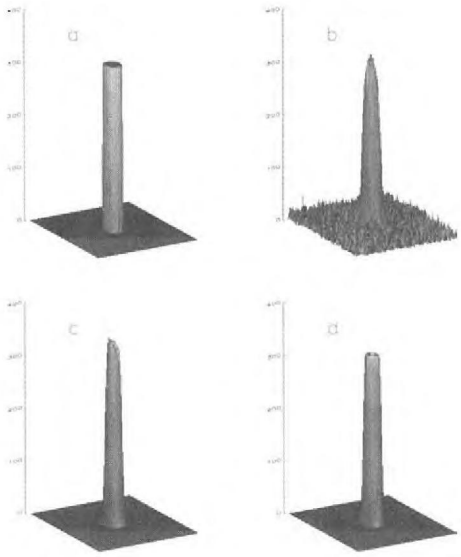


FIGURE 10.7 Image restoration based on a modified Hopfield neural network using artificial data. (a) Original image with a signal simulating a cylindrical source of activity, (b) blurred image by the PSF and additive noise, and (c) restoration result achieved by the modified Hopfield neural network. (Reprinted from [201] with permission from Kluwer.)

as edge information and local correlation values, describes each discrete pixel location within an $N \times N$ image. Let $\mathbf{p}^i$ with $1 \leq i \leq N^2$ be such an M-dimensional feature vector for each image pixel. If we assume L a priori known underlying classes, then each pixel belongs to one of these L classes.

10.7.1 Formulation of the Classification Problem

Given a known number of classes L, assign each pixel of the observed image to one of these classes.

Since a priori information has to be incorporated in the image models, a special optimization function for the L-class segmentation problem is defined:

$$J(X) = \sum_{i=1}^{N^2} J(\mathbf{p}^i, \mathbf{m}_1, \ldots, \mathbf{m}_L, \mathbf{x}_i) - \lambda f(X) \tag{10.18}$$

with $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N^2}\}$ being the set of binary L-dimensional class vectors $\mathbf{x}_i$. In addition, we impose that $x_{il} \in \{0, 1\}$ and $\sum_l x_{il} = 1$ for $1 \leq i \leq N^2$, $1 \leq l \leq L$. $\mathbf{m}_l$ with $l = 1, \ldots, L$ represent the cluster centers. $f(X) = f(\{\mathbf{x}_1, \ldots, \mathbf{x}_{N^2}\})$ describe the prior available information about the underlying image, with λ being a regularization parameter emphasizing the a priori information with respect to the data-independent term $J(\mathbf{p}^i, \mathbf{m}_1, \ldots, \mathbf{m}_L, \mathbf{x}^i)$.

The minimum of the function $J(X)$ represents the optimal solution of the classification problem. As a result, we obtain the required L cluster centers $\mathbf{m}_l$ with $1 \leq l \leq L$.

The optimization function, in its general form, can be simplified and becomes at the same time more feasible for applications. A good choice is [75]

$$J(X) = \sum_{l=1}^{L} ||(\mathbf{p}_i - \mathbf{m}_l||_2^2 x_{il} + \lambda \sum_{j \in \eta_i} \mathbf{x}_j^T \mathbf{x}_i \tag{10.19}$$

η_i is the set of all pixels which are neighbors of pixel i. A first-order neighborhood contains the four nearest neighbors of a pixel.

10.7.2 Algorithm for Iterative Clustering

The number of neurons in the Hopfield neural network corresponds to the number of pixels in the image. The energy of an $N \times N$-neuron Hopfield neural network is defined as

$$E = -\sum_{i=1}^{N^2} \sum_{j=1}^{N^2} w_{ij} x_i(t) x_j(t) - \sum_{i=1}^{N^2} b_i x_i(t) \tag{10.20}$$

w_{ij} are the weights, and x_i is the state of the ith neuron. b_i describes the bias input to the ith neuron.

The energy of a stable Hopfield neural network is decreasing over time. This basic fact can be used for solving the L-class pixel classification problem based on eq. (10.18). The weights and the bias inputs can be determined from eqs. (10.18), (10.19), and (10.20):

$$w_{ij} = 2\lambda, \quad \forall j \in \eta_i, i \neq j \tag{10.21}$$

and

$$b_l^i = \frac{1}{L} \sum_{l=1}^{L} \delta_l^i - \frac{8\lambda}{L} - \delta_l^i \tag{10.22}$$

with $\delta_l^i \equiv ||(\mathbf{p}_i - \mathbf{m}_l)||_2^2$.

The activation function is determined by

$$x_i = \begin{cases} 1, & \text{if } \tau_i > 0 \\ 0, & \text{if } \tau_i \leq 0 \end{cases} \tag{10.23}$$

where τ_i is given by eq. (7.57).

The *optimization algorithm of the Hopfield neural network using a priori image information* is iterative and described as follows [75]:

ALGORITHM 3.

1. **Initialization:** Choose random values for the cluster centers $\mathbf{m}_l$ and the neuron outputs x_i.
2. **Forward computation part I:** At each iteration k and for each neuron i compute: (a) the input to the neuron using eqs. (10.21) and (10.22) and (b) the new state based on eq. (10.23).
3. **Forward computation part II:** If $x_i(k) \neq x_i(k-1) \forall i$ go to step (2), else go to step (4).
4. **Adaptation:** Compute new cluster centers $\{\mathbf{m}_l\}$ using $x_i(k)$, with $i = 1, \ldots, N^2$. Go to step (2).
5. **Continuation:** Repeat until the cluster centers do not change.

EXAMPLE 10.7.1: The neural model was applied in [75] to segment masses in mammograms. Each pixel of the ROI image describing extracted masses belongs to either the mass or the background tissue and defines such a two-class classification problem. The gray levels of the pixels are used as the input feature. Figures 10.8 and 10.9 show the segmentation results obtained with a Hopfield network without ($\lambda = 0$) and with a priori information ($\lambda \neq 0$). As expected, including a priori information yields a smoother segmentation compared to $\lambda = 0$.

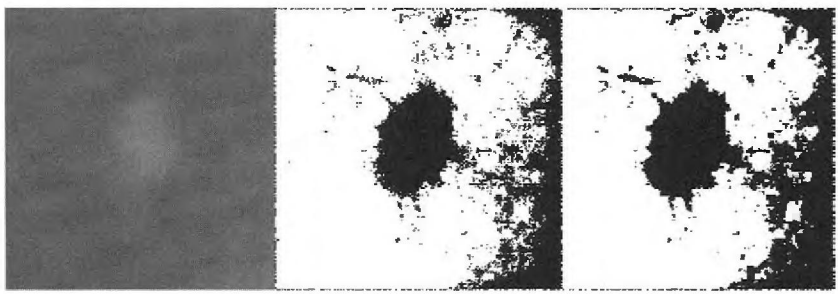


FIGURE 10.8 Segmentation results of a two-class classification problem: (left) original ROI, (center) segmentation result using a neural network with $\lambda = 0$, (right) segmentation result using a neural network with a priori information $\lambda = 1$. (Reprinted from [75] with permission from IEEE.)

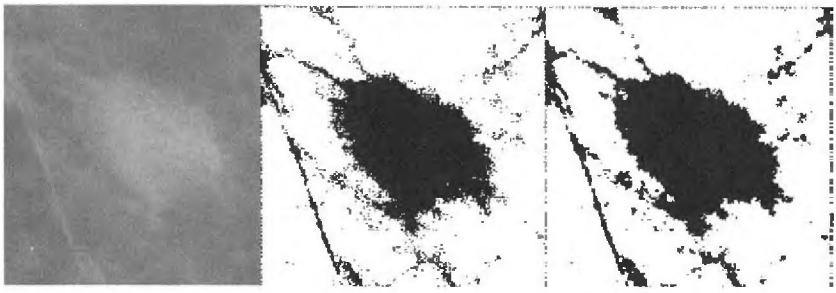


FIGURE 10.9 Segmentation results of a two-class classification problem: (left) original ROI, (center) segmentation result using a neural network with $\lambda = 0$, (right) segmentation result using a neural network with a priori information $\lambda = 2$. (Reprinted from [75] with permission from IEEE.)

10.8 HOPFIELD NEURAL NETWORK FOR TUMOR BOUNDARY DETECTION

The extraction of tumor boundaries in 3D images poses a challenging problem for bioimaging. In MRI, they provide useful insights in the internal structures of the human body. Several attempts have been made to apply neural network architectures to brain image analysis [9, 37, 254]. One of the most efficient approaches is described in [254]. It yields results comparable to those of standard snakes, and at the same time, it is less computationally intensive. The flow diagram in Figure 10.10 describes the implemented diagram from [254]. In the first step, the image is enhanced; and the second step involves initial boundary detection and searching grids estimation; while the last step detects the tumor boundaries using the Hopfield neural network.

The boundary detection is also formulated as an optimization problem that seeks the boundary points to minimize an energy functional based on an active contour model. A modified Hopfield network is constructed to solve the optimization problem.

10.8.1 Mathematical Model of the Active Contour

The basic active contour model is an energy-minimizing spline. For any point $v(s) = (x(s), y(s))$ of the contour, an energy function can be defined as [107]

$$E = \int_0^1 \alpha||v'(s)||^2 + \beta||v''(s)||^2 - P(v(s))ds \qquad (10.24)$$

The first two terms describe the internal energy of the contour, while the final term refers to the image forces. The first- and second-order derivatives of $v(s)$ are approximated as follows:

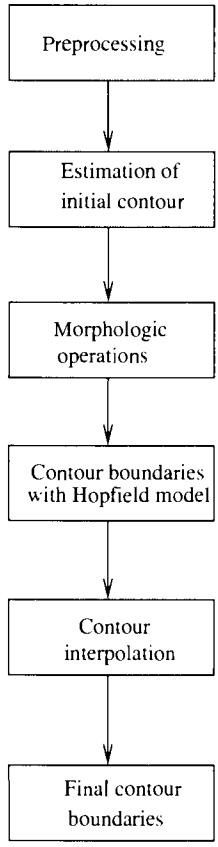


FIGURE 10.10 Block diagram of the tumor boundary detection method.

$$\left|\frac{dv_i}{ds}\right|^2 \approx |v_i - v_{i-1}|^2 = (x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 \tag{10.25}$$

and

$$\left|\frac{d^2v_i}{ds^2}\right|^2 \approx |v_{i-1} - 2v_i + v_{i+1}|^2 = (x_{i-1} - 2x_i + x_{i+1})^2 + (y_{i-1} - 2y_i + y_{i+1})^2 \tag{10.26}$$

x_i and y_i are the x- and y-coordinates of the ith boundary point v_i. In [254], the contour has to be attracted to edge points, and therefore the image forces depend on the gradient $g(v_i)$ of the image at point v_i

$$P(v_i) = -g(v_i) \tag{10.27}$$

The basic idea proposed in [254] is that the boundary is detected by iteratively deforming an initial approximation of the boundary through minimizing the following energy functional:

$$\begin{aligned} E_{Snake} &= \sum_{i=1}^{N} \{\alpha[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2] \\ &\quad + \beta[(x_{i-1} - 2x_i + x_{i+1})^2 + (y_{i-1} - 2y_i + y_{i+1})^2] \\ &\quad - \gamma g_i\} \end{aligned} \tag{10.28}$$

Before employing the active contour model, an initial contour must be estimated. In [254], the contour for the first slice is estimated based on morphological operations.

10.8.2 Algorithm for Boundary Detection

Contour detection in an image is achieved by employing a binary Hopfield neural network that implements a two-dimensional $N \times M$ neuronal lattice. N is equal to the number of sampling points of the initial contour, while M equals the number of grid points along each grid line. Each neuron is described by a point (i, k) with $1 \leq i \leq N$ and $1 \leq k \leq M$ and is considered as a hypothetical boundary point. Each neuron output is either equal to 1 (shows presence of boundary elements) or equal to 0 (shows absence of boundary elements), see Figure 10.11.

The energy function of this special Hopfield neural network is

$$E = -\frac{1}{2} \sum_{i}^{N} \sum_{k}^{M} \sum_{j}^{N} \sum_{l}^{M} w_{ikjl} x_{ik} x_{jl} - \sum_{i}^{N} \sum_{k}^{M} b_{ik} x_{ik} \tag{10.29}$$

x_{ik} is the neuron state at point (i, k), and w_{ikjl} is the synaptic weight connecting a neuron at point (i, k) with one at point (j, l). The propagation rule for the neuron at (i, k) is

$$x_{ik}(t+1) = \sum_{j}^{N} \sum_{l}^{M} w_{ikjl} x_{jl}(t) + b_{ik} \tag{10.30}$$

and the corresponding activation function is

$$x_{ik} = \begin{cases} 1, & \text{if } x_{ik} = \max(x_{ih}; h = 1, \ldots, M) \\ 0, & \text{else} \end{cases} \tag{10.31}$$

Equation (10.31) constrains the M mutually exclusive neurons representing a boundary point such that only one neuron in each row is firing while the other are resting.

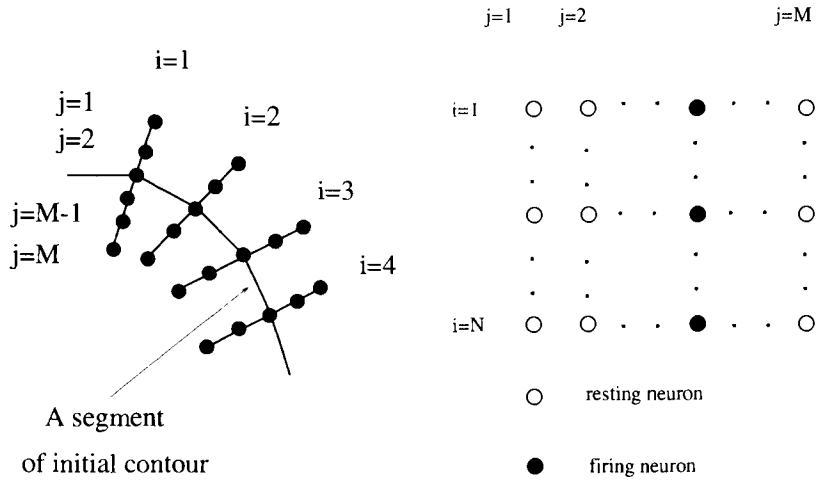


FIGURE 10.11 Correspondence between grid points and network architecture [254]. (a) Segment of the initial contour and search grid points; (b) network row-column assignment.

By comparing eqs. (10.28) and (10.29), the connection weights w_{ikjl} and neurons inputs b_{ik} are determined by

$$w_{ikjl} = \begin{cases} -[(4\alpha + 12\beta) - (2\alpha + 8\beta)\delta_{i+1j} \\ \quad - (2\alpha + 8\beta)\delta_{i-1j} + 2\beta\delta_{i+2j} \\ \quad +2\beta\delta_{i-2j}] \cdot [x_{ik}x_{jl} + y_{ik}y_{jl}], & \text{if} \quad i = j \\ [(2\alpha + 8\beta)\delta_{i+1j} + (2\alpha + 8\beta)\delta_{i-1j} \\ \quad - 2\beta\delta_{i+2j} - 2\beta\delta_{i-2j}] \cdot [x_{ik}x_{jl} \\ \quad + y_{ik}y_{jl}], & i \neq j \end{cases} \tag{10.32}$$

and

$$b_{ik} = \gamma g_{ik} \tag{10.33}$$

Based on eqs. (10.30) and (10.31), the next state of each neuron can be determined. While the neuron states change, the optimal boundary points are detected. Convergence is achieved if one and only one of the outputs of the neurons in each row is equal to 1. The location of the firing neurons corresponds to the detected boundary points.

EXAMPLE 10.8.1: Figure 10.12 [254] shows the performance of the described network on a slice of T1-weighted MR axial brain image.

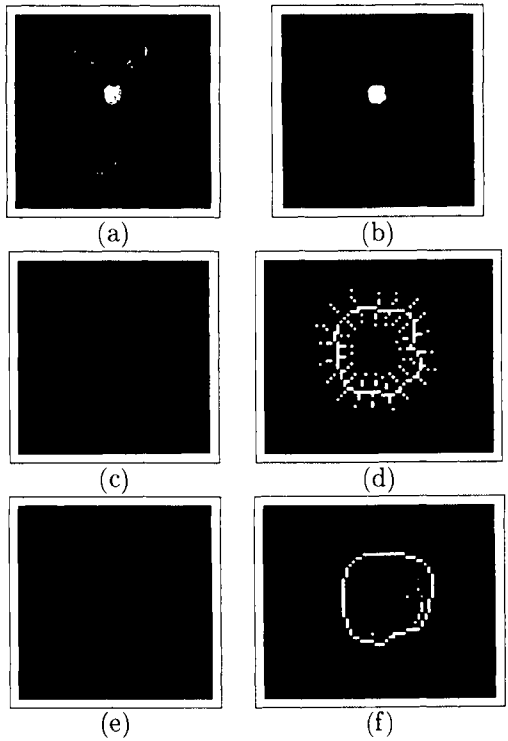


FIGURE 10.12 Example of a detected contour: (a) initial slice 39 (thresholded); (b) dilation of the original image; (c) initial contour; (d) search grid points and initial contour; (e) example of slice 49 containing a brain tumor; and (f) detected contour of slice 41 from (e). (Reprinted from [254] with permission from IEEE.)

10.9 CASCADED SELF-ORGANIZED NEURAL NETWORK FOR IMAGE COMPRESSION

10.9.1 Image Compression

Digital image processing enables easy image retrieval, efficient storage, rapid image transmission for off-site diagnoses, and the maintenance of large image banks for purposes of teaching and research. To use digital signal techniques, analog signals such as X-rays must be converted into a digital format, or they must be already available in digital form. The sampling process produces a loss of information and can lead to a deterioration of the signal. On the other hand, because of technological improvements in analog-to-digital converters, the quantities of digital information produced can overwhelm available resources. For example, a typical digitized mammogram with 4500×3200 pixels with 50-micron spot size and and 12-bit-per-pixel depth is approximately 38 megabytes of data. Extensive medical studies using digitized data format can easily require unacceptably long transmission times and can lead to major data management problems in local disk storage. A solution solely based on advancing technologies for transmission and storage is not sufficient. Data compression is desirable and often essential for improving cost and time for storage and communication. The typical digital compression system has three main steps: (1) signal decomposition based on a Fourier and wavelet transform, (2) coefficient quantization, and (3) lossless or entropy coding such as Huffman or arithmetic coding. Decompression is defined as the inverse operation, and if quantization is employed, decompression is lossy in the sense that the original image cannot be perfectly reconstructed. Figure 10.13 shows a compression scheme applied in digital mammography, where the transmission time of four images per patient per screening is critical. Lossless or invertible compression allows perfect reconstruction of a digital image, but the obtained compression ratios range from 2:1 to 3:1 on still-frame gray scale medical images. For many medical applications this modest compression is not adequate. Lossy compression can provide excellent quality at a fraction of the bit rate. The bit rate of a compression system is the average number of bits produced by the encoder for each image pixel. If the original image has 12 bits per pixel (bpp) and the compression method yields a rate of R bpp, then the compression ratio is given by 12:R.

The first studies in medical lossy compression were based on applying variations on the standard discrete cosine transform (DCT) coding algorithm combined with scalar quantization and lossless coding. These are variations of the international standard Joint Photographic Experts Group (JPEG) compression algorithm [34]. The American College of Radiology–National Electrical Manufacturers Association (ACR–NEMA) standard has not yet specifically named a standard compression technique, but transform coding methods are suggested. New studies of efficient lossy image

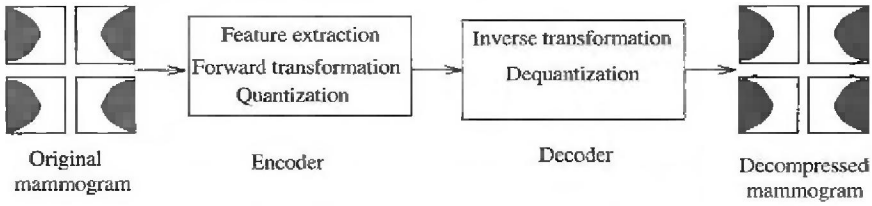


FIGURE 10.13 Scheme of the compression mechanism illustrated for digital mammography. Four images per patient are produced by a single screening.

compression algorithms have been based on subband or wavelet coding in combination with scalar or vector quantization [211, 243]. The main benefits of these techniques over traditional Fourier-type decompositions are better concentration of energy and decorrelation for a larger class of signals. However, the main disadvantage represents the errors introduced by a resolution-based variable filter length. They are not well-localized and represent a ringing artifact. In [21] and [189] a compression algorithm (Shapiro's embedded zero-tree algorithm [217]) was used based on the subband/pyramid/wavelet encoding. New forms of multiresolution vector quantization algorithms were investigated in [47] and [141]. The vector generalizations of the embedded zero-tree wavelet technique used in [47] introduced additional complexity and made this approach less attractive than the scalar wavelet coding selected. The nonwavelet multiresolution technique proposed in [141] yielded significant improvements at reasonable complexity for low-resolution images, e.g., for reduced-size medical images for progressive viewing during the displaying of the full image.

The main concern to be addressed in conjunction with lossy image compression is to specify the type and amount of distortion necessary for enough accuracy for diagnostic, educational, archival, or other purposes.

The tools for measuring image quality and their performance have been surveyed in two papers [1, 61]. Besides these general surveys, there are several investigations that focus on evaluating the quality of compressed medical images. A detailed study [46] compares three approaches in this field and notes that (1) SNR is quick and cheap to obtain, but does not take into account the medical nature of the images; (2) receiver operating characteristic (ROC) analyses are serious attempts to capture the medical interest of the images through their diagnostic values; and (3) subjective rating shows a different trend from the actual diagnostic quality, which can reassure physicians that diagnostic utility is retained even when a compressed image is perceptually distinguishable from the original.

10.9.2 Neural Network Architecture

Transform-based algorithms for lossy compression transform an image to concentrate signal energy into a few coefficients, quantize the coefficients, and then entropy code them after assigning special symbols for runs of zeros. The use of principal component analysis (PCA) for image coding is known to be an optimal scheme for data compression based on the exploitation of correlation between neighboring pixels or groups of pixels [168]. Additional data compression can be achieved by using vector quantization subsequent to a robust PCA [35, 168, 224, 234]. The basic idea of this improved compression scheme is visualized in Figure 10.14.

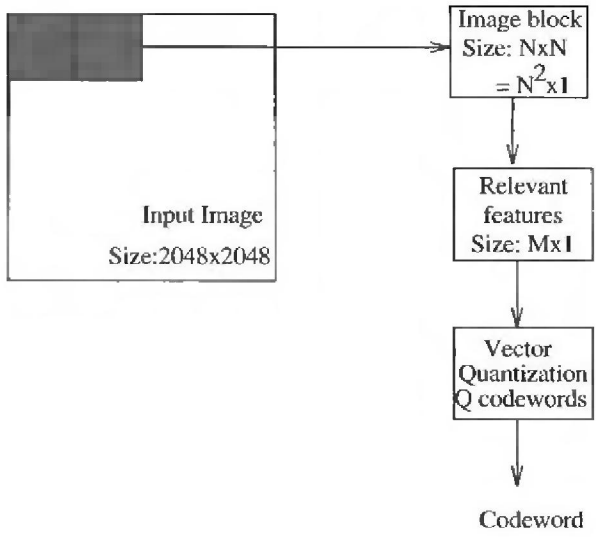


FIGURE 10.14 Compression scheme combining PCA with vector quantization.

An image is divided in blocks of the size $N \times N$. M relevant features, where $M < N$, are extracted from each block and then quantized based on a vector quantization (VQ) algorithm. VQ is defined as a mapping that assigns each feature vector to one of Q codewords from a finite subset, called a codebook. In the encoding process, a distance measure is evaluated to determine the closest codeword for each input vector. Then, the address corresponding to the codeword is assigned to every feature vector and transmitted. Compression is achieved by transmitting the index of the codeword instead of the vector itself.

Figure 10.15 shows a cascaded architecture of self-organizing neural networks for lossy image compression combining a PCA-type neural network implementing Oja's algorithm with a vector quantizer realized by the "neural-gas" network. Oja's algorithm is applied to image blocks that are

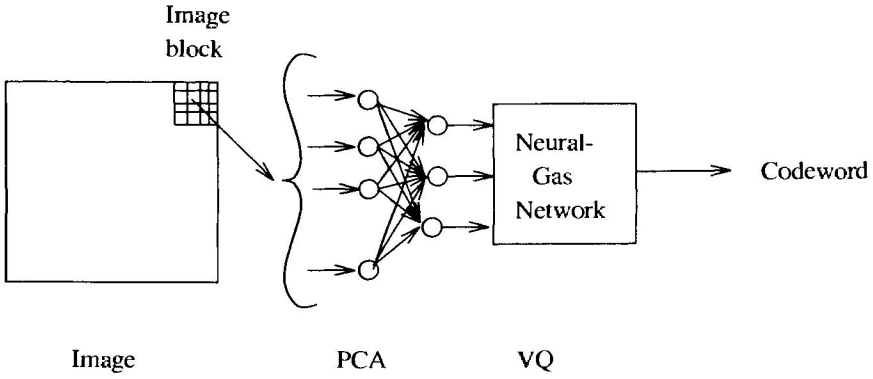
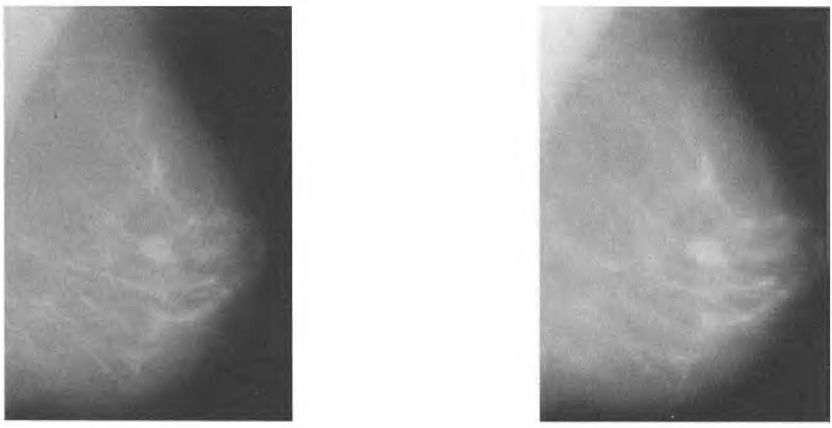


FIGURE 10.15 Architecture of the combined network (Oja's algorithm and "neural gas") for compression.

treated like data vectors and performs a projection of the data vector on a lower dimensional space. Additional data compression can be achieved by using the "neural-gas" network on the coder output to produce its quantized version. The combined network for compression is shown in Figure 10.15.

As an indicator for comparison of information loss due to image compression, we choose the "classical" approach for image quality. Image quality is typically quantified objectively by average distortion or peak signal-to-noise ratio (PSNR). The PSNR definition is $10 \log_{10} 256^2/\text{MSE}$, where MSE denotes the average squared error;



(a) (b)

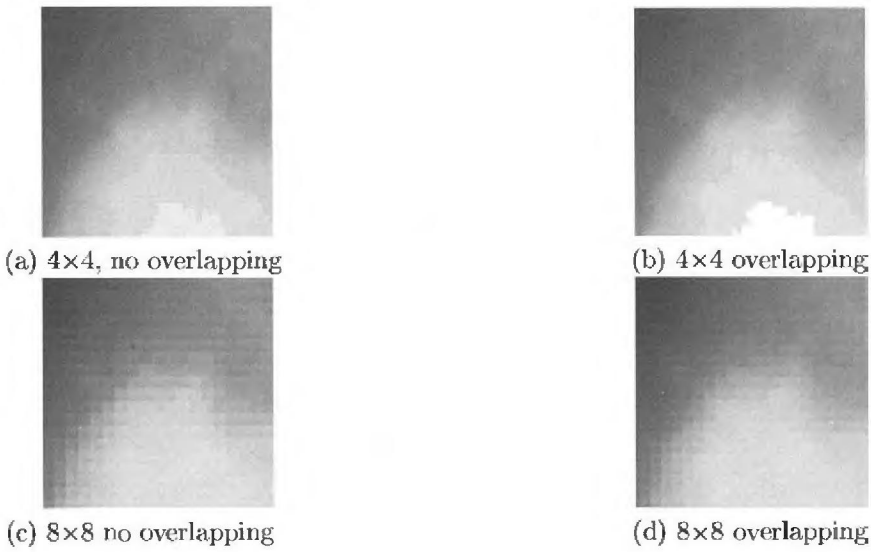
FIGURE 10.16 Original mammogram (a) and (b) same mammogram compressed by the proposed algorithm, compression rate = 64, PSNR = 23.5dB.

$$\text{MSE} = \frac{1}{1024 \times 1024} \sum_{i=1}^{1024} \sum_{j=1}^{1024} (x_{i,j} - y_{i,j})^2 \tag{10.34}$$

with $x_{i,j}$ the gray value of the original and $y_{i,j}$ that of the decompressed image for the pixel at the position i, j in a 1024×1024 image.

EXAMPLE 10.9.1: This neural architecture can be applied to compress digital mammograms. To study the efficiency of our algorithm, mathematical phantom features are blended into clinically proved cancer-free mammograms. The influence of the neural compression method on the phantom features and the mammographic image is not perceptible to a mammogram reader up to a compression rate of 48:1. Figure 10.16 shows the achieved compression results for a block-size of 8×8, 4 PCA components at a compression ratio of 64 and PSNR = 23.5. Oja's algorithm [178] was chosen for computing the principal components.

The quantization results for a 1024 codebook are shown in Figure 10.17. Again the advantage of the overlapping over nonoverlapping is evident. With overlapping and a smaller block-size annoying effects such as blocking are less accentuated.



(a) 4×4, no overlapping (b) 4×4 overlapping

(c) 8×8 no overlapping (d) 8×8 overlapping

FIGURE 10.17 Cut-outs of the quantized test image. Compression results with and without overlapping depending on different block-sizes.

XI

NEURAL-BASED COMPUTER-AIDED DIAGNOSIS SYSTEMS IN BREAST CANCER DETECTION

11.1 INTRODUCTION

According to the American Cancer Society (www.cancer.org), excluding cancers of the skin, breast cancer is the most common cancer among women, accounting for one out of every three cancer diagnoses in the United States. Today, a woman has a one in eight chance of developing breast cancer. In 2003, approximately 200,000 new cases of invasive breast cancer are expected to be diagnosed, and 50,000 women are expected to die from this disease. Breast cancer accounts for 31% of all cancers detected and 17% of all cancer deaths, and ranks as the second leading cause of death from cancer among women in the United States.

Mammography is the most sensitive widely used technique for detecting breast cancer, with a reported sensitivity of 85-95% for detecting small lesions. Most noninvasive ductal carcinomas are characterized by tiny nonpalpable calcifications detected at screening mammography [171]. Traditional mammography is essentially analog photography using X-rays in place of light and analog film for display. To take advantage of modern digital technologies such as storage, transmission, and digital computer processing, analog images have to be converted into a digital format. Digital

images are easily distributed to multiple sites, either in-hospital or off-site, and thus personnel costs are dramatically reduced. Currently only 30% of American women get regular mammograms, and the storage problems will become overwhelming if because of greater awareness or wider insurance coverage, this number increases. Digital image processing techniques support easy image retrieval, efficient storage, rapid image transmission for off-site diagnoses, and the maintenance of large image banks for educational and research endeavors. Basic image processing operations such as filtering, enhancement, classification, and fusion are supported leading to better screening, diagnosis, research, and treatment. It has been suggested that as normally viewed, mammograms display only about 3% of the information they detect. Although mammography is widely recognized as the best method for the early detection of breast cancer in asymptomatic women, 10% to 30% of women who have breast cancer and undergo mammography are diagnosed as having no malignancy, and in most cases of these false-negative mammograms the cancer was visible in retrospective studies. This significant observer error evolves either from human factors such as a fatigue, misinterpretation, or distraction, or from image factors such as obscuration by a dense breast. To this end, systems for computer-aided diagnosis (CAD) may assist the radiologist in the detection of abnormalities and thus provide a second opinion. It has been shown that the performance of radiologists can be increased when they are provided with the results of a CAD system. In other words, more cancers can be detected and at the same time fewer are missed with the same biopsy rate. In an automated prescreening situation, CAD systems are employed to sort out some fraction of the normal mammograms with extremely high reliability. Only the remaining mammograms with breast cancer actually present will undergo manual reading by a radiologist. By looking only at a few (approximately 0.5%) abnormal mammograms, the radiologist's time can be better spent examining suspicious and difficult-to-diagnose cases. It is evident that CAD systems assisting radiologists in reading mammograms must be adopted in the clinical environment. However, every CAD system requires images in digital format.

A primary breast carcinoma can metastize when it consists of a relatively small number of cells, far below our present threshold of detection. The importance of diagnosis of breast cancer at an early stage is critical to patient survival. The cure rate for breast lesions of 0.5 cm or less in diameter can approach 100%. Despite advances and improvements in mammography and mammographic screening programs, the detection of minimal breast cancer (those cancers 1.0 cm or less in diameter) remains difficult. Spiculated masses are arguably the most important type of lesion since (1) most breast carcinomas have the mammographic appearance of a spiculated mass and (2) their perception is difficult, especially when they are small [228].

Computerized methods for the detection of microcalcifications have demonstrated more progress than those for masses [171]. Microcalcifications are ideal targets for computer detection algorithms because of the lack of coexisting normal structures that have the same appearance. Masses, on the other hand, are more difficult to detect because they can be simulated or obscured by normal breast parenchyma [71].

As already stated, CAD systems have been developed to improve breast cancer detection based on lesion detection and classification in clinical screening [105, 109, 122, 191, 256]. Although the clinical role of the CAD system is still debatable, by incorporating the expert knowledge of radiologists, the CAD systems provide a second opinion in detecting abnormalities and making diagnostic decisions. It has been proven that the performance of radiologists can be improved by providing them with the results of a CAD system [12]. Hence, the importance and enormous effort to develop CAD systems to assist radiologists in reading mammograms.

Mammographic image analysis can be divided into three parts: (1) preprocessing such as enhancement of mammographic features, (2) feature selection and extraction, thus enabling detection and localization of suspicious areas, and (3) classification. Figure 11.1 visualizes the described procedure.

Although different classification approaches can be employed for breast cancer diagnosis, neural network techniques have proven to be superior compared to conventional methods [52, 159, 103, 255, 210, 246, 253]. The nonlinear tumor nature and the architectural flexibility of neural networks make them extremely appealing for computer-aided mammographic image analysis and interpretation.

Computer-aided methods for detecting clustered microcalcifications have been investigated using many different techniques. A statistical method for detection of microcalcifications was presented in [104]; a computer-based method is described in [246]. Wavelet-based methods for microcalcification detection are described in [223]. Neural and fuzzylogic based approaches are proposed in [38].

Because of the spiculated appearance of masses and also because of their larger size compared to microcalcifications, feature-based approaches are adopted in many CAD systems [105, 122, 191]. In [109] spiculated tumors were detected based on local edge characteristics and Laws texture features. A fuzzy classifier was employed in [27] to detect malignant masses based on morphological features. Local edge characteristics and texture properties for mass detection were used in [109], whereas only texture features in conjunction with a linear classifier were used in [240] to distinguish between masses and normal tissue. In [256] a statistical model was developed to describe and detect the abnormal pattern of linear structures of spiculated lesions. An approach based on using the orientation map of line-like structures for detecting stellate distortions was described in [105].

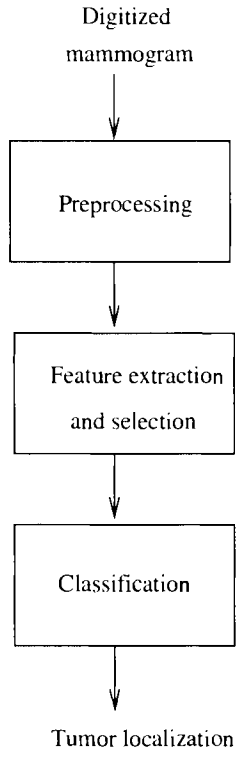


FIGURE 11.1 Flow diagram of a CAD system for breast cancer detection.

Other techniques involve the Hough spectrum [251] or a combination of breast tissue composition information [190].

This chapter describes some of the most important CAD systems known in literature for breast cancer detection. A CAD system for microcalcifications analysis using gray-level image structure features [52] and one for detection of clustered microcalcifications [249] is described. Also a CAD system for mass detection based on a convolution neural network [210] and a modular neural network is presented [139, 140].

11.2 SEGMENTATION OF MASS AND NORMAL BREAST TISSUE

An approach for image segmentation using suboctave wavelet representation and a dynamic resource-allocating neural network is described in [255]. The algorithm is applied to identify regions of masses in mammographic

images with varied degrees of difficulty. In the process of image segmentation, each mammographic image having a mass is first decomposed into wavelet representations of suboctave frequency bands.

A feature vector for each pixel through the scale space is constructed based on this representation from fine to coarse scales. The feature vectors are used to drive a neural network classifier of dynamic resource allocation for segmentation [193].

The use of the wavelet transform for the detection of masses is relatively new. Multiresolution texture features can effectively classify masses from normal tissue. The methods for detection of microcalcifications and masses have relied on the extraction of features from critically sampled wavelet representations. Although these representations have been used successfully in compression, they are not suitable for detection and enhancement tasks because of aliasing effects introduced during analysis. To achieve an optimal local and global image enhancement, an approach is chosen that is unique in the use of overcomplete and continuous wavelet representations for the extraction of features for mass detection.

In [125] it was shown that regions corresponding to masses can be identified reliably through multiscale features of a continuous wavelet transform. Overcomplete wavelet representations are more desirable for accomplishing detection and enhancement because (1) they avoid the aliasing effects introduced by critically sampled representations, and (2) they yield a shift-invariant representation. Continuous wavelet representations are also useful because they yield wavelet representations at nondyadic scales. It was suggested that this property is particularly useful in mammography because often one does not know "a priori" at which scale the pertinent information for diagnosis is localized.

11.2.1 Feature Extraction

Prior to neural network-based evaluation each mammogram is preprocessed. To accomplish this the regions corresponding to localization of the mass in a continuous wavelet representation are segmented. Three-dimensional scale space feature extraction is accomplished by processing and combining information from each suboctave frequency band. The output of the neural network determines the probability of each pixel belonging to a mass. Based on a subsequent threshold procedure, a binary mask is produced segmenting the local lesion within the sampled window. The steps of the multiscale mass segmentation are illustrated in Figure 11.2.

11.2.2 Neural Network Architecture and Classification Results

The RBNN has not predefined architecture but evolves during learning, is rotation- and shift-invariant, and has a minimal number of problem oriented computational units able to detect the most relevant features that emerge within each level of analysis. This network provides a method of

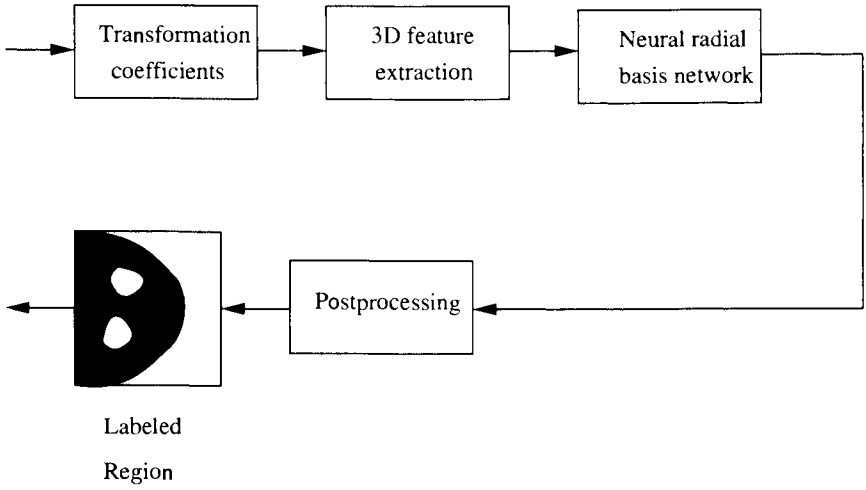


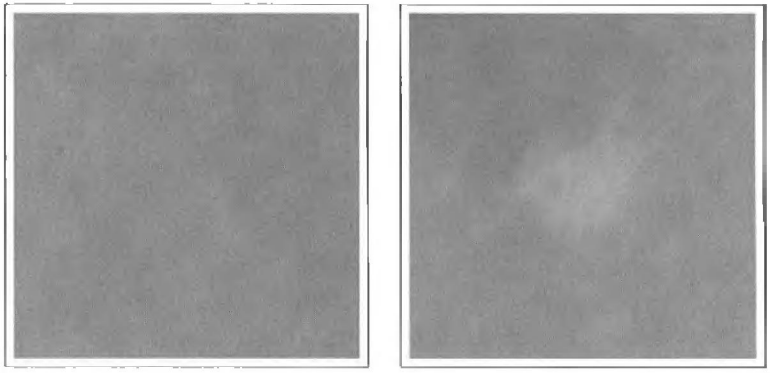
FIGURE 11.2 Multiscale mass segmentation.

detecting (beningn and malignant) masses within a cropped area, driven by continuous scale-space representation. The architecture of an RBNN is simple and consists of three layers: an input layer, a hidden layer, and an output layer. The dimension of the input layer is equal to that of the feature vector (here the 3D feature vector describing a mass), while the output layer is one-dimensional (suspicious mass or not). The number of neurons in the hidden layer will be determined automatically during the learning process and will be optimal in the sense of determining the contour of the masses in breast tissue.

In Figure 11.3 are shown a region of a healthy tissue and another containing a mass.

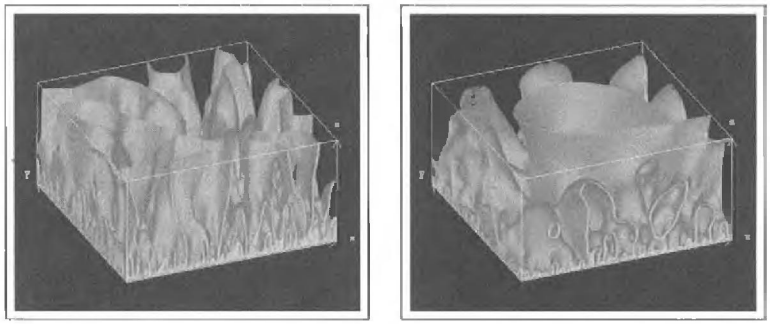
An RBNN is ideal carrying out the classification of scale space representations since the underlying cluster basis of the network takes the form of a hyperellipsoid. The decay of coefficients within a continuous wavelet analysis assumes structural forms (clusters) that are well matched to the form of a hyperellipsoid. Figure 11.4a and b shows the zero crossings of 3D continuous wavelet coefficients for a healthy tissue and a mass.

Several test images have been used to test the neural network trained through the three patterns. Figure 11.5 shows four experimental results of mass segmentation within cropped regions. There are four displays for each test image. The first one is an original image, the second is the smoothed



(a) (b)

FIGURE 11.3 Sample input for regions (a) without mass and (b) with spiculated mass (9 mm) for continuous scale visualization. (Images courtesy of Dr. A. Laine, Columbia University.)



(a) (b)

FIGURE 11.4 3D visualization of the zero crossings of continuous wavelet coefficients of a region (a) without mass and (b) with spiculated mass (9 mm). (Images courtesy of Dr. A. Laine, Columbia University.)

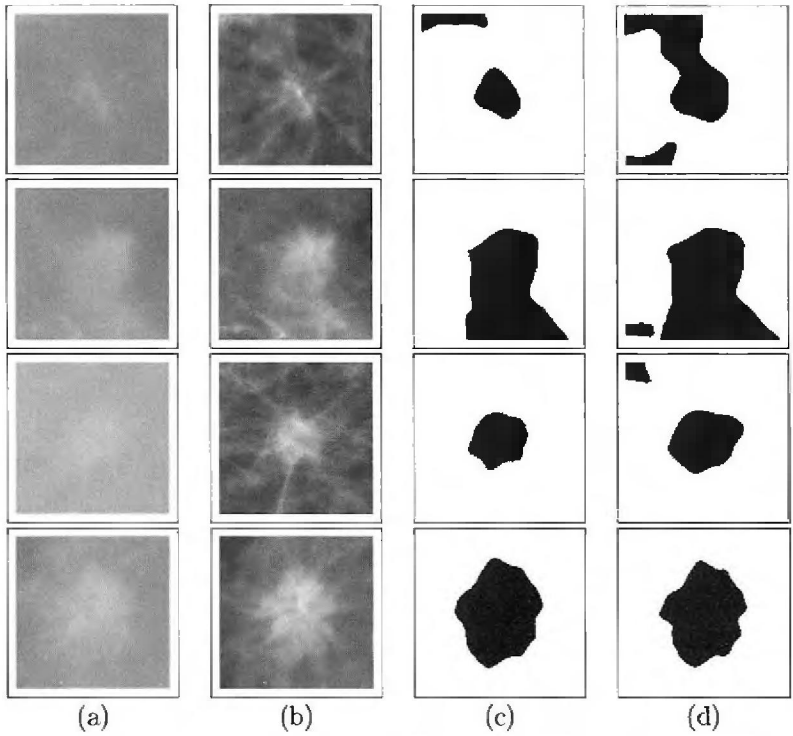


FIGURE 11.5 Experimental results of image segmentation. Four test cases are shown. There are four displays for each test image. (a) Original image; (b) smoothed and enhanced image; (c) segmented result; (d) result of a statistical classifier, shown here for comparison purposes. (Reprinted from [255] with permission from IEEE.)

and enhanced image, the third shows the segmented result, and the fourth is the result of a statistical classification based on local mean and standard deviation, which is shown here for comparison purposes. It can be seen that there is an improvement for the segmented results produced by the RBNN.

For the study, a set of test images was chosen consisting of 25 randomly selected sample images containing 0 to 2 cm masses. The masses in the database have varied degrees of perceptual difficulty. The mean recognition rate was 92.2%. The standard deviation of the recognition rates was 3.12% [255]. The maximum and minimum recognition rates are 97.67% and 84.47%, respectively. Figure 11.6 shows the distribution of the algorithm's recognition rates when tested on sample images.

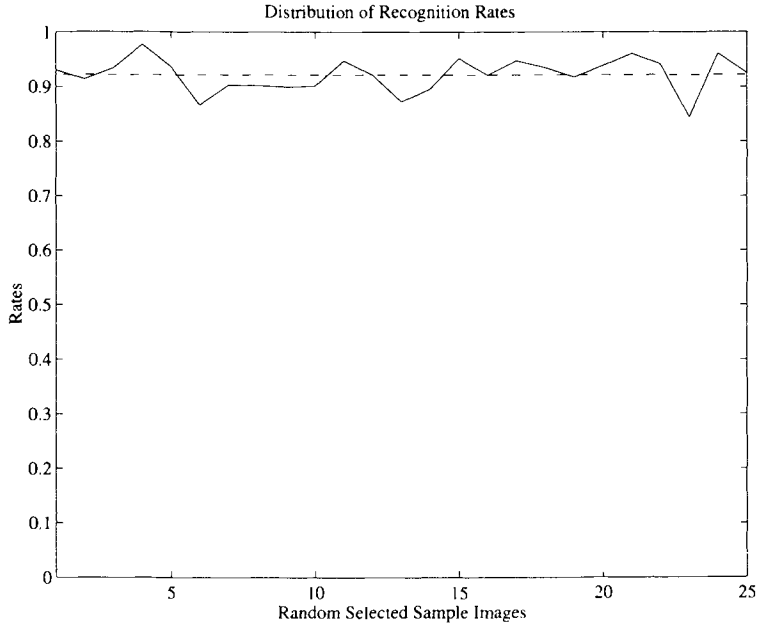


FIGURE 11.6 The distribution of recognition rates from the segmentation algorithm when tested on a set of sample images.

11.3 CLASSIFICATION OF MASS AND NORMAL BREAST TISSUE

A convolutional neural network is employed in [210] to classify ROIs as either masses or normal tissue. This very interesting approach using spatial domain and texture images is described in this section.

Image texture features were previously used for classification of masses and normal tissue on mammograms [240]. It was proven that image texture contains useful information for discriminating between masses and normal tissue. This idea was also followed in [210]. Local texture features permit more detailed characterization of texture changes within an ROI. However, within a selected ROI, there are different texture statistics: the region inside a mass, the transition region between mass and tissue, and the surrounding tissue. By determining the texture for an entire ROI, an average of the texture features for the different regions will result.

The key idea employed by [210] was to preserve these feature differences by determining the features in different subregions inside the ROI. This is accomplished by moving the center of the subregion on a rectangular grid over the ROI, and considering each computed feature as the pixel value of a texture-image at that grid location. This procedure is visualized

in Figure 11.7. This compression mechanism is achieved by a transformation from an input space (digitized image) onto an output space (texture image). Thus, a group of pixels is replaced by a texture feature.

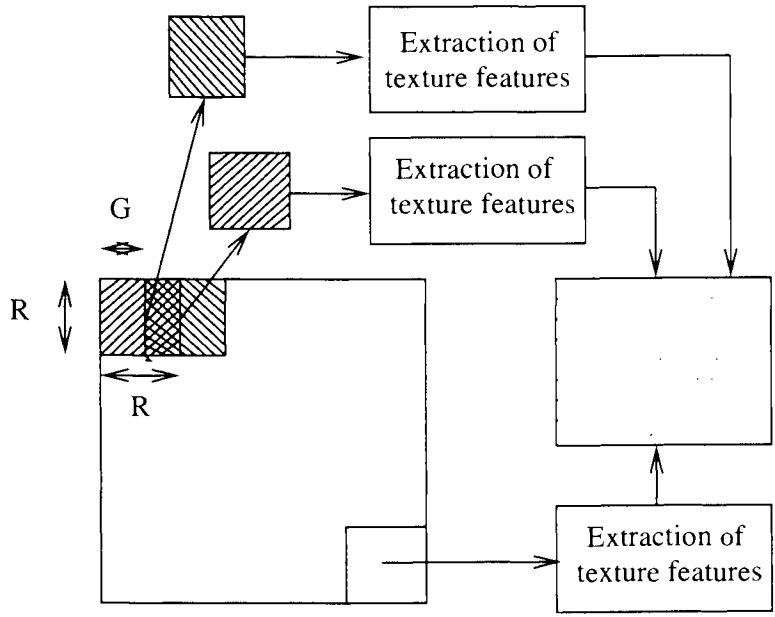


FIGURE 11.7 Computation of texture-images [210].

The texture images were then used as input into the convolutional neural network described in Chapter II, Section 2.4.5.

11.3.1 Feature Extraction

The data for the study in [210] consisted of manually extracted ROIs from 168 mammograms and included 85 benign and 83 malignant masses. Four different ROIs, each with 256×256 pixels, were extracted from each mammography. One of the ROIs contained the true verified mass while the others contained normal breast tissue.

The classification of mass and normal breast tissue was done by the CNN by using the following images:

- **Subsampled images:** The easiest classification method would be to use the background-corrected images as a direct input to the CNN. However, this will result in a huge computational cost. The image size is reduced by averaging adjacent pixels and subsampling. The image size was reduced to 16×16 and 32×32. In the first case, averaging involved nonoverlapping boxes of size 16×16 to obtain

16×16 subsampled images, while in the second case boxes of 8×8 resulted in 32×32 subsampled images.

- **Gray-level difference statistics:** These first-order statistical features represent the contrast, angular second moment, entropy, and mean, and roughly measure the coarseness of the texture elements in an image.
- **Spatial gray-level dependence:** These second-order statistical features represent the entropy, difference entropy, and correlation and are useful in distinguishing mass ROIs from normal tissue.

11.3.2 Convolutional Neural Network Architecture and Classification Results

The difference between a CNN and an ordinary neural network applied to image classification is that the CNN uses as an input an image instead of extracting relevant image features. Thus, the CNN is an adequate candidate for the classification of masses and normal tissue on mammograms. The CNN architecture employed for classifying mass and nonmass ROIs is shown in Figure 11.8. It is a three-layer network with three input images and a single output node. The number of hidden layer groups $N(2)$, and the weight kernel size between the input layer and the hidden layer $S_w(1)$ are not fixed. The first image is a 16×16 averaged-subsampled image; the second and the third have the same size as the first but are texture images. The second is a mean-texture image while the third is a correlation texture-image.

The classification results with the three input images derived from an ROI, as described above, are summarized in Table 11.1. The average test A_z was of 0.87 which corresponds to an average true-positive fraction of 90% at a false positive fraction of 31%. It is evident that the CNN architecture has almost no effect on the A_z. In other words, the choice of the CNN input images is more important than the choice of the CNN architecture.

TABLE 11.1 CNN classification performance with three input images derived from an ROI [210].

Kernel size	Number of groups	Training A_z	Test A_z
6	3	0.84	0.83
8	3	0.90	0.84
10	3	0.90	0.87
12	3	0.91	0.86
10	4	0.89	0.87
10	6	0.89	0.87
10	8	0.91	0.87

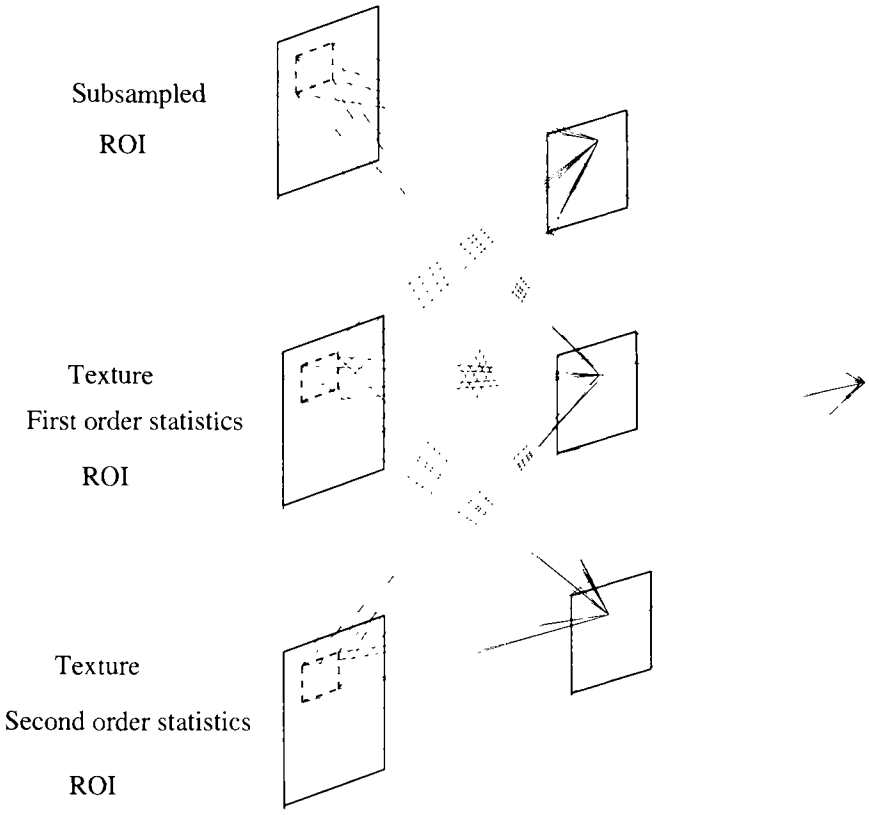


FIGURE 11.8 CNN architecture used for ROI classification with average-subsampled ROI, mean texture image, and correlation texture image [210].

11.4 INTERACTIVE USER INTERFACE-BASED CAD SYSTEM FOR MASS DETECTION

Although CAD systems support radiologists in their diagnostic decisions, some basic problems are still unresolved concerning the application of CAD systems. These are: (1) breast cancer is partly missed on mammograms because the optical density and contrast of cancer are not optimal for the human observer; (2) CAD seems to be more influenced by other criteria than human perception; (3) strategies for solving problems for human and for machine observers are quite different, and (4) decision making by the CAD systems is not always transparent to the user [139, 140]. To alleviate these problems, human intelligence has to be integrated in CAD systems. Figure 11.9 shows such an integrated CAD system.

In [140], a mass detection approach based on an interactive user interface to implement the complementary roles of the CAD in clinical practice

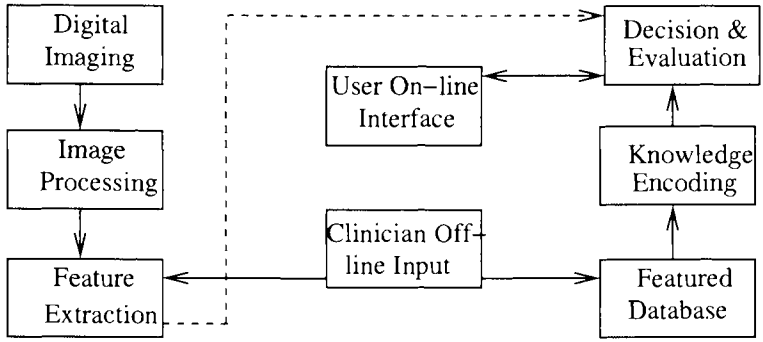


FIGURE 11.9 CAD system with integrated human intelligence [139].

was proposed. The main steps of computer-assisted mass detection are: (1) construction of the featured knowledge database; (2) mapping of the classified and/or unclassified data points in the database; and (3) development of an intelligent user interface. The clinical goal is to eliminate through feature discrimination the number of false positives that correspond to normal dense tissues with mass-like appearances.

The framework of the method described in [140] is illustrated in Figure 11.10.

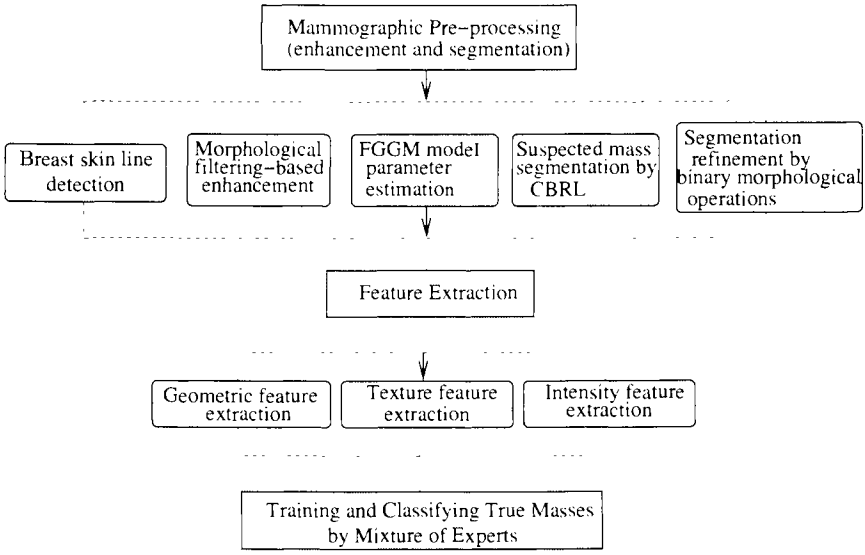


FIGURE 11.10 The flow diagram of mass detection in digital mammograms [139].

11.4.1 Knowledge Database Construction

The knowledge database constructed in multidimensional feature space combines intensity, geometric, and texture features. Table 11.2 summarizes these features.

TABLE 11.2 Description of intensity, geometric, and texture features [140].

Feature sub-space	Features
Intensity features	1. Contrast measure of ROIs 2. Standard derivation inside ROIs 3. Mean gradient of ROI boundary
Geometric features	1. Area measure 2. Circularity measure 3. Deviation of the normalized radial length 4. Boundary roughness
Texture features	1. Energy measure 2. Correlation of co-occurrence matrix 3. Inertia of co-occurrence matrix 4. Entropy of co-occurrence matrix 5. Inverse difference moment 6. Sum average 7. Sum entropy 8. Difference entropy 9. Fractal dimension of surface of ROI

11.4.2 Data Mapping for Decision Making

The optimal mapping of the data points is obtained by training a neural network. A probabilistic modular neural network (PMNN) consists of a set of simple class subnets with local mixture clusters, each one of which represents a specific region of the knowledge space. The architecture of the PMNN used in [140] is shown in Figure 11.11.

The PMNN consists of two subnets each having several neurons or clusters. The outputs of class subnets represent the inputs of a probability winner processor. This processor assigns an input vector to the correct class based on the highest probability achieved by a subnet corresponding to this class. The learning rule is unsupervised, the subnets are trained individually, and no mutual information is exchanged between the classes. For mass detection, one modular expert is trained to detect true masses, while the other is trained to detect false masses. After training, the network can distinguish between true and false masses.

11.4.3 Intelligent User Interface

The utility of the CAD system in [140] was improved by an intelligent user interface. Its major functions are interactive visual explanation of the

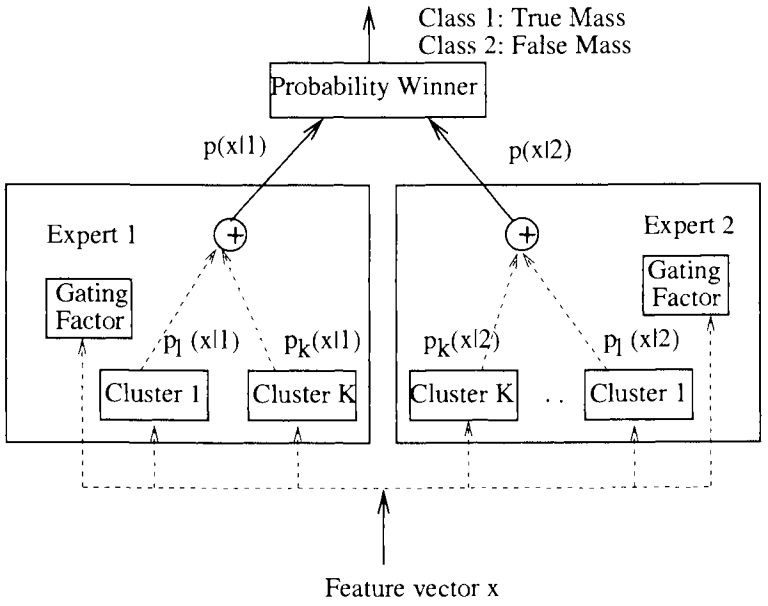


FIGURE 11.11 The architecture of the PMNN [140].

CAD decision making process, on-line retrieval of optimally displayed raw data, and supervised upgrade of the knowledge database with expert organized representative cases. A probabilistic principal component projection scheme is used to reduce the dimensionality of the feature space for natural human perception and thus leads to a hierarchical visualization algorithm.

The integrated system was evaluated in real case detection. In [140] it was reported that the system was able to reduce the number of suspicious masses with a sensitivity of 84% at 1.6 false positive findings/mammogram based on the testing data set containing 46 mammograms (23 of them had biopsy-proven masses).

11.5 MICROCALCIFICATION ANALYSIS SYSTEM USING IMAGE STRUCTURE FEATURES

The presence of microcalcifications represents one of the primary signs of breast cancer. However, it is still a difficult task to distinguish between benign and malignant microcalcifications associated with breast cancer [103].

Most techniques applied to the computerized analysis of mammographic microcalcifications employ extracted shape features from the segmented regions of microcalcifications. Since poorly defined microcalcification features affect the accuracy of the analysis system, shape features might not always represent the best solution. In [131] it was determined

that the intensity variations and texture information in the ROI provide important diagnostic information about the underlying biological process for the benign and malignant tissue, and therefore they should be considered by the CAD system. Although the second-order histogram statistics gives an excellent overall representation of the global nature of the texture (coarseness and skewing patterns), it does not provide sufficient local information in the form of spatial frequency-based features. An approach for providing better details on the texture is the wavelet transform [127]. Moreover, wavelet packets are able to detect specific scale and orientation sensitivities of the textures. Texture signatures such as entropy and energy measures represent robust features for discriminating natural textures.

A new technique for microcalcification analysis using non-shape features is described in [52]. Second-order gray-level histogram statistics is employed to represent global texture, while wavelet decomposition is used to describe local texture. Also, the cluster formation is considered by incorporating information about the number of the microcalcifications, and their average size and distance. The best feature set for characterizing microcalcifications was determined by a GA. A subsequent neural network performs the microcalcifications classification. Figure 11.12 illustrates the block diagram of the microcalcification analysis system. For each ROI, local and global features and also cluster formation features are extracted, forming a 40-dimensional feature vector. A GA reduces the dimension of this feature vector from 40 to 20. This new feature vector of reduced dimension forms the input to the subsequent neural classifier.

11.5.1 Feature Extraction

The data set used in [52] was based on 191 "difficult-to-diagnose" mammographic microcalcification cases which were digitized. The following features were extracted from the digitized ROIs:

- Global texture features from second-order histogram statistics: All 10 image structure features described in Chapter II, Section 2.4.5, were included in the microcalcification feature vector.
- Local texture features from wavelet packets: The wavelet analysis was performed on rectangular regions containing the microcalcifications. Daubechies wavelets provide a good modality for extracting texture information with varying spatial frequency [127]. The energy and entropy features were computed for the original image and for the level 1 decomposition according to Figure 4.4.
  The energy E is given by

$$E = \frac{\sum_i \sum_j x_{ij}^2}{L \times B} \tag{11.1}$$

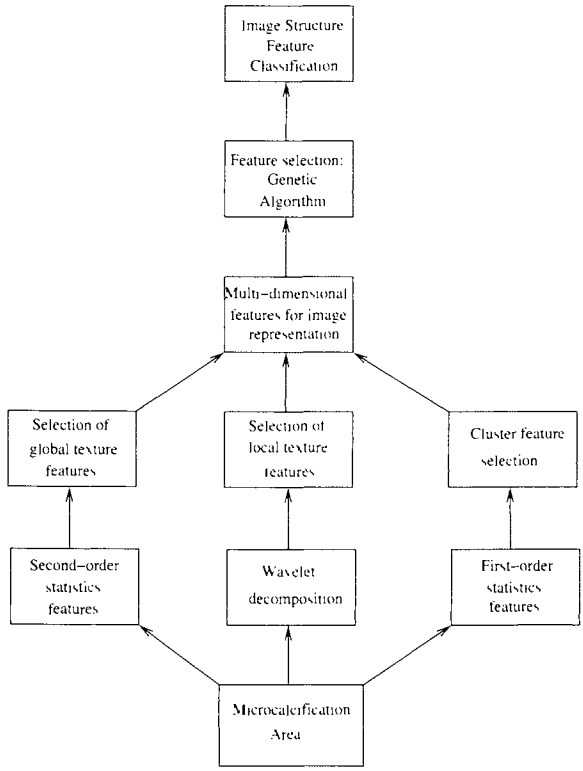


FIGURE 11.12 Block diagram of the microcalcification analysis system [52].

where x_{ij} is the computed wavelet packet value at the ith row and jth column of the wavelet packet. L is the length and B the breadth; at the same time these are the dimensions of the wavelet packet. The entropy H is defined as

$$H = -\sum_i \sum_j \left[\frac{x_{ij}^2}{N}\right] \log \left[\frac{x_{ij}^2}{N}\right] \tag{11.2}$$

with the N

$$N = \sum_i \sum_j x_{ij}^2 \tag{11.3}$$

- Cluster feature representation: From the relevant regions, features representing number of microcalcifications, average size, distance, and geometry of clusters were considered, among others.

A total of 40 features are included in the feature vector describing the complete microcalcification structure information.

11.5.2 Relevant Feature Selection and Classification Results

The global search ability of the GA forms the basis for the selection of the best feature subset for microcalcification classification. The feature vector was thus reduced from 40 to 20 dimensions. The new reduced-dimension vector represents the input of a feedforward neural network with two hidden layers. The different neural architectures used for classification are summarized in Table 11.3.

TABLE 11.3 Neural network architectures to be used in connection with the selected set of features [52].

Architecture	First hidden layer	Second hidden layer
A	10 neurons	2 neurons
B	30 neurons	2 neurons
C	40 neurons	2 neurons

The results of the neural network-based classification using 80 training and 111 test cases are presented in Table 11.4. "Max" represents the best area under the ROC curve, "Ave" the average of the areas over all the partitions, and "Dev" the standard deviation over all the partitions.

TABLE 11.4 ROC performance of different neural networks [52].

Architecture	Max	Ave	Dev
A	0.81	0.72	0.06
B	0.86	0.81	0.05
C	0.82	0.78	0.05

It can be easily seen that considering more neurons in the first hidden layer does not improve the classification results. In [52] it was also shown that the neural network classifier performed better than linear classifiers and the kNN classifier.

11.6 CAD SYSTEM FOR CLUSTERED MICROCALCIFICATION DETECTION

Clustered microcalcifications represent an important early sign of breast cancer in a mammogram. They have a higher X-ray attenuation than normal breast tissue and appear as a group of small localized granular bright spots in the mammogram.

It is interesting to point out that microcalcifications represent relatively high-frequency components being embedded in the background of

low-frequency components and very high-frequency noise in the mammograms [249]. Because of its frequency and space localization, the wavelet transform is an excellent candidate for extracting relevant information on microcalcifications and discarding irrelevant detection components. In [249], the mammogram is decomposed up to level 4 using a Daubechies orthogonal wavelet of length four. In [249] it was determined that the reconstructed images from level 1 are more sensitive to background noise, while those from level 4 are more sensitive to low-frequency background in the mammogram. Thus only the images reconstructed from levels 2 and 3 are left, describing information pertinent to microcalcifications. These wavelet features and some additional gray-level statistical features form the input to the subsequent neural network.

A system for the automatic detection of clustered microcalcifications is described in [249]. The CAD system consists of two steps: the segmentation of potential microcalcification pixels and detection of individual microcalcification objects using feature extraction and pattern classification techniques. Figure 11.13 shows the flow chart of the CAD system for the automatic detection of clustered microcalcifications in digitized mammographic films.

To determine whether a segmented potential individual microcalcification object is a true positive (TP) or a false positive (FP), Karssemeijer's criterion [104] is adopted in [249]. A TP is counted if two or more objects are within a distance of 1 cm. An FP is counted if none of the found objects is within this distance.

11.6.1 Detection of Potential Microcalcification Pixels

The pixels that might belong to individual microcalcification objects are first segmented out. This is accomplished by using a set of mixed features of two wavelet features from levels 2 and 3, and two gray-level statistics features consisting of the median contrast and the normalized gray-level value (contrast-to-noise ratio). In other words, each pixel in a mammogram is represented by a mixed feature vector of the four above-mentioned features. This vector is then the input to a multilayer feedforward neural network that generates a likelihood map of the mammogram. The value in the likelihood map describes the likelihood that the represented pixel belongs to a microcalcification. A large value corresponds to a microcalcification pixel. The likelihood map is thresholded in order to generate a binary map with zeros for normal pixels and ones for microcalcification pixels. The threshold was chosen such that major individual microcalcifications are considered by the thresholded image. Connected pixels in the binary image are considered part of the individual microcalcification object, while single pixels are considered as being noise and thus are excluded.

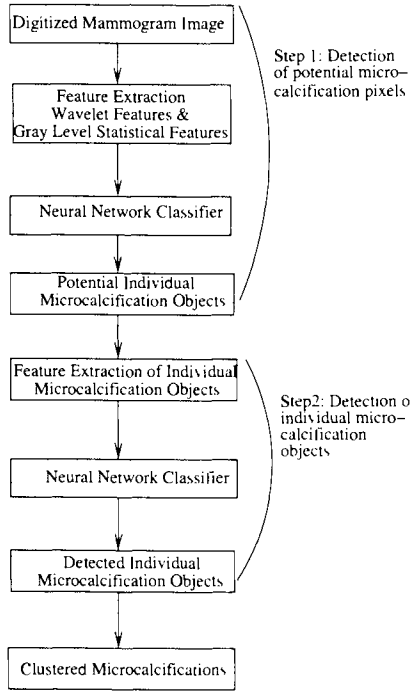


FIGURE 11.13 Flow chart of the CAD system for the automatic detection of clustered microcalcifications [249].

11.6.2 Detection of Individual Microcalcification Objects

Because of noise, pixels describing dense breast tissue and blood vessels may also exhibit a similarity to potential microcalcification pixels and are therefore called false individual microcalcification objects. It is evident that they decrease the performance of the CAD system and must therefore be eliminated [249].

The potential individual microcalcification objects are classified as true or false microcalcification objects using a set of 31 features [249]. The classification relevance of these features is analyzed using a general regression neural network (GRNN) via sequential backward and forward methods. The GRNN is a memory-based feedforward network [220] based on the estimation of the probability density function. Its main advantage is that only one pass of learning is needed for accurate classification and a relatively small number of iterations. The detected true individual microcalcification objects form a cluster.

A free-response operating characteristic (FROC) curve is chosen in [249] for evaluating the performance of the system. The authors applied their CAD system to a database of 40 mammograms containing 105 micro-

calcification clusters. They achieved a 90% mean true positive detection rate at the cost of 0.5 false positive per image, when 15 relevant features selected by the sequential backward selection method out of the initial 31 features are considered.

REFERENCES

1. A. Ahumada. Computational image quality metrics: A review. *Society for Information Display Digest of Technical Papers*, pages 305–308, 1993.

2. A. N. Akansu and R. A. Haddad. *Multiresolution Signal Decomposition.* Academic Press, 1992.

3. M. Akay. *Time Frequency and Wavelets in Biomedical Signal Processing.* IEEE Press, 1998.

4. A. Aldroubi and M. Unser. *Wavelets in Medicine and Biology.* CRC Press, 1996.

5. D. Alkon, K. Blackwell, G. Barbor, A. Rigle. and T. Vogel. Pattern-recognition by an artificial network derived from biological neuronal systems. *Biological Cybernetics*, 62:363–379, 1990.

6. S. Amari. Competitive and cooperative aspects in dynamics of neural excitation and self-organization. *Competition Cooperation Neural Networks*, 20:1–28, 1982.

7. S. Amari. Field theory of self-organizing neural nets. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-13, No. 9 & 10:741–748, 1983.

8. S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. *NIPS96*, 8:757–763, 1996.

9. S. Amartur, D. Piraino, and Y. Takefuji. Optimization neural networks for the segmentation of magnetic resonance iamges. *IEEE Trans. Med. Imaging*, 11:215–220, 1992.
10. H. Andrews. *Mathematical Techniques in Pattern Recognition.* Wiley, 1972.
11. Y. Anzai, S. Minoshima, G. Wolf, and R. Wahl. Head and neck cancer: Detection of recurrence with 3d principal component analysis at dynamic fgd pet. *Radiology*, 212:285–290, 1999.
12. S. Astley, I. Hutt, and S. Adamson. Automation in mammography: Computer vision and human perception. *Int. J. Pattern Recognit. Artificial Intell.*, 7:1313–1338, 1993.
13. H. Attias. Independent factor analysis. *Neural Computation*, 11:803–851, 1999.
14. W. Backfrieder, R. Baumgartner, M. Samal, E. Moser, and H. Bergmann. Quantification of intensity variations in functional MR images using rotated principal components. *Phys. Med. Biol.*, 41:1425–1438, 1996.
15. P. Bandettini, E. Wong, R. Hinks, R. Tikofski, and J. Hyde. Time course epi of human brain function during task activation. *Magn. Reson. Med.*, 25:390–397, 1992.
16. B. Batchelor. *Practical Approach to Pattern Classification.* Plenum Press, 1974.
17. R. Baumgartner, L. Ryder, W. Richter, R. Summers, M. Jarmasz, and R. Somorjai. Comparison of two exploratory data analysis methods for fmri: fuzzy clustering versus principal component analysis. *Magn. Reson. Imaging*, 18:89–94, 2000.
18. A. Beghdadi and A. Le Negrate. Contrast enhancement technique based on local detection of edges. *Comp. Vision Graph. Image Processing*, 46:162–174, 1989.
19. A. Bell and T. Sejnowski. Fast blind separation based on information theory. *International Symposium on Nonlinear Theory and Applications*, pages 43–47, 1995.
20. A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.
21. B. Betts, A. Aiyer, J. Li, D. Ikeda, R. Birdwell, R. M. Gray, and R. A. Olshen. Management and lesion detection effects of lossy image compression on digitized mammograms. *Proceedings of the Fourth International Workshop on Digital Mammography*, 8, 1998.
22. J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, 1981.
23. J. Bezdek, L. Hall, and L. Clarke. Review of MR image segmentation techniques using pattern recognition. *Med. Phys.*, 20:1033–1048, 1993.

24. J. Boxerman, P. Bandettini, K. Kwong, and J. Baker. The intravascular contribution to fmri signal change: Monte carlo modeling and diffusion-weighted studies in vivo. *Magn. Reson. Med.*, 34:4–10, 1995.
25. P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover Publications, 1966.
26. D. Broomhead and D. Lowe. Multivariable function interpolation and adaptive networks. *Complex Syst.*, 2:321–355, 1988.
27. D. Brzakovic, X. Luo, and P. Brzakovic. An approach to automated detection of tumors in mammography. *IEEE Trans. Med. Imaging*, 9:233–234, 1990.
28. H. Bunke and A. Sanfeliu. *Syntactic and Structural Pattern Recognition: Theory and Applications.* World Scientific, 1990.
29. C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transform.* Prentice Hall, 1998.
30. P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Communic.*, 18:532–540, 1983.
31. J. Canny. A computational approach to edge detection. *IEEE Trans. PAMI*, 8:679–698, 1986.
32. J. F. Cardoso. Blind signal separation: Statistical principles. *Proc. IEEE*, 9:2009–2025, 1998.
33. J. F. Cardoso. Multidimensional independent component analysis. *Proc. IEEE ICASSP*, 1998.
34. K. R. Castleman. *Digital Image Processing.* Prentice Hall, 1996.
35. W. Chang, H. Soliman, and A. Sung. A vector quantization neural network to compress still monochromatic images. *Proceedings of the IEEE Joint International Conference on Neural Networks in Orlando*, pages 4163–4168, 1994.
36. C. Chatterjee, V. Roychowdhhurry, and E. Chong. On relative convergence properties of principal component algorithms. *IEEE Trans. Neural Networks*, 9:319–329, 1998.
37. C. Chen, E. Tsao, and W. Lin. Medical image segmentation by a constraint satisfaction neural network. *IEEE Trans. Nucl. Sci.*, 38:678–686, 1991.
38. H. Cheng, Y. Lui, and R. Feiimanis. A novel approach to microcalcification detection using fuzzy logic techniques. *IEEE Trans. Med. Imaging*, 17:442–450, 1998.
39. Z. Cho, J. Jones, and M. Singh. *Foundations of Medical Imaging.* Wiley, 1993.
40. N. Chomsky. *Aspects of the Theory of Syntax.* MIT Press, 1965.
41. Leon O. Chua. Cellular neural networks: Theory. *IEEE Trans. on Circ. and Syst.*, 35:1257–1272, 1988.

42. K. Chuang, M. Chiu, C. Lin, and J. Chen. Model-free functional mri analysis using kohonen clustering neural network and fuzzy c-means. *IEEE Trans. on Med. Imaging*, 18:1117–1128, 8 1999.

43. E. Ciaccio, S. Dunn, and M. Akay. Biosignal pattern recognition and interpretation systems: Part i. *IEEE Eng. in Med. Biol.*, 13:89–97, 1993.

44. A. M. Cohen and S. Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Trans. on Syst. Man Cybernet.*, SMC-13:815–826, 9 1983.

45. P. Comon. Independent component analysis, a new concept? *Sig. Proc.*, 36:287–314, 1994.

46. P. C. Cosman, R. M. Gray, and R. A. Olshen. Evaluating quality of compressed medical images: snr, subjective rating, and diagnostic accuracy. *Proc. IEEE*, 82:919–932, 1994.

47. P. C. Cosman, R. M. Gray, and M. Vetterli. Vector quantazation of image subbands: A survey. *IEEE Trans. Image Processing*, 5:202–225, 1996.

48. G. Hinton D. Rumelhart and J. McClelland. *A General Framework for Parallel Distributed Processing.* Cambridge Press, 1986.

49. R. Dave. Fuzzy shell clustering and applications to circle detection in digital images. *Int. J. of Gen. Syst.*, 16:343–355, 9 1990.

50. R. Dave and K. Bhaswan. Adaptive fuzzy c–shells clustering and detection of ellipses. *IEEE Trans. Neural Networks*, 3:643–662, 1992.

51. C. Von der Malsburg. Self-organization of orientation sensitive cells in striata cortex. *Kybernetik*, 14:85–100, 1973.

52. A. P. Dhawan, Y. Chitre, C. Kaiser, and M. Moskowitz. Analysis of mammographic microcalcifications using gray-level image structure features. *IEEE Trans. Med. Imaging*, 15:246–259, 1996.

53. A. P. Dhawan and E. LeRoyer. Mammographic feature enhancement by computerized image processing. *Comp. Methods Programs Biomed.*, pages 23–25, 1988.

54. K. I. Diamantaras and S. Y. Kung. *Principal Component Neural Networks.* Wiley, 1996.

55. S. Dickson. *Investigation of the Use of Neural Networks for Computerised Medical Image Analysis.* Ph.D. thesis, University of Bristol, 1998.

56. S. Dodel, J. Herrmann, and T. Geisel. Localization of brain activity-blind separation for fmri data. *Neurocomputing*, 33:701–708, 2000.

57. D. L. Donoho. De-noising by soft-thresholding. *IEEE Trans. on Information Theory*, 41:613–627, 1995.

58. D. L. Donoho and I. M. Johnstone. De-noising by soft-thresholding. *Biometrika*, 81:425–455, 1994.

59. R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

60. D. Dumitrescu, B. Lazzerini, and L. Jain. *Fuzzy Sets and Their Application to Clustering and Training.* CRC Press, 2000.

61. A. M. Eskicioglu and P. S. Fisher. A survey of quality measures for gray scale image compression. *Proc. Space and Earth Data Compression Workshop*, pages 49–61, 1993.

62. J. Fan. *Overcomplete Wavelet Representations with Applications in Image Processing.* Ph.D., thesis University of Florida, 1997.

63. M. A. T. Figueiredo and J. M. N. Leitao. Sequential and parallel image restoration: Neural network implementations. *IEEE Trans. Neural Networks*, 3:789–801, 1994.

64. H. Fisher and J. Hennig. Clustering of functional mr data. *Proc. ISMRM 4rd Annu. Meeting*, 96:1179–1183, 1996.

65. D. B. Fogel. *Evolutionary Computation.* IEEE Press, 1995.

66. J. Frahm, K. Merboldt, and W. Hanicke. Functional mri of human brain activation at high spatial resolution. *Magn. Reson. Med.*, 29:139–144, 1992.

67. K. S. Fu. *Syntactical Methods in Pattern Recognition.* Academic Press, 1974.

68. K. S. Fu. *Syntactic Pattern Recognition with Applications.* Prentice Hall, 1982.

69. K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.

70. D. Gabor. Theory of communication. *J. IEE*, 93:429–457, 1946.

71. M. L. Giger, R. M. Nishikawa, K. Doi, F. Yin, C. J. Vyborny, R. A. Schmidt, C. E. Metz, Y. Wu, H. MacMahon. and H. Yoshimura. Development of a "smart" workstation for use in mammography. *SPIE Proceedings Series*, pages 101–103, 1991.

72. M. Girolami and C. Fyfe. An extended exploratory pursuit network with linear and nonlinear anti-hebbian lateral connections applied to the cocktail party problem. *Neural Networks*, 10:1607–1618, 1997.

73. M. Girolami and C. Fyfe. Stochastic ICA contrast maximisation using Oja's nonlinear PCA algorithm. *Int. J. of Neural Syst.*, 8:661–679, 1997.

74. R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Prentice Hall, 2002.

75. S.S. Gopal, B. Sahiner, H. P. Chan, and N. Petrick. Neural network based segmentation using a priori image models. *Proceedings of the*

IEEE Joint International Conference on Neural Networks in Houston, 3:2456–2459, 1997.

76. S. Grossberg. On learning and energy-entropy dependence in recurrent and nonrecurrent signed networks. *J. of Statist. Phys.*, 1:319–350, 1969.
77. S. Grossberg. Adaptive pattern classification and universal recording. *Biol. Cybernet.*, 23:121–134, 1976.
78. S. Grossberg. Competition, decision and consensus. *J. Math. Anal. Appl.*, 66:470–493, 1978.
79. L. Hall, A. Bensaid, L. Clarke, R. Velthuizen, M. Silbiger, and J. Bezdek. A comparison of neural network and fuzzy clustering techniques in segmenting magnetic resonance images of the brain. *IEEE Trans. on Neural Networks*, 5:672–682, 1992.
80. R. M. Haralick and L. Shapiro. *Computer and Robot Vision.* Addison-Wesley, 1992.
81. S. Haring, M. Viergever, and N. Kok. Kohonen networks for multiscale image segmentation. *Image Vision Comput.*, 12:339–344, 1994.
82. Eric J. Hartman, James D. Keeler, and Jacek M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Comput.*, 2:210–215, 1990.
83. R. Hathaway and J. Bezdek. Optimization of clustering criteria by reformulation. *IEEE Trans. on Fuzzy Syst.*, 3:241–245, 1992.
84. S. Haykin. *Neural Networks.* Maxwell MacMillan, 1994.
85. D. O. Hebb. *The Organization of Behavior.* Wiley, 1949.
86. J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.
87. J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Scie. USA*, 79:2554–2558, 1982.
88. J. J. Hopfield and D. W. Tank. Computing with neural circuits: A model. *Science*, 233:625–633, 1986.
89. P. Huber. Projection pursuit. *The Annals of Statistics*, 13:435–475, 11, 1985.
90. A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. on Neural Networks*, 10:626–634, 5, 1999.
91. A. Hyvarinen and P. Hoyer. Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural Comput.*, 12:1705–1720, 2000.
92. A. Hyvarinen, P. Hoyer and M. Inki. The Independence Assumption: Analyzing the Independence of Components by Topography. In

Girolami, M., editor, *Advances in Independent Component Analysis*, Springer, 45–62, 2000.

93. A. Hyvarinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13:411–430, 2000.
94. M. I. Jordan and R. A. Jacobs. Hierarchical mixture of experts and the em algorithm. *Neural Comput.*, 6:181–214, 1994.
95. C. Jutten and J. Herault. Blind separation of sources. *Signal Processing*, 24:1–10, 1991.
96. A. Kalukin, M. Van Geet, and R. Swennen. Principal component analysis of multienergy X-ray computed tomography of mineral samples. *IEEE Trans. on Nucl. Sci.*, 47:1729–1736, 2000.
97. N. Kambhatla, S. Haykin, and R. Dony. Image compression using klt, wavelets and an adaptive mixture of principal components model. *J. VLSI Signal Processing Syst.*, 18:287–297, 1998.
98. I. Kapouleas. Segmentation and feature extraction for magnetic resonance brain image analysis. *ICPR90*, 1:583–590, 1990.
99. N. Karayiannis. A methodology for constructing fuzzy algorithms for learning vector quantization. *IEEE Trans. Neural Networks*, 8:505–518, 1997.
100. N. Karayiannis and P. Pai. Fuzzy algorithms for learning vector quantization. *IEEE Trans. Neural Networks*, 7:1196–1211, 5 1996.
101. J. Karhunen and J. Joutsensalo. Representation and separation of signals using nonlinear PCA type learning. *Neural Networks*, 7:113–127, 1994.
102. J. Karhunen, E. Oja, L. Wang, R. Vigario, and J. Joutsensalo. A class of neural networks for independent component analysis. *IEEE Trans. on Neural Networks*, 8:486–504, 1997.
103. N. Karssemeijer. Stochstic model for automated detection of calcification in digital mammograms. *Image Vision Comput.*, 10:369–375, 1992.
104. N. Karssemeijer. Adaptive noise equalization and recognition of microcalcification clusters in mammograms. *Int. J. Pattern Recognition Artificial Intelligence*, pages 1357–1376, 1993.
105. N. Karssemeijer and G. te Brake. Detection of stellate distortions in mammogram. *IEEE Transaction on Medical Imaging*, 15:611–619, 8 1996.
106. R. L. Kashyap and A. Khotanzad. A model based method for rotation invariant texture classification. *IEEE Trans. on PAMI*, 8:472–481, 1986.
107. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour model. *Int. J. on Comp. Vision*, 1:321–331, 1988.
108. A. G. Katsaggelos. *Digital Image Processing.* Springer Verlag, 1991.

109. W. P. Kegelmeyer, J. M. Pruneda, and P. D. Bourland. Computer-aided mammographic screening for spiculated lesions. *Radiology*, 191:331–337, 1994.

110. D. N. Kennedy, P. A. Filipek, and V. S. Caviness. Anatomic segmentation and volumetric calculations in nuclear magnetic resonace imaging. *IEEE Trans. Med. Imaging*, 1:1-7, 1989.

111. H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

112. V. Kim and L. Yaroslavskii. Rank algorithms for picture processing. *Comp. Vision Graph. Image Processing*, 35:234–259, 1986.

113. J. Kittler. Feature selection and extraction. *Handbook of Pattern Recognition Image Processing*, 1:59–83, 1986.

114. T. Kohonen. Self-organized formation of topologically correct feature maps. *Biol. Cybernet.*, 43:59-69. 1982.

115. T. Kohonen. *Self-Organization and Associative Memory*. Springer Verlag, 1988.

116. T. Kohonen. Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map. *Biol. Cybernet.*, 75:281–291, 1996.

117. I. Koren, A. Laine, and F. Taylor. Image fusion using steerable dyadic wavelet transform. *ICIP95*, 3:232–235, 1995.

118. I. Koren, A. Laine, and F. Taylor. Enhancement via fusion of mammographic features. *ICIP98*, 3:722-726, 1998.

119. B. Kosko. Adaptive bidirectional associative memory. *Appl. Optics*, 26:4947–4960, 1987.

120. B. Kosko. *Neural Networks and Fuzzy Systems*. Prentice Hall, 1992.

121. C. Kotropoulos, X. Magnisalis, I. Pitas, and M. G. Strintzis. Nonlinear ultrasonic image processing based on signal-adaptive filters and self-organizing neural networks. *IEEE Trans. on Image Processing*, 3:65–77, 1994.

122. M. Kupinski and M. Giger. Automated seeded lesion segmentation on digital mammograms. *IEEE Trans. on Med. Imaging*, 17:510–517, 8, 1998.

123. K. Kwong. Functional magnetic resonance imaging with echo planar imaging. *Magn. Reson. Q.*, 11:1-20, 1992.

124. K. Kwong, J. Belliveau, and D. Chesler. Dynamic magnetic resonance imaging of human brain activity during primary sensor stimulation. *Proc. Nat. Acad. Scie. USA*, 89:5675–5679, 8 1992.

125. A. Laine, W. Huda, D. Chen, and J. Harris. Segmentations of masses using continous scale representations. *Proceedings of the Third International Workshop on Digital Mammography*, 1996.

126. A. Laine, I. Koren, J. Fan, and F. Taylor. A steerable dyadic wavelet transform and interval wavelets for enhancement of digital mammography. *SPIE Proceedings Series*, 2491:736–749, 1995.

127. A. F. Laine and J. Fan. Texture discrimination by wavelet packet signatures. *IEEE Trans. on PAMI*, 15:1186–1191, 1993.

128. A. F. Laine, J. Fan, and W. Yang. Wavelets for contrast enhancement of digital mammography. *IEEE Eng. Med. Biol.*, 15:536–550, 1995.

129. A. F. Laine, A. Meyer-Bäse, W. Huda. J. Honeyman, and B. Steinbach. Local enhancement of masses via multiscale analysis. Technical Report 4, University of Florida. 1996.

130. A. F. Laine, S. Schuler, J. Fan. and W. Huda. Mammographic feature enhancement by multiscale analysis. *IEEE Trans. on Med. Imaging*, 13:725–740, 1994.

131. M. Lanyi. *Diagnosis and Differential Diagnosis of Breast Calcifications.* Springer Verlag, 1986.

132. K. Laws. Rapid texture identification. *SPIE Proceedings Series 238*, pages 376–380, 1980.

133. K. I. Laws. *Textured Image Segmentation.* Ph.D. thesis, University of Southern California, 1980.

134. R. S. Ledley and L. B. Lusted. Reasoning foundations of medical diagnosis. *MD Comput.*, 8:300–315, 1991.

135. Sukhan Lee and Rhee M. Kil. A gaussian potential function network with hierarchically self-organizing learning. *Neural Networks*, 4:207–224, 1991.

136. T. W. Lee, M. Girolami, and T. J. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. *Neural Comput.*. 11:409–433, 1999.

137. C. Leondes. *Image Processing and Pattern Recognition.* Academic Press, 1998.

138. M. D. Levine. *Vision in Man and Machine.* McGraw-Hill, 1985.

139. H. Li, Y. Wang, S. Lo, and M. Freedman. Computerized radiographic mass detection-part 1. *IEEE Trans. Med. Imaging*, 20:289–301, 2001.

140. H. Li, Y. Wang, S. Lo, and M. Freedman. Computerized radiographic mass detection–part 2. *IEEE Trans. Med. Imaging*, 20:302–313, 2001.

141. J. Li, N. Chaddha, and R. M. Gray. Multiresolution tree structured vector quantization. *Asilomar Conference on Signals, Systems and Computer*, 1996.

142. Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. Communications*, 28:84–95, 1980.

143. R. P. Lipmann. An introduction to computing with neural networks. *IEEE ASSP Mag.*, 4:4–22. 1987.

144. S. B. Lo, H. P. Chan, J. Lin, M. T. Freedman, and S. K. Mun. Artificial convolution neural network for medical image pattern recognition. *Neural Networks*, 8:1201–1214, 1995.

145. S. B. Lo, J. Lin, M. T. Freedman, and S. K. Mun. Application of artificial neural networks to medical image pattern recognition: Detection of clustered microcalcifications on mammograms and lung cancer on chest radiographs. *J. VLSI Signal Processing*, 18:263–274, 1998.

146. C. Looney. *Pattern Recognition Using Neural Networks.* Oxford University Press, 1997.

147. J. Lu, D. Healy, and J. Weaver. Contrast enhancement of medical images using multiscale edge representation. *Opt. Eng.*, 33:2151–2161, 1994.

148. A. Macovski. *Medical Imaging Systems.* Prentice Hall, 1983.

149. J. Makhoul. Linear prediction: A tutorial review. *Proc. IEEE*, 63:561–580, 1975.

150. S. Mallat. Zero-crossings of a wavelet transform. *IEEE Trans. Information Theory*, 37:1019–1033, 1991.

151. S. Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, 1997.

152. S. Mallat and S. Zhong. Characterization of signals from multiscale edges. *IEEE Trans. on PAMI*, 14:710–732, 1992.

153. J. Mao and A. K. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25:173–188, 1992.

154. J. A. Marshall. Adaptive perceptual pattern recognition by self-organizing neural networks. *Neural Networks*, 8:335–362, 11995.

155. T. Martinetz, S. Berkovich, and K. Schulten. Neural gas network for vector quantization and its application to time-series prediction. *IEEE Trans. on Neural Networks*, 4:558–569, 1993.

156. K. Matsuoka and M. Kawamoto. A neural network that self-organizes to perform three operations related to principal component analysis. *Neural Networks*, 7:753–765, 1994.

157. M. McKeown, T. Jung, S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski. Analysis of fmri data by blind separation into independent spatial components. *Human Brain Mapping*, 6:160–188, 1998.

158. M. McKeown, T. Jung, S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski. Spatially independent activity patterns in functional magnetic resonance imaging data during the stroop color-naming task. *Proc. Natl. Acad. Sci. USA*, 95:803–810, 1998.

159. D. Meersman, P. Scheunders, and D. VanDyck. Detection of microcalcifications using neural networks. *Digital Mammography '96*, pages 287–290, 1996.

160. W. Meisel. *Computer-Oriented Approaches to Pattern Recognition.* Academic Press, 1972.

161. C. E. Metz. ROC methodology in radiologic imaging. *Invest. Radiol.*, 21:720–733, 7 1986.

162. A. Meyer-Bäse. On the existence and stability of solutions in self-organizing cortical maps. *IEICE Trans. Fund. Electron., Communic. Comp. Scie.*, E82-A No.9:1883–1887, 1999.

163. A. Meyer-Bäse, F. Ohl, and H. Scheich. Singular perturbation analysis of competitive neural networks with different time-scales. *Neural Comput.*, pages 545–563, 1996.

164. Z. Michalewicz. *Genetic Algorithms.* Springer Verlag, 1995.

165. L. Miclet. *Structural Methods in Pattern Recognition.* Springer Verlag, 1986.

166. J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Comput.*, 1:281–295, 1989.

167. B. Nabet and R. Pinter. *Sensory Neural Networks.* CRC Press, 1991.

168. A. N. Netravali and B. G. Haskell. *Digital Pictures. Representation and Compression.* Plenum Press, 1988.

169. S. Ngan and X. Hu. Analysis of fmri imaging data using self-organizing mapping with spatial connectivity. *Magn. Reson. Med.*, 41:939–946, 1999.

170. N. J. Nilsson. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems.* McGraw-Hill, 1965.

171. R. M. Nishikawa, M. L. Giger, K. Doi, C. J. Vyborny, R. A. Schmidt, C. E. Metz, Y. Wu, Yin F.-F, Y. Jiang, Z. Huo, P. Lu, W. Zhang, T. Ema, U. Bick, J. Papaioannou, and R. H. Nagel. Computer-aided detection and diagnosis of masses and clustered microcalcifications from digital mammograms. *SPIE Proceedings Series 1905*, pages 422–432, 1993.

172. S. Ogawa, T. Lee, and B. Barrere. The sensitivity of magnetic resonance image signals of a rat brain to changes in the cerebral venous blood oxygenation activation. *Magn. Reson. Med.*, 29:205–210, 1993.

173. S. Ogawa, D. Tank, and R. Menon. Intrinsic signal changes accompanying sensory stimulation: Functional brain mapping with magnetic resonance imaging. *Proc. Nat. Acad. Scie. USA*, 89:5951–5955, 1992.

174. M. Ogiela and R. Tadeusiewicz. Syntactic reasoning and pattern recognition for analysis of coronary artery images. *Artificial Intelligence Med.*, 670:1–15, 2002.

175. M. R. Ogiela and R. Tadeusiewicz. Advances in syntactic imaging techniques for perception of medical images. *J. Imaging Sci.*, 49:113–120, 2001.

176. M. R. Ogiela and R. Tadeusiewicz. New aspects of using the structured graph-grammar based technique for recognition of selected medical images. *J. Digital Imaging*, 14:231–232, 2001.

177. E. Oja. A simplified neural model as a principal component analyzer. *J. Math. Biol.*, 15:267–273, 1982.

178. E. Oja. Neural networks, principal components, and subspaces. *Int. J. Neural Syst.*, 1:61–68, 1989.

179. E. Oja. Self-organizing maps and computer vision. *Neural Networks Percept.*, 1:368–385, 1992.

180. E. Oja, H. Ogawa, and J. Wangviwattana. Learning in nonlinear constrained hebbian networks. *Artificial Neural Networks*, 1:385–390, 1991.

181. T. Otto, A. Meyer-Bäse, M. Hurdal. D. Sumners, D. Auer, and A. Wismüller. Model-free functional mri analysis using transformation-based methods. *Proceeding of SPIE, Vol. 5102*, 156–167, 2003.

182. J. K. Paik and A. K. Katsagellos. Image restoration using a modified Hopfield network. *IEEE Trans. on Image Processing*, 1:49–63, 1992.

183. N. Pal, J. Bezdek, and E. Tsao. Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Trans. on Neural Networks*, 4:549–557, 1993.

184. S. Pal, A. Ghosh, and M. Kundu. *Soft Computing for Image Processing.* Springer, 2000.

185. S. Pal and S. Mitra. *Neuro-Fuzzy Pattern Recognition.* J. Wiley, 1999.

186. A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 1986.

187. J. Park and I. Sandberg. Universal approximation using radial-basis-function networks. *Neural Comput.*, 3:247–257, 1991.

188. F. Pedersen, M. Bergstrom, E. Bengtsson, and B. Langstrom. Principal component analysis of dynamic positron emission tomography images. *Eur. J. of Nucl. Med.*, 21:1285–1292, 1994.

189. S. M. Perlmutter, P. C. Cosman, R. M. Gray, R. A. Olshen, D. Ikeda, C. N. Adams, B. J. Betts, M. Williams, K. O. Perlmutter, J. Li, A. Aiyer, L. Fajardo, R. Birdwell, and B. L. Daniel. Image quality in lossy compressed digital mammograms. *Signal Processing*, 180–210, 1997.

190. N. Petrick, H. Chan, B. Sahiner, M. Helvie, M. Goodsitt, and D. Adler. Computer-aided breast mass detection: False positive reducing using breast tissue composition. *Excerpta Med.*, 1119:373–378, 1996.

191. N. Petrick, H. Chan, B. Sahiner, and D. Wei. An adaptive density-weighted contrast enhancement filter for mammographic breast mass detection. *IEEE Trans. on Med. Imaging*, 15:59–67, 1996.

192. D. T. Pham and P. Garat. Blind separation of mixture of independent sources through a quasimaximum likelihood approach. *IEEE Trans. on Signal Processing*, 45:1712–1725, 1997.

193. John Platt. A resource-allocating network for function interpolation. *Neural Comp.*, 3:213–225, 1991.

194. T. Poggio and F. Girosi. Extensions of a theory of networks for approximations and learning: Outliers and negative examples. *Touretky's Connectionist Summer School*, 3:750–756, 6, 1990.

195. Tomaso Poggio and Federico Girosi. Networks and the best approximation property. *Biol. Cybernet.*, 63:169–176. 1990.

196. Tomaso Poggio and Federico Girosi. Networks for approximation and learning. *Proc. IEEE*, 78:1481–1497, 1990.

197. Tomaso Poggio, Michael Jones, and Federico Girosi. Regularization theory and neural networks architectures. *Neural Comput.*, 7:219–269, 1995.

198. W. K. Pratt. *Digital Image Processing.* Wiley, 1978.

199. F. P. Preparata and M. I. Shamos. *Computational geometry: An Introduction.* Springer Verlag, 1988.

200. P. Pudil, J. Novovicova, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Lett.*, 15:1119–1125, 1994.

201. W. Qian, H. Li, M. Kallergi, D. Song, and L. Clarke. Adaptive neural network for nuclear medicine image restoration. *J. VLSI Signal Processing Syst.*, 297–315, 1998.

202. L. R. Rabiner and R. Schafer. Digital representation of speech signals. *Proceedings of the IEEE*, 63:662–677. 1975.

203. G. X. Ritter and J. N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra.* CRC Press, 1996.

204. G. P. Robinson, A. C. F. Colchester, and L. D. Griffin. Model-based recognition of anatomical objects from medical images. *Image and Vision Comput.*, 12:499–507, 1994.

205. F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.*, 65:386–408, 1958.

206. Z. Roth and Y. Baram. Multidimensional density shaping by sigmoids. *IEEE Trans. on Neural Networks*, 7:1291–1298, 1996.

207. Ali Saberi and Hassan Khalil. Quadratic-type lyapunov functions for singularly perturbed systems. *IEEE Trans. on Automatic Control*, 542–550, 1984.

208. E. Säckinger, B. E. Boser. J. Bromley, Y. LeCun, and L. D. Jackel. Application of a neural network chip to high-speed character recognition. *IEEE Trans. on Neural Networks*, 3:498–505, 1992.

209. A. Saha, D. S. Christian, D. S. Tang, and C. L. Wu. Oriented nonradial basis functions for image coding and analysis. *Touretky's Connectionist Summer School*, 2:728–734, 1991.

210. B. Sahiner, H. P. Chan, N. Petrick, D. Wei, M. A. Helvie, D. Adler, and M. M. Goodsitt. Classification of mass and normal breast tissue: A convolution neural network classifier with spatial domain and texture images. *IEEE Trans. on Med. Imaging*, 15:598–610, 1996.

211. A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circ. Syst. for Video Technol.*, 6:243–250, 1996.

212. S. Saito and K. Nakato. *Fundamentals of Speech Signal Processing.* Academic Press, 1985.

213. P. Sajda, C. D. Spence, J. C. Pearson, and R. M. Nishikawa. Exploiting context in mammograms: A hierarchical neural network for detecting microcalcifications. *SPIE Proceedings Series 2710*, pages 733–742, 1996.

214. T. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 12:459–473, 1989.

215. G. Scarth, M. McIntrye, B. Wowk, and R. Samorjai. Detection novelty in functional imaging using fuzzy clustering. *Proc. SMR 3rd Annu. Meeting*, 95:238–242, 1995.

216. R. Schalkoff. *Pattern Recognition.* Wiley, 1992.

217. J. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Processing*, 41:3445–3462, 1993.

218. L. Shapiro and G. Stockman. *Computer Vision.* Prentice-Hall, 2001.

219. W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Lett.*, 10:335–347, 1989.

220. D. Specht. A general regression neural network. *IEEE Trans. Neural Networks*, 2:568–597, 1991.

221. S. D. Stearns and D. R. Hush. *Digital Signal Analysis.* Prentice-Hall, 1990.

222. M. Stone. Cross-validatory choice and assessment of statistical predictions. *J. R. Statist. Soc.*, 111–133, 1974.

223. R. Strickland and H. Hahn. Wavelet transform for detecting microcalcifications in mammograms. *IEEE Trans. Med. Imaging*, 15:218–229, 1996.

224. M. Strintzis. Optimal pyramidal and subband decomposition for hierarchical coding of noisy and quantized images. *IEEE Trans. Image Processing*, 7:155–166, 1998.

225. A. Sudjianto, M. Hassoun, and G. Wasserman. Extension of principal component analysis for nonlinear feature extraction. *SPIE Proceedings Series 3210*, pages 1433–1438, 1996.

226. Y. Sun, J. G. Li, and S. Y. Wu. Improvement on performance of modified hopfield neural network for image restoration. *IEEE Trans. Image Processing*, 4:688–692, 1995.

227. J. Sychra, P. Bandettini, N. Bhattacharya, and Q. Lin. Synthetic images by subspace transforms i. principal components images and related filters. *Med. Phys.*, 21:193–201, 1994.

228. L. Tabar and P. B. Dean. *Teaching Atlas of Mammography.* Thieme Inc., 1985.

229. S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Academic Press, 1998.

230. A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems.* W. H. Winston, 1977.

231. A. Toet. Multiscale contrast enhancement with application to image fusion. *Opt. Eng.*, 31:1026–1030, 1992.

232. P. Trahanias and E. Skordalakis. Syntactic pattern recognition of the ecg. *IEEE Trans. PAMI*, 12:648–657, 1990.

233. O. Tsujii, A. Hasegawa, C. Wu, B. Lo, M. Freedman, and S. Mun. Classification of microcalcifications in digital mammograms for the diagnosis of breast cancer. *SPIE Proceedings Series 2710*, 794–804, 1996.

234. D. Tzovaras and M. Strintzis. Use of nonlinear principal component analysis and vector quantization for image coding. *IEEE Trans. Image Processing*, 7:1218–1223, 1998.

235. P. P. Vaidyanathan. *Multirate Systems and Filterbanks.* Prentice Hall, 1993.

236. R. Vigario, V. Jousmaki, M. Hamalainen, R. Hari, and E. Oja. Independent component analysis for identification of artifacts in magnetoencephalographic recordings. *NIPS98*, 10:229–235, 1998.

237. D. Walnut. *An Introduction to Wavelet Analysis.* Birkhäuser, 2001.

238. D. C. Wang, A. H. Vagnucci, and C. C. Li. Digital image enhancement: A survey. *IEEE Trans. Information Theory*, 24:363–381, 1983.

239. S. Webb. *The Physics of Medical Imaging.* Adam Hilger, 1990.

240. D. Wei, H. P. Chan, M. A. Helvie, B. Sahiner, N., D. D. Adler, and M. M. Goodsitt. Classification of mass and normal breast tissue on digital mammograms: Multiresolution texture analysis. *Med. Phys.*, 22:1501–1513, 1995.

241. D. Whitley. Genetic algorithm tutorial. *Statist. Comput.*, 4:65–85, 1994.

242. D. J. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self-organization. *Proc. R. Soc. London B*, 194:431–445, 1976.

243. J. W. Woods. *Subband Image Coding.* Kluwer Academic Publishers, 1991.

244. K. S. Woods. *Automated Image Analysis Techniques for Digital Mammography.* Ph.D. thesis. University of South Florida, 1994.

245. K. S. Woods, J. L. Solka, C. E. Priebe, C. C. Doss, K. W. Boyer, and L. P. Clarke. Comparative evaluation of pattern recognition techniques for detection of microcalcifications. *SPIE Proceedings Series 1905*, pages 841–852, 1993.

246. Y. Wu, M. Giger, K. Doi, and R. M. Nishikawa. Computerized detection of clustered microcalcifications in digital mammograms: Applications of artificial neural networks. *Med. Phys.*, 19:555–560, 1992.

247. W. Yan, U. Helmke, and J. Moore. Global analysis of oja's flow for neural networks. *IEEE Trans. Neural Networks*, 5:674–883, 1994.

248. T. Y. Young and K. S. Fu. *Handbook of Pattern Recognition and Image Processing.* Academic Press. 1986.

249. S. Yu and L. Guan. A CAD system for the automatic detection of clustered microcalcifications in digitized mammogram films. *IEEE Trans. Med. Imaging*, 19:115- 126, 2000.

250. L. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.

251. M. Zhang, M. Giger, C. Vyborny, and K. Doi. Mammographic texture analysis for the detection of spiculated lesions. *Excerpta Med.*, 1119:347–351, 1996.

252. W. Zhang, M. Giger, K. Doi, Y. Wu. R. Nishikawa, and R. Schmidt. Computerized detection of masses in digital mammograms using a shift-invariant artificial neural network. *Med. Phys.*, 21:517–524, 1994.

253. W. Zhang, Y. Wu, M. Giger, K. Doi, and R. M. Nishikawa. Computerized detection of clustered microcalcifications in digital mammograms using a shift-invariant neural network. *Med. Phys.*, 21:517–524, 1994.

254. Y. Zhu and H. Yan. Computerized tumor boundary detection using a hopfield neural network. *Proc. IEEE Joint Int. Conf. Neural Networks Houston*, 3:2467–2472, 1997.

255. X. Zong, A. Meyer-Bäse, and A. Laine. Multiscale segmentation through a radial basis neural network. *IEEE Int. Conf. on Image Processing*, pages 400–403, 1997.

256. R. Zwiggelaar, T. Parr, J. Schumm, and I. Hutt. Model-based detection of spiculated lesions in mammograms. *Med. Image Anal.*, 3:39–62, 1999.

INDEX

A

Activation function, 189
Active contour model, 321
Adaptive fuzzy n-shells, 307
Adaptive Principal Component Extraction, 244
Affine wavelet, 74
Alphabet, 157
Alternating optimization technique, 292
Analysis bank, 51
Analysis modulation matrix, 60
Attributed graph, 33
Autoregressive (AR) process, 20

B

Backpropagation, 192
Bayesian estimation, 139
Best-approximation, 207
Between-class scatter matrix, 45
Binary decision trees, 154
Biorthogonal filter bank, 83
Bit rate, 341
Bit string, 123
Blind source separation, 255
Branch and bound algorithm, 42
Building blocks, 132

C

Canny edge detector, 105
Central moments, 19
Chromosome, 122
Classification, 190
  pixel-based, 225
  region-based, 226
Closure set, 158
Clustering, 137
  agglomerative, 138
  c-means, 137
  divisive, 138
  hierarchical, 138
  k-means, 137
Co-occurrence matrix, 38
Cocktail party problem, 255
Codebook, 201
Codewords, 201
Committee machines, 195
Competitive learning law, 240
Complement, 284
Compression ratio, 341
Computer-aided diagnosis (CAD), 8
Computer-assisted tomography (CAT), 9
Confusion matrix, 228
Conjugate quadrature mirror filters, 61

Constant-Q subband tree, 63
Content addressable memory, 241
Continuous wavelet transform, 71
Contrast enhancement, 108
Contrast function, 260
Convolution neural networks, 322
Crisp set, 283
Cross-modulation, 60
Cross-validation, 194
Crossover, 124
Crowding, 132
Curse of dimensionality, 216

D

Decimation, 52
Delaunay triangulation, 203
Denoising, 110
Directed graph, 33
Discrete cosine transform, 31
Discrete Fourier transform, 29
Discrete sine transform, 32
Discrete wavelet transform, 71
Discriminant function, 142
Dissimilarity graph, 165
Dissimilarity, 291
Downsampling, 51

E

Early stopping, 195
Error-correction learning, 215
Evaluation function, 125
Exhaustive search, 42

F

Fast ICA, 267
Feasibility threshold, 134
Feature extraction, 16
Feature map, 197
Feature selection, 16
Feature, 14
Filter bank, 51
Finite-state machine, 160
Fitness function, 125
Fitness value, 125
Floating search method, 44
Functional magnetic resonance imaging (fMRI), 12
Fundamental wavelet equation, 79
Fuzzifier, 294
Fuzzy clustering, 288
Fuzzy learning vector quantization, 313
Fuzzy neural network, 287
Fuzzy partition, 285
Fuzzy set, 284

G

Generalized adaptive fuzzy n-means, 299
Generalized adaptive fuzzy n-shells, 306
Generalized fuzzy n-means, 293
Generalized fuzzy n-shells, 300, 302
Generalized Hebbian Algorithm, 242
Genetic algorithms, 121
Geometric moment, 18
Gradient-descent procedure, 215
Grammar, 33
Grammatical inference, 162

H

Haar wavelet, 80
Hebbian learning law, 239
Heisenberg Uncertainty Principle, 68
Hidden layers, 189
Hierarchical mixture of experts, 196
Hierarchical pyramid neural network, 325
Hopfield neural network, 219
Hotelling transform, 25

I

Image fusion, 115
Image restoration, 328
Image segmentation, 332
Inadequacy, 292
Independent component analysis, 254
Independent subspace, 272
Infomax, 266
Input layer, 189
Instance, 130
Interpolation network, 205
Interpolation, 52
Invariant-feature subspaces, 271

K

K-nearest-neighbor density estimation, 150
Karhunen-Loeve transformation, 25
Kullback divergence, 263
Kurtosis, 261

L

Language, 158
Laplacian pyramid, 64
Lattice of neurons, 198
Laws texture energy, 40
Learning vector quantization, 201
Levinson-Durbin-algorithm, 21
Likelihood function, 140
Linear Oja's learning algorithm, 247

Loglikelihood function, 146

M

Magnetic resonance imaging (MRI), 11
Max-min feature selection, 42
Maximum likelihood estimation, 139
Membership degree, 283
Membership function, 283
Membership matrix, 290
Mexican-hat, 72
Mirror filter, 57
Mixture of experts, 196
Modified Hopfield network, 331
Moment, 18
Multiscale edge detection, 105
Multiscale signal decomposition, 63
Multiresolution, 76
Mutation, 124

N

Negentropy, 262
Neighborhood function, 200
Nesting effect, 44
Neural-gas network, 202
Neurodynamics, 238
Neurons, 188
Non-ambiguous, 284
Nonlinear classification, 191
Nonlinear Oja's learning algorithm, 247
Nonparametric estimation, 148
Norm-induced distance, 290
Normalized central moments, 19
Nuclear medicine, 10

O

Offspring, 123
Orthogonal filter bank, 83
Orthonormal filters, 61
Output layer, 189

P

Parametric estimation, 139
Parsing, 157
Partition matrix, 296
Parzen windows, 148
Pattern vector, 17
Peak signal-to-noise ratio, 344
Perfect reconstruction filter bank, 52
Pixel labeling, 223
  contextual, 224
  non-contextual, 223
Population, 123
Positron emission tomography (PET), 11
Potential functions, 151
Principal component analysis, 25
Problem factorization, 327
Propagation rule, 189

Q

Quadrature mirror filter, 58

R

Radial-basis functions, 210
Radial-basis neural networks, 204
Ranking order curves, 228
Receiver operating characteristic, 167
Receptive field, 210, 323
Regularization network, 209
Regularization term, 208
Relational graph, 33
Relative contribution, 312

S

Scaling functions, 76
Schema
  defining length, 131
  order, 131
Schemata, 130
Selection, 124
Self-organizing map, 197
Sensitivity, 168
Sentences, 158
Sequential backward selection, 44
Sequential forward selection, 43
Short-time Fourier transform, 67
Single photon emission computed tomography (SPECT), 10
Singular value decomposition, 29
Spatial width, 323
Specificity, 168
Spectral theorem, 26
Stochastic approximation, 210
String matching, 157
Structural pattern recognition, 156
Subband decomposition, 51
Subspace decomposition, 28
Symbols, 157
Synaptic connections, 188
Synaptic dynamics, 239
Syntactic pattern recognition, 156
Syntax, 159
Synthesis bank, 51

T

Texture energy map, 40
Texture, 34
Thresholding, 109

Topographic independent component analysis, 271
Topological ordering, 202
Topology-preserving map, 203
Transformation radial basis neural network, 216
Two-channel filter bank, 58

U

Ultrasound, 12
Ultrasonic signature, 13
Universal approximator, 207
Universe of discourse, 283
Unsharp masking, 113
Upsampling, 51

V

Vector quantization, 201
Voronoi quantizer, 201

W

Wavelet functions, 78
Wavelet transform, 69
Weight kernel, 323
Winning neuron, 198
Within-class scatter matrix, 45

X

X-rays, 9
XOR problem, 191